МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра інформаційних технологій

«До захисту допущено»

В.о. завідувача кафедри
Світлана ВАЩЕНКО

КВАЛІФІКАЦІЙНА РОБОТА
на здобуття освітнього ступеня бакалавр

зі спеціальності 122 «Комп'ютерні науки»,

освітньо-професійної програми «Інформаційні технології проектування»

на тему: Web-додаток моніторингу електроспоживання комунальними та промисловими об'єктами

Здобувача групи ІТ-91 Сенчі Егора Сергійовича
(шифр групи) (прізвище, ім'я, по батькові)

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

_____ Егор СЕНЧА
(підпис) (Ім'я та ПРІЗВИЩЕ здобувача)

Керівник к.н.т, доц. Анна МАРЧЕНКО _____
(посада, науковий ступінь, вчене звання, Ім'я та ПРІЗВИЩЕ) (підпис)

Суми – 2023

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра інформаційних технологій
Спеціальність 122 «Комп'ютерні науки»
Освітньо-професійна програма «Інформаційні технології проектування»

ЗАТВЕРДЖУЮ
в.о. заф. каф

_____ Ващенко С.М.
«__» _____ 2023 р.

З А В Д А Н Н Я
НА КВАЛІФІКАЙІЙНЕ РОБОТУ БАКАЛАВРА СТУДЕНТУ

Сенчі Егору Сергійовичу

1 Тема роботи *Web-додаток моніторингу електроспоживання комунальними та промисловими об'єктами*

керівник роботи *Марченко Анна Вікторівна, к.т.н., доцент*,

затверджені наказом по університету від «29»травня 2023 р. №0588-VI

2 Строк подання студентом роботи «7» червня 2023 р.

3 Вхідні дані до роботи Співбесіда із замовником

4 Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Аналіз предметної області, моделювання та проєктування web-додатку, розробка web-додатку

5 Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Актуальність, Постановка задачі, Результат аналізу аналогів, Інструменти та технології, Діаграма процесів в нотації IDEFO. Концептуальний рівень, Діаграма бізнес-процесу в нотації IDEFO. 1-ий рівень декомпозиції, Діаграма варіантів використання, Логічна модель даних, Алгоритм обробки даних електроспоживання, Алгоритм агрегації, Архітектура серверної частини, Демонстрація отримання даних споживання у табличній формі, Демонстрація отримання даних споживання у графічній формі, Демонстрація перегляду списку об'єктів, Демонстрація редагування об'єкту, Демонстрація додавання об'єкту, Акт впровадження

6. Консультанти розділів роботи:

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7.Дата видачі завдання 8 лютого 2023

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Отримання завдання на проєктування	Вересень 2022	
2	Аналіз аналогів	Вересень 2022	
3	Аналіз та вибір інструментів	Вересень 2022	
4	Загальний дизайн архітектури додатку	Вересень – Жовтень 2022	
5	Дизайн серверної частини додатку	Жовтень 2022	
6	Розробка макетів користувацької частини	Листопад 2022	
7	Розробка додатку	Листопад 2022 – Травень 2023	
8	Тестування	Травень 2023	
9	Оформлення документації	Травень 2023	
10	Захист роботи	Червень 2023	

Студент _____ Сенча Є.С.
(підпис)

Керівник роботи _____ к.т.н., доц. Марченко А.В.
(підпис)

РЕФЕРАТ

Тема кваліфікаційної роботи бакалавра «Web-додаток моніторингу електроспоживання комунальними та промисловими об’єктами».

Пояснювальна записка складається зі вступу, 4 розділів, висновків, списку використаних джерел із 16 найменувань, додатків. Загальний обсяг роботи – 225 сторінок, у тому числі 48 сторінок основного тексту, 2 сторінки списку використаних джерел, 174 сторінок додатків.

У ході виконання роботи було проведено дослідження проблем і актуальних питань у сфері споживання електроенергії промисловими чи комунальними об’єктами. Також було проаналізовано існуючі продукти на ринку, що слугують для вирішення цих питань та проблем.

На основі цього аналізу було сформовано ключову мету web-додатку. Для його реалізації було обрано оптимальний набір інструментів та технологій, шляхом аналізу підходів, що використовуються зараз.

У ході реалізації додатку було проведено проведено проєктування його основних процесів, визначено основні варіанти використання. На основі цього було реалізовано сам додаток та уведено в експлуатацію.

Практичне значення додатку полягає у наданні можливості просто та гнучно відслідковувати електроспоживання у режимі реального часу.

Ключові слова: енергоспоживання, web-додаток, Python, Golang, моніторинг електроспоживання.

ЗМІСТ

ВСТУП

В сучасному світі питання енергоефективності та збереження електроенергії набувають все більшої актуальності. Зростання населення, розвиток технологій та промисловості призводять до збільшення попиту на електроенергію. Одночасно з цим, природні ресурси планети - обмежені, а проблеми зміни клімату вимагають пошуку енергоефективних рішень.

Саме тому, моніторинг електроспоживання стає невід'ємною частиною раціонального використання електроенергії. Здійснення контролю над споживанням дозволяє ефективно управляти енергетичними системами, виявляти непотрібні витрати, забезпечуючи можливість їх оптимізації.

Моніторинг електроспоживання надає користувачам важливі дані щодо їхньої енергетичної ефективності. Він дозволяє виявити основні джерела споживання енергії, встановити пікові навантаження, спостерігати за змінами у споживанні та реагувати на них. Крім того, моніторинг є важливим інструментом для ідентифікації потенційних проблем у системі енергозабезпечення та вчасного виявлення несправностей.

Web-додаток для моніторингу електроспоживання має ряд переваг, які роблять його привабливим та ефективним інструментом для контролю енергетичних ресурсів. Ось кілька причин, чому саме web-додаток є вигідним вибором:

- Доступність: web-додаток можна використовувати з будь-якого пристрою, що має доступ до Інтернету, такого як комп'ютер, смартфон або планшет. Він не потребує встановлення окремого програмного забезпечення і може бути запущений з будь-якого браузера, забезпечуючи легкий доступ до даних про електроспоживання.
- Реальний час: web-додаток забезпечує миттєвий доступ до інформації про споживання електроенергії у режимі реального часу. Користувачі можуть відстежувати своє електроспоживання в режимі онлайн, що дозволяє вчасно виявляти потенційні проблеми та займатися енергозбереженням.

– Аналітика та статистика: web-додаток забезпечує можливість аналізувати дані про електроспоживання на основі збережених даних за весь час роботи. Він надасть користувачам детальну статистику, графіки та звіти, що допомагають виявити тенденції, зрозуміти пікові навантаження та виявити можливості для підвищення енергоефективності.

Метою роботи є розроблення web-додатку для моніторингу електроспоживання комунальними і промисловими об’єктами в рамках виконання замовлення Сумського державного університету. Під час розробки будуть вирішені наступні задачі: аналіз аналогів, аналіз та вибір інструментів реалізації, проєктування системи та її впровадження.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд останніх досліджень і публікацій

Провівши аналіз наявних ресурсів на тему електроспоживання, було знайдено розгорнуту статтю про домашній енерго-моніторинг [1]. Електро-монітор для дому - це прилад, що встановлюється перед окремими об'єктами-споживачами, або відразу на вхід мережі. У цих випадках, електроспоживання відслідковується або точково (поприладно, чи погрупово), або відразу загальне споживання.

Мета використання таких систем - це зменшення витрат на електроспоживання. Це може досягатися наступним чином:

- Ідентифікації очевидного надмірного споживання, наприклад, у разі увімкнутого обігрівача у непотрібний час.
- Моніторингу пасивного споживання приладами, що не використовуються, але залишаються увімкненими у мережу.
- Підвищення обізнаності у сфері відповідального споживання
- Проведення енергозатратних операцій у години зменшеного споживання, наприклад, уночі

Особливо важливим є ведення моніторингу електроспоживання в режимі реального часу для крупних комунальних та промислових об'єктів, закладів дошкільної, шкільної, професійної та вищої освіти.

Тож, цільовою аудиторією веб-сайту по моніторингу електроспоживання будуть споживачі, що споживають багато та мають розгалужену електричну мережу із великою кількістю приладів: підприємства, державні установи, заводи тощо. За рахунок гнучкої фільтрації отримання даних, можна буде зручно та ефективно відслідковувати споживання, адже, наприклад, на заводі, це зробити набагато важче аніж вдома.

1.2 Аналіз програмних продуктів - аналогів

Загальнодоступними є лише додатки для приватних користувачів, тож аналізувати будемо їх, так як додатки для крупних підприємств у вільний доступ лише рекламні матеріали. Одним із найбільш популярних є мобільний застосунок Sense Energy Monitor [2]. Він має приємний дизайн, інформативну та зручну інфографіку щодо поточного споживання.

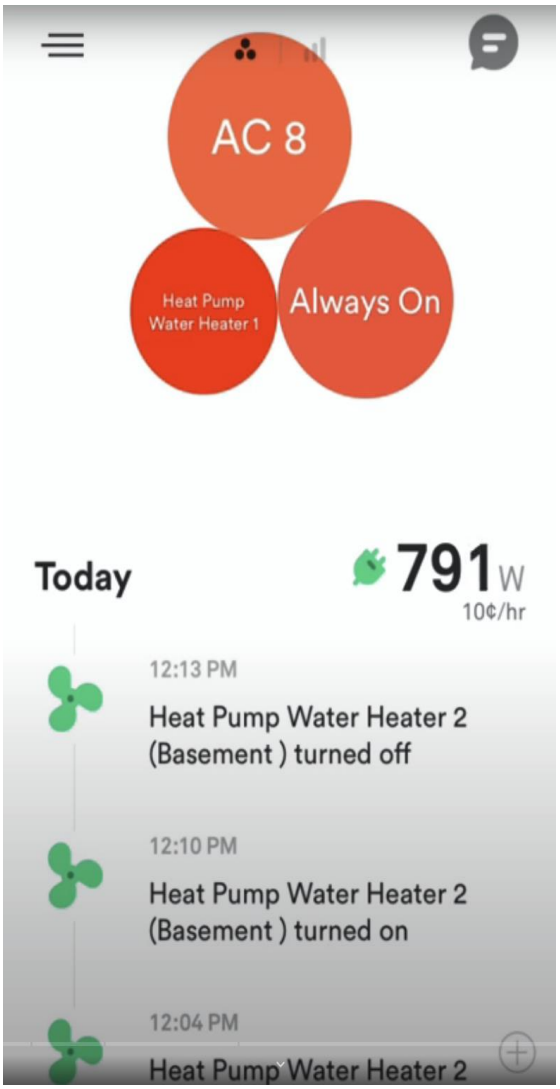


Рисунок 1.1 - Інфографіка поточного споживання

Також він має репрезентативний графік щодо попереднього споживання, із можливістю переглядати різні відрізки часу. На графіку зображується як

загальне споживання, так і прилади, що призвели до зміни споживання, що дуже зручно для побутових користувачів.

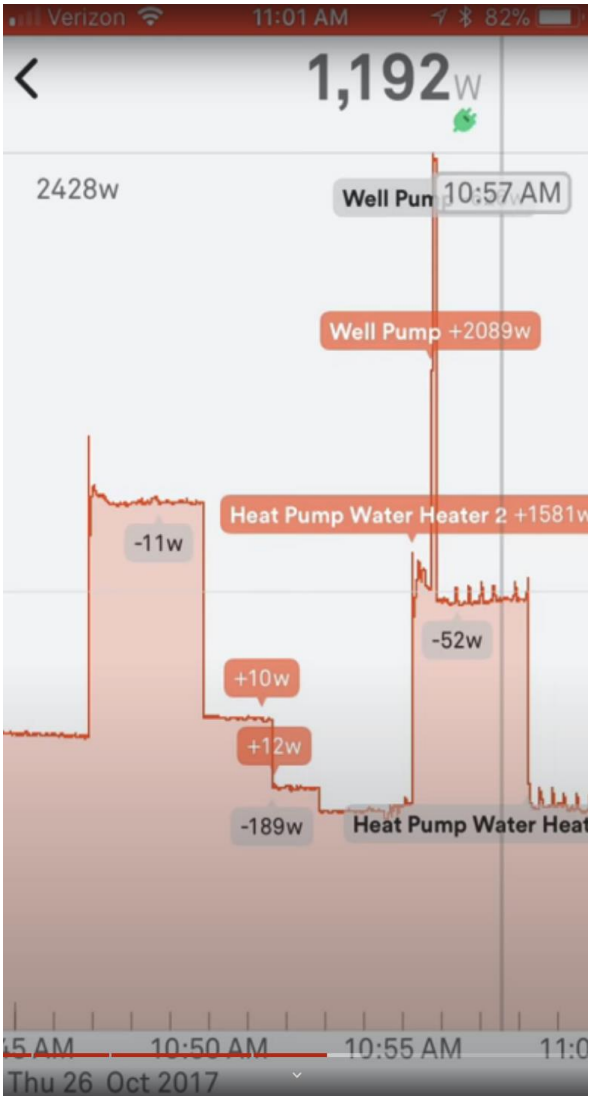


Рисунок 1.2 - Графік споживання

Із мінусів додатку це його формат - із екрану мобільного телефону не дуже зручно проглядати великий діапазон часу, та відсутність будь-якого групування приладів, наприклад я не можу подивитись споживання саме мого бойлера.

Наступним аналогом є Generac orientation [3], від компанії Generac Power Systems, яка спеціалізується на різноманітних шляхах забезпечення будинків електроенергією, від генераторів до сонячних панелей із акумуляторами. Його головний екран також є доволі репрезентативним, висвітлюючи три основні

джерела електроенергії користувачів додатку - сонячні панелі, акумулятори та загальна мережа.

Рисунок 1.3 - Головний екран Generac Orientation

Інший доволі корисний функціонал - перегляд платежів за електроенергію, що дозволяє дуже зручно порівнювати та відслідковувати фінансову сторону та, можливо, задуматись над оптимізацією споживання.

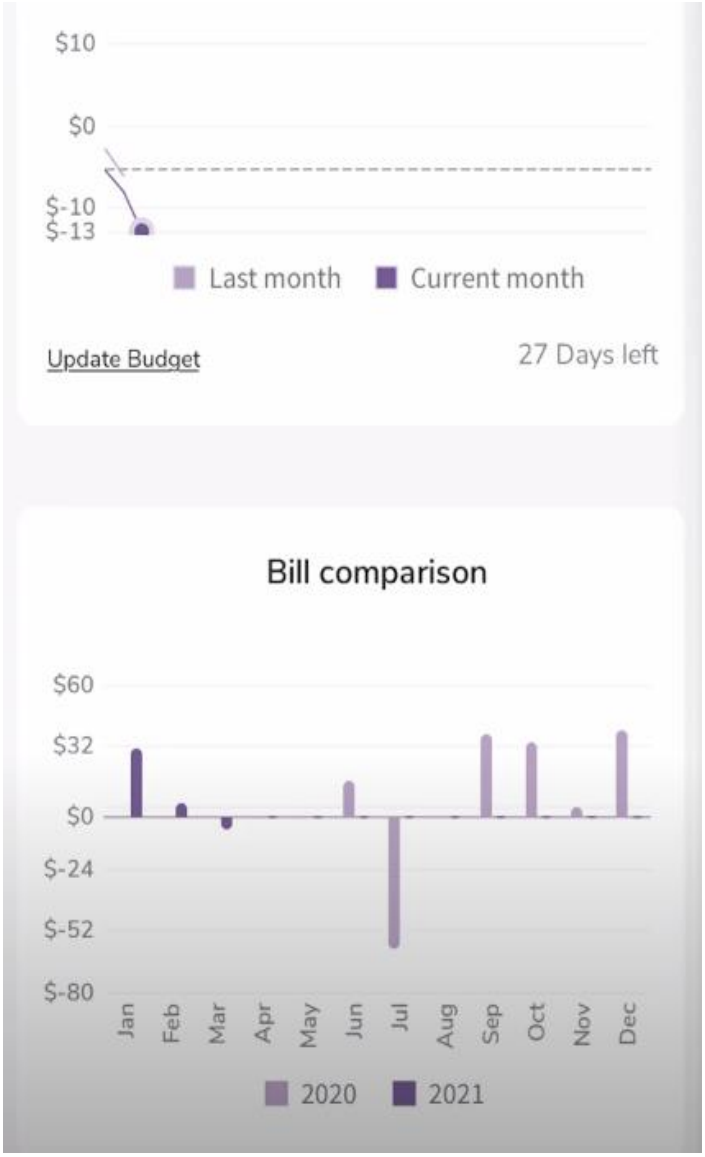


Рисунок 1.4 - Generac Orientation сторінка платежів

Цей додаток дозволяє переглядати виключно загальне споживання/генерацію електроенергії, що я безсумнівним мінусом для цільової аудиторії, що була визначена раніше. Також він є зосередженим навколо сонячних панелей, тож не є доцільним для користувачів без них.

Останнім розглянемо додаток Emporia [4], від компанії Emporia Energy. Головний екран має трохи застарілий дизайн, але, з іншої сторони, надає детальні дані щодо відносного та абсолютного споживання по кожній з ліній споживання,

що є дуже зручним. Також тут одразу пропонується вибір інтервалу споживання та різних об’єктів.

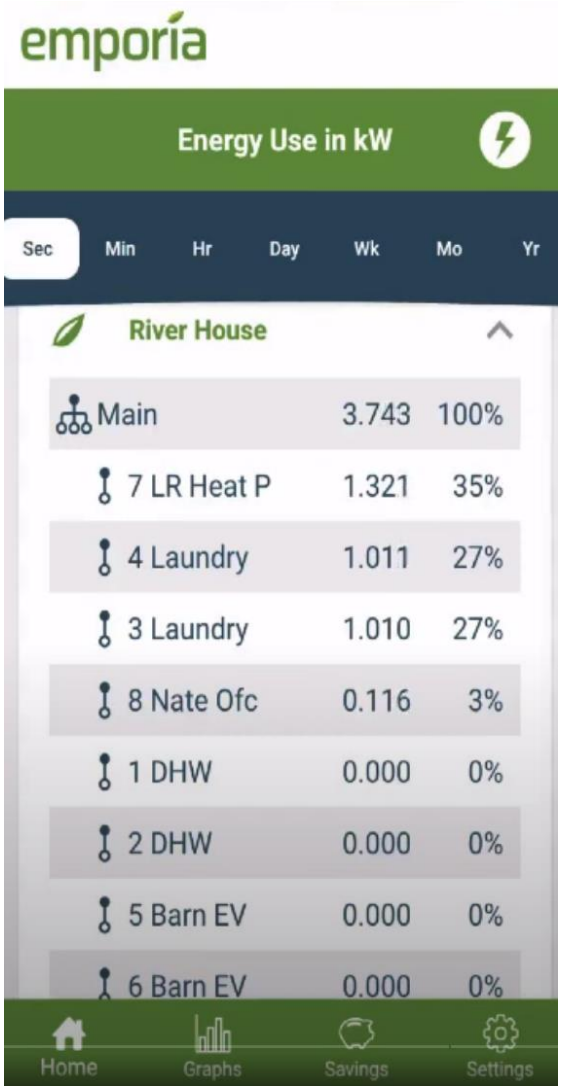


Рисунок 1.5 - Головний екран emporia

Також присутній графік споживання, із агрегованими даними за обраний період, але відсутня можливість переглядати графік полінійно, за аналогією із головним екраном.

Рисунок 1.6 - Графік споживання у Emporia

Проаналізувавши ці додатки, можна підбити підсумок, зробивши порівняльну таблицю.

Таблиця 1.1 - Порівняння додатків

Характеристика/Додаток	Sense	Generac	Emporia	Web-додаток, що розроблюється
Веб-сайт	-	-	+	+
Мобільний додаток	+	+	+	-

Продовження таблиці 1.1

Сучасний дизайн	+	+	-	+
Гнучкість перегляду	-	-	+	+
Перегляд платежів	-	+	+	-

Тож, при наявності усіх перерахованих вище властивостей, web-додаток надасть можливість легко та зручно відслідковувати дані споживання у режимі реального часу об’єктами крупних комунальних та промислових закладів, що значно спрощує аналіз та оптимізацію витрат на електроспоживання.

1.3 Постановка задачі web-додатку

Метою дослідження є розробка web-додатку для збору та відслідковування даних електроспоживання як у реальному часі, так і на певному відрізку часу.

Актуальна задача, яку вирішує впровадження web-додатку, є моніторинг електроспоживання в режимі реального часу крупних об’єктів громадського, промислового призначення, сфер обслуговування, навчальних закладів різного рівня.

Функціонал розроблюваного web-додатку:

- отримання показників електроспоживання від датчиків, встановлених на різних об’єктах закладів;
- агрегація (погодинна) показників електроспоживання та збереження їх в базі даних для подальшого перегляду;
- фільтрація відображення агрегованих показників електроспоживання за різними параметрами;
- відображення агрегованих показників в табличній та графічній формах;

- реєстрація користувачів та розмежування прав доступу перегляду показників електроспоживання дозволених об’єктів відповідно до ролей користувачів;
- адміністрування апаратної частини системи (реєстрація датчиків електроспоживання для об’єктів закладів, комбінування окремих датчиків в концентратори) ;
- адміністрування інформаційної частини системи (регулювання прав доступу груп користувачів, окремих користувачів, реєстрація закладів та їх об’єктів, електроспоживання яких підлягатиме моніторингу) ;

Кінцевий продукт буде веб-сайтом, що реалізовуватиме увесь функціонал із вимог.

Існує декілька основних мов та технологій для створення веб-сайтів [5]:

- php. Популярна мова для створення веб-сайтів, але вже досить застаріла, що тягне за собою можливу необхідність використовувати інтерфейси, що були спроєктовані без урахування сучасних потреб. Не дивлячись на це, все ще непогана опція, так як на ринку багато розробників із знанням цієї мови і тому її можна обрати для проєктів помірної складності або малобюджетних [6];
- мови родини C. Компільовані мови завжди виграють по швидкості у інтерпритованих, але, зазвичай, є значно складнішими у розробці, що тягне за собою підвищену вартість. Мають значно меншу популярність аніж інші опції, тому їх доцільно використовувати у проєктах із пріоритетом на швидкодію;
- python. Розповсюджений інструмент з дуже гарною варіативністю бібліотек для розробки веб-сайтів із фокусом на різноманітних аспектах. Через це тягне швидкість та, відповідно, нижчу вартість розробки. Але він є інтерпритованим, що автоматично означає меншу швидкість роботи [7]

Обираючи найбільш оптимальний інструмент, було проаналізоване необхідне програмне забезпечення, ключові характеристики аналогів та проведено дослідження ринку.

Було обрано саме Python, адже швидкодія для проєкту, не є ключовим параметром (детальний опис завдання на проєктування наведено у додатку в додатку А) , на відміну від простоти створення та підтримки. Результати планування проєкту приведені у додатку Б.

Для реалізації поставленої мети дослідження необхідно виконати такі задачі:

- аналіз програм-аналогів;
- визначення функціональних вимог;
- вибір засобів реалізації;
- проектування web-додатку;
- реалізація web-додатку;
- впровадження в тестову експлуатацію;

2 МОДЕЛЮВАННЯ ТА ПРОЄКТУВАННЯ WEB-ДОДАТКУ

2.1 Функціональне моделювання web-додатку в IDEF0

IDEF0 - це графічний методологія моделювання процесів, що використовується для впровадження систем та розробки програмного забезпечення. Ці методи використовуються для створення моделей даних, симуляцій, об’єктно-орієнтованого аналізу та отримання інформації.

IDEF0 була спроєктована та розроблена повітряними силами США у середині 1970-х років. Вона була розроблена як стандартний метод документування та аналізу бізнес процесів. Зараз ця методологія використовується як регламентований підхід до аналізу підприємств, передаючи моделі процесів “як є”, та моделювання діяльності бізнес груп [8].

Діаграма А-0 верхнього рівня - найбільш абстрактна, зображує систему у цілому. До неї входять єдиний функціональний блок, що найбільш широко описує головну функцію системи, та різноманітні вхідні та вихідні дані, що задіяні у її функціонуванні. Діаграму А-0 зображено на рисунку 2.1.

Рисунок 2.1 – Діаграма бізнес-процесів в нотації IDEF0. Конептуальний рівень

Вхідними даними системи є наступні джерела:

- параметри запиту від користувача;
- інформація про користувача.

Контролюючими механізмами є:

- алгоритм агрегації даних;
- алгоритм обробки даних електроспоживання;
- алгоритм виведення даних.

Механізми, що використовуються для виконання роботи:

- Django (web-фреймворк, що є каркасом серверної частини сайту);
- PostgreSQL (база даних);
- додаток обробки даних електроспоживання;
- додаток агрегації даних електроспоживання.

Вихідними даними системи є дані споживання відповідно до параметрів запиту.

Діаграма А-1 має меншу ступінь абстракції, та репрезентує роботу системи шляхом розбиття її основного процесу на підпроцеси, відображаючи

послідовність цих процесів, разом із використанням чи взаємодією із вхідними даними, контролюючими механізмами, механізмами для виконання роботи та вихідними даними цими підпроцесам. Декомпозиція функціональної моделі web-додатку представлена на рисунку 2.2.

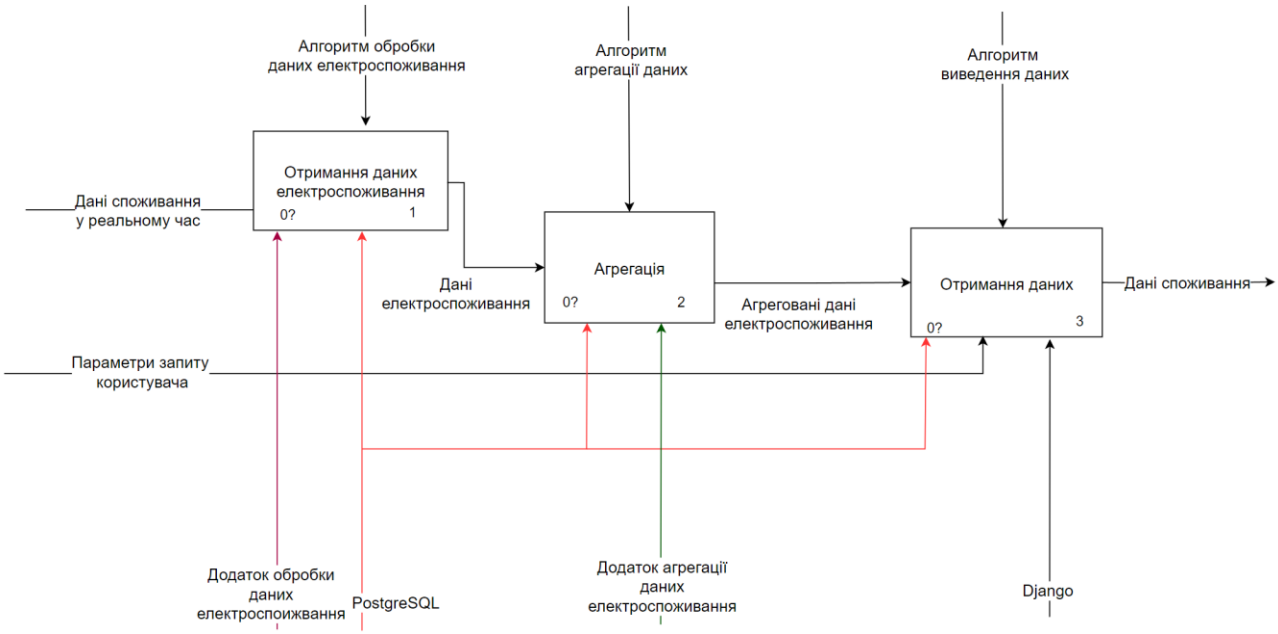


Рисунок 2.2 – Декомпозиція бізнес процесу web-додатку

2.2 Проектування інформаційної системи

Діаграма варіантів використання UML є основною формою вимог до системи/програмного забезпечення, що розробляється. Варіанти використання вказують на очікувану поведінку (що), а не на точний спосіб її здійснення (як). Випадки використання, як тільки визначені, можуть позначатися як текстовим, так і візуальним представленням (тобто діаграмою варіантів використання).

Ключова концепція моделювання варіантів використання полягає в тому, що воно допомагає проектувати систему з точки зору кінцевого користувача. Це ефективна техніка для передачі інформації про поведінку системи в термінах користувача шляхом визначення всієї зовнішньо видимої поведінки системи [9].

Діаграма варіантів використання в нотації UML представлена на рисунку 2.3.

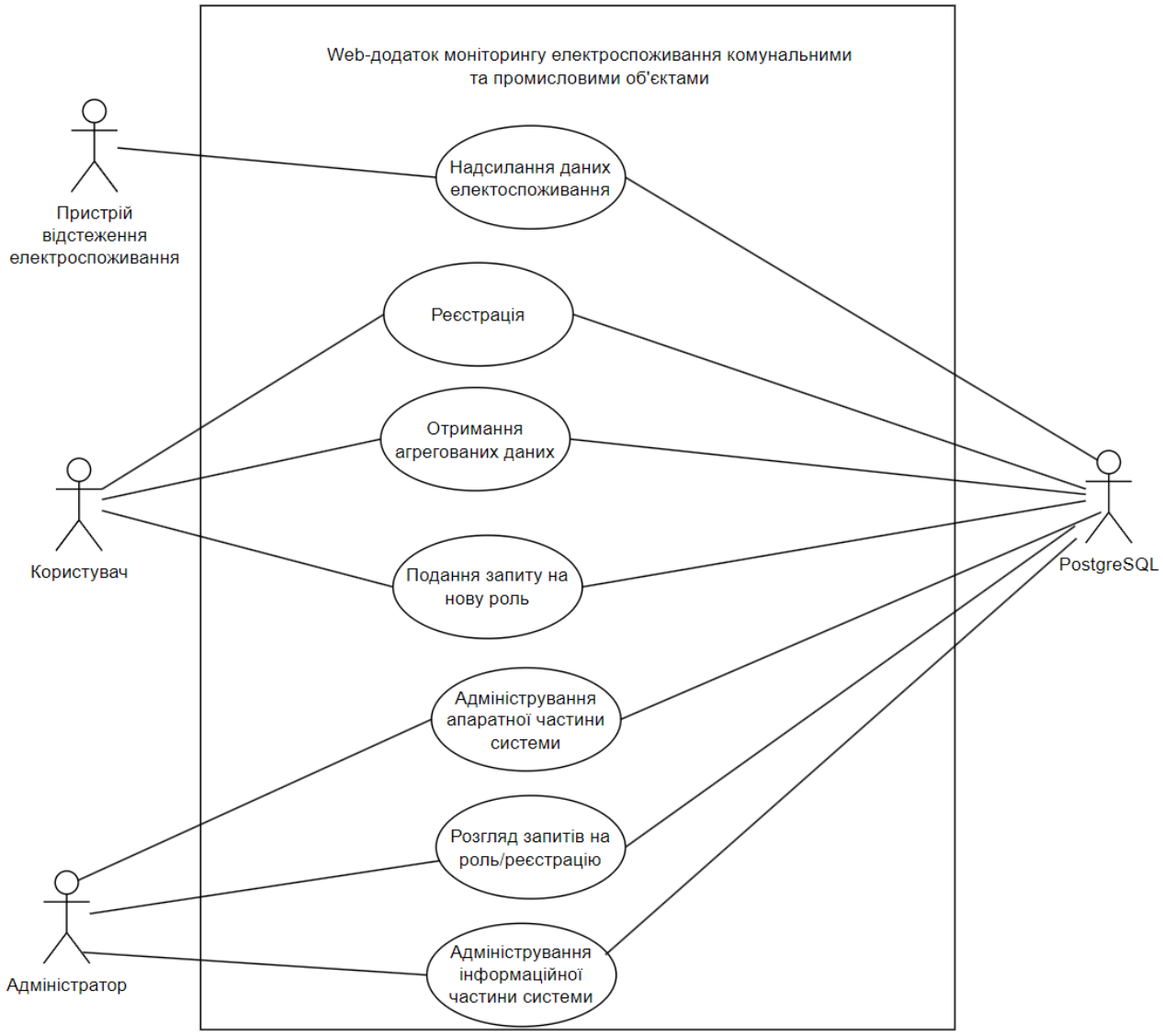


Рисунок 2.3 – Діаграма варіантів використання

У системі присутні 3 актори:

- користувач додатку;
- адміністратор;
- база даних (PostgreSQL) ;
- пристрій відстеження електроспоживання.

2.3 Проєктування моделі бази даних

Існує два основних підходи до зберігання даних реляційний та нереляційний підхід, відповідно. У кожного є свої переваги та недоліки, що мають визначальний вплив на структуру системи, її ефективність, складність проєктування та розробки. Через це вибір правильного підходу є одним із найголовніших завдань на початкових етапах створення системи, адже, у разі помилки, зазвичай, виправляти її буде дуже складно, бо підхід зберігання даних є однією із ключових складових фундаменту проєкту.

Реляційна база даних — це тип бази даних, яка зберігає та надає доступ до точок даних, пов'язаних одна з одною. Реляційні бази даних базуються на реляційній моделі - інтуїтивно зрозумілому, простому способу представлення даних у таблицях. У реляційній базі даних кожен рядок у таблиці є записом з унікальним ідентифікатором, який називається ключем. Стовпці таблиці містять атрибути даних, і кожен запис зазвичай має значення для кожного атрибута, що полегшує встановлення зв'язків між точками даних [14].

Нереляційні бази даних відрізняються від традиційних реляційних баз даних тим, що вони зберігають дані не в табличній формі. Натомість, нереляційні бази даних можуть базуватися на таких структурах даних, як документи. Документ може бути дуже детальним, водночас містити низку різних типів інформації в різних форматах. Ця здатність зберігати та організовувати різні типи інформації поруч робить нереляційні бази даних набагато більш гнучкими, ніж реляційні бази даних [15].

Рисунок 2.4 – Приклад документу для нереляційної бази даних

Для реалізації системи, було обрано реляційній тип зберігання даних, адже оперувати будемо лише завчасно визначеними моделями даних. Згідно із нею було створено логічну модель даних. Логічна модель даних наведена на рисунку 2.5.

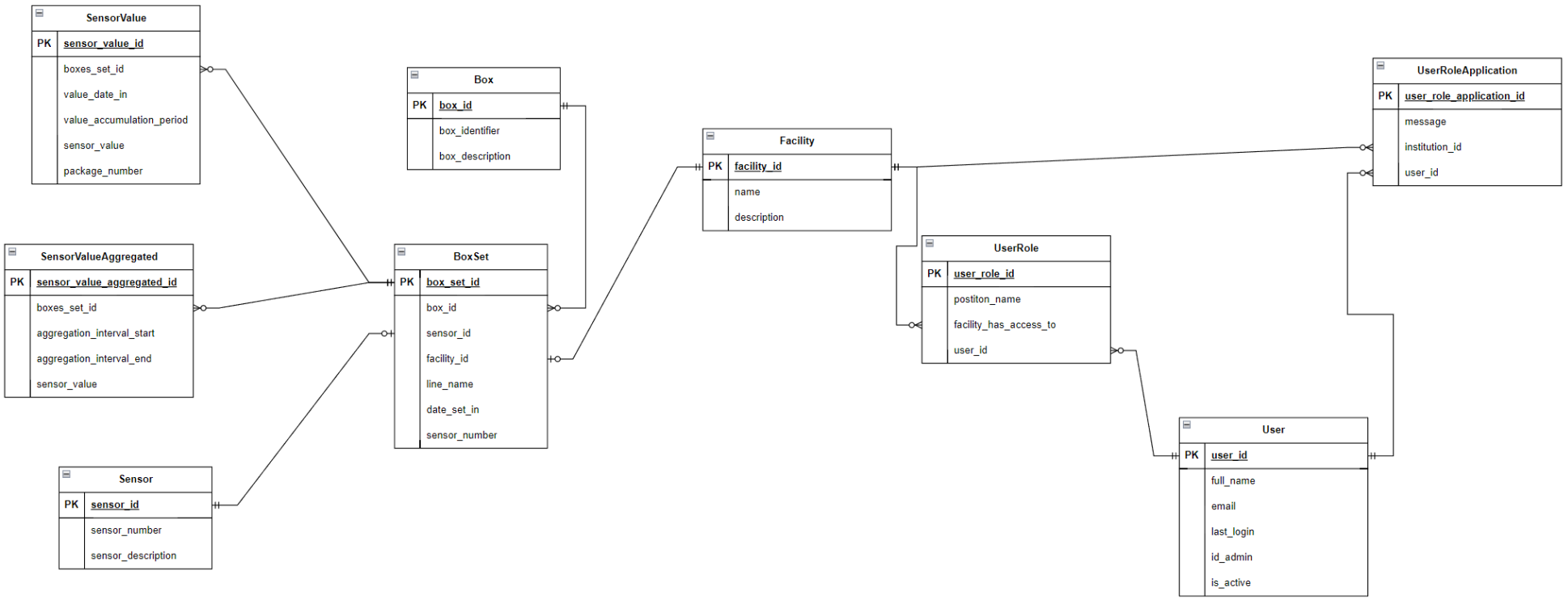


Рисунок 2.5 – Логічна модель даних

Таблиця 2.1 – Опис сутностей моделі даних

Номер	Назва таблиці	Назва поля	Опис
1	Facility		Таблиця призначення для збереження даних про установи, електроспоживання яких моніториться засобами web-додатку
		facility_id	Первинний ідентифікатор установи
		name	Назва установи чи об’єкту
		description	Опис установи чи об’єкту
2	Box		Таблиця для збереження даних про пристрої, що відслідковують та надсилають дані електроспоживання
		box_id	Первинний ідентифікатор пристрою

Продовження таблиці 2.1

		box_identifier	Ідентифікатор пристрою у системі
		box_description	Опис пристрою
3	Sensor		Таблиця використовується для збереження даних про сенсори відслідковування електроспоживання
		sensor_id	Первинний ідентифікатор сенсору
		sensor_number	Номер сенсору
		sensor_description	Опис сенсору
4	BoxSet		Таблиця слугує для збереження інформації про набори сенсори та пристрій
		box_set_id	Первинний ідентифікатор набору
		box_id	Зовнішній ключ пристрою
		sensor_id	Зовнішній ключ сенсору

Продовження таблиці 2.1

		facility_id	Зовнішній ключ об’єкту, до якого прив’язаний набір
		line_name	Назва лінії
		date_set_in	Дата встановлення
		sensor_number	Номер сенсору у наборі
5	SensorValue		Таблиця зберігає інформацію електроспоживання
		sensor_value_id	Первинний ідентифікатор запису електроспоживання
		boxes_set_id	Зовнішній ключ набору до якого відносяться дані
		value_date_in	Дата наджодження даних
		value_accumulation_period	Час акумуляції даних
		sensor_value	Накопичене значення
		package_number	Номер пакету
6	SensorValueAggregated		Таблиця зберігає агреговані дані електроспоживання

Продовження таблиці 2.1

		sensor_value_aggregated_id	Первинний ідентифікатор запису агрегованих даних
		boxes_set_id	Зовнішній ключ набору, якому відповідає запис
		aggregation_interval_start	Початок інтервалу агрегації
		aggregation_interval_end	Кінець інтервалу агрегації
		sensor_value	Значення електроспоживання
7	User		Таблиця зберігає інформацію щодо користувачів
		user_id	Первинний ідентифікатор користувача
		full_name	Повне ім’я
		email	Електронна пошта користувача
		last_login	Дата останньої авторизації
		is_admin	Чи є адміністратором

Продовження таблиці 2.1

8	UserRole		Таблиця зберігає дані про рівень доступу користувача
		user_role_id	Первинний ідентифікатор рівню доступу
		position_name	Назва позиції
		facility_has_access_to	Зовнішній ключ об’єкту, до якого має доступ
		user_id	Зовнішній ключ користувача
9	UserRoleApplication		Таблиця зберігає запити на роль від користувачів
		user_role_application_id	Первинний ключ запиту на роль
		message	Повідомлення від користувача
		institution_id	Зовнішній ключ установи
		user_id	Зовнішній ключ користувача

3 РОЗРОБКА WEB-ДОДАТКУ

3.1 Реалізація моделі даних

На основі логічної моделі даних було реалізовано базу даних у СУБД PostgeSQL. Основною складністю цього розділу був вибір правильного підходу зберігання ієрархічних даних таблиці facilities, адже взаємозв'язки між об'ктами найкраще відображує структура даних «дерево». Було проаналізовано наступні підходи до зберігання та створено порівняльну таблицю.

Таблиця 3.1 – Порівняння підходів збереження ієрархічних даних

Метод	Вибірка	Додавання	Оновлення	Видалення	Цілісність посилань
Зв'язний список (Adjacency list)	Складно	Легко	Легко	Легко	+
Плоска таблиця (Flat table)	Легко	Складно	Легко	Легко	Неможлово визначити
Зв'язна таблиця (Bridge table)	Легко	Легко	Складно	Складно	+
Вкладений набір (Nested set)	Легко	Складно	Складно	Складно	-

Продовження таблиці 3.1

Нумерація шляхів (Path enumeration)	Легко	Легко	Легко	Складно	-

Проаналізувавши варіант використання цієї таблиці, було обрано підхід зв’язного списку, адже у нього найшвидша вибірка та зручна перевірка на підлеглість об’єкту дерева. Складність інших операцій буде компенсовати використанням заздалегідь визначених програмних рішень, за рахунок створення параметризованих функцій, прибираючи необхідність кожен раз піклуватися про внутнішню будову даних. Схема отриманої реалізації зображена на рисунку 3.1.

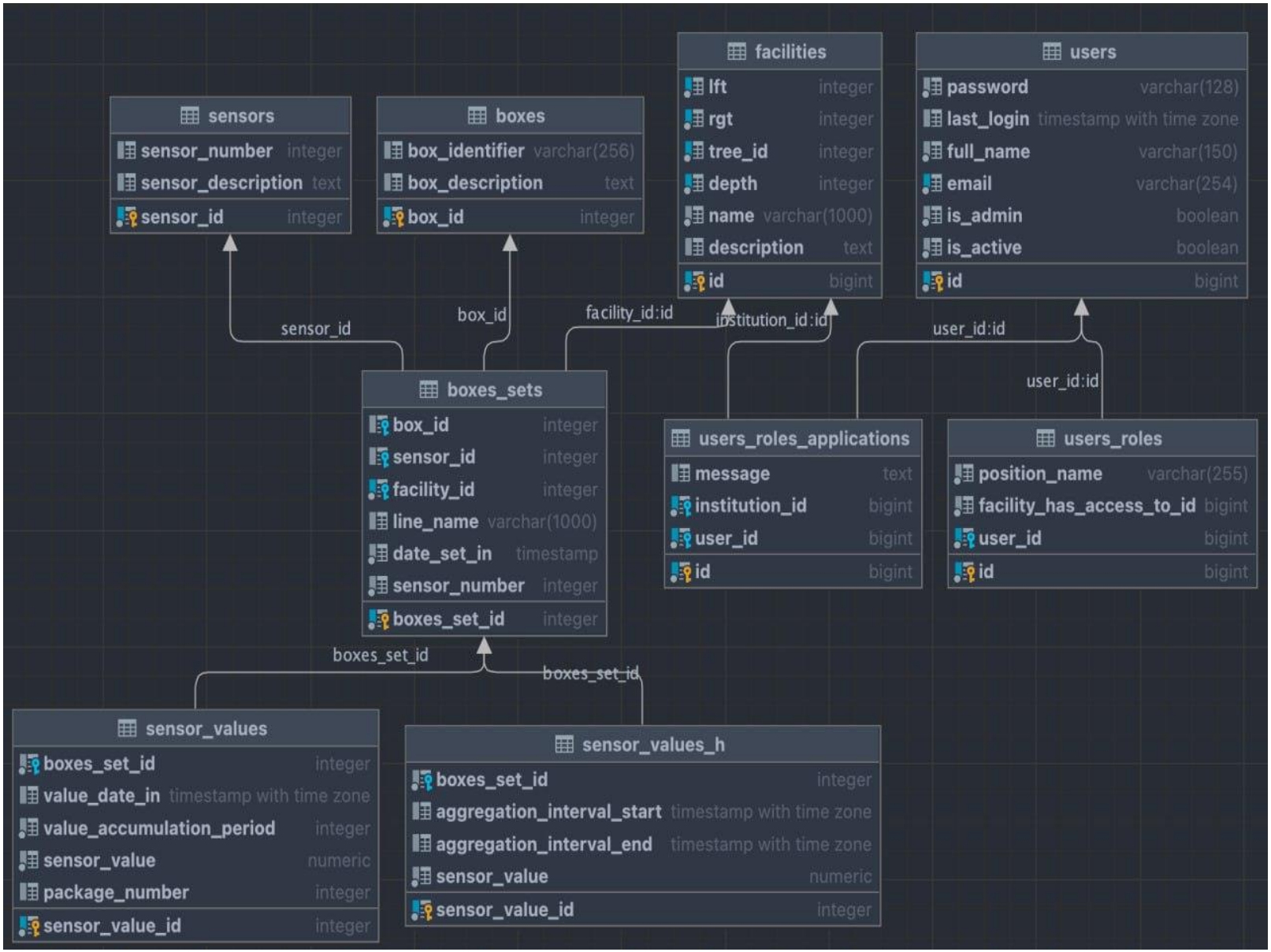


Рисунок 3.1 - Схема бази даних, реалізованої засобами СУБД PostgreSQL

3.2 Алгоритм обробки даних

Дані електроспоживання надсилаються спеціальними приладами шляхом TCP-повідомлень, тож було розроблено алгоритм для обробки бінарних даних. Для його реалізації було обрано мову GoLang, разом із фреймворком TcpServer. Цю мову було обрано через швидкість її роботи, та легкість вивчення, адже вона була розроблена, орієнтуючись на простоту використання та розуміння. Обраний фреймворк забезпечує зручний каркас для функції-обробника повідомлень. UML-діаграму станів його роботи зображено на рисунку 3.2.

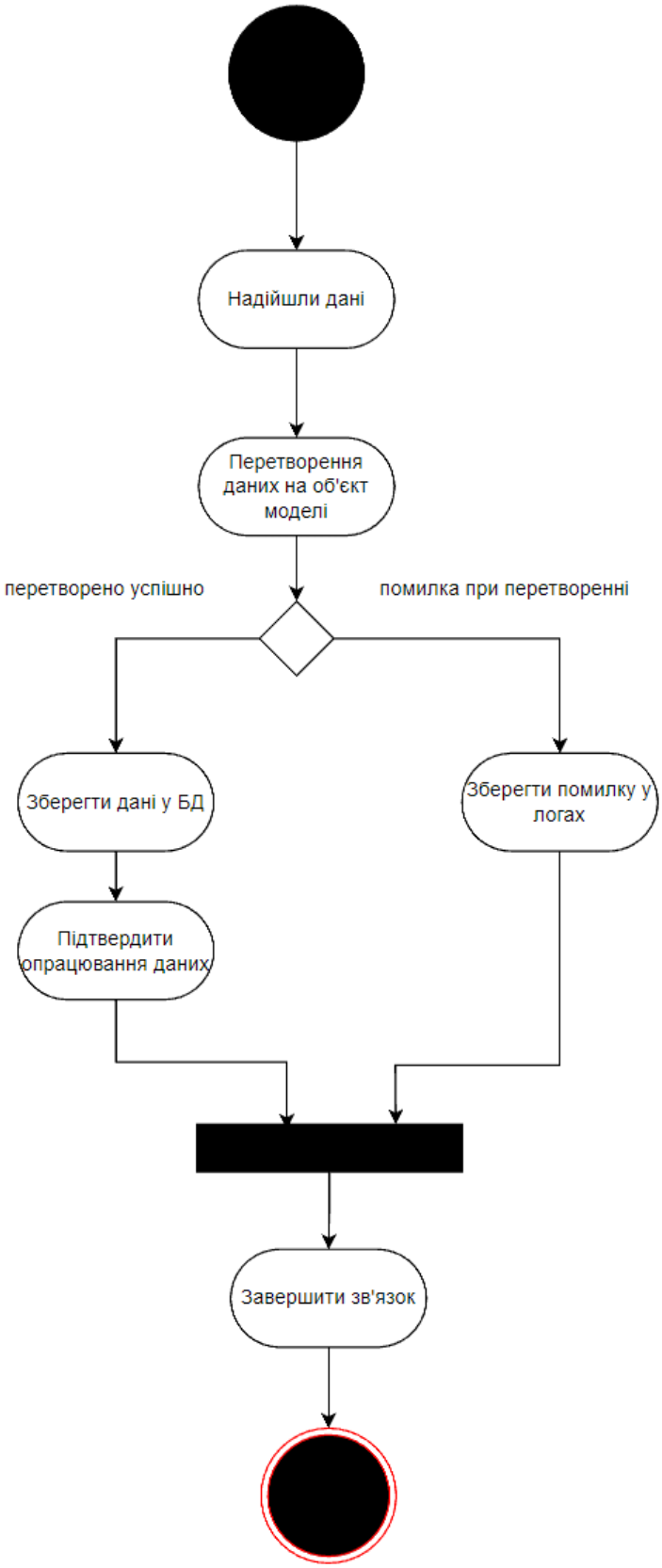


Рисунок 3.2 - UML-діаграма станів алгоритму обробки даних

3.2 Алгоритм агрегації даних електроспоживання

Для пришвидшення отримання агрегованих даних, було вирішено створити алгоритм для агрегації даних, щоб користувач лише отримував необхідні дані, а не очікував поки дані буде загреговано під його запит. Для реалізації цього алгоритму було обрано мову GoLang, адже вона легка у використанні та для розуміння і, одночасно, достатньо швидка, адже необхідно агрегувати десятки тисяч записів. UML-діаграма станів його роботи зображена на рисунку 3.3.

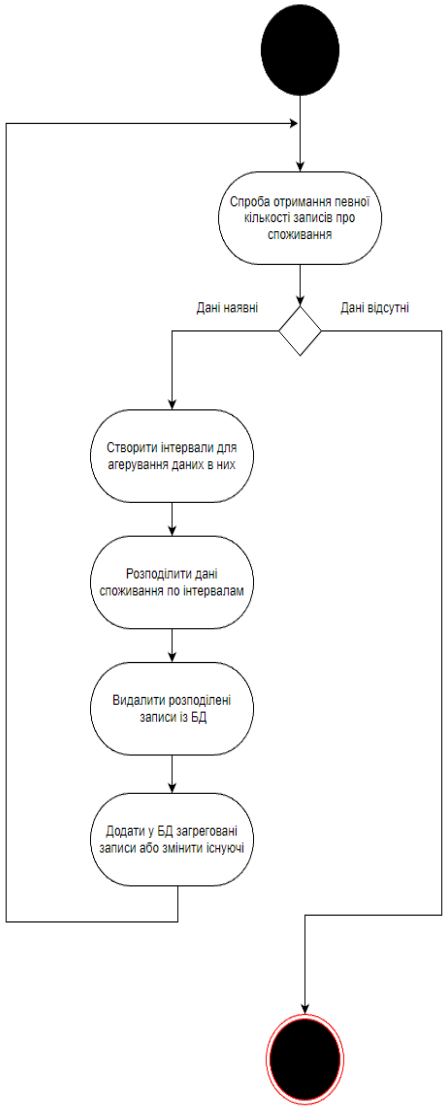


Рисунок 3.3 - UML-діаграма станів алгоритму агрегації даних електроспоживання

3.3 Архітектура web-додатку

Для додатку було обрано MVT (model - view - template) архітектуру. Її суть полягає у поділу на три шари із чітко визначеними зонами відповідальності, що спрощує розробку та зберігає зрозумілий розподіл функціоналу.

Перший із них це модель - він визначає структуру разом із інтерфейсом даних, і є відповідальним за додавання, редагування, видалення та оновлення даних в БД. Дуже важливо, щоб зберігався розподіл функціоналу між шарами, бо, подекуди, присутня помилка додавання бізнес-логіки у шар моделі, при використанні цієї архітектури.

Наступний шар - це вигляд, що є відповідальним за обробку запитів від користувача та надання йому відповідей. Він відповідальний за всю бізнес-логіку додатку та оперує Моделями, разом із даними від користувача. Після виконання необхідних операцій формується відповідь із використанням наступного шару.

Шаблон - це останній шар архітектури MVT, і він є набором шаблонів під відповіді, що наповнюються даними у шарі вигляду. Зазвичай шаблонами є текстові файли HTML, XML, тощо [16].

Для реалізації додатку було обрано саме цю архітектуру через її простоту, адже шарів небагато та у кожного є чітко визначена зона відповідальності та розповсюдженість, що спростить розробку.

3.4 Програмна реалізація

3.4.1 Реалізація серверної частини

Для реалізації серверної сторони було обрано мову Python, разом із web-фреймворком Django. Цей фреймворк є найбільш популярним для розробки web-додатків на обраній мові та має цілу низку характеристик, що стануть у нагоді:

- фіксована архітектура;
- велика кількість включеного найбільш розповсюдженого функціоналу;
- популярність (велика кількість сторонніх бібліотек та велика вірогідність знаходження вже готового рішення на проблему);
- зручна ORM (object-relational mapping), що майже повністю прибирає необхідність користуватися чистим SQL, та надає зручний та більш простий інтерфейс роботи із моделями даних;
- система шаблонів спрямована на повторне використання елементів;
- легкість маштабування.

3.4.2 Реалізація користувацької частини

Для реалізації користувацької частини було вирішено використовувати поєднання HTML, CSS та JS. Для полегшення стилізації зовнішнього вигляду було обрано CSS фреймворк Bootstrap, що значно полегшує розробку, шляхом надання готових рішень. Для покращення досвіду користувачів та кращої інтерактивності було використано JavaScript бібліотеку Jquery для маніпуляції елементами HTML-дерева та асинхронного звернення до серверної частини у випадках, коли треба оновити певну частину даних без оновлення цілої сторінки.

3.3 Використання web-додатку

Заходячи на сайт, користувач бачить вікно входу, яке також має посилання на реєстрацію.

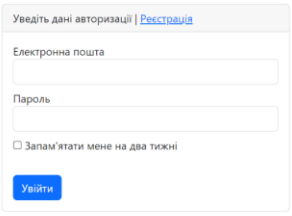


Рисунок 3.4 - Вікно логіну

Натискаючи на посилання для реєстрації, користувач бачить наступну форму. Заповнюємо дані та відправляємо заявку.

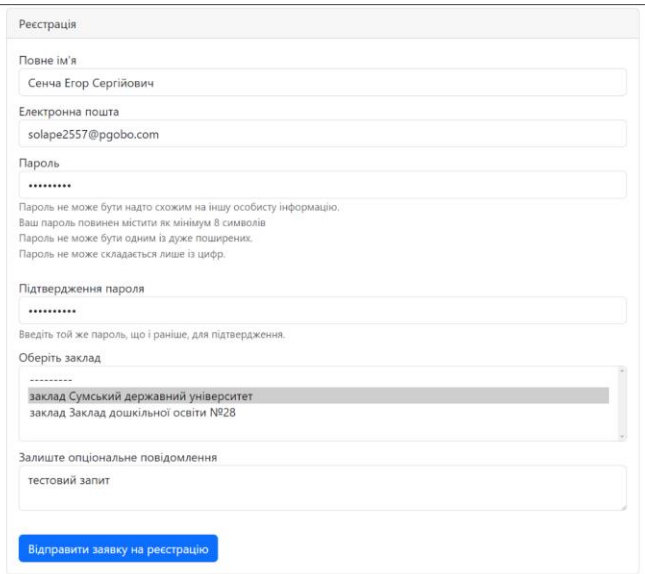


Рисунок 3.5 - Вікно реєстрації

У разі коректності даних, користувач бачить повідомлення про успішну реєстрацію та йому надсилається електронний лист.

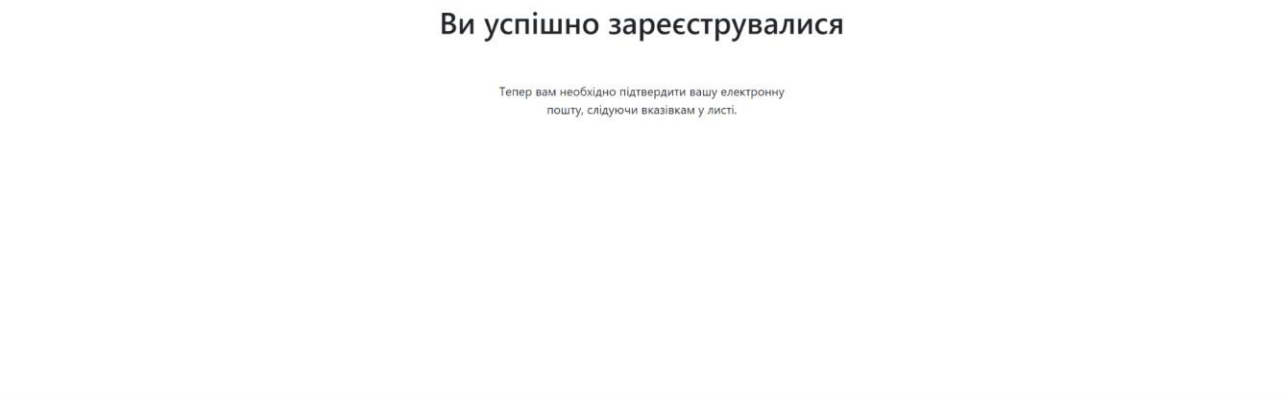


Рисунок 3.5 - Сторінка підтвердження реєстрації

Лист із підтвердженням пошти зображено на рисунку

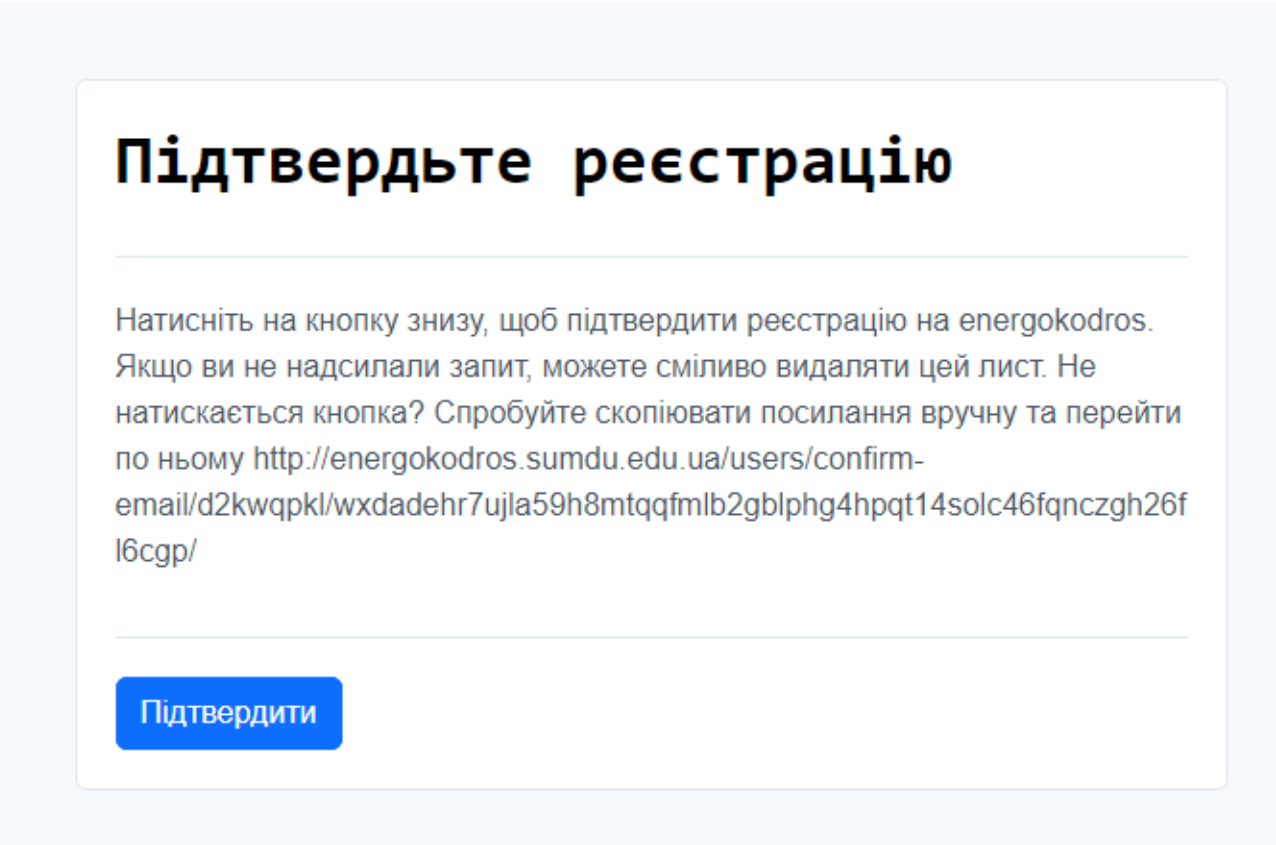


Рисунок 3.6 - Лист підтвердження реєстрації

Натискаючи на кнопку підтвердження, користувача направляє на наступну сторінку.

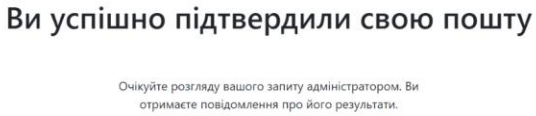


Рисунок 3.7 - Сторінка підтвердження електронної пошти

Після підтвердження пошти, у адміністратора з’являється заявка до розгляду.

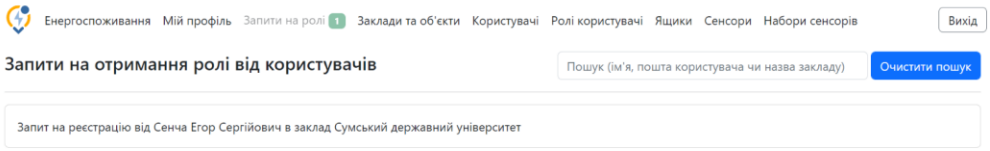


Рисунок 3.8 - Сторінка розгляду запитів на реєстрацію та отримання ролей

У формі розгляду запиту на реєстрацію необхідно ввести назву позиції та обрати об’єкт, до якого користувач матиме доступ. Її зображено на рисунку 3.9.

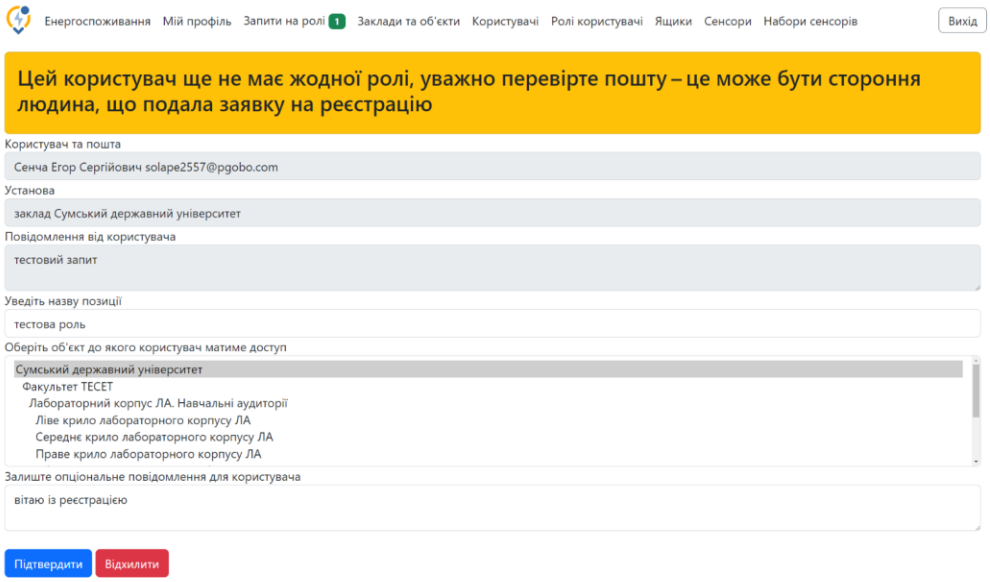


Рисунок 3.9 - Сторінка розгляду запиту на реєстрацію

Адміністратор отримує сповіщення відповідно до рішення відносно запиту.

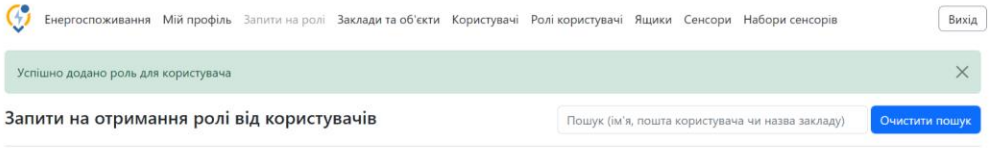


Рисунок 3.10 - Повідомлення про результат розгляду

А користувач отримує лист із результатом розгляду.

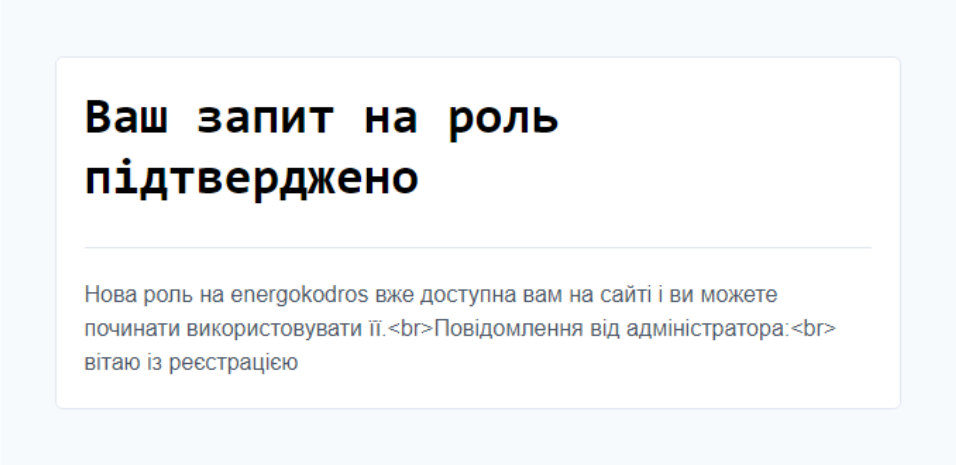


Рисунок 3.11 - Лист про результати розгляду реєстрації

Основною сторінкою сайту є сторінка перегляду даних електроспоживання, головними компонентами якої є форма параметрів запиту та результати запиту у табличній чи графічній формах. Приклади запитів та результатів наведено на рисунках 3.12 - 3.13.

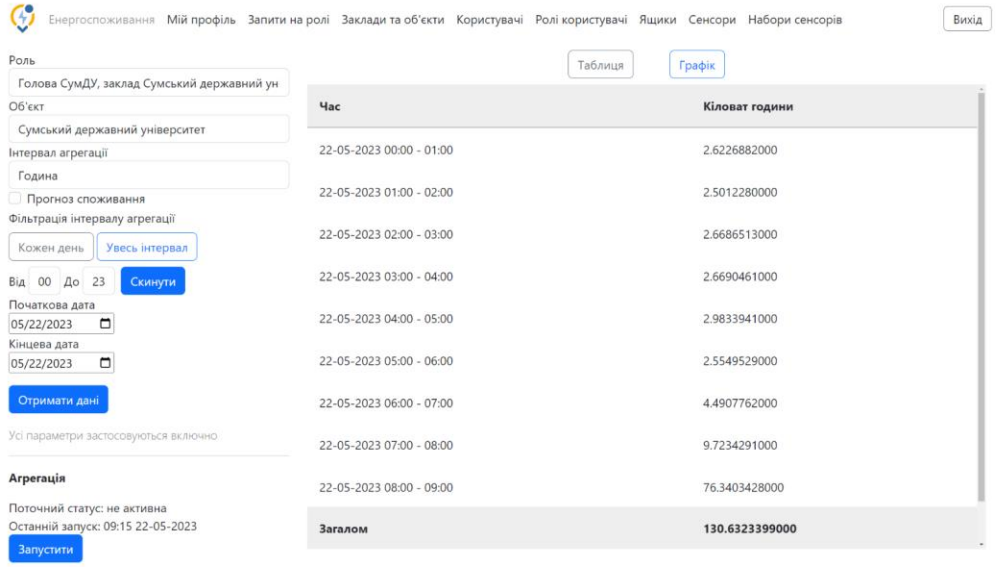


Рисунок 3.12 - Результати запиту із агрегацією погодинно у табличній формі

Рисунок 3.13 - Результати запиту із агрегацією поденно у графічній формі

Також кожному користувачеві, окрім сторінки електроспоживання, доступна сторінка його профілю із загальною інформацією.

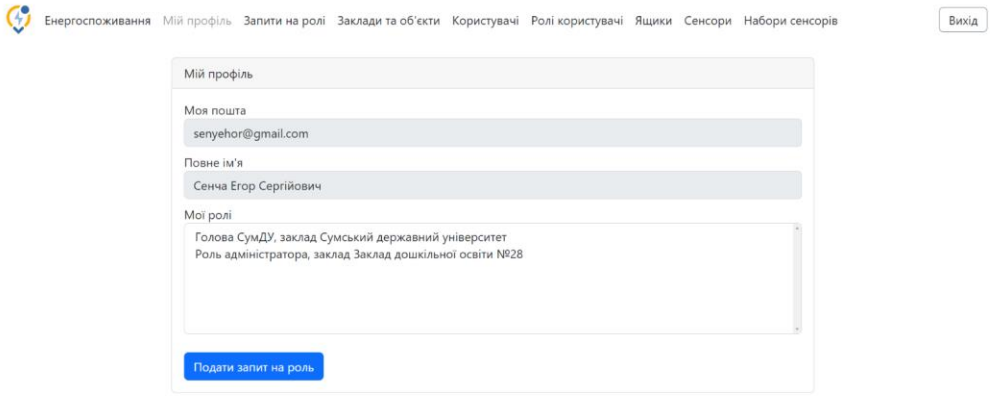


Рисунок 3.14 - Сторінка профілю

Адміністратору доступні сторінки із перегляду та управління різними об’єктами системи. Так як ці сторінки є уніфікованими, розглянемо на прикладі управління закладами та об’єктами. Основна сторінка має список об’єктів із можливістю пошуку по ключовим властивостям та зручною розбивкою посторінково.

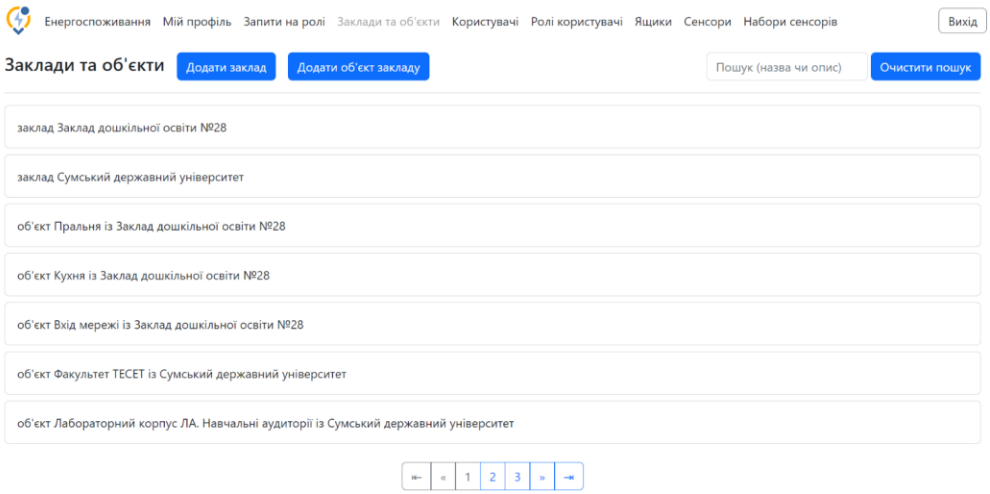


Рисунок 3.15 - Сторінка закладів та об’єктів

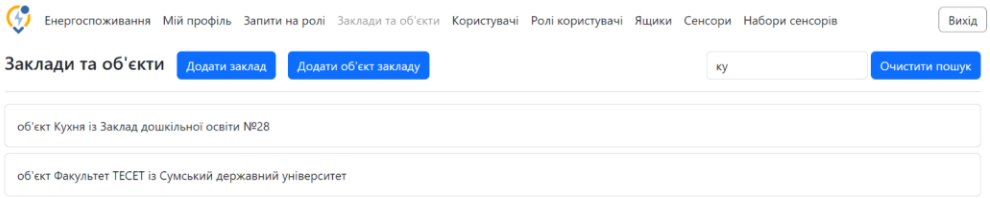


Рисунок 3.16 - Пошук по ключовим параметрам

При натисканні на об’єкт вас направить на сторінку із інформацією про нього, можливістю редагування чи видалення.

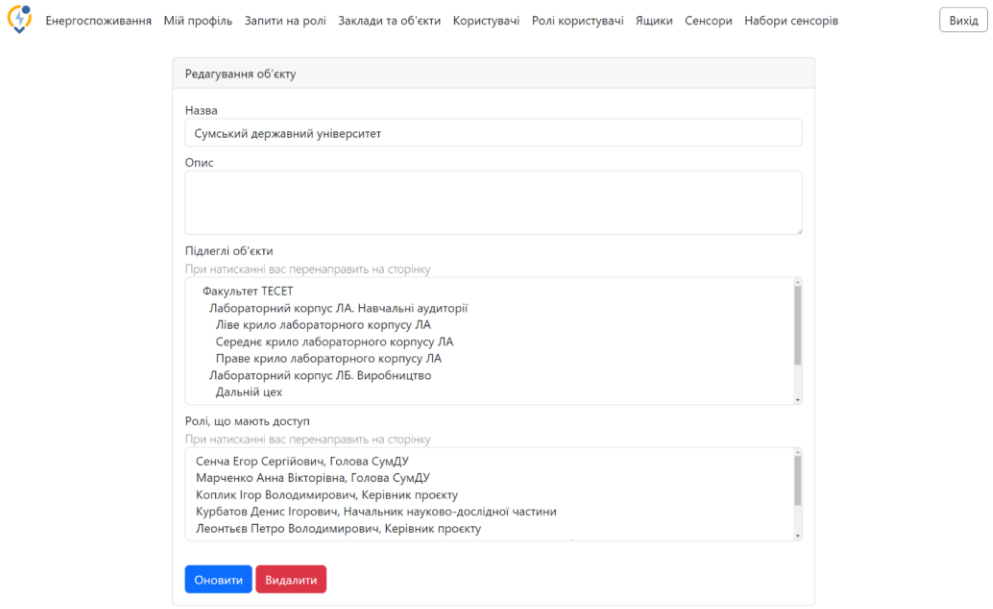


Рисунок 3.17 - Сторінка редагування об’єкту

Додавання об’єктів виконується шляхом використання наступної форми.

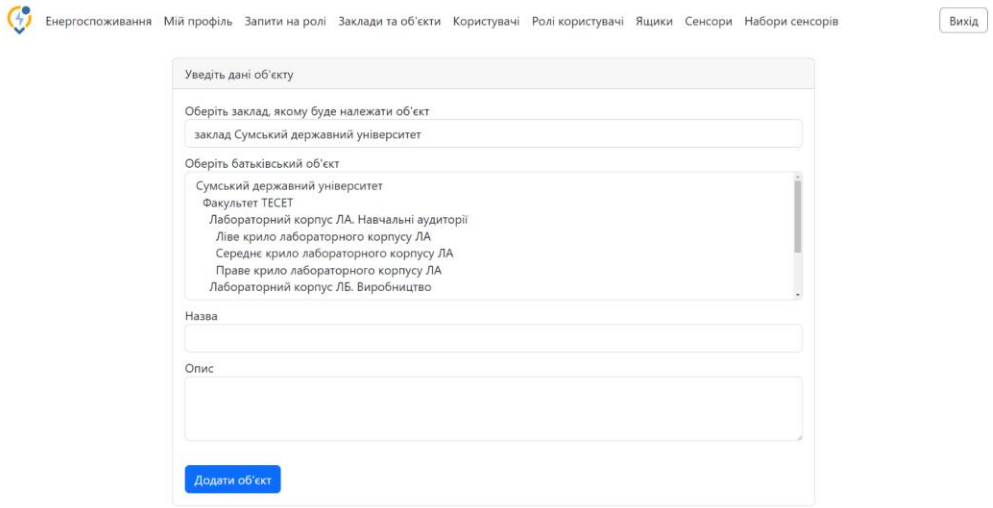


Рисунок 3.18 - Сторінка додавання нового об’єкту

А додавання нового ящику відбувається покроково. Спочатку додаємо інформацію про сам ящик.

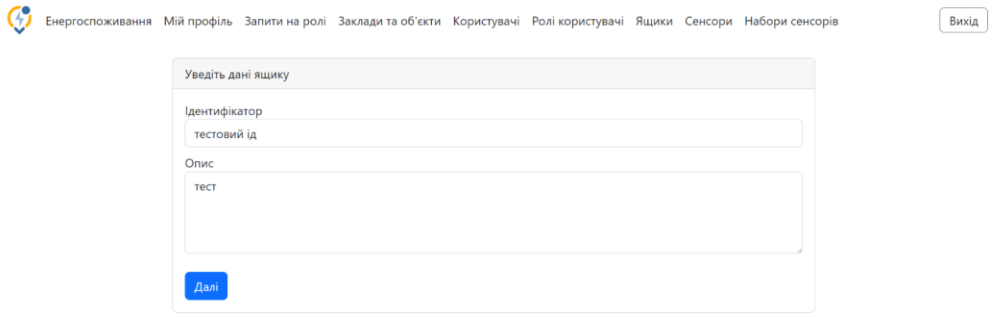


Рисунок 3.19 - Форма інформації про ящик

Потім заповнюємо інформацію щодо сенсорів.

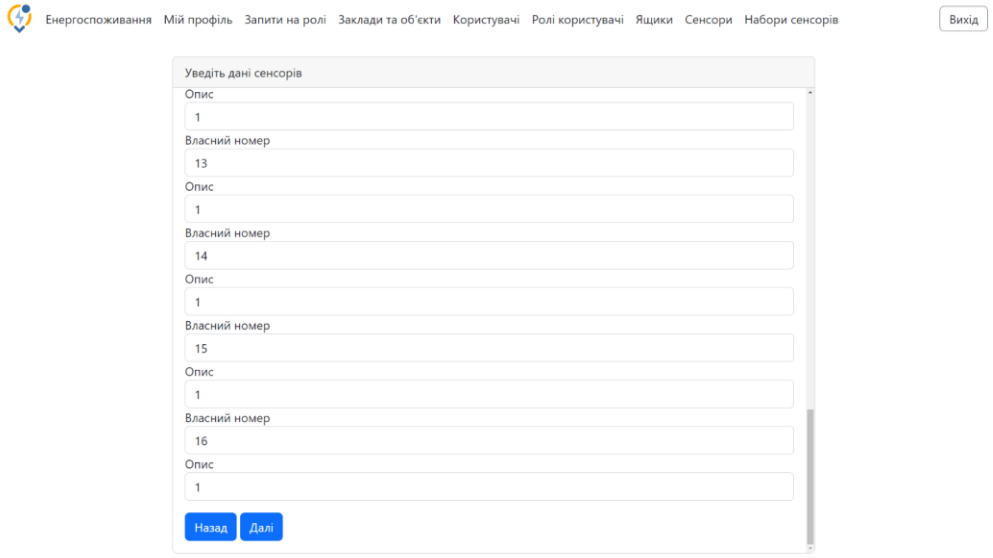


Рисунок 3.20 - Форма інформації про сенсори

І вже після цього зв’язуємо кожну лінію із необхідним об’єктом.

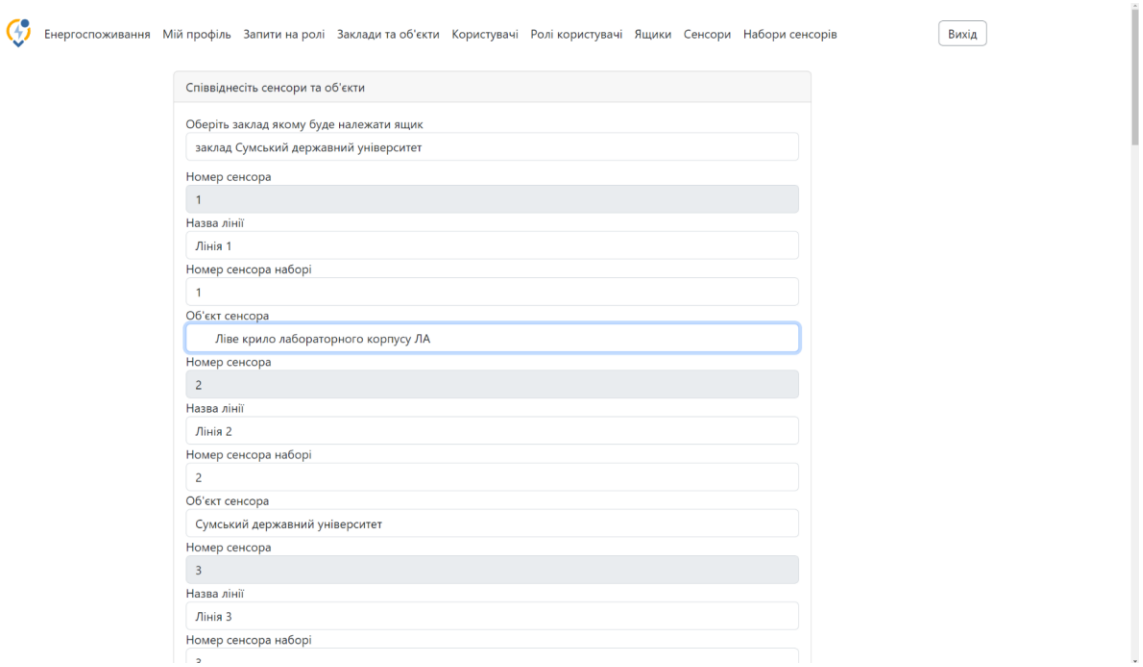


Рисунок 3.21 - Форма співставлення сенсорів та об’єктів

ВИСНОВКИ

Результатом виконання кваліфікаційної роботи бакалавра є розроблений та впроваджений в експлуатацію web-додаток моніторингу електроспоживання об'єктами комунальної та промислової власності.

Під час виконання дипломної роботи було проаналізовано публікації щодо актуальності моніторингу електроспоживання, і на основі цього було сформовано цільову аудиторію web-додатку, разом із її основними потребами. Також було розглянуто програмні продукти-аналоги, і, виходячи із цього, сформовано порівняльну таблицю, яка допомогла краще зрозуміти переваги та недоліки web-додатку, що створюється, відносно вже існуючих рішень.

На основі цього було сформовано мету роботи, розглянуто наявні інструменти та засоби реалізації, та обрано найбільш оптимальні із них. Метою розробки є створення web-додатку для отримання та перегляду даних електроспоживання, разом із адмініструванням об’єктів системи. Було обрано мову Python та фреймворк Django для реалізації серверної частини додатку.

Під час проєктування, для опису основних процесів системи було створено відповідні діаграми, використовуючи нотацію IDEF0. Також було створено UML-діаграму варіантів використання системи. Наприкінці проєктування була розроблена логічна модель бази даних та обрано архітектуру додатку.

У результаті реалізації було розроблено та уведено в експлуатацію web-додаток, що відповідає всім функціональним вимогам із додатку А, разом із допоміжними додатками для обробки та агрегації даних електроспоживання.

Додаток впроваджений в роботу Науково-дослідного інституту енергоефективних технологій СумДУ (акт впровадження наведений у додатку В).

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Чи дійсно ви потребуєте монітор електроспоживання [Електронний ресурс] – Доступ до ресурсу: https://www.nytimes.com/wirecutter/reviews/home-energy-monitor/ (Дата звернення 13.05.2023)

2. Завантажте додаток. [Електронний ресурс] – Доступ до ресурсу: https://sense.com/app/ (Дата звернення 13.05.2023)

3. Web-додаток. [Електронний ресурс] – Доступ до ресурсу: https://www.generac.com/pwrview-apps (Дата звернення 13.05.2023)

4. Web-додаток. [Електронний ресурс] – Доступ до ресурсу: https://www.emporiaenergy.com/smart-home-energy-management-app (Дата звернення 13.05.2023)

5. Найкращі мови для web-розробки [Електронний ресурс] – Доступ до ресурсу: https://www.computerscience.org/bootcamps/guides/programming-languages-web-development/ (Дата звернення 13.05.2023)

6. Що таке PHP? [Електронний ресурс] – Доступ до ресурсу: https://freehost.com.ua/ukr/faq/wiki/chto-takoe-php/ (Дата звернення 13.05.2023)

7. Переваги мови Python [Електронний ресурс] – Доступ до ресурсу: https://disted.edu.vn.ua/courses/learn/7649 (Дата звернення 13.05.2023)

8. Що таке IDEF - Визначення, Методи та Переваги [Електронний ресурс] – Доступ до ресурсу: https://www.edrawsoft.com/what-is-idef.html (Дата звернення 13.05.2023)

9. Що таке діаграма використання? [Електронний ресурс] – Доступ до ресурсу: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/ (Дата звернення 13.05.2023)

10. Що таке SMART-цілі і навіщо вони потрібні [Електронний ресурс] – Доступ до ресурсу: https://brainlab.com.ua/uk/blog-uk/shho-take-smart-czili-i-navishho-vony-potribni (Дата звернення 13.05.2023)

11. Структура розбивки роботи [Електронний ресурс] – Доступ до ресурсу: https://www.projectmanager.com/guides/work-breakdown-structure (Дата звернення 13.05.2023)

12. Ризики проекту: аналіз, оцінка та стратегії управління [Електронний ресурс] – Доступ до ресурсу: https://skillsetter.io/blog/risk-management-ua (Дата звернення 13.05.2023)

13. Що таке шифрування паролів та коли достатньо? [Електронний ресурс] – Доступ до ресурсу: https://teampassword.com/blog/what-is-password-encryption-and-how-much-is-enough (Дата звернення 13.05.2023)

14. Що таке реляційна база даних? [Електронний ресурс] – Доступ до ресурсу: https://www.oracle.com/database/what-is-a-relational-database/ (Дата звернення 13.05.2023)

15. Що таке нереляційна база даних? [Електронний ресурс] – Доступ до ресурсу: https://www.mongodb.com/databases/non-relational (Дата звернення 13.05.2023)

16. Що таке MVT структура у Django [Електронний ресурс] – Доступ до ресурсу: https://www.educative.io/answers/what-is-mvt-structure-in-django (Дата звернення 13.05.2023)

ДОДАТОК А. ТЕХНІЧНЕ ЗАВДАННЯ

ТЕХНІЧНЕ ЗАВДАННЯ

на розробку інформаційної системи
«Web-додаток моніторингу електропоспоживання комунальними та промисловими об'єктами»

ПОГОДЖЕНО:

Доцент кафедри комп’ютерних наук

_____ Марченко А.В.

Студент групи ІТ-91

_____ Сенча Є.С.

Суми 2023

1. ПРИЗНАЧЕННЯ І МЕТА СТВОРЕННЯ WEB-ДОДАТКУ

Призначення web-додатку

Web-додаток буде надавати функціонал для отримання та перегляду даних щодо енергоспоживання, адміністрування web-додатком та адміністраторським управлінням об’єктами системи та групами користувачів. Також web-додаток має опрацьовувати та зберігати дані електроспоживання.

Мета створення web-додатку

Головна мета проекту – надавати змогу гнучко та зручно отримувати дані щодо енергоспоживання як у реальному часі, так і за певний проміжок у минулому. Аналіз отриманих агрегованих погодинно показників електроспоживання дозволить визначити об’єкти, що споживають забагато електроенергії та коригувати витрати закладів.

Цільова аудиторія

Цільовою аудиторією даного проекту є компанії та підприємства, що є досить великими та мають розгалужену систему об’єктів споживання електричної системи, та потребують моніторингу електроспоживання в режимі реального часу як окремих об’єктів (наприклад, цехами чи корпусами) так і в цілому

2. ВИМОГИ ДО WEB-ДОДАТКУ

2.1. Вимоги до web-додатку в цілому

2.1.1. Вимоги до структури й функціонування web-додатку

Метою дослідження є створення web-додатку, який надає змогу моніторити електроспоживання у реальному часі. Впровадження web-додатку дозволить проводити моніторинг електроспоживання в режимі реального часу на великих об’єктах громадського, промислового та обслуговування, а також в навчальних закладах різного рівня.

2.1.2. Вимоги до модераторів та користувачів

Користувачі повинні мати лише базові вміння працювати із веб-сторінками.

2.1.3. Вимоги до збереження інформації

Уся інформація надана у web-додатку повинна зберігатися у базі даних, реалізованій засобами системи управління базами даних Postgresql. Частина даних щодо авторизації користувачів (паролі) повинна зберігатися зашифрованою. [13]

2.1.4. Вимоги до розмежування доступу

Web-додаток має бути загальнодоступним у мережі Інтернет. Призначення web-додатку зумовлює реалізацію розмежування прав доступу між двома групами користувачів: адміністратор та зареєстрований користувач.

Адміністратор повинен мати можливість створювати, видаляти та редагувати дані, також розглядати запити на реєстрацію чи отримання нових ролей користувачами. Для окремих груп користувачів адміністратор повинен

визначати рівні доступу до перегляду показників електроспоживання відповідних об’єктів та закладів в цілому.

Зареєстрований користувач повинен мати доступ до даних електроспоживання відповідно до рівня його доступу (визначається можливістю отримати дані по електроспоживанню об’єкта та усих підлеглих йому об’єктів) та можливість подавати запити на отримання ролей.

2.2. Структура web-додатку

2.2.1. Загальні інформація про структуру web-додатку

До структури web-додатку входять усі його web-сторінки, які є загальнодоступними (енергоспоживання та отримання ролей) та сторінки адміністратора. Перелік сторінок web-додатку наступний:

- Енергоспоживання: форма для запиту даних та самі дані у табличній/графічній формах
- Отримання ролей: форма із списком установ та повідомленням до адміністратора
- Логін: введення даних свого аккаунту щоб отримати доступ
- Реєстрація
- Сторінка установ та об’єктів: пошук по існуючим установам та об’єктам, маніпуляції ними та додавання нових
- Сторінка запитів на нові ролі та реєстрацію: список із запитів, при натисканні перенаправляє на сторінку розгляду запиту на роль
- Запит на роль: дані користувача, чи є він зареєстрованим, та повідомлення від нього
- Датчики та сенсори: список датчиків на сенсорів та маніпуляція ними.

2.2.2. Навігація

Для зручної навігації повинно бути створене меню, що забезпечить швидке переміщення користувача по всім доступним сторінкам web-додатку. Меню має бути закріплене і розташовуватися зверху (у шапці) на кожній сторінці.

2.2.3. Дизайн та структура меню

Дизайн web-додатку має бути виконаний у мінімалістичному та сучасному стилі. Кольори та схему оформлення визначає замовник проекту та реалізує виконавець.

Види і розміри шрифтів повинні бути комфортними для перегляду. Інформаційні блоки, графічні матеріали та інші елементи web-сторінок повинні мати зручне і логічне розташування. Шаблон майбутнього програмного продукту зображено на рисунку А.1

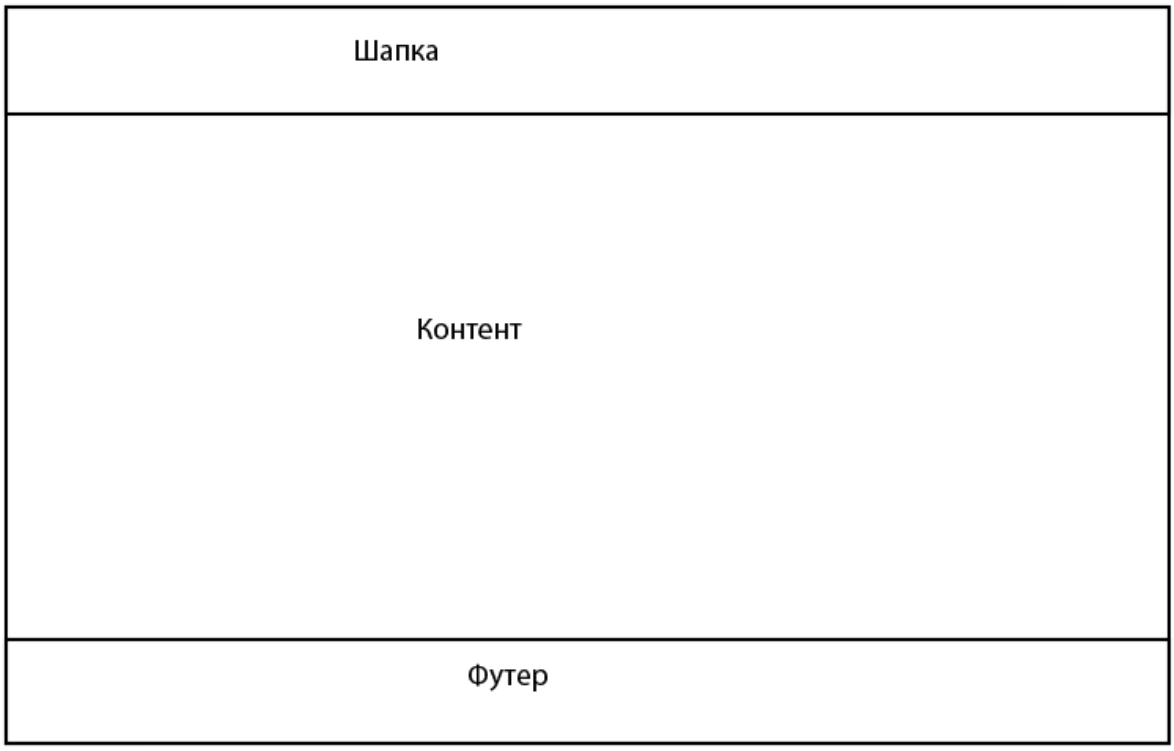


Рисунок А.1 – Схема головної сторінки

2.3. Вимоги до функціонування системи

Готовий додаток має задовольняти наступні функціональні вимоги:

- реєстрація та підтвердження реєстрації нових користувачами адміністраторами;
- надання інформації щодо електроспоживання відповідно до заданих параметрів запиту;
- додавання, видалення та редагування об’єктів системи;
- надання доступу до інформації згідно рівням доступу;
- формування pdf звіту по електроспоживанню по заданим параметрам.

2.4. Вимоги до видів забезпечення

2.4.1. Вимоги до інформаційного забезпечення

При реалізації веб-сайту буде використано наступні інструменти:

- Python3
- Django, gunicorn, celery
- Redis
- Git
- Docker, docker compose
- Html, JS, CSS
- Jquery
- Ubuntu, SSH, SLL
- PyCharm, Sublime text
- pdfkit

2.4.2. Вимоги до лінгвістичного забезпечення

Весь текст у web-додатку має бути українською, та має бути перебачено функціонал для майбутнього перекладу сайту.

2.4.3. Вимоги до програмного забезпечення

Для забезпечення стабільної роботи web-додатку веб-браузер користувача має бути актуальним (оновлення не раніше 2015 року).

3. Склад і зміст робіт із створення web-додатку

Відповідно до логічності створення та пріорітетності функціоналу, створення web-додатку було поділено на наступні етапи.

Таблиця А.2 - Етапи створення web-додатку

№	Склад і зміст робіт	**Строк розробки**
1	Розробка загальної структури додатку	23 дні
2	Розробка макетів фронтенду	20 днів
3	Розробка архітектури бекенду	13 днів
4	Розробка шаблонів сторінок реєстрації та логіну	5 днів
5	Розробка бекенд-функціоналу для логіну та реєстрації	22 днів
6	Розробка шаблонів сторінок для розгляду запитів на роль та реєстрацію	9 днів
7	Розробка бекенд-функціоналу для розгляду запитів на роль	13 днів
8	Розробка шаблонів сторінок для закладів та об’єктів	13 днів
9	Розробка бекенд-функціоналу для об’єктів та закладів	17 днів
10	Розробка шаблонів сторінок для енергоспоживання	28 днів
11	Розробка бекенд-функціоналу для отримання даних енергоспоживання	5 днів
12	Розробка шаблонів сторінок для подання запиту на нову роль	5 днів
13	Розробка бекенд-функціоналу для розгляду запитів на роль	13 днів

4. Потреби користувача

Потреби користувача, визначені на основі рішення замовника, представлені у таблиці А.1.

Таблиця А.1 – Потреби користувача

ID	Потреби користувача	Джерело
UN-01	Реєстрація	Користувач
UN-02	Логін	Користувач, Адмін
UN-03	Отримання даних електроспоживання	Користувач
UN-04	Подання запиту на отримання нової ролі	Користувач
UN-05	Перегляд запитів на реєстрацію та прийняття рішень по ним	Адмін
UN-06	Перегляд об’єктів та закладів	Адмін
UN-07	Редагування додавання та видалення об’єктів та закладів	Адмін
UN-08	Перегляд датчиків та сенсорів	Адмін
UN-09	Перегляд користувачів та їх даних	Адмін

5. Вимоги до складу і змісту робіт із введення web-додатку у використання

Реалізований функціонал web-додатку має бути протестований та затверджений замовником, сам додаток розміщено на веб-хостингу Сумського державного університету energokodros.sumdu.edu.ua.

ДОДАТОК Б. ПЛАНУВАННЯ РОБІТ

Головною метою проєкту є надання зручного доступу до перегляду показників електроспоживання згідно із рівнями доступу та параметрами запиту, шляхом створення web-додатку, який би надавав необхідний функціонал.

Для отримання доступу до перегляду агрегованих даних користувач спочатку має зареєструватись, вказуючи, до об'єкту якої установи він хоче мати доступ, після чого адміністратор підтверджує або відхиляє запит на реєстрацію, обираючи об'єкт доступу для користувача, у разі підтвердження. Також користувач може подавати запити на нові ролі. Адміністратор має доступ до перегляду агрегованих показників будь-яких установ, та до функціоналу із перегляду, створення, редагування та видалення будь-яких сутностей системи.

Завдяки цьому додатку великі споживачі електроенергії зможуть детально відслідковувати своє споживання та мати доступ до статистики та різноманітних фільтрів та налаштувань. web-додаток повинен бути впроваджений в експлуатацію відповідно календарному плану, визначеному в ТЗ замовника. Географія впровадження – Україна, із можливістю подальшого розширення географії користувачів.

Результат деталізації методом SMART розміщено у таблиці Б.1. [10]

Таблиця Б.1 - Деталізація мети проєкту методом SMART

Specific	Готовий сайт
Measurable	На сайт можна зайти і виконати будь-які дії із вимог
Achievable	Необхідні знання для розробки веб-інтерфейсів та знання реляційних баз даних
Relevant	Продукт надасть змогу відслідковувати та, виходячи з цього, заощаджувати електроенергію
Time-framed	Кінцевий термін закінчення бета-тестування - 20 квітня 2023 року

ПЛАНУВАННЯ ЗМІСТУ РОБІТ

WBS – це графічне подання згрупованих елементів проекту у вигляді пакета робіт, які ієрархічно пов’язані з продуктом проекту. На верхньому першому рівні WBS фіксується продукт проекту. Він повинен відповідати продукту проекту. Наступний II рівень відповідає діям або основним заходам для досягнення продукту проекту.

Потім триває розбивка цих дій доти, поки не відбувається виконання дій елементарних робіт. Елементарні роботи – це роботи, які мають один чіткий результат, який використовується при прийнятті цієї роботи; на які призначений один конкретний відповідальний; на неї можна обчислити витрати праці і тривалість виконання. Зазвичай декомпозиція завершується тоді, коли для розкриття змісту потрібні вузькі фахівці, що знають технологічні особливості їх виконання.

OBS структура проекту – організаційна структура виконавців (організацій) проекту. Визначається за переліком пакетів робіт нижнього рівня кожної гілки WBS-структури. Організаційна структура представляє собою графічне відображення учасників проекту та їх відповідальних осіб, які задіяні в реалізації проекту. [11]

На верхньому рівні OBS розташована команда проекту. На наступному рівні фіксуються виконавці: організації, відділи тощо. Потім, рівнем нижче, для кожного виконавця вказують прізвища конкретних осіб, які будуть відповідати за виконання елементарних робіт WBS.

Діаграма Ганта - один з найбільш популярних інструментів управління проектами. Це набір графічних гістограм, які фіксують терміни, взаємозв'язок і віхи реалізації окремих складових проекту.

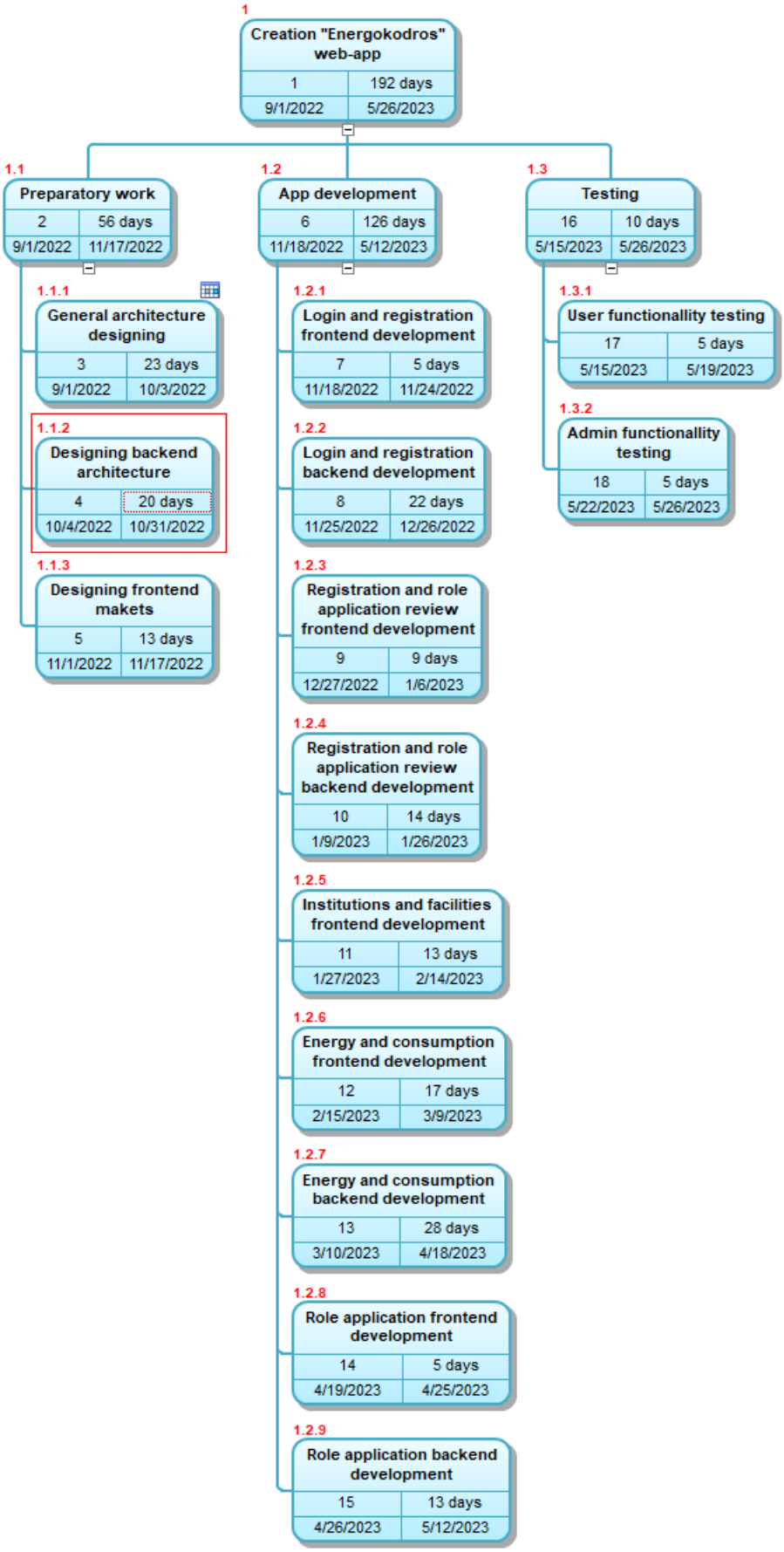


Рисунок Б.1 - Діаграма WBS

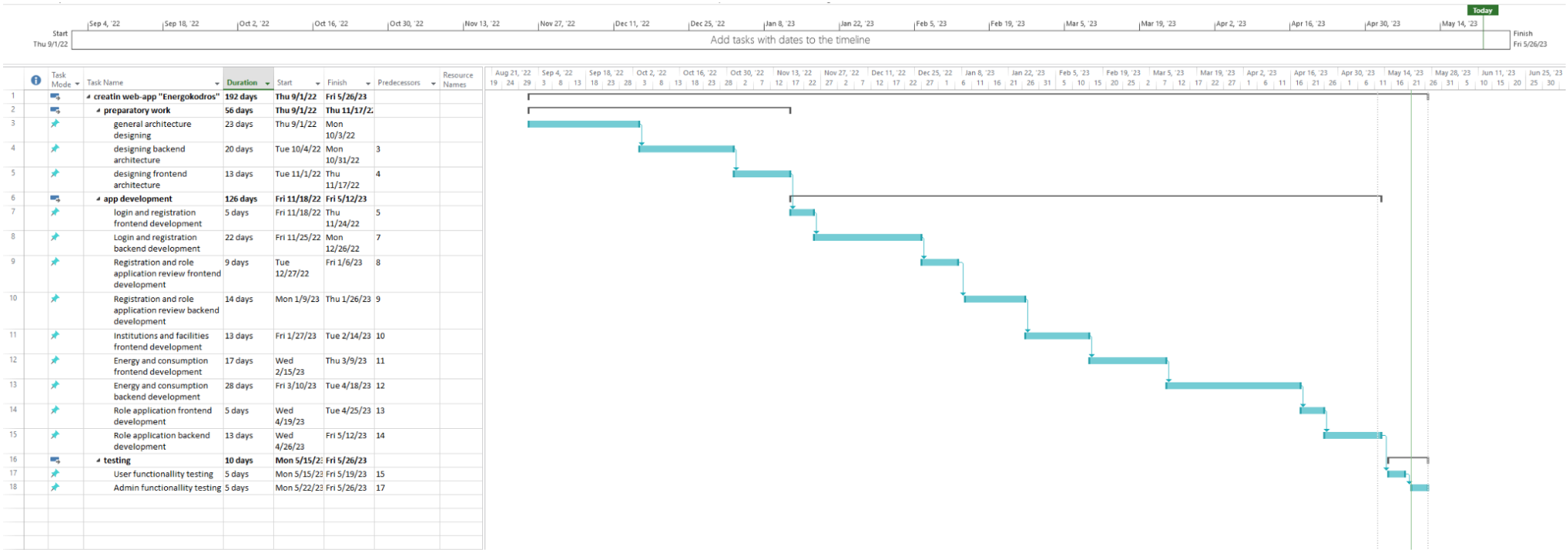
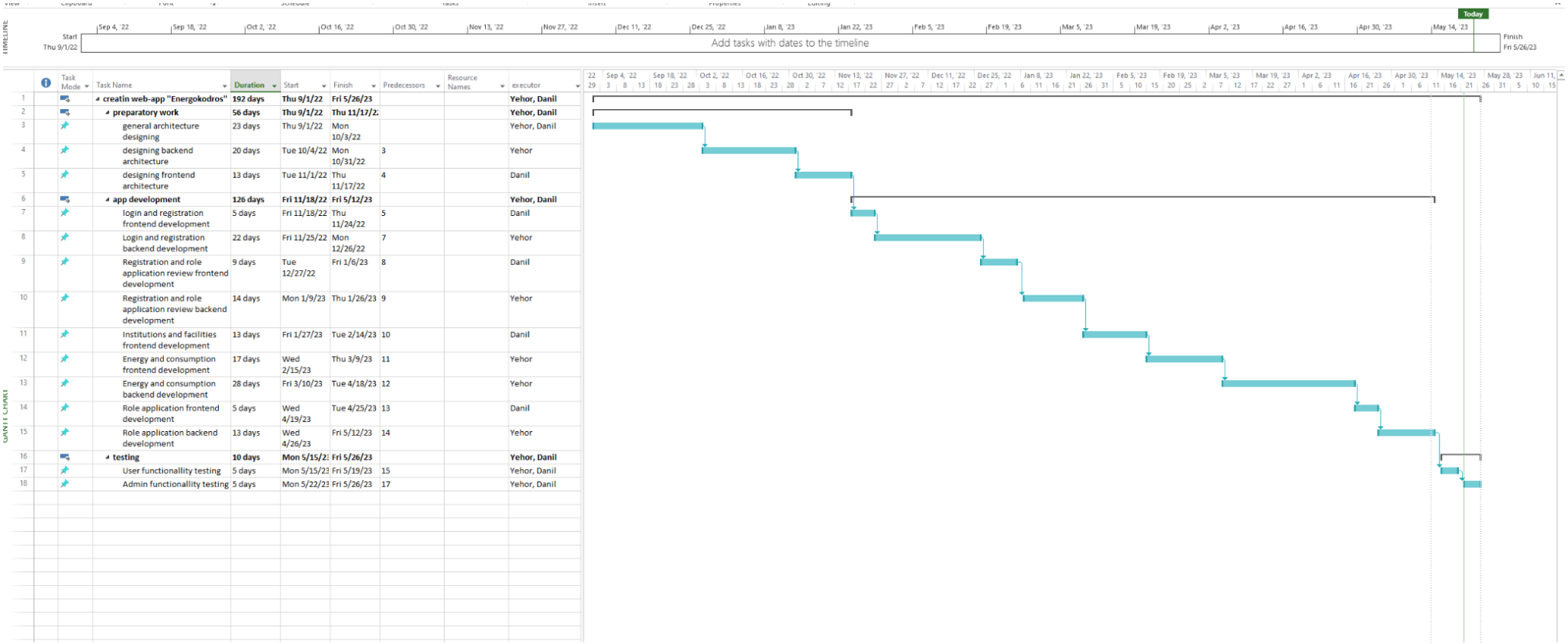


Рисунок Б.2 - OBS структура проєкту



	Task Mode	Task Name	Duration	Start	Finish	Predecessors	Resource Names	executor
1		**creatin web-app "Energokodros"**	**192 days**	**Thu 9/1/22**	**Fri 5/26/23**			**Yehor, Danil**
2		**preparatory work**	**56 days**	**Thu 9/1/22**	**Thu 11/17/2**			**Yehor, Danil**
3		general architecture designing	23 days	Thu 9/1/22	Mon 10/3/22			Yehor, Danil
4		designing backend architecture	20 days	Tue 10/4/22	Mon 10/31/22	3		Yehor
5		designing frontend architecture	13 days	Tue 11/1/22	Thu 11/17/22	4		Danil
6		**app development**	**126 days**	**Fri 11/18/22**	**Fri 5/12/23**			**Yehor, Danil**
7		login and registration frontend development	5 days	Fri 11/18/22	Thu 11/24/22	5		Danil
8		Login and registration backend development	22 days	Fri 11/25/22	Mon 12/26/22	7		Yehor
9		Registration and role application review frontend development	9 days	Tue 12/27/22	Fri 1/6/23	8		Danil
10		Registration and role application review backend development	14 days	Mon 1/9/23	Thu 1/26/23	9		Yehor
11		Institutions and facilities frontend development	13 days	Fri 1/27/23	Tue 2/14/23	10		Danil
12		Energy and consumption frontend development	17 days	Wed 2/15/23	Thu 3/9/23	11		Yehor
13		Energy and consumption backend development	28 days	Fri 3/10/23	Tue 4/18/23	12		Yehor
14		Role application frontend development	5 days	Wed 4/19/23	Tue 4/25/23	13		Danil
15		Role application backend development	13 days	Wed 4/26/23	Fri 5/12/23	14		Yehor
16		**testing**	**10 days**	**Mon 5/15/2**	**Fri 5/26/23**			**Yehor, Danil**
17		User functionallity testing	5 days	Mon 5/15/23	Fri 5/19/23	15		Yehor, Danil
18		Admin functionallity testing	5 days	Mon 5/22/23	Fri 5/26/23	17		Yehor, Danil

Рисунок Б.3 - Діаграма Ганта

Далі проведемо аналіз потенціальних проблем, які можуть трапитися під час реалізації проекту чи вже під час його життя. [12]

Таблиця Б. 2 - Ідентифікація ризиків

Номер	Назва	Ймовірність (0,1-0,9)	Вплив (0,05-0,8)	Ранг
1	Непорозуміння між розробниками та замовником	0,2	0,3	0,06
2	Поява продукту-конкурента	0,1	0,4	0.04
3	Недоліки ТЗ	0,3	0,2	0.06
4	Низька кваліфікація розробників	0,1	0,6	0.06
5	Хвороби працівників	0,1	0,3	0.03
6	Фундаментальні зміни в ТЗ	0,1	0,6	0.06
7	Помилки у плануванні розробки	0,6	0,2	0.12
8	Невірно обраний технологічний стек	0,1	0,6	0.06
9	Некваліфікованість розробників	0,3	0,5	0.15

Для оцінки проблем буде використовуватися матриця ймовірності та ризику. Ризики, розташовані в зеленій зоні, вважатимуться прийнятними, жовта зона позначатиме виправданий ризик, а червона зона буде позначати недопустимі проблеми. Зелена оцінка вказує на те, що проблема може мати легкий негативний вплив на загальну роботу. Жовта оцінка позначатиме, що існує ризик, але застосування додаткових стратегій може допомогти уникнути проблем. Щодо червоної оцінки, необхідно якомога швидше відкинути такі проблеми або значно знизити можливість завдання шкоди проекту.

Таблиця Б.3 – Матриця ймовірності та впливу

Ймовірність	Вплив загрози (ризику)				
	Дуже малий 0,05	Малий 0,1	Середній 0,2	Великий 0,4	Дуже великий 0,8
0,9					
0,7					
0,5			R7		
0,3			R3	R4, R6	R9
0,1			R1, R5	R2, R8	

Таблиця Б.4 - Шкала оцінювання за рівнем ризику

№	Назва	Межі	Ризики, що входять
1	Прийнятні	$0,005 \leq R \leq 0,05$	1, 2,5,8
2	Виправдані	$0,05 < R \leq 0,14$	3,4,6,7
3	Недопустимі	$0,14 < R \leq 0,72$	9

Виходячи із попереднього аналізу ризиків, створимо таблицю реагування на них.

Таблиця Б.5 - Протидія ризикам

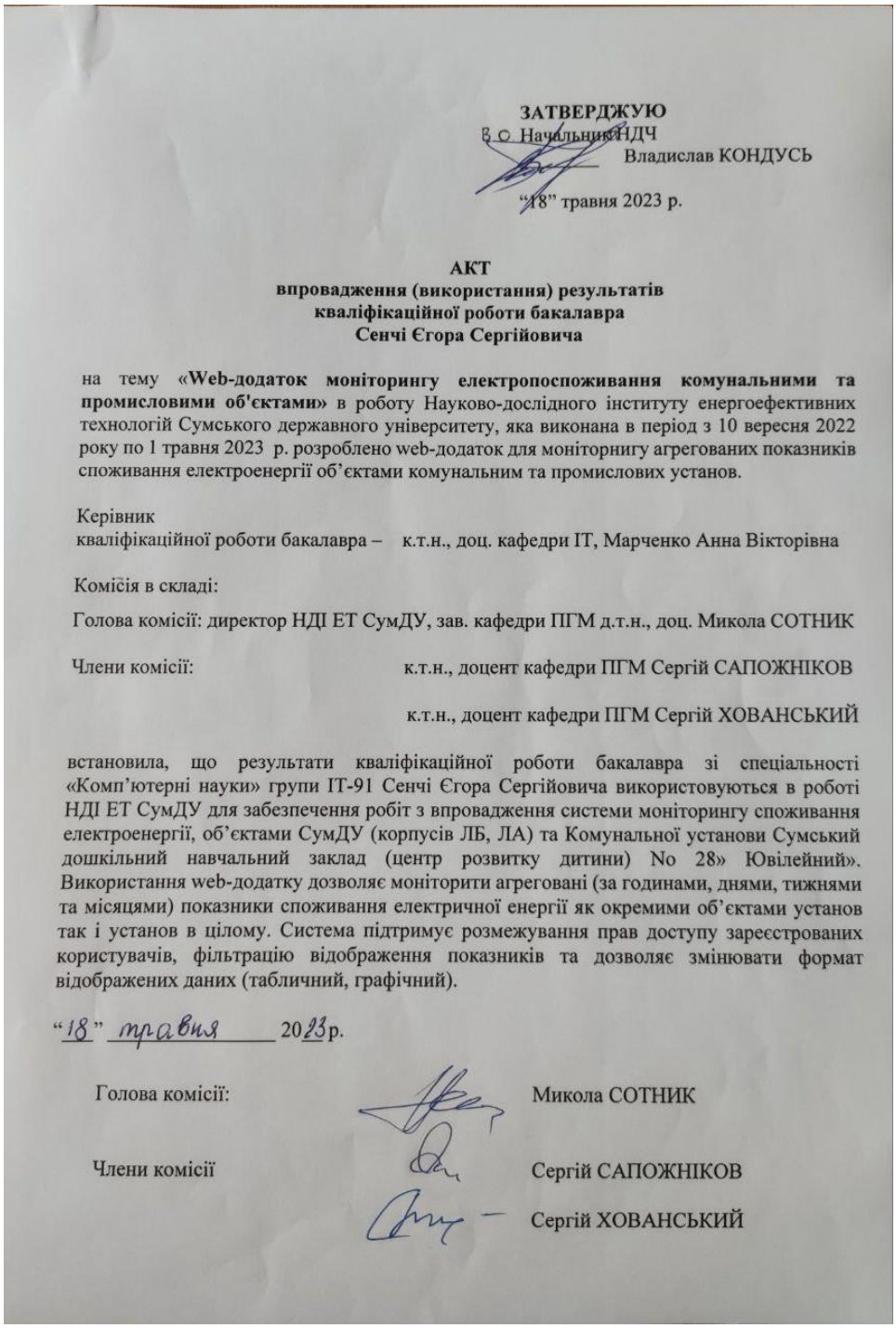
ID	Статус ризику	Опис ризику	Ймовірність виникнення	Вплив	Ранг ризику	План А	Тип стратегії реагування	План Б
RS_1	Відкритий	Непорозуміння між розробниками та замовником	Низька	Середній	3	Детально та прискіпливо підходити до діалогу із замовником	Попередження	Аналізувати потреби замовника, а не просто слідувати його запитам
RS_2	Відкритий	Поява продукту-конкурента	Низька	Середній	3	Фокусуватися на конкурентних перевагах та розробляти якісний продукт	Прийняття	Аналізувати продукт-конкурент та фокусуватися на своїх перевагах
RS_3	Відкритий	Недоліки ТЗ	Середня	Низька	4	Розробляти гнучко та легкозмінно	Попередження	Виправляти помилки
RS_4	Закритий	Низька кваліфікація розробників	Низька	Великий	5	Переконуватися у кваліфікації виконавців	Попередження	Найняти інших людей
RS_5	Відкритий	Хвороби працівників	Низька	Середній	3	Внести корективи у графік	Прийняття	Змінювати строки розробки
RS_6	Відкритий	Фундаментальні зміни в ТЗ	Низька	Високий	6	Робити із урахуванням можливих змін	Прийняття	Перероблювати, враховуючи зміни
RS_7	Відкритий	Помилки у плануванні розробки	Середня	Середній	5	Залишати додатковий час на непрораховані роботи	Прийняття	Перероблювати, враховуючи зміни

Продовження таблиці Б.5

RS_8	Закритий	Невірно обраний технологічний стек	Низька	Високий	6	Провести детальний аналіз наявних інструментів Писати максимально незалежний код	Попередження	Міграція на інші інструменти
RS_9	Відкритий	Некваліфікованість розробників	Низька	Високий	6	Ретельно відбирати персонал	Попередження	Наймати інших людей

Додаток В

Акт впровадження



ЗАТВЕРДЖУЮ
В.о. Начальника НДЧ
Владислав КОНДУСЬ
"18" травня 2023 р.

АКТ
впровадження (використання) результатів
кваліфікаційної роботи бакалавра
Сенчі Єгора Сергійовича

на тему **«Web-додаток моніторингу електропоспоживання комунальними та промисловими об'єктами»** в роботу Науково-дослідного інституту енергоефективних технологій Сумського державного університету, яка виконана в період з 10 вересня 2022 року по 1 травня 2023 р. розроблено web-додаток для моніторнигу агрегованих показників споживання електроенергії об'єктами комунальним та промислових установ.

Керівник
кваліфікаційної роботи бакалавра – к.т.н., доц. кафедри ІТ, Марченко Анна Вікторівна

Комісія в складі:

Голова комісії: директор НДІ ЕТ СумДУ, зав. кафедри ПГМ д.т.н., доц. Микола СОТНИК

Члени комісії: к.т.н., доцент кафедри ПГМ Сергій САПОЖНІКОВ

к.т.н., доцент кафедри ПГМ Сергій ХОВАНСЬКИЙ

встановила, що результати кваліфікаційної роботи бакалавра зі спеціальності «Комп'ютерні науки» групи ІТ-91 Сенчі Єгора Сергійовича використовуються в роботі НДІ ЕТ СумДУ для забезпечення робіт з впровадження системи моніторингу споживання електроенергії, об'єктами СумДУ (корпусів ЛБ, ЛА) та Комунальної установи Сумський дошкільний навчальний заклад (центр розвитку дитини) No 28» Ювілейний». Використання web-додатку дозволяє моніторити агреговані (за годинами, днями, тижнями та місяцями) показники споживання електричної енергії як окремими об'єктами установ так і установ в цілому. Система підтримує розмежування прав доступу зареєстрованих користувачів, фільтрацію відображення показників та дозволяє змінювати формат відображених даних (табличний, графічний).

"18" травня 2023 р.

Голова комісії: Микола СОТНИК

Члени комісії Сергій САПОЖНІКОВ

Сергій ХОВАНСЬКИЙ

Рисунок В.1 – Акт впровадження

Додаток Г

Програмний код

Структура проєкту

Рисунок Г.1 – Структура проєкту

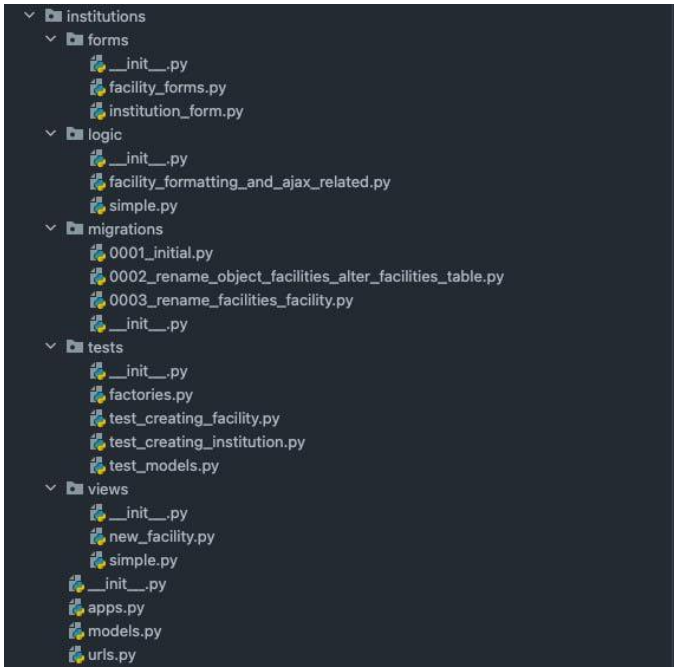


Рисунок Г.2 – Структура проєкту

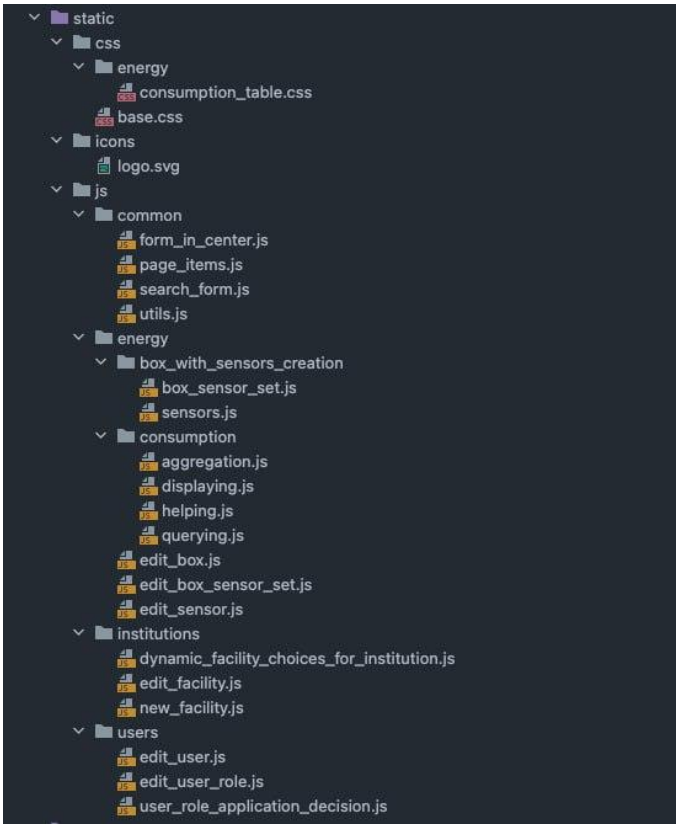


Рисунок Г.3 – Структура проєкту

Рисунок Г.4 – Структура проєкту

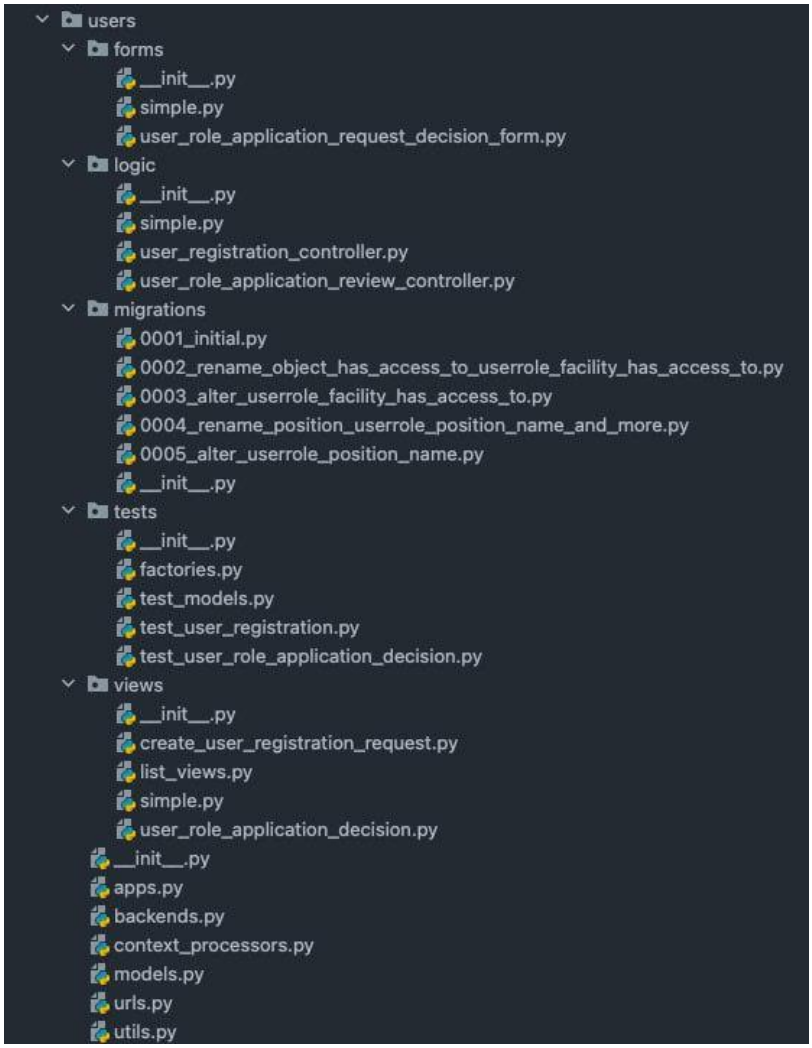


Рисунок Г.5 – Структура проєкту

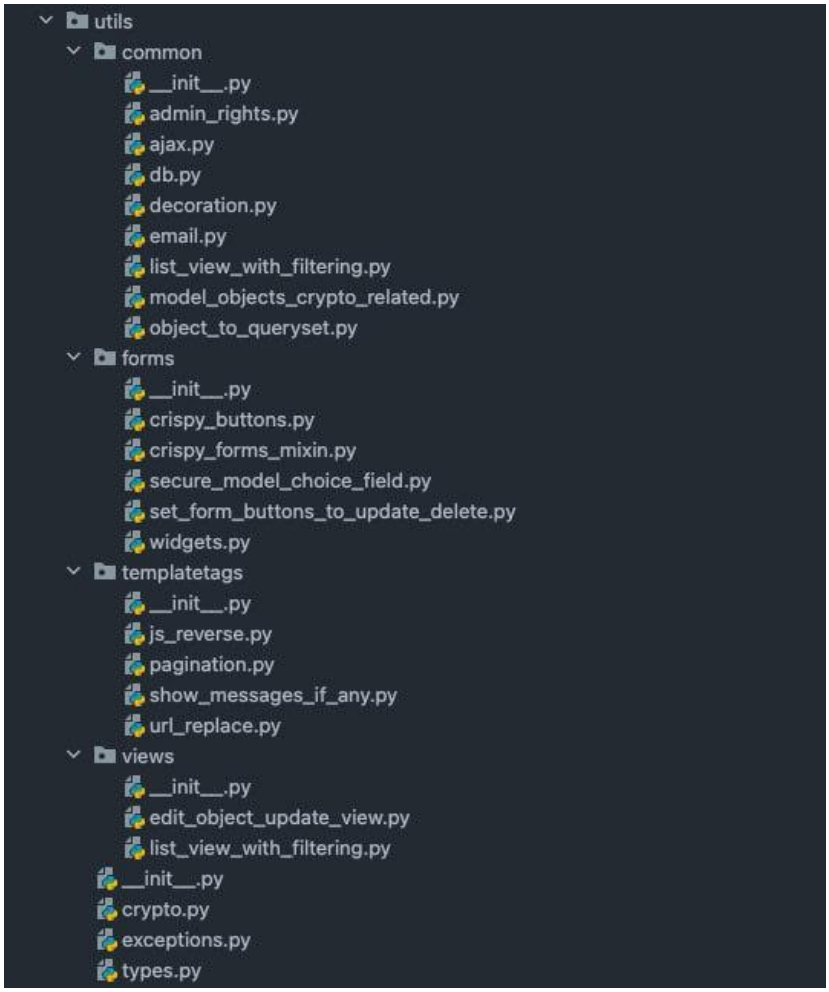


Рисунок Г.6 – Структура проєкту

Файл: gunicorn.conf.py

```
import os

bind = f"[::]:{os.getenv('APP_PORT')}"
workers = 3
# Whether to send Django output to the error log
capture_output = True
loglevel = "info"
redirect_stderr = True
```

Файл: manage.py

```
#!/usr/bin/env python
"""Django's command-line utility for administrative tasks."""
import os
import sys


def main():
    """Run administrative tasks."""
    os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'energokodros.settings')
```

```
    try:
        from django.core.management import execute_from_command_line
    except ImportError as exc:
        raise ImportError(
            "Couldn't import Django. Are you sure it's installed and "
            "available on your PYTHONPATH environment variable? Did you "
            "forget to activate a virtual environment?"
        ) from exc
    execute_from_command_line(sys.argv)


if __name__ == '__main__':
    main()
```

Файл: institutions/models.py

```
from django.db import models
from django.urls import reverse_lazy
from django.utils.translation import gettext as _
from treebeard.ns_tree import NS_Node, NS_NodeManager


class FacilityManager(NS_NodeManager):
    def get_institutions(self):
        # we separate a regular facility (that have a parent facility)
        # and institutions, that are on top of hierarchy
        return self.model.get_root_nodes()


class Facility(NS_Node):
    name = models.CharField(
        _("назва об'єкту"),
        max_length=1000,
        blank=True,
        null=False,
        db_column='name'
    )
    description = models.TextField(
        _("опис об'єкту"),
        blank=True,
        null=True,
        db_column='description'
    )

    objects = FacilityManager()

    class Meta:
        db_table = 'facilities'
        verbose_name = _("Об'єкт")
        verbose_name_plural = _("Об'єкти")

    def __str__(self):
        if self.is_root():
            return _(f'заклад {self.name}')
        return _(f"об'єкт {self.name} із {self.get_institution().name}")

    def get_absolute_url(self):
        return reverse_lazy('edit-facility', kwargs={'pk': self.pk})

    def get_institution(self) -> 'Facility':
        return self.get_root()
```

```
    # methods below are not implemented for nested set in django_treebeard, so
    # currently they are just 'stubbed'
    @classmethod
    def find_problems(cls):
        raise NotImplementedError

    @classmethod
    def fix_tree(cls):
        raise NotImplementedError
```

Файл: institutions/__init__.py

Файл: institutions/apps.py

```
from django.apps import AppConfig


class InstitutionsConfig(AppConfig):
    default_auto_field = 'django.db.models.BigAutoField'
    name = 'institutions'
```

Файл: institutions/urls.py

```
from django.urls import path

from institutions.models import Facility
from institutions.views import (
    CreateFacilityView, CreateInstitutionView, FacilitiesListView,
    get_institution_facilities_choices, EditFacilityView,
)
from utils.common import redirect_to_object_pk_in_post

urlpatterns = [
    path(
        'institutions/',
        FacilitiesListView.as_view(),
        name='facilities-list'
    ),
    path(
        'new-institution/',
        CreateInstitutionView.as_view(),
        name='new-institution'
    ),
    path(
        'new-facility/',
        CreateFacilityView.as_view(),
        name='new-facility'
    ),
    path(
        'get-institution-facilities/',
        get_institution_facilities_choices,
        name='get-institution-facilities'
    ),
    path(
        'edit-facility/<hashed_int:pk>',
        EditFacilityView.as_view(),
        name='edit-facility'
    )
]
```

```
helper_urls = [
    path(
        'edit-facility-pk-in-post/',
        redirect_to_object_pk_in_post(Facility, 'edit-facility'),
        name='edit-facility-pk-in-post'
    )
]

urlpatterns += helper_urls
```

Файл: institutions/migrations/__init__.py

Файл:
institutions/migrations/0002_rename_object_facilities_alter_facilities_table.py

```
# Generated by Django 4.0.5 on 2022-07-25 09:11

from django.db import migrations


class Migration(migrations.Migration):

    dependencies = [
        ('users', '0001_initial'),
        ('institutions', '0001_initial'),
    ]

    operations = [
        migrations.RenameModel(
            old_name='Object',
            new_name='Facilities',
        ),
        migrations.AlterModelTable(
            name='facilities',
            table='facilities',
        ),
    ]
```

Файл: institutions/migrations/0003_rename_facilities_facility.py

```
# Generated by Django 4.0.5 on 2022-07-26 11:28

from django.db import migrations


class Migration(migrations.Migration):

    dependencies = [
        ('users', '0001_initial'),
        ('institutions',
'0002_rename_object_facilities_alter_facilities_table'),
    ]

    operations = [
        migrations.RenameModel(
            old_name='Facilities',
            new_name='Facility',
        ),
```

```
    ]
```

Файл: institutions/migrations/0001_initial.py

```
# Generated by Django 4.0.5 on 2022-07-23 10:21

from django.db import migrations, models


class Migration(migrations.Migration):

    initial = True

    dependencies = [
    ]

    operations = [
        migrations.CreateModel(
            name='Object',
            fields=[
                ('id', models.BigAutoField(auto_created=True, primary_key=True,
serialize=False, verbose_name='ID')),
                ('lft', models.PositiveIntegerField(db_index=True)),
                ('rgt', models.PositiveIntegerField(db_index=True)),
                ('tree_id', models.PositiveIntegerField(db_index=True)),
                ('depth', models.PositiveIntegerField(db_index=True)),
                ('name',      models.CharField(blank=True,      db_column='name',
max_length=1000, verbose_name="назва об'єкту")),
                ('description',                      models.TextField(blank=True,
db_column='description', null=True, verbose_name="опис об'єкту")),
            ],
            options={
                'verbose_name': "Об'єкт",
                'verbose_name_plural': "Об'єкти",
                'db_table': 'objects',
            },
        ),
    ]
```

Файл: institutions/forms/institution_form.py

```
from django.forms import Textarea
from django.utils.translation import gettext_lazy as _

from institutions.models import Facility
from utils.forms import create_primary_button, CrispyModelForm


class InstitutionForm(CrispyModelForm):
    class Meta:
        model = Facility
        fields = ('name', 'description')
        labels = {
            'name':        _('Назва'),
            'description': _('Опис')
        }
        widgets = {
            'description': Textarea(attrs={'rows': 3})
        }
        buttons = (create_primary_button(_('Додати установу')),)
```

```
    def save(self, commit=True):
        if self.errors:
            # pass raising exception to base class
            super().save()
        Facility.add_root(instance=self.instance)
        return self.instance
```

Файл: institutions/forms/__init__.py

```
from .facility_forms import NewFacilityForm, FacilityEditForm
from .institution_form import InstitutionForm
```

Файл: institutions/forms/facility_forms.py

```
from django import forms
from django.db.models import Model
from django.utils.translation import gettext_lazy as _

from institutions.logic import (
    get_facility_name_space_padded_according_to_nesting,
    label_from_user_role_for_facility_roles,
)
from institutions.models import Facility
from users.models import UserRole
from utils.forms import (
    create_primary_button, CrispyModelForm, SecureModelChoiceField,
    SelectWithFormControlClass, UPDATE_DELETE_BUTTONS_SET,
)
from utils.views import AdditionalSetupRequiredFormMixin


class NewFacilityForm(CrispyModelForm):
    # based on this field choice user will be given corresponding facilities
choices
    # to set as parent for a new facility
    institution = SecureModelChoiceField(
        queryset=Facility.objects.get_institutions(),
        required=False,
        label=_("Оберіть заклад, якому буде належати об'єкт"),
        empty_label=None
    )
    # this field should be populated based on institution choice by js
    parent_facility = SecureModelChoiceField(
        queryset=Facility.objects.all(),
        required=True,
        label=_("Оберіть батьківський об'єкт"),
        widget=SelectWithFormControlClass(attrs={'size': 7}),
        empty_label=None
    )

    class Meta:
        model = Facility
        fields = ('name', 'description')
        labels = {
            'name':        _('Назва'),
            'description': _('Опис'),
        }
        widgets = {
            'description': forms.Textarea(attrs={'rows': 3}),
        }
        fields_order = ('institution', 'parent_facility') + fields
```

```
        buttons = (create_primary_button(_("Додати об'єкт")),)


class FacilityEditForm(CrispyModelForm, AdditionalSetupRequiredFormMixin):
    # info only field, qs is filled in custom method
    descendants = SecureModelChoiceField(
        queryset=Facility.objects.none(),
        label=_("Підлеглі об'єкти"),
        empty_label=None,
        required=False,
        disabled=True,
        widget=SelectWithFormControlClass(attrs={'size': 7}),

label_from_instance_function=get_facility_name_space_padded_according_to_nesting
    )
    # info only field, qs is filled in custom method
    roles_that_have_access_to_this_facility = SecureModelChoiceField(
        queryset=UserRole.objects.none(),
        label=_('Ролі, що мають доступ'),
        empty_label=None,
        required=False,
        disabled=True,
        widget=SelectWithFormControlClass(attrs={'size': 5}),
        label_from_instance_function=label_from_user_role_for_facility_roles
    )

    class Meta:
        model = Facility
        fields = ('name', 'description')
        labels = {
            'name':        _('Назва'),
            'description': _('Опис'),
        }
        widgets = {
            'description': forms.Textarea(attrs={'rows': 3}),
            'institution': forms.Textarea(attrs={'rows': 1})
        }
        buttons = UPDATE_DELETE_BUTTONS_SET

    def additionally_setup(self, obj: Model):
        # noinspection PyTypeChecker
        facility: Facility = obj
        self.fields['descendants'].queryset = facility.get_descendants()
        self.fields['roles_that_have_access_to_this_facility'].queryset        =
facility.users_roles


Файл: institutions/tests/test_creating_institution.py

from django.test import TestCase
from django.urls import reverse_lazy
from faker import Faker

from institutions.models import Facility
from users.tests.factories import create_admin_client


class InstitutionCreationTest(TestCase):
    def setUp(self):
        self.client = create_admin_client()
        self.creation_url = reverse_lazy('new-institution')
        fake = Faker()
        self.institution_name = fake.name()
```

```
        self.institution_description = fake.text(max_nb_chars=200)

    def test_creating_institution(self):
        resp = self.client.post(
            self.creation_url,
            data={
                'name':        self.institution_name,
                'description': self.institution_description
            },
            follow=True
        )
        self.assertEqual(
            resp.status_code,
            200,
            'invalid return code with correct data'
        )
        self.assertTrue(
            Facility.objects.filter(
                name=self.institution_name,
                description=self.institution_description
            ).count() == 1,
            'Institution not created or created multiple'
        )
```

Файл: institutions/tests/test_creating_facility.py

```
from django.test import TestCase
from django.urls import reverse
from faker import Faker

from users.tests.factories import create_admin_client
from institutions.models import Facility
from utils.common import hash_id


class FacilityCreationTest(TestCase):
    def setUp(self):
        self.client = create_admin_client()
        self.creation_url = reverse('new-facility')
        fake = Faker()
        self.parent_facility: Facility = Facility.add_root(
            name=fake.name, description=fake.text(max_nb_chars=200)
        )
        self.facility_name = fake.name()
        self.facility_description = fake.text(max_nb_chars=200)

    def test_creating_facility(self):
        parent_facility_id = hash_id(
            self.parent_facility
        )
        response = self.client.post(
            self.creation_url,
            data={
                'name':            self.facility_name,
                'description':     self.facility_description,
                'parent_facility': parent_facility_id
            },
            follow=True
        )
        self.assertEqual(
            response.status_code,
            200,
```

```
            'invalid response code with correct data'
        )
        created_facility = Facility.objects.get(
            name=self.facility_name,
            description=self.facility_description,
        )
        self.parent_facility.refresh_from_db()
        self.assertTrue(
            created_facility.is_child_of(self.parent_facility),
            'parent facility is set incorrectly'
        )
```

Файл: institutions/tests/__init__.py

Файл: institutions/tests/factories.py

```
import factory
from factory import django

from institutions.models import Facility


class InstitutionFactory(django.DjangoModelFactory):
    """by default creates root node"""
    name = factory.Sequence(lambda n: f'institution {n}')
    description = factory.Sequence(lambda n: f'institution {n} description')

    @classmethod
    def _create(cls, model_class: 'Facility', *args, **kwargs):
        return model_class.add_root(**kwargs)

    class Meta:
        model = Facility
```

Файл: institutions/tests/test_models.py

```
from django.test import TestCase

from institutions.models import Facility


class FacilityTest(TestCase):
    def test_creating_facility_set(self):
        root: Facility = Facility.add_root(name='root')
        child: Facility = root.add_child(name='root child')
        child_child: Facility = child.add_child(name='child child')
        root.refresh_from_db()
        self.assertTrue(
            child.is_child_of(root),
            'child is not child of root'
        )
        self.assertTrue(
            child_child.is_descendant_of(root),
            'child_child is not descendant of root'
        )
```

Файл: institutions/logic/__init__.py

```
from .facility_formatting_and_ajax_related import (
    get_facility_name_space_padded_according_to_nesting,
    make_facility_and_descendants_choices_with_padding_according_to_nesting,
    make_institution_facilities_choices_with_padding_according_to_nesting,
)
from .simple import (
    get_all_descendants_of_facility_with_self,
get_all_facilities_of_facility_institution,
    label_from_user_role_for_facility_roles,
)
```

Файл: institutions/logic/simple.py

```
from django.db.models import QuerySet

from institutions.models import Facility
from users.models import UserRole


def label_from_user_role_for_facility_roles(role: UserRole) -> str:
    return f'{role.user.full_name}, {role.position_name}'


def get_all_facilities_of_facility_institution(facility: Facility):
    return facility.get_tree(facility.get_institution())


def get_all_descendants_of_facility_with_self(facility: Facility) -> QuerySet:
    return facility.get_tree(facility)
```

Файл: institutions/logic/facility_formatting_and_ajax_related.py

```
from django.utils.safestring import mark_safe

from institutions.models import Facility
from utils.common import compose_secure_choices_for_queryset
from utils.common.model_objects_crypto_related import Choices


def  get_facility_name_space_padded_according_to_nesting(facility:  Facility)  ->
str:
    return mark_safe('  ' * (facility.depth - 1) + facility.name)


def
make_institution_facilities_choices_with_padding_according_to_nesting(institutio
n: Facility) \
        -> Choices:
    if not institution.is_root():
        raise ValueError('provided objects is a facility, not institution')
    return
make_facility_and_descendants_choices_with_padding_according_to_nesting(institut
ion)


def
make_facility_and_descendants_choices_with_padding_according_to_nesting(facility
: Facility) \
        -> Choices:
    return compose_secure_choices_for_queryset(
        facility.get_tree(facility),
```

```
        get_facility_name_space_padded_according_to_nesting
    )
```

Файл: institutions/views/new_facility.py

```
from django.contrib import messages
from django.http import JsonResponse
from django.urls import reverse_lazy
from django.utils.translation import gettext_lazy as _
from django.views.generic import FormView

from institutions.forms import NewFacilityForm
from institutions.logic import (
    make_institution_facilities_choices_with_padding_according_to_nesting,
)
from institutions.models import Facility
from utils.common import get_object_by_hashed_id_or_404
from utils.common.admin_rights import admin_rights_and_login_required


@admin_rights_and_login_required
def get_institution_facilities_choices(request) -> JsonResponse:
    institution: Facility = get_object_by_hashed_id_or_404(  # noqa
        Facility,
        request.POST.get('institution_id')
    )
    formatted_choices_ordered                                                    =
make_institution_facilities_choices_with_padding_according_to_nesting(
        institution
        )
    return JsonResponse(formatted_choices_ordered, safe=False)


@admin_rights_and_login_required
class CreateFacilityView(FormView):
    template_name = 'institutions/new_facility.html'
    form_class = NewFacilityForm
    success_url = reverse_lazy('facilities-list')

    def form_valid(self, form):
        parent_facility: Facility = form.cleaned_data['parent_facility']
        new_facility = form.save(commit=False)
        parent_facility.add_child(instance=new_facility)
        messages.success(self.request, _("Об'єкт успішно додано"))
        return super().form_valid(form)
```

Файл: institutions/views/__init__.py

```
from .new_facility import CreateFacilityView, get_institution_facilities_choices
from .simple import (
    CreateInstitutionView, FacilitiesListView,
    EditFacilityView,
)
```

Файл: institutions/views/simple.py

```
from django.contrib import messages
from django.urls import reverse_lazy
from django.utils.translation import gettext_lazy as _
from django.views.generic import FormView
```

```
from institutions.forms import FacilityEditForm, InstitutionForm
from institutions.models import Facility
from utils.common import admin_rights_and_login_required
from utils.views import ListViewWithFiltering
from utils.views.edit_object_update_view import EditDeleteObjectUpdateView


@admin_rights_and_login_required
class FacilitiesListView(ListViewWithFiltering):
    queryset = Facility.objects.all()
    fields_order_by_before_pk = ('depth',)
    filter_fields = ('name', 'description')
    template_name = 'institutions/facility_list.html'


@admin_rights_and_login_required
class CreateInstitutionView(FormView):
    form_class = InstitutionForm
    template_name = 'institutions/new_institution.html'
    success_url = reverse_lazy('facilities-list')

    def form_valid(self, form):
        form.save()
        messages.success(
            self.request,
            _('Установу успішно додано')
        )
        return super().form_valid(form)


@admin_rights_and_login_required
class EditFacilityView(EditDeleteObjectUpdateView):
    model = Facility
    form_class = FacilityEditForm
    template_name = 'institutions/edit_facility.html'
    success_url = reverse_lazy('facilities-list')
```

Файл: energy/models.py

```
from django.core.validators import MaxValueValidator, MinValueValidator
from django.db import models
from django.urls import reverse_lazy
from django.utils.translation import gettext as _

from energokodros.settings import SENSOR_COUNT_PER_BOX
from institutions.models import Facility


class Box(models.Model):
    box_id = models.AutoField(primary_key=True)
    identifier = models.CharField(
        max_length=256,
        blank=False,
        null=False,
        db_column='box_identifier'
    )
    description = models.TextField(
        blank=False,
        null=False,
        db_column='box_description'
    )
```

```
    class Meta:
        db_table = 'boxes'

    def __str__(self):
        return _(f'Ящик {self.identifier}')

    def get_absolute_url(self):
        return reverse_lazy('edit-box', kwargs={'pk': self.pk})


class Sensor(models.Model):
    sensor_id = models.AutoField(primary_key=True)
    sensor_number = models.IntegerField(blank=False, null=False)
    sensor_description = models.TextField(blank=False, null=False)

    class Meta:
        db_table = 'sensors'

    def __str__(self):
        return _(f'Сенсор {self.sensor_number} ящику {self.set.box.identifier}')

    def get_absolute_url(self):
        return reverse_lazy('edit-sensor', kwargs={'pk': self.pk})


class BoxSensorSet(models.Model):
    _SENSOR_STARTING_NUMBER = 1
    boxes_set_id = models.AutoField(primary_key=True)
    box = models.ForeignKey(
        Box,
        models.CASCADE,
        null=False,
        blank=False,
        related_name='sensor_sets'
    )
    sensor = models.OneToOneField(
        Sensor,
        models.CASCADE,
        null=False,
        related_name='set'
    )
    facility = models.ForeignKey(
        Facility,
        models.CASCADE,
        null=False,
        blank=False,
        related_name='box_sensors_set'
    )
    line_name = models.CharField(
        max_length=1000,
        blank=False,
        null=False
    )
    date_set_in = models.DateTimeField(
        auto_now_add=True,
        null=False
    )
    sensor_number_in_set = models.IntegerField(
        null=False,
        blank=False,
        validators=(
            MinValueValidator(_SENSOR_STARTING_NUMBER),
```

```
            MaxValueValidator(SENSOR_COUNT_PER_BOX)
        ),
        db_column='sensor_number'
    )

    class Meta:
        db_table = 'boxes_sets'

    def __str__(self):
        return _(f'{self.sensor} is {self.box}')

    def get_absolute_url(self):
        return reverse_lazy('edit-box-sensor-set', kwargs={'pk': self.pk})
```

Файл: energy/__init__.py

Файл: energy/apps.py

```
from django.apps import AppConfig


class EnergyConfig(AppConfig):
    default_auto_field = 'django.db.models.BigAutoField'
    name = 'energy'
```

Файл: energy/urls.py

```
from django.urls import path

from energy import views
from energy.models import Box, BoxSensorSet, Sensor
from utils.common import redirect_to_object_pk_in_post

urlpatterns = [
    path(
        'aggregated-consumption/',
        views.ConsumptionPageView.as_view(),
        name='aggregated-consumption'
    ),
    path(
        'get-facilities-list-for-role/',
        views.get_facilities_choices_for_role,
        name='get-facilities-list-for-role'
    ),
    path(
        'get-aggregated-consumption-for-facility/',
        views.get_consumption_with_total_consumption,
        name='get-aggregated-consumption-for-facility'
    ),
    path(
        'create-box-with-sensors/',
        views.BoxWithSensorsCreateView.as_view(),
        name='create-box-with-sensors'
    ),
    path(
        'box-list',
        views.BoxListView.as_view(),
        name='box-list',
    ),
```

```
    path(
        'edit-box/<hashed_int:pk>',
        views.BoxEditDeleteView.as_view(),
        name='edit-box'
    ),
    path(
        'sensor-list',
        views.SensorListView.as_view(),
        name='sensor-list'
    ),
    path(
        'edit-sensor/<hashed_int:pk>',
        views.SensorEditDeleteView.as_view(),
        name='edit-sensor'
    ),
    path(
        'box-sensor-set-list',
        views.BoxSensorSetListView.as_view(),
        name='box-sensor-set-list'
    ),
    path(
        'edit-box-sensor-set/<hashed_int:pk>',
        views.BoxSensorSetEditDeleteView.as_view(),
        name='edit-box-sensor-set'
    )
]

ajax_url = [
    path(
        'edit-sensor-pk-in-post',
        redirect_to_object_pk_in_post(Sensor, 'edit-sensor'),
        name='edit-sensor-pk-in-post',
    ),
    path(
        'edit-box-pk-in-post',
        redirect_to_object_pk_in_post(Box, 'edit-box'),
        name='edit-box-pk-in-post'
    ),
    path(
        'edit-box-sensor-set-pk-in-post',
        redirect_to_object_pk_in_post(BoxSensorSet, 'edit-box-sensor-set'),
        name='edit-box-sensor-set-pk-in-post'
    ),
    path(
        'run-aggregation/',
        views.run_aggregation,
        name='run-aggregation'
    ),
    path(
        'get-aggregation-status-and-last-time-run/',
        views.get_aggregation_status_and_last_time_run,
        name='get-aggregation-status-and-last-time-run'
    )
]

urlpatterns += ajax_url
```

Файл: energy/migrations/0005_rename_box_number_box_box_identifier_and_more.py

```
# Generated by Django 4.1.3 on 2022-12-19 11:13

import django.core.validators
```

```
import django.db.models.deletion
from django.db import migrations, models


class Migration(migrations.Migration):
    dependencies = [
        ('energy', '0004_alter_boxsensorsset_date_set_in_and_more'),
    ]

    operations = [
        migrations.RenameField(
            model_name='box',
            old_name='box_number',
            new_name='box_identifier',
        ),
        migrations.AlterField(
            model_name='boxsensorsset',
            name='sensor',
            field=models.OneToOneField(
                on_delete=django.db.models.deletion.CASCADE,   related_name='+',
to='energy.sensor'
                ),
        ),
        migrations.AlterField(
            model_name='boxsensorsset',
            name='sensor_number',
            field=models.IntegerField(
                validators=[django.core.validators.MinValueValidator(1),
                            django.core.validators.MaxValueValidator(16)]
                ),
        ),
    ]
```

Файл: energy/migrations/0011_alter_boxsensorset_sensor.py

```
# Generated by Django 4.1.5 on 2023-02-13 08:50

import django.db.models.deletion
from django.db import migrations, models


class Migration(migrations.Migration):
    dependencies = [
        ('energy', '0010_alter_box_box_description_and_more'),
    ]

    operations = [
        migrations.AlterField(
            model_name='boxsensorset',
            name='sensor',
            field=models.OneToOneField(
                on_delete=django.db.models.deletion.CASCADE, related_name='set',
to='energy.sensor'
                ),
        ),
    ]
```

Файл: energy/migrations/0003_alter_boxsensorsset_facility.py

```
# Generated by Django 4.1.1 on 2022-10-01 12:35
```

```
from django.db import migrations, models
import django.db.models.deletion


class Migration(migrations.Migration):

    dependencies = [
        ('institutions', '0003_rename_facilities_facility'),
        ('energy', '0002_alter_boxsensorsset_facility'),
    ]

    operations = [
        migrations.AlterField(
            model_name='boxsensorsset',
            name='facility',
            field=models.ForeignKey(blank=True,                          null=True,
on_delete=django.db.models.deletion.CASCADE, to='institutions.facility'),
        ),
    ]
```

Файл: energy/migrations/__init__.py

Файл: energy/migrations/0009_rename_boxsensorsset_boxsensorset.py

```
# Generated by Django 4.1.5 on 2023-01-23 10:46

from django.db import migrations


class Migration(migrations.Migration):
    dependencies = [
        ('institutions', '0003_rename_facilities_facility'),
        ('energy', '0008_alter_boxsensorsset_sensor_number_in_set'),
    ]

    operations = [
        migrations.RenameModel(
            old_name='BoxSensorsSet',
            new_name='BoxSensorSet',
        ),
    ]
```

Файл: energy/migrations/0008_alter_boxsensorsset_sensor_number_in_set.py

```
# Generated by Django 4.1.3 on 2023-01-16 13:08

import django.core.validators
from django.db import migrations, models


class Migration(migrations.Migration):
    dependencies = [
        ('energy', '0007_alter_boxsensorsset_box_and_more'),
    ]

    operations = [
        migrations.AlterField(
            model_name='boxsensorsset',
            name='sensor_number_in_set',
```

```
            field=models.IntegerField(db_column='sensor_number',

validators=[django.core.validators.MinValueValidator(1),

django.core.validators.MaxValueValidator(16)]
                                      ),
        ),
    ]
```

Файл: energy/migrations/0002_alter_boxsensorsset_facility.py

```
# Generated by Django 4.1.1 on 2022-10-01 12:18

from django.db import migrations, models
import django.db.models.deletion


class Migration(migrations.Migration):

    dependencies = [
        ('institutions', '0003_rename_facilities_facility'),
        ('energy', '0001_initial'),
    ]

    operations = [
        migrations.AlterField(
            model_name='boxsensorsset',
            name='facility',
            field=models.OneToOneField(blank=True,                    null=True,
on_delete=django.db.models.deletion.CASCADE, to='institutions.facility'),
        ),
    ]
```

Файл: energy/migrations/0012_alter_boxsensorset_box.py

```
# Generated by Django 4.1.5 on 2023-02-14 12:40

import django.db.models.deletion
from django.db import migrations, models


class Migration(migrations.Migration):
    dependencies = [
        ('energy', '0011_alter_boxsensorset_sensor'),
    ]

    operations = [
        migrations.AlterField(
            model_name='boxsensorset',
            name='box',
            field=models.ForeignKey(
                on_delete=django.db.models.deletion.CASCADE,
related_name='sensor_sets',
                to='energy.box'
                ),
        ),
    ]
```

Файл: energy/migrations/0010_alter_box_box_description_and_more.py

```
# Generated by Django 4.1.5 on 2023-01-23 11:07

from django.db import migrations, models


class Migration(migrations.Migration):
    dependencies = [
        ('energy', '0009_rename_boxsensorsset_boxsensorset'),
    ]

    operations = [
        migrations.AlterField(
            model_name='box',
            name='box_description',
            field=models.TextField(db_column='box_description'),
        ),
        migrations.RenameField(
            model_name='box',
            old_name='box_description',
            new_name='description',
        ),
        migrations.AlterField(
            model_name='box',
            name='box_identifier',
            field=models.CharField(db_column='box_identifier', max_length=256),
        ),
        migrations.RenameField(
            model_name='box',
            old_name='box_identifier',
            new_name='identifier',
        ),
    ]
```

Файл: energy/migrations/0006_alter_boxsensorsset_sensor_number_and_more.py

```
# Generated by Django 4.1.3 on 2022-12-22 10:10

import django.core.validators
from django.db import migrations, models


class Migration(migrations.Migration):
    dependencies = [
        ('energy', '0005_rename_box_number_box_box_identifier_and_more'),
    ]

    operations = [
        migrations.AlterField(
            model_name='boxsensorsset',
            name='sensor_number',
            field=models.IntegerField(db_column='sensor_number',

validators=[django.core.validators.MinValueValidator(1),

django.core.validators.MaxValueValidator(16)]
                                      ),
        ),
        migrations.RenameField(
            model_name='boxsensorsset',
            old_name='sensor_number',
            new_name='sensor_number_in_set',
        ),
```

```
    ]
```

Файл: energy/migrations/0001_initial.py

```
# Generated by Django 4.1.1 on 2022-10-01 12:17

from django.db import migrations, models
import django.db.models.deletion


class Migration(migrations.Migration):

    initial = True

    dependencies = [
        ('institutions', '0003_rename_facilities_facility'),
    ]

    operations = [
        migrations.CreateModel(
            name='Box',
            fields=[
                ('box_id', models.AutoField(primary_key=True, serialize=False)),
                ('box_number', models.CharField(max_length=256)),
                ('box_description', models.TextField()),
            ],
            options={
                'db_table': 'boxes',
            },
        ),
        migrations.CreateModel(
            name='Sensor',
            fields=[
                ('sensor_id',                 models.AutoField(primary_key=True,
serialize=False)),
                ('sensor_number', models.IntegerField()),
                ('sensor_description', models.TextField()),
            ],
            options={
                'db_table': 'sensors',
            },
        ),
        migrations.CreateModel(
            name='BoxSensorsSet',
            fields=[
                ('boxes_set_id',              models.AutoField(primary_key=True,
serialize=False)),
                ('line_name', models.CharField(max_length=1000)),
                ('date_set_in', models.DateTimeField(blank=True, null=True)),
                ('sensor_number', models.IntegerField()),
                ('box',
models.ForeignKey(on_delete=django.db.models.deletion.CASCADE,
to='energy.box')),
                ('facility',     models.OneToOneField(blank=True,     null=True,
on_delete=django.db.models.deletion.DO_NOTHING, to='institutions.facility')),
                ('sensor',
models.OneToOneField(on_delete=django.db.models.deletion.RESTRICT,
to='energy.sensor')),
            ],
            options={
                'db_table': 'boxes_sets',
            },
```

```
        ),
    ]
```

Файл: energy/migrations/0004_alter_boxsensorsset_date_set_in_and_more.py

```
# Generated by Django 4.1.1 on 2022-10-02 08:47

from django.db import migrations, models
import django.db.models.deletion


class Migration(migrations.Migration):

    dependencies = [
        ('institutions', '0003_rename_facilities_facility'),
        ('energy', '0003_alter_boxsensorsset_facility'),
    ]

    operations = [
        migrations.AlterField(
            model_name='boxsensorsset',
            name='date_set_in',
            field=models.DateTimeField(auto_now_add=True),
        ),
        migrations.AlterField(
            model_name='boxsensorsset',
            name='facility',
            field=models.ForeignKey(on_delete=django.db.models.deletion.CASCADE,
related_name='box_sensors_set', to='institutions.facility'),
        ),
        migrations.AlterField(
            model_name='boxsensorsset',
            name='sensor',

field=models.OneToOneField(on_delete=django.db.models.deletion.RESTRICT,
related_name='+', to='energy.sensor'),
        ),
    ]
```

Файл: energy/migrations/0007_alter_boxsensorsset_box_and_more.py

```
# Generated by Django 4.1.3 on 2023-01-16 12:53

import django.core.validators
import django.db.models.deletion
from django.db import migrations, models


class Migration(migrations.Migration):
    dependencies = [
        ('energy', '0006_alter_boxsensorsset_sensor_number_and_more'),
    ]

    operations = [
        migrations.AlterField(
            model_name='boxsensorsset',
            name='box',
            field=models.ForeignKey(
                on_delete=django.db.models.deletion.CASCADE,
related_name='box_sensor_sets',
                to='energy.box'
```

```
                ),
        ),
        migrations.AlterField(
            model_name='boxsensorsset',
            name='sensor_number_in_set',
            field=models.IntegerField(
                db_column='sensor_number',
validators=[django.core.validators.MinValueValidator(0),

django.core.validators.MaxValueValidator(0)]
                ),
        ),
    ]
```

Файл: energy/forms/energy_consumption_display_page_control_form.py

```
from django.forms import BooleanField, ChoiceField, Form
from django.utils.translation import gettext_lazy as _

from energy.logic.aggregated_consumption.models import AggregationIntervalSeconds
from institutions.models import Facility
from users.models import User, UserRole
from utils.forms import CrispyFormMixin, SecureModelChoiceField

AGGREGATION_INTERVAL_CHOICES = (
    (AggregationIntervalSeconds.ONE_HOUR.value, _('Година')),
    (AggregationIntervalSeconds.ONE_DAY.value, _('День')),
    (AggregationIntervalSeconds.ONE_WEEK.value, _('Тиждень')),
    (AggregationIntervalSeconds.ONE_MONTH.value, _('Місяць')),
    (AggregationIntervalSeconds.ONE_YEAR.value, _('Рік')),
)


class EnergyConsumptionDisplayPageControlForm(CrispyFormMixin, Form):
    """must be created for user using corresponding method,
    and some fields are added in the template"""
    # correct qs for role is set in __init__
    role = SecureModelChoiceField(
        label=_('Роль'),
        queryset=UserRole.objects.all(),
        empty_label=None,
    )
    # should be populated via ajax depending on chosen role,
    # qs is not as facility validation is done when querying for facilities list
for role
    # field is used just for label and select widget
    facility_to_get_consumption_for = SecureModelChoiceField(
        label=_("Об'єкт"),
        queryset=Facility.objects.none(),
    )
    aggregation_interval_seconds = ChoiceField(
        label=_('Інтервал агрегації'),
        choices=AGGREGATION_INTERVAL_CHOICES,
    )
    include_forecast = BooleanField(
        label=_('Прогноз споживання')
    )

    def __init__(self, user: User, *args, **kwargs):
        super().__init__(*args, **kwargs)
        self.__set_correct_roles_choices(user)
```

```
    def __set_correct_roles_choices(self, user: User):
        self.fields['role'].queryset = user.roles


Файл: energy/forms/__init__.py

from .box_with_sensors_creation import (
    BoxFormNoRelationFields,          BoxSensorSetForBoxWithSensorsCreationForm,
BoxSensorSetFormset,
    ChooseInstitutionForm,
    SensorFormNoRelationFields, SensorsFormset,
)
from             .energy_consumption_display_page_control_form             import
EnergyConsumptionDisplayPageControlForm
from .simple import (BoxForm, BoxSensorSetForm, SensorForm)


Файл: energy/forms/box_with_sensors_creation.py

from django.forms import BaseFormSet, CharField, Form, formset_factory, Textarea,
TextInput
from django.utils.translation import gettext_lazy as _

from energokodros.settings import SENSOR_COUNT_PER_BOX
from energy.models import Box, BoxSensorSet, Sensor
from institutions.models import Facility
from utils.forms import (
    CrispyFormMixin, CrispyModelForm, SecureModelChoiceField,
)


class SensorFormNoRelationFields(CrispyModelForm):
    sensor_description = CharField(
        label=_('Опис'),
        required=True,
        max_length=255,
    )

    class Meta:
        model = Sensor
        fields = ('sensor_number', 'sensor_description')
        labels = {
            'sensor_number': _('Власний номер'),
        }


class BoxFormNoRelationFields(CrispyModelForm):
    class Meta:
        model = Box
        fields = ('identifier', 'description')
        labels = {
            'identifier':  _('Ідентифікатор'),
            'description': _('Опис')
        }
        widgets = {
            'description': Textarea({'rows': '4'})
        }


__SensorsFormsetBase = formset_factory(
    SensorFormNoRelationFields,
    # setting extra to zero as actual forms count will depend on data or initial
    extra=0,
```

```
    min_num=SENSOR_COUNT_PER_BOX,
    max_num=SENSOR_COUNT_PER_BOX,
)


class BoxSensorSetForBoxWithSensorsCreationForm(CrispyModelForm):
    # sensor number must be prepopulated from created sensors
    sensor_number = CharField(
        label=_('Номер сенсора'),
        required=False,
        widget=TextInput(attrs={'readonly': 'readonly'})
    )
    facility = SecureModelChoiceField(
        label=_("Об'єкт сенсора"),
        # qs set to all to accept any facility,
        # correct choices must be set via ajax
        queryset=Facility.objects.all(),
        required=True,
        empty_label=None
    )

    class Meta:
        model = BoxSensorSet
        fields = ('line_name', 'sensor_number_in_set')
        labels = {
            'line_name':            _('Назва лінії'),
            'sensor_number_in_set': _('Номер сенсора наборі'),
        }
        fields_order = ('sensor_number',) + fields + ('facility',)


__BoxSensorSetFormset = formset_factory(
    BoxSensorSetForBoxWithSensorsCreationForm,
    # setting extra to zero as actual forms count will depend on data or initial
    extra=0,
    min_num=SENSOR_COUNT_PER_BOX,
    max_num=SENSOR_COUNT_PER_BOX,
)


class __MustBeUsedWithInitialOrData:
    def __init__(self: BaseFormSet, *args, **kwargs):
        super().__init__(*args, **kwargs)
        if not any((self.initial, self.data)):
            raise ValueError('must be used either with data or initial')


class SensorsFormset(__MustBeUsedWithInitialOrData, __SensorsFormsetBase):
    pass


class BoxSensorSetFormset(__MustBeUsedWithInitialOrData, __BoxSensorSetFormset):
    pass


class ChooseInstitutionForm(CrispyFormMixin, Form):
    institution = SecureModelChoiceField(
        queryset=Facility.objects.get_institutions(),
        required=False,
        label=_("Оберіть заклад якому буде належати ящик"),
        empty_label=None,
    )
```

Файл: energy/forms/simple.py

```
from django.db.models import Model
from django.utils.translation import gettext_lazy as _

from energy.forms.box_with_sensors_creation import (
    BoxFormNoRelationFields,
    BoxSensorSetForBoxWithSensorsCreationForm, SensorFormNoRelationFields,
)
from energy.models import Box, BoxSensorSet, Sensor
from institutions.logic import (
    get_all_facilities_of_facility_institution,
get_facility_name_space_padded_according_to_nesting,
)
from institutions.models import Facility
from utils.common.object_to_queryset import object_to_queryset
from utils.forms import (
    SecureModelChoiceField, SelectWithFormControlClass,
    UPDATE_DELETE_BUTTONS_SET,
)
from utils.views import AdditionalSetupRequiredFormMixin


class SensorForm(SensorFormNoRelationFields, AdditionalSetupRequiredFormMixin):
    box = SecureModelChoiceField(
        queryset=Box.objects.none(),
        label=_('Ящик'),
        empty_label=None,
        disabled=True
    )
    set = SecureModelChoiceField(
        queryset=BoxSensorSet.objects.none(),
        label=_('Набір'),
        empty_label=None,
        disabled=True
    )

    class Meta(SensorFormNoRelationFields.Meta):
        fields_order = SensorFormNoRelationFields.Meta.fields + ('box', 'set')
        buttons = UPDATE_DELETE_BUTTONS_SET

    def additionally_setup(self, obj: Model):
        # noinspection PyTypeChecker
        sensor: Sensor = obj
        self.fields['box'].queryset = object_to_queryset(sensor.set.box)
        self.fields['set'].queryset = object_to_queryset(sensor.set)


class BoxForm(BoxFormNoRelationFields, AdditionalSetupRequiredFormMixin):
    sensor_sets = SecureModelChoiceField(
        queryset=BoxSensorSet.objects.none(),
        disabled=True,
        label=_('Набори'),
        label_from_instance_function=lambda           _set:           _(f'Набір
{_set.sensor_number_in_set}'),
        widget=SelectWithFormControlClass(attrs={'size': 10}),
        empty_label=None
    )

    class Meta(BoxFormNoRelationFields.Meta):
        buttons = UPDATE_DELETE_BUTTONS_SET
```

```
    def additionally_setup(self, obj: Model):
        # noinspection PyTypeChecker
        box: Box = obj
        self.fields['sensor_sets'].queryset = box.sensor_sets


class                BoxSensorSetForm(BoxSensorSetForBoxWithSensorsCreationForm,
AdditionalSetupRequiredFormMixin):
    # sensor number is model choice field in order to have sensor pk in form
    # so user can be easily redirected to sensor page
    sensor_number = SecureModelChoiceField(
        label=_('Номер сенсора'),
        # correct sensor must be set in additionally_setup
        queryset=Sensor.objects.all(),
        label_from_instance_function=lambda x: x.sensor_number,
        empty_label=None,
        disabled=True,
        required=False
    )
    facility = SecureModelChoiceField(
        label=_("Об'єкт сенсора"),
        # qs set to all to accept any facility,
        # correct choices must be set in additionally_setup
        queryset=Facility.objects.all(),
        empty_label=None,

label_from_instance_function=get_facility_name_space_padded_according_to_nesting
    )

    class Meta(BoxSensorSetForBoxWithSensorsCreationForm.Meta):
        buttons = UPDATE_DELETE_BUTTONS_SET
        fields_order   =   ('line_name',   'sensor_number_in_set',   'facility',
'sensor_number')

    def additionally_setup(self, obj: Model):
        # noinspection PyTypeChecker
        box_sensor_set: BoxSensorSet = obj
        self.fields['sensor_number'].queryset                                  =
object_to_queryset(box_sensor_set.sensor)
        self.fields['facility'].queryset = \
            get_all_facilities_of_facility_institution(box_sensor_set.facility)
        self.fields['facility'].initial = box_sensor_set.facility


Файл: energy/tests/__init__.py



Файл: energy/tests/factories.py

from dataclasses import dataclass, field

import factory
from django.utils.decorators import classonlymethod
from factory import django

from energy.models import Box, BoxSensorSet, Sensor
from institutions.models import Facility
from institutions.tests.factories import InstitutionFactory


class BoxFactory(django.DjangoModelFactory):
    identifier = factory.Sequence(lambda n: f'{n}')
```

```
    description = factory.Sequence(lambda n: f'box number {n} description')

    class Meta:
        model = Box


class SensorFactory(django.DjangoModelFactory):
    sensor_number = factory.Sequence(lambda n: f'{n}')
    sensor_description = factory.Sequence(lambda n: f'sensor number {n}
description')

    class Meta:
        model = Sensor


@dataclass
class BoxSensorsSetData:
    facility: Facility
    box: Box
    sensors: list[Sensor] = field(default_factory=list)


class BoxSensorsSetManualFactory:
    """this is not a DjangoModelFactory, due to application logic"""
    __SENSORS_PER_BOX = 16

    @classonlymethod
    def generate(cls, facility: Facility = None,
                 initial_number_for_subfactories_sequences: int = 1) ->
BoxSensorsSetData:
        box = BoxFactory(__sequence=initial_number_for_subfactories_sequences)
        if not facility:
            facility =
InstitutionFactory(__sequence=initial_number_for_subfactories_sequences)
        data = BoxSensorsSetData(facility, box)
        for i in range(cls.__SENSORS_PER_BOX):
            sensor = SensorFactory()
            BoxSensorSet(
                box=box,
                sensor=sensor,
                facility=facility,
                line_name=f'line number {i + 1}',
                sensor_number=i + 1
            ).save()
            data.sensors.append(sensor)
        return data
```

Файл: energy/logic/box_identifier_validation.py

```
from enum import Enum

REGEX_GROUPS_SEPARATOR = '-'


class CharactersCountForBoxIdentifierParts:
    CHARACTERS_PER_BOX_ORDINAL_NUMBER_FOR_BOX_IDENTIFIER = 2
    CHARACTERS_PER_CITY_FOR_BOX_IDENTIFIER = 2
    CHARACTERS_PER_INSTITUTION_FOR_BOX_IDENTIFIER = 2
    CHARACTERS_PER_INSTITUTION_NUMBER_FOR_BOX_IDENTIFIER = 2
    CHARACTERS_PER_SENSORS_COUNT_FOR_BOX_IDENTIFIER = 2
```

```
class BoxIdentifierRegexGroups(str, Enum):
    CITY = 'city'
    INSTITUTION = 'institution'
    INSTITUTION_NUMBER = 'institution_number'
    BOX_ORDINAL_NUMBER = 'box_ordinal_number'
    SENSORS_COUNT = 'sensors_count'


def create_regex_for_letter_count(group_name: str, count: int) -> str:
    return f'(?P<{group_name}>[A-Za-z]{{{count}}})'


def create_regex_for_number_count(group_name: str, count: int) -> str:
    return f'(?P<{group_name}>[0-9]{{{count}}})'


def compose_regex() -> str:
    groups = BoxIdentifierRegexGroups
    _ = CharactersCountForBoxIdentifierParts
    city = create_regex_for_letter_count(
        groups.CITY,
        _.CHARACTERS_PER_CITY_FOR_BOX_IDENTIFIER
    )
    institution = create_regex_for_letter_count(
        groups.INSTITUTION,
        _.CHARACTERS_PER_INSTITUTION_FOR_BOX_IDENTIFIER
    )
    institution_number = create_regex_for_number_count(
        groups.INSTITUTION_NUMBER,
        _.CHARACTERS_PER_INSTITUTION_NUMBER_FOR_BOX_IDENTIFIER
    )
    box_ordinal_number = create_regex_for_number_count(
        groups.BOX_ORDINAL_NUMBER,
        _.CHARACTERS_PER_BOX_ORDINAL_NUMBER_FOR_BOX_IDENTIFIER
    )
    sensors_count = create_regex_for_number_count(
        groups.SENSORS_COUNT,
        _.CHARACTERS_PER_SENSORS_COUNT_FOR_BOX_IDENTIFIER
    )
    return REGEX_GROUPS_SEPARATOR.join(
        (
            city,    institution,    institution_number,    box_ordinal_number,
sensors_count
        )
    )
```

Файл: energy/logic/__init__.py

```
from .ajax import get_facilities_formatted_choices_for_user_role
```

Файл: energy/logic/ajax.py

```
from institutions.logic import \
    make_facility_and_descendants_choices_with_padding_according_to_nesting
from users.models import UserRole
from utils.common import Choices


def get_facilities_formatted_choices_for_user_role(role: UserRole) -> Choices:
    return
make_facility_and_descendants_choices_with_padding_according_to_nesting(
```

```
        role.facility_has_access_to
        )


Файл: energy/logic/aggregated_consumption/parameters_parsers.py

import time
from dataclasses import fields
from datetime import date, datetime
from typing import Any, Callable, Iterable, Type, TypeAlias

from energy.logic.aggregated_consumption.exceptions import (
    AggregationIntervalDoesNotFitPeriod,                    FutureFilteringDate,
InvalidHourFilteringMethod,
    PeriodStartGreaterThanEnd,
    QueryParametersInvalid, StartHourGreaterThanEndHour,
)
from energy.logic.aggregated_consumption.models import AggregationIntervalSeconds
from energy.logic.aggregated_consumption.parameters import (
    AnyQueryParameters, CommonQueryParameters,
    EnergyConsumptionQueryRawParameters,
OneHourAggregationIntervalQueryParameters,
)
from energy.logic.aggregated_consumption.simple import \
    parse_str_parameter_to_int_with_correct_exception
from institutions.models import Facility
from utils.common import get_object_by_hashed_id_or_404

AnyQueryParametersParser: TypeAlias = '_CommonQueryParametersParser'


class ParameterParser:
    def __init__(self, raw_parameters: dict[str, str]):
        self.__raw_parameters                                                =
EnergyConsumptionQueryRawParameters(**raw_parameters)

    def get_parameters(self) -> AnyQueryParameters:
        aggregation_interval = self.__extract_aggregation_interval()
        parser                                                               =
self.__get_parser_for_aggregation_interval(aggregation_interval)
        _ = self.__raw_parameters
        return parser(
            aggregation_interval=aggregation_interval,

facility_pk_to_get_consumption_for_or_all_descendants_if_any=_.pop('facility_pk'
),
            period_end_epoch_seconds=_.pop('period_end_epoch_seconds'),
            period_start_epoch_seconds=_.pop('period_start_epoch_seconds'),
            **_
        ).get_parameters()

    def __extract_aggregation_interval(self) -> AggregationIntervalSeconds:
        aggregation_interval_seconds                                         =
parse_str_parameter_to_int_with_correct_exception(
            self.__raw_parameters.pop('aggregation_interval_seconds')
        )
        return AggregationIntervalSeconds(aggregation_interval_seconds)

    def __get_parser_for_aggregation_interval(
            self, aggregation_interval: AggregationIntervalSeconds) \
            -> Type[AnyQueryParametersParser] | Callable:
        return
_AGGREGATION_INTERVAL_TO_PARAMETERS_PARSER_MAPPING[aggregation_interval]
```

```
class _CommonQueryParametersParser:
    def __init__(
            self, *,
            aggregation_interval: AggregationIntervalSeconds,
            facility_pk_to_get_consumption_for_or_all_descendants_if_any: str,
            period_start_epoch_seconds: str,
            period_end_epoch_seconds: str, **kwargs
    ):
        self.__facility_to_get_consumption_for_or_all_descendants_if_any = \

self.__parse_facility(facility_pk_to_get_consumption_for_or_all_descendants_if_a
ny)
        self.__aggregation_interval = aggregation_interval
        self.__set_period_boundaries(period_start_epoch_seconds,
period_end_epoch_seconds)
        self._validate()

    def get_parameters(self) -> CommonQueryParameters:
        return CommonQueryParameters(
            facility_to_get_consumption_for_or_all_descendants_if_any=
            self.__facility_to_get_consumption_for_or_all_descendants_if_any,
            aggregation_interval=self.__aggregation_interval,
            period_start=self._period_start,
            period_end=self._period_end
        )

    def _validate(self):
        self.__check_period_is_valid()
        self._check_period_contains_at_least_one_aggregation_interval()

    def __set_period_boundaries(self, period_start_epoch_seconds: str,
                                period_end_epoch_seconds: str):
        _ = parse_str_parameter_to_int_with_correct_exception
        period_start_epoch_seconds = _(period_start_epoch_seconds)
        period_end_epoch_seconds = _(period_end_epoch_seconds)
        self._period_start                                                    =
self.__convert_epoch_seconds_to_date(period_start_epoch_seconds)
        self._period_end                                                      =
self.__convert_epoch_seconds_to_date(period_end_epoch_seconds)

    def __check_period_is_valid(self):
        if self._period_start > self._period_end:
            raise PeriodStartGreaterThanEnd

    def _check_period_contains_at_least_one_aggregation_interval(self):
        period_length_seconds          =          int((self._period_end          -
self._period_start).total_seconds())
        if period_length_seconds < self.__aggregation_interval:
            raise AggregationIntervalDoesNotFitPeriod

    def __convert_epoch_seconds_to_date(self, epoch_seconds: int) -> date:
        if epoch_seconds < 0:
            raise QueryParametersInvalid
        if int(time.time()) < epoch_seconds:
            # future timestamp (senseless filtering for future dates)
            raise FutureFilteringDate
        return datetime.fromtimestamp(epoch_seconds).date()

    def __parse_facility(self, facility_pk: str) -> Facility:
        # noinspection PyTypeChecker
        return get_object_by_hashed_id_or_404(Facility, facility_pk)
```

```
class __AllowQueryingForCurrentDayParser(_CommonQueryParametersParser):
    def _check_period_contains_at_least_one_aggregation_interval(
            self: _CommonQueryParametersParser):
        if self._period_start == self._period_end:
            return
        super()._check_period_contains_at_least_one_aggregation_interval()


class
_OneHourAggregationIntervalQueryParametersParser(__AllowQueryingForCurrentDayPar
ser):
    """one hour aggregation interval has additional parameters"""
    __HOURS: Iterable[int] = range(24)

    def __init__(
            self, *, hours_filtering_start_hour: str = None,
            hours_filtering_end_hour: str = None, hour_filtering_method: str =
None,
            **kwargs
    ):

self.__check_aggregation_interval_is_on_hour(kwargs.get('aggregation_interval'))
        self.__hours_filtering_start_hour = hours_filtering_start_hour
        self.__hours_filtering_end_hour = hours_filtering_end_hour
        self.__hour_filtering_method = hour_filtering_method
        if self.__check_hour_filtering_options_are_set():
            self.__set_hours_filtering_range(hours_filtering_start_hour,
hours_filtering_end_hour)
        super().__init__(**kwargs)
        self._validate()

    def get_parameters(self) -> OneHourAggregationIntervalQueryParameters:
        _ = super().get_parameters()
        base_kwargs = {field.name: getattr(_, field.name) for field in
fields(_.__class__)}
        return OneHourAggregationIntervalQueryParameters(
            **base_kwargs,
            hours_filtering_start_hour=self.__hours_filtering_start_hour,
            hours_filtering_end_hour=self.__hours_filtering_end_hour,
            hours_filtering_method=self.__hour_filtering_method
        )

    def _validate(self):
        super()._validate()
        if self.__check_hour_filtering_options_are_set():
            self.__check_hours_filtering_range_is_correct()

    def __check_aggregation_interval_is_on_hour(self, aggregation_interval: Any):
        if aggregation_interval is not AggregationIntervalSeconds.ONE_HOUR:
            raise ValueError('this class should work only with one hour
aggregation interval')

    def __check_hours_filtering_range_is_correct(self):
        if self.__hours_filtering_start_hour not in self.__HOURS:
            raise QueryParametersInvalid
        if self.__hours_filtering_end_hour not in self.__HOURS:
            raise QueryParametersInvalid
        if self.__hour_filtering_method == \

OneHourAggregationIntervalQueryParameters.HourFilteringMethods.EVERY_DAY:
```

```
            if                self.__hours_filtering_start_hour                >
self.__hours_filtering_end_hour:
                raise StartHourGreaterThanEndHour

    def __set_hours_filtering_range(self, hours_filtering_start_hour: str,
                                    hours_filtering_end_hour: str):
        self.__hours_filtering_start_hour                                        =
parse_str_parameter_to_int_with_correct_exception(
            hours_filtering_start_hour
        )
        self.__hours_filtering_end_hour                                          =
parse_str_parameter_to_int_with_correct_exception(
            hours_filtering_end_hour
        )

    def __set_hour_filtering_method(self):
        try:
            self.__hour_filtering_method = \
                OneHourAggregationIntervalQueryParameters.HourFilteringMethods(
                    self.__hour_filtering_method
                )
        except ValueError as e:
            raise InvalidHourFilteringMethod from e

    def __check_hour_filtering_options_are_set(self):
        return self.__hours_filtering_start_hour is not None \
            and self.__hours_filtering_end_hour is not None \
            and self.__hour_filtering_method is not None


class
_OneDayAggregationIntervalQueryParametersParser(__AllowQueryingForCurrentDayPars
er):
    pass


_AGGREGATION_INTERVAL_TO_PARAMETERS_PARSER_MAPPING: \
    dict[
        AggregationIntervalSeconds,
        Type[AnyQueryParametersParser]] = \
    {
        AggregationIntervalSeconds.ONE_HOUR:
_OneHourAggregationIntervalQueryParametersParser,
        AggregationIntervalSeconds.ONE_DAY:
_OneDayAggregationIntervalQueryParametersParser,
        AggregationIntervalSeconds.ONE_WEEK:  _CommonQueryParametersParser,
        AggregationIntervalSeconds.ONE_MONTH: _CommonQueryParametersParser,
        AggregationIntervalSeconds.ONE_YEAR:  _CommonQueryParametersParser
    }
```

Файл: energy/logic/aggregated_consumption/forecast.py

```
import os
from datetime import datetime
from typing import Callable

from            energy.logic.aggregated_consumption.exceptions            import
ForecastForParametersDoesNotExist
from energy.logic.aggregated_consumption.formatters import (
    format_forecast, RawAggregatedDataFormatter,
)
from energy.logic.aggregated_consumption.models import AggregationIntervalSeconds
```

```
from energy.logic.aggregated_consumption.parameters import AnyQueryParameters
from energy.logic.aggregated_consumption.types import (
    ConsumptionForecast, ConsumptionWithConsumptionForecast, RawConsumption,
    RawConsumptionForecast, RawConsumptionTime,
)
from institutions.models import Facility


class ConsumptionForecaster:
    """
    currently this is just a stub returning hardcoded data,
    but will be a forecasting AI
    """
    format_forecast: Callable[[RawConsumptionForecast], ConsumptionForecast] = \
        staticmethod(format_forecast)

    def __init__(
            self, parameters: AnyQueryParameters, consumption: RawConsumption,
            raw_aggregation_data_formatter: RawAggregatedDataFormatter):
        self.__parameters = parameters
        self.__consumption = consumption
        self.__raw_aggregation_data_formatter = raw_aggregation_data_formatter

    def                  get_consumption_with_forecast(self)                  ->
ConsumptionWithConsumptionForecast:
        _ = self.__raw_aggregation_data_formatter
        return [
            (
                _.format_time(date),
                _.format_consumption(consumption),
                self.format_forecast(self.__get_forecast_for_date(date))
            )
            for date, consumption in self.__consumption
        ]

    def     __get_forecast_for_date(self,     date:     RawConsumptionTime)     ->
RawConsumptionForecast:
        if self.__check_is_kindergarten_and_interval_is_one_hour():
            return self.__get_forecast_for_kindergarten(date)
        raise ForecastForParametersDoesNotExist

    def     __get_forecast_for_kindergarten(self,     date:     datetime)     ->
RawConsumptionForecast:
        day_number = date.weekday()
        hour = date.hour
        forecast_source = self.__get_forecast_source_for_kindergarten(
            self.__check_day_is_working(day_number)
        )
        return forecast_source[hour]

    def __check_is_kindergarten_and_interval_is_one_hour(self) -> bool:
        is_kindergarten = self.__check_facility_is_kindergarten()
        interval_is_one_day = self.__check_aggregation_interval_is_one_hour()
        return is_kindergarten and interval_is_one_day

    def __check_aggregation_interval_is_one_hour(self) -> bool:
        return             self.__parameters.aggregation_interval            ==
AggregationIntervalSeconds.ONE_HOUR

    def __check_facility_is_kindergarten(self) -> bool:
        return
self.__parameters.facility_to_get_consumption_for_or_all_descendants_if_any == \
            Facility.objects.get(pk=os.getenv('KINDERGARTEN_ID'))
```

```
    def __get_forecast_source_for_kindergarten(self, is_day_working: bool) \
            -> dict[int, RawConsumptionForecast]:
        if is_day_working:
            return
KINDERGARTEN_CONSUMPTION_FORECAST_KILOWATT_HOUR_FOR_WORKING_DAY_BY_HOUR
        return
KINDERGARTEN_CONSUMPTION_FORECAST_KILOWATT_HOUR_FOR_WEEKEND_BY_HOUR

    def __check_day_is_working(self, day_number_from_zero: int) -> bool:
        return day_number_from_zero in range(5)


KINDERGARTEN_CONSUMPTION_FORECAST_KILOWATT_HOUR_FOR_WORKING_DAY_BY_HOUR = {
    0:  1.57,
    1:  1.57,
    2:  1.57,
    3:  1.57,
    4:  1.57,
    5:  1.57,
    6:  1.57,
    7:  18.412,
    8:  24.283,
    9:  48.1394,
    10: 47.7198,
    11: 20.7058,
    12: 20.5874,
    13: 21.704,
    14: 21.663,
    15: 18.3154,
    16: 19.4908,
    17: 10.0008,
    18: 10.8984,
    19: 7.2084,
    20: 2.69,
    21: 1.57,
    22: 1.57,
    23: 1.57
}

KINDERGARTEN_CONSUMPTION_FORECAST_KILOWATT_HOUR_FOR_WEEKEND_BY_HOUR = {
    0:  1.57,
    1:  1.57,
    2:  1.57,
    3:  1.57,
    4:  1.57,
    5:  1.57,
    6:  1.57,
    7:  1.682,
    8:  1.682,
    9:  1.682,
    10: 1.682,
    11: 1.682,
    12: 1.682,
    13: 1.682,
    14: 1.682,
    15: 1.682,
    16: 1.682,
    17: 1.682,
    18: 1.682,
    19: 2.802,
    20: 2.690,
    21: 1.57,
```

```
    22: 1.57,
    23: 1.57
}
```

Файл: energy/logic/aggregated_consumption/controller.py

```
from energy.logic.aggregated_consumption.exceptions import QueryParametersInvalid
from energy.logic.aggregated_consumption.forecast import ConsumptionForecaster
from            energy.logic.aggregated_consumption.formatters            import
RawAggregatedDataFormatter
from        energy.logic.aggregated_consumption.parameters_parsers        import
ParameterParser
from             energy.logic.aggregated_consumption.queriers             import
AggregatedConsumptionQuerier
from energy.logic.aggregated_consumption.types import (
    ConsumptionWithConsumptionForecast,
    ConsumptionWIthConsumptionForecastWithTotalConsumption,
    ConsumptionWithTotalConsumption, RawConsumption,
)
from           users.logic           import           check_role_belongs_to_user,
check_role_has_access_for_facility
from users.models import User, UserRole
from utils.common import get_object_by_hashed_id_or_404
from utils.types import StrStrDict


class AggregatedEnergyConsumptionController:

    def __init__(self, user: User, parameters: StrStrDict):
        role = self.__extract_role(parameters)
        self.__include_forecast = self.__extract_include_forecast(parameters)
        self.__parameters = ParameterParser(parameters).get_parameters()
        # noinspection PyTypeChecker
        self.__check_user_is_role_owner_and_role_has_access_to_facility(
            user,
            role
        )

    def get_consumption_with_optional_forecast_and_total_consumption(self) -> \
            ConsumptionWithTotalConsumption \
            | ConsumptionWIthConsumptionForecastWithTotalConsumption \
            | tuple[None, None]:
        querier = self.__create_consumption_querier()
        total_consumption = querier.get_total_consumption()
        if self.__include_forecast:
            raw_consumption = querier.get_raw_consumption()
            if raw_consumption:
                consumption_with_forecast                                    =
self.__get_consumption_with_forecast(
                    raw_consumption,
                    querier.formatter
                )
            else:
                consumption_with_forecast = None
            return consumption_with_forecast, total_consumption
        consumption = querier.get_consumption()
        return consumption, total_consumption

    def __get_consumption_with_forecast(
            self, raw_consumption: RawConsumption,
            raw_consumption_formatter: RawAggregatedDataFormatter
    ) -> ConsumptionWithConsumptionForecast:
```

```
        forecaster          =          self.__create_forecaster(raw_consumption,
raw_consumption_formatter)
        return forecaster.get_consumption_with_forecast()

    def __create_forecaster(
            self, raw_consumption: RawConsumption,
            raw_aggregation_data_formatter:    RawAggregatedDataFormatter)    ->
ConsumptionForecaster:
        return ConsumptionForecaster(
            self.__parameters, raw_consumption, raw_aggregation_data_formatter
        )

    def __create_consumption_querier(self) -> AggregatedConsumptionQuerier:
        return AggregatedConsumptionQuerier(self.__parameters)

    def __extract_role(self, parameters: dict[str, str]) -> UserRole:
        # noinspection PyTypeChecker
        return get_object_by_hashed_id_or_404(
            UserRole,
            parameters.pop('role_pk', None)
        )

    def __check_user_is_role_owner_and_role_has_access_to_facility(
            self, user: User, role: UserRole):
        # noinspection PyTypeChecker
        check_role_belongs_to_user(user, role)
        # noinspection PyTypeChecker
        check_role_has_access_for_facility(
            role,
self.__parameters.facility_to_get_consumption_for_or_all_descendants_if_any
        )

    def __extract_include_forecast(self, parameters: StrStrDict) -> bool:
        include_forecast = parameters.pop('include_forecast', None)
        if include_forecast == 'true':
            return True
        if include_forecast == 'false':
            return False
        raise QueryParametersInvalid('include_forecast must be true or false')
```

Файл: energy/logic/aggregated_consumption/formatters.py

```
import calendar
from datetime import datetime

from django.utils.translation import gettext as _

from energy.logic.aggregated_consumption.types import (
    ConsumptionForecast,           ConsumptionTime,           ConsumptionValue,
RawConsumptionForecast,
    RawConsumptionTime,         RawConsumptionValue,         RawTotalConsumption,
TotalConsumption,
)


class RawAggregatedDataFormatter:
    @staticmethod
    def format_time(time: RawConsumptionTime) -> ConsumptionTime:
        return time

    @staticmethod
    def format_consumption(consumption: RawConsumptionValue) -> ConsumptionValue:
```

```
        return f'{consumption:.10f}'

    @staticmethod
    def format_total_consumption(total_consumption: RawTotalConsumption) \
            -> TotalConsumption:
        return f'{total_consumption:.10f}'


class OneHourFormatter(RawAggregatedDataFormatter):
    @staticmethod
    def format_time(time: datetime) -> ConsumptionTime:
        # time contains start hour as we use aggregation_interval_start in select
        end_hour = str(time.hour + 1)
        return time.strftime(f'%d-%m-%Y %H:%M - {end_hour.zfill(2)}:%M')


class OneMonthFormatter(RawAggregatedDataFormatter):
    @staticmethod
    def format_time(time: str) -> ConsumptionTime:
        def __get_year_and_month_from_time(_time: str) -> tuple[str, str]:
            parts = _time.split('-')
            return parts[0], parts[1]

        def __get_month_name(month_number_from_zero: int) -> str:
            return _(calendar.month_name[month_number_from_zero])

        year, month = __get_year_and_month_from_time(time)
        month_name = __get_month_name(int(month))
        return f'{year} {month_name}'


def format_forecast(forecast: RawConsumptionForecast) -> ConsumptionForecast:
    return f'{forecast:.10f}'
```

Файл: energy/logic/aggregated_consumption/models.py

```
from datetime import timedelta
from enum import IntEnum


class AggregationIntervalSeconds(IntEnum):
    ONE_HOUR = int(timedelta(hours=1).total_seconds())
    ONE_DAY = int(timedelta(days=1).total_seconds())
    ONE_WEEK = int(timedelta(weeks=1).total_seconds())
    ONE_MONTH = int(timedelta(weeks=4).total_seconds())
    ONE_YEAR = int(timedelta(weeks=4 * 12).total_seconds())
```

Файл: energy/logic/aggregated_consumption/queriers.py

```
from abc import ABC
from datetime import timedelta
from enum import IntEnum
from typing import Iterable, Type, TypeAlias, TypedDict

from django.db import connection

from            energy.logic.aggregated_consumption.exceptions            import
FacilityAndDescendantsHaveNoSensors
from energy.logic.aggregated_consumption.formatters import (
    OneHourFormatter,
    OneMonthFormatter, RawAggregatedDataFormatter,
```

```
)
from energy.logic.aggregated_consumption.models import AggregationIntervalSeconds
from energy.logic.aggregated_consumption.parameters import (
    AnyQueryParameters,                                    CommonQueryParameters,
OneHourAggregationIntervalQueryParameters,
)
from energy.logic.aggregated_consumption.types import (
    Consumption, ConsumptionWithTotalConsumption, RawConsumption,
    RawConsumptionWithRawTotalConsumption, RawQueryRows, RawTotalConsumption,
    TotalConsumption,
)
from energy.models import BoxSensorSet

AnyQuerier: TypeAlias = '_AggregatedConsumptionQuerierBase'


class AggregatedConsumptionQuerier:
    def __init__(self, parameters: AnyQueryParameters):
        self.__parameters = parameters
        self.__raw_consumption = None
        self.__total_consumption = None
        self.__querier = self.__get_querier_type()(self.__parameters)
        self.__query()

    def get_consumption(self) -> Consumption | None:
        if self.__raw_consumption:
            return self.__querier.format_consumption(
                self.__raw_consumption
            )
        return None

    def get_raw_consumption(self) -> RawConsumption | None:
        return self.__raw_consumption

    def get_total_consumption(self) -> TotalConsumption | None:
        if self.__total_consumption:
            return
self.formatter.format_total_consumption(self.__total_consumption)
        return None

    def __query(self):
        _ = self.__querier.get_raw_consumption_with_total_consumption()
        if not _:
            return
        self.__raw_consumption, self.__total_consumption = _

    def __get_querier_type(self) -> Type[AnyQuerier]:
        return
_AGGREGATION_INTERVAL_TO_QUERIER_MAPPING[self.__parameters.aggregation_interval]

    @property
    def formatter(self) -> RawAggregatedDataFormatter:
        return self.__querier.formatter


class _AggregatedConsumptionQuerierBase(ABC):
    SELECT_PART: str = None
    GROUP_BY_PART: str = None
    ORDER_BY_PART: str = None

    __CONSUMPTION_CTE_START = 'WITH consumption AS ('
    __CONSUMPTION_EXPRESSION = 'TRUNC(SUM(sensor_value), 7)'
    __CONSUMPTION_QUERY_SELECT_WITH_FROM = 'SELECT {select}, ' + f"""
```

```
            {__CONSUMPTION_EXPRESSION} AS consumption
            FROM sensor_values_h
    """
    __QUERY_WHERE_INTERVAL = """
        WHERE aggregation_interval_start >= '{aggregation_interval_start}'
        AND aggregation_interval_end <= '{aggregation_interval_end}'
    """
    __QUERY_WHERE_BOXES_SETS = """
        AND boxes_set_id IN ({box_set_ids})
    """
    __QUERY_GROUP_BY_AND_ORDER_BY = """
        GROUP BY {group_by}
        ORDER BY {order_by}
    """
    __CONSUMPTION_CTE_END = '),'
    __TOTAL_CONSUMPTION_CTE = """
        total_consumption AS (SELECT SUM(consumption) as total_consumption FROM
consumption)
    """
    __MAIN_SELECT = "SELECT * FROM consumption, total_consumption;"

    formatter: RawAggregatedDataFormatter = RawAggregatedDataFormatter()

    class __RawAggregatedConsumptionDataIndexes(IntEnum):
        TIME_PART = 0
        CONSUMPTION_PART = 1

    class __ConsumptionWithTotalConsumptionRowsIndexes(IntEnum):
        TIME_PART = 0
        CONSUMPTION_PART = 1
        TOTAL_CONSUMPTION_PART = 2

    class __BoxesSetIdSubqueryParameters(TypedDict):
        boxes_set_id_subquery: str
        id_or_ids_to_filter: str | int

    def __init__(self, parameters: AnyQueryParameters):
        if self.__class__ == _AggregatedConsumptionQuerierBase:
            raise NotImplementedError('this class must be subclassed')
        self.__parameters = parameters

    def get_formatted_consumption_with_total_consumption(self) \
            -> ConsumptionWithTotalConsumption | None:
        _ = self.get_raw_consumption_with_total_consumption()
        if _:
            raw_consumption, raw_total_consumption = _
            return (
                self.format_consumption(raw_consumption),
                self.format_total_consumption(raw_total_consumption)
            )
        return None

    def get_raw_consumption_with_total_consumption(self) \
            -> RawConsumptionWithRawTotalConsumption | None:
        with connection.cursor() as cursor:
            cursor.execute(self.__compose_query())
            rows = cursor.fetchall()
        if rows:
            return
self.__split_rows_into_raw_consumption_and_total_consumption(rows)
        return None

    def __split_rows_into_raw_consumption_and_total_consumption(
```

```
            self, rows: RawQueryRows
    ) -> RawConsumptionWithRawTotalConsumption:
        _ = self.__ConsumptionWithTotalConsumptionRowsIndexes
        raw_aggregated_consumption = (
            (
                row[_.TIME_PART],
                row[_.CONSUMPTION_PART]
            )
            for row in rows
        )
        raw_total_consumption = self.__extract_raw_total_consumption(rows)
        # raw_aggregate_consumption types inside are recognized as Any, so disable
type checker
        # noinspection PyTypeChecker
        return raw_aggregated_consumption, raw_total_consumption

    def __extract_raw_total_consumption(self, rows: RawQueryRows) ->
RawTotalConsumption:
        _ = self.__ConsumptionWithTotalConsumptionRowsIndexes
        # take any row, as total consumption is the same in every row
        return rows[0][_.TOTAL_CONSUMPTION_PART]

    def __compose_query(self) -> str:
        return ' '.join(
            (
                self.__CONSUMPTION_CTE_START,
                self.__compose_select(),
                self._compose_where(),
                self.__compose_group_by_and_order_by(),
                self.__CONSUMPTION_CTE_END,
                self.__TOTAL_CONSUMPTION_CTE,
                self.__MAIN_SELECT
            )
        )

    def __compose_select(self) -> str:
        return self.__CONSUMPTION_QUERY_SELECT_WITH_FROM.format(
            select=self.SELECT_PART
        )

    def _compose_where(self) -> str:
        return ' '.join(
            (
                self._compose_where_interval(),
                self.__compose_where_boxes_sets()
            )
        )

    def _compose_where_interval(self) -> str:
        # in order to filter aggregation_interval_end correctly, period end must
be shifted,
        # as we want to include all the intervals behind period end,
        #  having aggregation_interval_end <= period_end_one_day_forward_shifted
soft
        # to include 23 - 00 interval
        period_end_one_day_forward_shifted = self.parameters.period_end +
timedelta(days=1)
        return self.__QUERY_WHERE_INTERVAL.format(
            aggregation_interval_start=self.parameters.period_start,
            aggregation_interval_end=period_end_one_day_forward_shifted,
        )

    def __compose_where_boxes_sets(self) -> str:
```

```
        return self.__QUERY_WHERE_BOXES_SETS.format(
            box_set_ids=self.__get_boxes_set_ids()
        )

    def __compose_group_by_and_order_by(self) -> str:
        return self.__QUERY_GROUP_BY_AND_ORDER_BY.format(
            group_by=self.GROUP_BY_PART,
            order_by=self.ORDER_BY_PART
        )

    def __get_boxes_set_ids(self) -> str:
        ids = self.__get_box_set_ids_for_facility()
        ids = (str(_id) for _id in ids)
        return ', '.join(ids)

    def __get_box_set_ids_for_facility(self) -> Iterable[int]:
        facility                                                    =
self.__parameters.facility_to_get_consumption_for_or_all_descendants_if_any
        facility_and_descendants = facility.get_tree(facility)
        ids = BoxSensorSet.objects. \
            only('pk'). \
            filter(facility__in=facility_and_descendants). \
            values_list('pk', flat=True)
        if ids:
            return ids
        raise FacilityAndDescendantsHaveNoSensors

    def format_consumption(self, raw_consumption: RawConsumption) \
            -> Consumption:
        _ = self.__RawAggregatedConsumptionDataIndexes
        return [
            (
                self.formatter.format_time(line[_.TIME_PART]),
                self.formatter.format_consumption(line[_.CONSUMPTION_PART])
            )
            for line in raw_consumption
        ]

    def           format_total_consumption(self,           raw_total_consumption:
RawTotalConsumption) \
            -> TotalConsumption:
        return self.formatter.format_total_consumption(raw_total_consumption)

    @property
    def parameters(self) -> CommonQueryParameters:
        return self.__parameters


class _OneHourQuerier(_AggregatedConsumptionQuerierBase):
    SELECT_PART = 'aggregation_interval_start AS time'
    GROUP_BY_PART = 'time'
    ORDER_BY_PART = GROUP_BY_PART

    __ADDITIONAL_HOURS_WHERE_FILTERS_FOR_FILTER_EVERY_DAY = """
        AND      EXTRACT(HOUR      FROM      aggregation_interval_start)      >=
{hours_filtering_start_hour}
        AND      EXTRACT(HOUR      FROM      aggregation_interval_start)      <=
{hours_filtering_end_hour}
    """

    formatter = OneHourFormatter()

    def _compose_where(self) -> str:
```

```
        base_where = super()._compose_where()
        if self.parameters.filter_every_day:
            return self.__add_hours_filter_to_base_where(base_where)
        return base_where

    def __add_hours_filter_to_base_where(self, base_where: str) -> str:
        return ' '.join(
            (
                base_where,
                self.__compose_additional_hours_where_filters()
            )
        )

    def __compose_additional_hours_where_filters(self) -> str:
        return
self.__ADDITIONAL_HOURS_WHERE_FILTERS_FOR_FILTER_EVERY_DAY.format(

hours_filtering_start_hour=self.parameters.hours_filtering_start_hour,
            hours_filtering_end_hour=self.parameters.hours_filtering_end_hour
        )

    @property
    def parameters(self) -> OneHourAggregationIntervalQueryParameters:
        # typehint correct parameters type for this querier
        # noinspection PyTypeChecker
        return super().parameters


class _OneDayQuerier(_AggregatedConsumptionQuerierBase):
    SELECT_PART = 'aggregation_interval_start::date AS date'
    GROUP_BY_PART = 'date'
    ORDER_BY_PART = GROUP_BY_PART


class _OneWeekQuerier(_AggregatedConsumptionQuerierBase):
    SELECT_PART = """
        DATE_TRUNC('week', aggregation_interval_start)::DATE || ' - ' ||
        (DATE_TRUNC('week', aggregation_interval_start) + '6 days')::DATE AS week
    """
    GROUP_BY_PART = "DATE_TRUNC('week', aggregation_interval_start)"
    ORDER_BY_PART = GROUP_BY_PART


class _OneMonthQuerier(_AggregatedConsumptionQuerierBase):
    SELECT_PART = """
        EXTRACT(YEAR FROM aggregation_interval_start) || '-' ||
        EXTRACT(MONTH FROM aggregation_interval_start) AS month
    """
    GROUP_BY_PART = """
        EXTRACT(YEAR FROM aggregation_interval_start),
        EXTRACT(MONTH FROM aggregation_interval_start)
    """
    ORDER_BY_PART = GROUP_BY_PART

    formatter = OneMonthFormatter()


class _OneYearQuerier(_AggregatedConsumptionQuerierBase):
    SELECT_PART = 'EXTRACT(YEAR FROM aggregation_interval_start) AS year'
    GROUP_BY_PART = 'year'
    ORDER_BY_PART = GROUP_BY_PART
```

```
_AGGREGATION_INTERVAL_TO_QUERIER_MAPPING:       dict[AggregationIntervalSeconds,
Type[AnyQuerier]] = \
    {
        AggregationIntervalSeconds.ONE_HOUR:  _OneHourQuerier,
        AggregationIntervalSeconds.ONE_DAY:   _OneDayQuerier,
        AggregationIntervalSeconds.ONE_WEEK:  _OneWeekQuerier,
        AggregationIntervalSeconds.ONE_MONTH: _OneMonthQuerier,
        AggregationIntervalSeconds.ONE_YEAR:  _OneYearQuerier
    }
```

Файл: energy/logic/aggregated_consumption/__init__.py

```
from .controller import AggregatedEnergyConsumptionController
from .exceptions import EnergyConsumptionExceptionWithMessage
from .simple import (
    convert_request_post_dict_to_regular_dict,
    show_no_roles_page_if_user_has_no_roles,
)
```

Файл: energy/logic/aggregated_consumption/types.py

```
from datetime import datetime
from decimal import Decimal
from typing import TypeAlias

RawConsumptionForecast: TypeAlias = float

RawConsumptionTime: TypeAlias = datetime | str
RawConsumptionValue: TypeAlias = Decimal
RawTotalConsumption: TypeAlias = Decimal
RawQueryRows      =      list[tuple[RawConsumptionTime,      RawConsumptionValue,
RawTotalConsumption]]

ConsumptionTime: TypeAlias = str
ConsumptionValue: TypeAlias = str
ConsumptionForecast: TypeAlias = str
TotalConsumption: TypeAlias = str

RawConsumption = list[tuple[RawConsumptionTime, RawConsumptionValue]]
RawConsumptionWithRawTotalConsumption = tuple[
    RawConsumption, RawTotalConsumption
]
Consumption: TypeAlias = \
    list[
        tuple[ConsumptionTime, ConsumptionValue]
    ]
ConsumptionWithConsumptionForecast: TypeAlias = \
    list[tuple[ConsumptionTime, ConsumptionValue, ConsumptionForecast]]
ConsumptionWithTotalConsumption = tuple[
    Consumption, TotalConsumption
]
ConsumptionWIthConsumptionForecastWithTotalConsumption = tuple[
    ConsumptionWithConsumptionForecast, TotalConsumption
]

HOUR: TypeAlias = int
```

Файл: energy/logic/aggregated_consumption/parameters.py

```
from dataclasses import dataclass
```

```
from datetime import date
from enum import Enum
from typing import TypeAlias, TypedDict, Union

from energy.logic.aggregated_consumption.models import AggregationIntervalSeconds
from energy.logic.aggregated_consumption.types import HOUR
from institutions.models import Facility

AnyQueryParameters: TypeAlias = 'CommonQueryParameters'


class EnergyConsumptionQueryRawParameters(TypedDict, total=False):
    facility_pk: str
    aggregation_interval_seconds: str
    period_start_epoch_seconds: str
    period_end_epoch_seconds: str
    # one hour interval specific filters
    hours_filtering_start_hour: str
    hours_filtering_end_hour: str
    hour_filtering_method: str


@dataclass(kw_only=True)
class CommonQueryParameters:
    facility_to_get_consumption_for_or_all_descendants_if_any: Facility
    aggregation_interval: AggregationIntervalSeconds
    period_start: date
    period_end: date

    @property
    def period_start(self) -> date:
        return self._period_start

    @period_start.setter
    def period_start(self, value: date):
        # noinspection PyAttributeOutsideInit
        self._period_start = value

    @property
    def period_end(self) -> date:
        return self._period_end

    @period_end.setter
    def period_end(self, value: date):
        # noinspection PyAttributeOutsideInit
        self._period_end = value


@dataclass(kw_only=True)
class OneHourAggregationIntervalQueryParameters(CommonQueryParameters):
    aggregation_interval = AggregationIntervalSeconds.ONE_HOUR
    hours_filtering_method: Union['HourFilteringMethods', None]
    hours_filtering_start_hour: HOUR | None
    hours_filtering_end_hour: HOUR | None

    class HourFilteringMethods(str, Enum):
        EVERY_DAY = 'filter-every-day'
        WHOLE_INTERVAL = 'filter-whole-interval'

    @property
    def filter_every_day(self) -> bool:
        return self.hours_filtering_start_hour is not None \
            and self.hours_filtering_end_hour is not None \
```

```
            and self.__hour_filtering_method_every_day

    @property
    def __hour_filtering_method_every_day(self) -> bool:
        return self.hours_filtering_method == self.HourFilteringMethods.EVERY_DAY

    @property
    def __hour_filtering_method_whole_interval(self) -> bool:
        return                    self.hours_filtering_method                    ==
self.HourFilteringMethods.WHOLE_INTERVAL
```

Файл: energy/logic/aggregated_consumption/exceptions.py

```
from django.utils.translation import gettext as _

from utils.exceptions import ExceptionWithMessage


class EnergyConsumptionExceptionWithMessage(ExceptionWithMessage):
    pass


class InvalidHourFilteringMethod(EnergyConsumptionExceptionWithMessage):
    message = _('Такого методу погодинної фільтрації не існує')


class QueryParametersInvalid(EnergyConsumptionExceptionWithMessage):
    message = _('Сталася помилка при опрацюванні параметрів')


class FutureFilteringDate(EnergyConsumptionExceptionWithMessage):
    message = _('Фільтрація по майбутнім датам заборонена')


class
AggregationIntervalDoesNotFitPeriod(EnergyConsumptionExceptionWithMessage):
    message = _('Обраний період не містить жодного інтервалу агрегації')


class PeriodStartGreaterThanEnd(EnergyConsumptionExceptionWithMessage):
    message = _('Початок періоду пізніше кінця')


class StartHourGreaterThanEndHour(EnergyConsumptionExceptionWithMessage):
    message = _('Початкова година фільтрації більша за кінцеву')


class ForecastForParametersDoesNotExist(EnergyConsumptionExceptionWithMessage):
    message = _('Для заданих параметрів прогноз відсутній')


class
FacilityAndDescendantsHaveNoSensors(EnergyConsumptionExceptionWithMessage):
    message = _("У обраного об'єкту немає прив'язаних сенсорів")
```

Файл: energy/logic/aggregated_consumption/simple.py

```
import functools

from django.http import HttpRequest, QueryDict
from django.shortcuts import render
```

```
from energy.logic.aggregated_consumption.exceptions import QueryParametersInvalid
from users.logic import check_user_has_no_roles
from utils.types import FuncView, StrStrDict


def    convert_request_post_dict_to_regular_dict(post_dict:    QueryDict)    ->
StrStrDict:
    return post_dict.dict()


def parse_str_parameter_to_int_with_correct_exception(value: str) -> int:
    try:
        return int(value)
    except ValueError as e:
        raise QueryParametersInvalid from e


def show_no_roles_page_if_user_has_no_roles(view: FuncView) -> FuncView:
    @functools.wraps(view)
    def wrapper(request: HttpRequest, *args, **kwargs):
        if check_user_has_no_roles(request.user):
            return render(request, 'energy/consumption/you_have_no_roles.html')
        return view(request, *args, **kwargs)

    return wrapper
```

Файл: energy/logic/run_aggregation/config.py

```
from datetime import timedelta

from energokodros.settings import env

MAX_AGGREGATION_START_TIME = timedelta(
    seconds=env.int('MAX_AGGREGATION_START_TIME_SECONDS')
)
```

Файл: energy/logic/run_aggregation/__init__.py

```
from    energy.logic.run_aggregation.redis_based    import    AGGREGATION_RUNNER,
AGGREGATION_STATE_RETRIEVER

AGGREGATION_RUNNER = AGGREGATION_RUNNER
AGGREGATION_STATE_RETRIEVER = AGGREGATION_STATE_RETRIEVER
```

Файл: energy/logic/run_aggregation/models_and_constants.py

```
from enum import Enum

LAST_TIME_AGGREGATION_WAS_RUN_FORMAT = '%Y-%m-%d %H:%M:%S %z'


class AggregationStates(str, Enum):
    IDLE = 'idle'
    RUNNING = 'running'
```

Файл: energy/logic/run_aggregation/exceptions.py

```
class RunAggregationExceptionBase(Exception):
```

```
    ...


class InvalidAggregatorState(RunAggregationExceptionBase):
    ...


class AggregationAlreadyRunning(RunAggregationExceptionBase):
    ...


class AggregationDidNotStart(RunAggregationExceptionBase):
    ...
```

Файл: energy/logic/run_aggregation/bases.py

```
from abc import ABC, abstractmethod
from datetime import datetime

from energy.logic.run_aggregation.models_and_constants import AggregationStates


class AggregationRunnerBase(ABC):
    @abstractmethod
    def run_aggregation(self):
        ...


class AggregationStateRetrieverBase(ABC):
    @abstractmethod
    def get_state(self) -> AggregationStates:
        ...

    @abstractmethod
    def get_last_time_aggregation_was_run(self) -> datetime | None:
        ...


class AggregationStartedCheckerBase(ABC):
    @abstractmethod
    def check_aggregation_started(self) -> bool:
        ...
```

Файл: energy/logic/run_aggregation/redis_based/runner.py

```
from redis.client import Redis

from energy.logic.run_aggregation.bases import AggregationRunnerBase
from energy.logic.run_aggregation.exceptions import (
    AggregationAlreadyRunning,
    AggregationDidNotStart,
)
from energy.logic.run_aggregation.models_and_constants import AggregationStates
from energy.logic.run_aggregation.redis_based.aggregation_started_checker import
\
    AggregationStartedChecker
from energy.logic.run_aggregation.redis_based.exceptions import (
    InvalidStartAggregationRequestReceiverCount,
)
from       energy.logic.run_aggregation.redis_based.state_retriever       import
RedisAggregationStateRetriever
```

```
class RedisAggregationRunner(AggregationRunnerBase):
    __EXPECTED_START_AGGREGATION_REQUEST_RECEIVER_COUNT = 1

    def __init__(
            self, r: Redis, start_aggregation_message: str,
            start_aggregation_channel:          str,          state_retriever:
RedisAggregationStateRetriever,
            aggregation_started_checker: AggregationStartedChecker
    ):
        self.__r = r
        self.__start_aggregation_message = start_aggregation_message
        self.__channel = start_aggregation_channel
        self.__state_retriever = state_retriever
        self.__aggregation_started_checker = aggregation_started_checker

    def run_aggregation(self):
        self.__ensure_aggregation_is_not_running()
        self.__send_start_aggregation_message()
        self.__ensure_aggregation_started()

    def __ensure_aggregation_started(self):
        if not self.__aggregation_started_checker.check_aggregation_started():
            raise AggregationDidNotStart

    def __send_start_aggregation_message(self):
        receiver_count = self.__r.publish(
            self.__channel,
            self.__start_aggregation_message
        )

self.__ensure_start_aggregation_message_receiver_count_is_correct(receiver_count
)

    def __ensure_aggregation_is_not_running(self):
        state = self.__state_retriever.get_state()
        if state == AggregationStates.RUNNING:
            raise AggregationAlreadyRunning

    def       __ensure_start_aggregation_message_receiver_count_is_correct(self,
receiver_count: int):
        if                              receiver_count                              !=
self.__EXPECTED_START_AGGREGATION_REQUEST_RECEIVER_COUNT:
            raise InvalidStartAggregationRequestReceiverCount(receiver_count)


Файл: energy/logic/run_aggregation/redis_based/state_retriever.py

from datetime import datetime

from redis.client import Redis

from energy.logic.run_aggregation.bases import AggregationStateRetrieverBase
from energy.logic.run_aggregation.exceptions import InvalidAggregatorState
from energy.logic.run_aggregation.models_and_constants import (
    AggregationStates,
    LAST_TIME_AGGREGATION_WAS_RUN_FORMAT,
)
from energy.logic.run_aggregation.redis_based.exceptions import ValueWasNotFound
from            energy.logic.run_aggregation.redis_based.utils            import
get_value_from_redis_as_str
```

```
class RedisAggregationStateRetriever(AggregationStateRetrieverBase):
    def __init__(self, r: Redis, state_key: str, aggregation_last_time_run_key:
str):
        self.__r = r
        self.__state_key = state_key
        self.__aggregation_last_time_run_key = aggregation_last_time_run_key

    def get_state(self) -> AggregationStates:
        raw_state = get_value_from_redis_as_str(
            self.__r,
            self.__state_key
        )
        try:
            return AggregationStates(raw_state)
        except ValueError as e:
            raise InvalidAggregatorState from e

    def get_last_time_aggregation_was_run(self, formatted=False) -> datetime |
str | None:
        try:
            raw_last_time_aggregation_was_run = get_value_from_redis_as_str(
                self.__r,
                self.__aggregation_last_time_run_key
            )
        except ValueWasNotFound:
            return None
        last_time_run = datetime.strptime(
            raw_last_time_aggregation_was_run,
            LAST_TIME_AGGREGATION_WAS_RUN_FORMAT
        )
        if formatted:
            return last_time_run.strftime('%H:%M %d-%m-%Y')
        return last_time_run
```

Файл: energy/logic/run_aggregation/redis_based/aggregation_started_checker.py

```
from datetime import datetime, timedelta

from redis.client import Redis

from energy.logic.run_aggregation.bases import AggregationStartedCheckerBase
from energy.logic.run_aggregation.redis_based.exceptions import (
    AggregationStartedConfirmationMessageInvalid, SubOrUnsubMessageTypeExpected,
)


class AggregationStartedChecker(AggregationStartedCheckerBase):
    def __init__(
            self, aggregation_start_results_channel: str,
            aggregation_started_successfully_message: str,
            aggregation_failed_message: str,
            r: Redis, max_start_time: timedelta
    ):
        self.__pubsub = r.pubsub()
        self.__aggregation_start_results_channel                                =
aggregation_start_results_channel
        self.__confirmation_message = aggregation_started_successfully_message
        self.__failure_message = aggregation_failed_message
        self.__max_start_time = max_start_time

    def check_aggregation_started(self) -> bool:
```

```
        # sub and unsub each time, as otherwise there are too many subscribers,
        # probably due to how gunicorn pre-forks
        self.__subscribe_to_channel()
        aggregation_started = self.__check_aggregation_started()
        self.__unsubscribe_from_channel()
        return aggregation_started

    def __check_aggregation_started(self):
        max_wait_time = datetime.now() + self.__max_start_time
        while datetime.now() < max_wait_time:
            if self.__get_aggregation_started():
                return True
        return False

    def __get_aggregation_started(self) -> bool:
        if message := self.__pubsub.get_message():
            data = message.get('data').decode()
            if data == self.__confirmation_message:
                return True
            if data == self.__failure_message:
                return False
            raise AggregationStartedConfirmationMessageInvalid
        return False

    def __subscribe_to_channel(self):
        self.__pubsub.subscribe(self.__aggregation_start_results_channel)
        self.__dump_message()

    def __unsubscribe_from_channel(self):
        self.__pubsub.unsubscribe(self.__aggregation_start_results_channel)
        self.__dump_message()

    def __dump_message(self):
        # timeout needed as sometimes program is too fast
        # and message is created after we try to dump it, so it clutters channel
        message = self.__pubsub.get_message(timeout=1)
        if message.get('type') not in ('subscribe', 'unsubscribe'):
            raise SubOrUnsubMessageTypeExpected
```

Файл: energy/logic/run_aggregation/redis_based/__init__.py

```
from redis.client import Redis

from energokodros.settings import env
from energy.logic.run_aggregation.config import MAX_AGGREGATION_START_TIME
from energy.logic.run_aggregation.redis_based.aggregation_started_checker import
(
    AggregationStartedChecker, AggregationStartedChecker,
)
from energy.logic.run_aggregation.redis_based.runner import (
    RedisAggregationRunner,
    RedisAggregationRunner,
)
from energy.logic.run_aggregation.redis_based.state_retriever import (
    RedisAggregationStateRetriever,
    RedisAggregationStateRetriever,
)

__REDIS = Redis(
    host=env('REDIS_HOST'),
    port=env.int('REDIS_PORT'),
    db=env.int('REDIS_DB'),
```

```
    password=env('REDIS_PASSWORD')
)

AGGREGATION_STATE_RETRIEVER = RedisAggregationStateRetriever(
    r=__REDIS,
    state_key=env('AGGREGATION_STATE_KEY'),
    aggregation_last_time_run_key=env('AGGREGATION_LAST_TIME_RUN_KEY')
)

__aggregation_started_checker = AggregationStartedChecker(
    aggregation_start_results_channel=env(
        'AGGREGATION_START_RESULTS_CHANNEL'
    ),
    aggregation_started_successfully_message=env(
        'AGGREGATION_STARTED_SUCCESSFULLY_MESSAGE'
    ),
    aggregation_failed_message=env(
        'AGGREGATION_FAILED_TO_START_MESSAGE'
    ),
    r=__REDIS,
    max_start_time=MAX_AGGREGATION_START_TIME
)

AGGREGATION_RUNNER = RedisAggregationRunner(
    r=__REDIS,
    start_aggregation_message=env('START_AGGREGATION_MESSAGE'),
    start_aggregation_channel=env('AGGREGATOR_CONTROLLER_REDIS_CHANNEL'),
    state_retriever=AGGREGATION_STATE_RETRIEVER,
    aggregation_started_checker=__aggregation_started_checker
)
```

Файл: energy/logic/run_aggregation/redis_based/utils.py

```
from redis.client import Redis

from energy.logic.run_aggregation.redis_based.exceptions import (
    ValueWasNotFound,
)


def get_value_from_redis_as_str(r: Redis, key: str) -> str:
    raw_value = r.get(key)
    if not raw_value:
        raise ValueWasNotFound
    if isinstance(raw_value, bytes):
        raw_value = raw_value.decode()
    return raw_value
```

Файл: energy/logic/run_aggregation/redis_based/exceptions.py

```
from energy.logic.run_aggregation.exceptions import RunAggregationExceptionBase


class InvalidStartAggregationRequestReceiverCount(RunAggregationExceptionBase):
    def __init__(self, actual_receiver_count: int):
        self.__actual_receiver_count = actual_receiver_count

    def __str__(self):
        return f'{super().__str__()}\n' \
               f'actual receiver count: {self.__actual_receiver_count}'
```

```
class ValueWasNotFound(RunAggregationExceptionBase):
    ...


class AggregationStartedConfirmationMessageInvalid(RunAggregationExceptionBase):
    ...

class SubOrUnsubMessageTypeExpected(RunAggregationExceptionBase):
    ...
```

Файл: energy/logic/box_with_sensors_creation/config_and_models.py

```
from energy.forms import BoxFormNoRelationFields, BoxSensorSetFormset,
SensorsFormset


class STEPS:
    BOX = 'box'
    SENSORS = 'sensors'
    BOX_SENSORS_SET = 'box_sensor_set'


FORMS = (
    (STEPS.BOX, BoxFormNoRelationFields),
    # default sensors count is max but can be adjusted by user
    (STEPS.SENSORS, SensorsFormset),
    # box_sensors_set should be created dynamically depending on sensor count from
previous step
    (STEPS.BOX_SENSORS_SET, BoxSensorSetFormset)
)

BOX_WITH_SENSORS_CREATION_TEMPLATES = {
    STEPS.BOX:             'energy/box_with_sensors_creation/box.html',
    STEPS.SENSORS:         'energy/box_with_sensors_creation/sensors.html',
    STEPS.BOX_SENSORS_SET:
'energy/box_with_sensors_creation/box_sensors_set.html',
}

FORMS_ORDER_FROM_ZERO = {
    STEPS.BOX:             0,
    STEPS.SENSORS:         1,
    STEPS.BOX_SENSORS_SET: 2
}
```

Файл: energy/logic/box_with_sensors_creation/initial_data_creation.py

```
from typing import TypeAlias

from django.utils.translation import gettext_lazy as _

from energokodros.settings import SENSOR_COUNT_PER_BOX
from utils.types import StrKeyDict

FormsetData: TypeAlias = list[StrKeyDict]


def create_initial_sensor_numbers_for_sensors_formset() -> FormsetData:
    return [
        {'sensor_number': sensor_number}
        for sensor_number in range(1, SENSOR_COUNT_PER_BOX + 1)
```

```
    ]


def create_initial_sensor_numbers_in_set_for_box_sensor_set_formset(
        sensors_formset_cleaned_data: FormsetData) -> FormsetData:
    return [
        {
            'sensor_number':        sensors_data['sensor_number'],
            'sensor_number_in_set': sensor_number_in_set,
            'line_name':            _(f'Лінія {sensor_number_in_set}')
        }
        for           sensor_number_in_set,           sensors_data           in
enumerate(sensors_formset_cleaned_data, start=1)
    ]
```

Файл: energy/logic/box_with_sensors_creation/__init__.py

```
from .config_and_models import BOX_WITH_SENSORS_CREATION_TEMPLATES, FORMS, STEPS
from .initial_data_creation import (
    create_initial_sensor_numbers_for_sensors_formset,
    create_initial_sensor_numbers_in_set_for_box_sensor_set_formset,
    FormsetData,
)
from    .simple    import    (create_box_sensor_sets_along_with_box_and_sensors,
get_forms_from_from_list)
```

Файл: energy/logic/box_with_sensors_creation/simple.py

```
from typing import Iterable

from django.db import IntegrityError
from django.forms import Form

from energokodros.settings import SENSOR_COUNT_PER_BOX
from energy.forms import (
    BoxFormNoRelationFields, BoxSensorSetForBoxWithSensorsCreationForm,
    BoxSensorSetFormset, SensorFormNoRelationFields, SensorsFormset,
)
from       energy.logic.box_with_sensors_creation.config_and_models       import
FORMS_ORDER_FROM_ZERO, STEPS
from energy.models import Box, BoxSensorSet, Sensor
from institutions.models import Facility


def create_box_sensor_sets_along_with_box_and_sensors(
        box_form: BoxFormNoRelationFields, sensors_formset: SensorsFormset,
        box_sensor_set_formset: BoxSensorSetFormset
):
    box = box_form.save()
    _ = create_box_sensor_set_form_and_sensor_form_match(box_sensor_set_formset,
sensors_formset)
    for box_sensor_set_form, sensor_form in _:
        sensor: Sensor = sensor_form.save()
        box_sensor_set: BoxSensorSet = box_sensor_set_form.save(commit=False)
        facility = box_sensor_set_form.cleaned_data['facility']
        fill_box_sensor_set_relations(box_sensor_set, box, facility, sensor)
        box_sensor_set.save()
    box.refresh_from_db()
    validate_sensors_count(box)
```

```
def create_box_sensor_set_form_and_sensor_form_match(
        box_sensor_set_formset:      BoxSensorSetFormset,      sensors_formset:
SensorsFormset
)          ->          Iterable[tuple[BoxSensorSetForBoxWithSensorsCreationForm,
SensorFormNoRelationFields]]:
    return zip(box_sensor_set_formset.forms, sensors_formset.forms)


def fill_box_sensor_set_relations(
        box_sensor_set: BoxSensorSet, box: Box, facility: Facility, sensor: Sensor
) -> BoxSensorSet:
    box_sensor_set.facility = facility
    box_sensor_set.sensor = sensor
    box_sensor_set.box = box
    return box_sensor_set


def get_forms_from_from_list(forms: list[Form]) -> \
        tuple[BoxFormNoRelationFields, SensorsFormset, BoxSensorSetFormset]:
    # noinspection PyTypeChecker
    return (
        forms[FORMS_ORDER_FROM_ZERO[STEPS.BOX]],
        forms[FORMS_ORDER_FROM_ZERO[STEPS.SENSORS]],
        forms[FORMS_ORDER_FROM_ZERO[STEPS.BOX_SENSORS_SET]],
    )


def validate_sensors_count(box: Box):
    if box.sensor_sets.count() != SENSOR_COUNT_PER_BOX:
        raise IntegrityError('Invalid sensor count')
```

Файл: energy/views/generic.py

```
from django.urls import reverse_lazy
from django.utils.translation import gettext_lazy as _

from energy.forms import BoxForm, BoxSensorSetForm, SensorForm
from energy.models import Box, BoxSensorSet, Sensor
from utils.common import admin_rights_and_login_required
from utils.views import EditDeleteObjectUpdateView, ListViewWithFiltering


@admin_rights_and_login_required
class BoxListView(ListViewWithFiltering):
    queryset = Box.objects.all()
    filter_fields = ('identifier', 'description')
    template_name = 'energy/box_list.html'


@admin_rights_and_login_required
class BoxEditDeleteView(EditDeleteObjectUpdateView):
    model = Box
    form_class = BoxForm
    success_url = reverse_lazy('box-list')
    template_name = 'energy/edit_box.html'
    EDIT_SUCCESS_MESSAGE = _('Ящик успішно відредаговано')


@admin_rights_and_login_required
class SensorListView(ListViewWithFiltering):
    queryset = Sensor.objects.all()
    filter_fields = ('sensor_number', 'sensor_description')
```

```
    template_name = 'energy/sensor_list.html'


@admin_rights_and_login_required
class SensorEditDeleteView(EditDeleteObjectUpdateView):
    model = Sensor
    form_class = SensorForm
    success_url = reverse_lazy('sensor-list')
    template_name = 'energy/edit_sensor.html'
    EDIT_SUCCESS_MESSAGE = _('Сенсор успішно відредаговано')


@admin_rights_and_login_required
class BoxSensorSetListView(ListViewWithFiltering):
    queryset = BoxSensorSet.objects.all()
    filter_fields = ('facility__name', 'line_name', 'box__identifier')
    template_name = 'energy/box_sensor_set_list.html'


@admin_rights_and_login_required
class BoxSensorSetEditDeleteView(EditDeleteObjectUpdateView):
    model = BoxSensorSet
    form_class = BoxSensorSetForm
    success_url = reverse_lazy('box-sensor-set-list')
    template_name = 'energy/edit_box_sensor_set.html'
    EDIT_SUCCESS_MESSAGE = _('Набір успішно відредаговано')


Файл: energy/views/__init__.py

from       .aggregated_consumption_view       import       ConsumptionPageView,
get_consumption_with_total_consumption
from .box_with_sensors_creation import BoxWithSensorsCreateView
from .generic import (
    BoxEditDeleteView,          BoxListView,          BoxSensorSetEditDeleteView,
BoxSensorSetListView,
    SensorEditDeleteView,
    SensorListView,
)
from .simple import (
    get_aggregation_status_and_last_time_run, get_facilities_choices_for_role,
    run_aggregation,
)


Файл: energy/views/box_with_sensors_creation.py

from django.contrib import messages
from django.db import transaction
from django.forms import Form
from django.shortcuts import redirect
from django.urls import reverse_lazy
from django.utils.decorators import method_decorator
from django.utils.translation import gettext_lazy as _
from formtools.wizard.views import SessionWizardView

from energy.forms import ChooseInstitutionForm
from energy.logic.box_with_sensors_creation import (
    BOX_WITH_SENSORS_CREATION_TEMPLATES,
create_box_sensor_sets_along_with_box_and_sensors,
    create_initial_sensor_numbers_for_sensors_formset,
    create_initial_sensor_numbers_in_set_for_box_sensor_set_formset,      FORMS,
FormsetData,
```

```
    get_forms_from_from_list, STEPS,
)
from utils.common import admin_rights_and_login_required
from utils.types import StrKeyDict


@admin_rights_and_login_required
class BoxWithSensorsCreateView(SessionWizardView):
    form_list = FORMS
    success_url = reverse_lazy('box-list')
    success_message = _('Успішно створено ящик та сенсори')

    @method_decorator(transaction.atomic)
    def done(self, form_list, **kwargs):
        box_form,        sensors_formset,        box_sensor_set_formset        =
self.__get_forms_from_form_list(
            form_list
        )
        create_box_sensor_sets_along_with_box_and_sensors(
            box_form, sensors_formset, box_sensor_set_formset
        )
        messages.success(
            self.request,
            self.success_message
        )
        return redirect(self.success_url)

    def __get_forms_from_form_list(self, forms: list[Form]):
        return get_forms_from_from_list(forms)

    def __create_box_sensor_set_initial(self) -> FormsetData:
        return create_initial_sensor_numbers_in_set_for_box_sensor_set_formset(
            self.get_cleaned_data_for_step(STEPS.SENSORS)
        )

    def get_context_data(self, form, **kwargs):
        ctx = super().get_context_data(form, **kwargs)
        if self.steps.current == STEPS.BOX_SENSORS_SET:
            ctx.update(self.__create_institutions_choice_context_dict())
        return ctx

    def __create_institutions_choice_context_dict(self) -> StrKeyDict:
        return {
            'choose_institution_form': ChooseInstitutionForm()
        }

    def get_template_names(self):
        return [BOX_WITH_SENSORS_CREATION_TEMPLATES[self.steps.current]]

    def get_form_initial(self, step):
        if step == STEPS.SENSORS:
            return create_initial_sensor_numbers_for_sensors_formset()
        if step == STEPS.BOX_SENSORS_SET:
            return self.__create_box_sensor_set_initial()
        return {}
```

Файл: energy/views/aggregated_consumption_view.py

```
from django.contrib.auth.decorators import login_required
from django.http import HttpRequest, JsonResponse
from django.views.generic import FormView
```

```
from energy.forms import EnergyConsumptionDisplayPageControlForm
from energy.logic.aggregated_consumption import (
    AggregatedEnergyConsumptionController,
convert_request_post_dict_to_regular_dict,
    EnergyConsumptionExceptionWithMessage,
show_no_roles_page_if_user_has_no_roles,
)
from utils.common.decoration import decorate_class_or_function_view


@decorate_class_or_function_view(login_required)
@decorate_class_or_function_view(show_no_roles_page_if_user_has_no_roles)
class ConsumptionPageView(FormView):
    template_name = 'energy/consumption/main_page.html'

    def get_form(self, form_class=None):
        return EnergyConsumptionDisplayPageControlForm(self.request.user)


def get_consumption_with_total_consumption(request: HttpRequest) -> JsonResponse:
    try:
        # noinspection PyTypeChecker
        parameters = convert_request_post_dict_to_regular_dict(request.POST)
        controller     =     AggregatedEnergyConsumptionController(request.user,
parameters)
        consumption_with_optional_forecast, total_consumption = \

controller.get_consumption_with_optional_forecast_and_total_consumption()
    except EnergyConsumptionExceptionWithMessage as e:
        return JsonResponse(e.message, status=400, safe=False)
    return JsonResponse(
        {
            'consumption_with_optional_forecast':
consumption_with_optional_forecast,
            'total_consumption':                  total_consumption
        },
        safe=False
    )
```

Файл: energy/views/simple.py

```
import logging

from django.contrib.auth.decorators import login_required
from django.http import HttpRequest, JsonResponse

from energy.logic import get_facilities_formatted_choices_for_user_role
from        energy.logic.run_aggregation        import        AGGREGATION_RUNNER,
AGGREGATION_STATE_RETRIEVER
from energy.logic.run_aggregation.exceptions import RunAggregationExceptionBase
from users.logic.simple import check_role_belongs_to_user
from users.models import UserRole
from utils.common import get_object_by_hashed_id_or_404

logger = logging.getLogger(__name__)


@login_required
def get_facilities_choices_for_role(request: HttpRequest) -> JsonResponse:
    # noinspection PyTypeChecker
    role: UserRole = get_object_by_hashed_id_or_404(
        UserRole, request.POST.get('role_id')
```

```
    )
    check_role_belongs_to_user(request.user, role)
    choices = get_facilities_formatted_choices_for_user_role(role)
    return JsonResponse(choices, safe=False)


@login_required
def run_aggregation(request: HttpRequest) -> JsonResponse:
    try:
        AGGREGATION_RUNNER.run_aggregation()
    except RunAggregationExceptionBase as e:
        logger.error(e, exc_info=True)
        return JsonResponse(
            {'message': 'unexpected error happened'},
            status=400
        )
    return JsonResponse({'message': 'aggregation started successfully'})


@login_required
def get_aggregation_status_and_last_time_run(request: HttpRequest):
    _ = AGGREGATION_STATE_RETRIEVER
    return JsonResponse(
        {
            'aggregation_status':        _.get_state(),
            'aggregation_last_rime_run':
_.get_last_time_aggregation_was_run(formatted=True)
        }
    )
```

Файл: utils/__init__.py

Файл: utils/types.py

```
from typing import Any, Callable, Tuple, TypeAlias

from django.http import HttpRequest, HttpResponse
from django.views import View as View_for_typehint

StrTuple: TypeAlias = Tuple[str, ...]
StrKeyDict: TypeAlias = dict[str, Any]
StrStrDict: TypeAlias = dict[str, str]
FuncView: TypeAlias = Callable[[HttpRequest, ...], HttpResponse]
ClassView: TypeAlias = View_for_typehint
ViewType: TypeAlias = FuncView | ClassView
ViewDecorator: TypeAlias = Callable[[ViewType], ViewType]
```

Файл: utils/crypto.py

```
import string

from hashids import Hashids

from energokodros.settings import MIN_HASH_LENGTH, SECRET_KEY

# custom alphabet to exclude url-unsafe symbols (/,% etc)
__ALPHABET = string.ascii_lowercase + string.digits
```

```
_HASHER = Hashids(salt=SECRET_KEY, min_length=MIN_HASH_LENGTH,
alphabet=__ALPHABET)

_REGEX = f'[{__ALPHABET}]' + '{%i,}' % MIN_HASH_LENGTH # noqa pylint:
disable=C0209


class IntHasher:
    regex = _REGEX

    @classmethod
    def hide_int(cls, val: int) -> str:
        if val < 0:
            raise ValueError('only positive integers are allowed')
        return _HASHER.encode(val)

    @classmethod
    def reveal_int(cls, val: str) -> int:
        res = _HASHER.decode(val)
        if len(res) != 1:
            raise ValueError(f'one integer return expected, probably wrong input,
got {res}')
        return res[0]

    def to_python(self, value: str) -> int:
        return self.reveal_int(value)

    def to_url(self, value: int) -> str:
        return self.hide_int(value)


class StringHasher:
    regex = _REGEX

    @classmethod
    def hide_str(cls, val: str) -> str:
        return _HASHER.encode(*(ord(c) for c in val))

    @classmethod
    def reveal_str(cls, val: str) -> str:
        try:
            return ''.join(chr(c) for c in _HASHER.decode(val))
        except ValueError as e:
            raise ValueError('can not transform to char, probably wrong input')
from e

    def to_python(self, value: str) -> str:
        return self.reveal_str(value)

    def to_url(self, value: str) -> str:
        return self.hide_str(value)
```

Файл: utils/exceptions.py

```
class ExceptionWithMessage(Exception):
    message: str
```

Файл: utils/templatetags/js_reverse.py

```
from django import template
from django.urls import reverse
```

```
from django.utils.safestring import mark_safe

register = template.Library()


@register.simple_tag
def create_url_name_id_p_tag_with_link_content(url_name: str) -> str:
    """provides a way to include links in template dynamically"""
    return mark_safe(f"<p id='{url_name}'>{reverse(url_name)}</p>")
```

Файл: utils/templatetags/show_messages_if_any.py

```
from django import template
from django.template.loader import render_to_string
from django.utils.safestring import mark_safe

register = template.Library()


@register.simple_tag(takes_context=True)
def show_messages(context: dict) -> str:
    messages = context.get('messages', None)
    output = ''
    if messages:
        for message in messages:
            output += render_to_string(
                'common/message.html',
                {
                    'message_class': message.tags,
                    'message_text':  message.message
                }
            )
    return mark_safe(output)
```

Файл: utils/templatetags/url_replace.py

```
from typing import Any

from django import template
from django.http import HttpRequest

register = template.Library()


@register.simple_tag
def url_replace(request: HttpRequest, field: str, value: Any):
    query_string = request.GET.copy()
    query_string[field] = value
    return query_string.urlencode()
```

Файл: utils/templatetags/__init__.py

Файл: utils/templatetags/pagination.py

```
from typing import Iterable

from django import template
from django.core.paginator import Paginator
```

```
from django.http import HttpRequest

from utils.templatetags.url_replace import url_replace

register = template.Library()

START_PAGE_NUMBER = 1
SIDE_NEIGHBOR_COUNT = 5
MAX_NEIGHBOR_COUNT = SIDE_NEIGHBOR_COUNT * 2
TOTAL_PAGE_ITEMS_COUNT = MAX_NEIGHBOR_COUNT + 1

INVISIBLE_PAGE_DELTA = SIDE_NEIGHBOR_COUNT + 1


class StartAndEndPagesIndexes:

    def __init__(self, current_page_number: int, num_pages: int):
        self.__start    =    max(START_PAGE_NUMBER,    current_page_number    -
SIDE_NEIGHBOR_COUNT)
        self.__end = min(num_pages, current_page_number + SIDE_NEIGHBOR_COUNT)
        self.__num_pages = num_pages

    def get_pages_indexes(self) -> Iterable[int]:
        self.__expand_pages_indexes_boundaries_if_needed()
        self.__shrink_pages_indexes_boundaries_if_needed()
        return self.__create_pages_indexes()

    def __create_pages_indexes(self) -> Iterable[int]:
        return range(self.__start, self.__end + 1)[:TOTAL_PAGE_ITEMS_COUNT]

    def __expand_pages_indexes_boundaries_if_needed(self):
        if self.__check_end_less_than_start_plus_max_neighbor_count():
            self.__set_end_to_start_plus_max_neighbor_count()
        elif self.__check_start_greater_than_end_minus_max_neighbor_count():
            self.__set_start_to_end_minus_max_neighbor_count()

    def __shrink_pages_indexes_boundaries_if_needed(self):
        if self.__start < START_PAGE_NUMBER:
            self.__end -= self.__start
            self.__start = START_PAGE_NUMBER
        elif self.__end > self.__num_pages:
            self.__end = self.__num_pages

    def __check_end_less_than_start_plus_max_neighbor_count(self) -> bool:
        return self.__end < self.__start_plus_max_neighbor_count

    def __set_end_to_start_plus_max_neighbor_count(self):
        self.__end = self.__start_plus_max_neighbor_count

    def __check_start_greater_than_end_minus_max_neighbor_count(self) -> bool:
        return self.__start > self.__end_minus_max_neighbor_count

    def __set_start_to_end_minus_max_neighbor_count(self):
        self.__start = self.__end_minus_max_neighbor_count

    @property
    def __start_plus_max_neighbor_count(self) -> int:
        return self.__start + MAX_NEIGHBOR_COUNT

    @property
    def __end_minus_max_neighbor_count(self) -> int:
        return self.__end - MAX_NEIGHBOR_COUNT
```

```
@register.filter(name='paginate_with_items_per_page_limit')
def paginate_with_items_per_page_limit(paginator: Paginator, current_page_number:
int):
    if __check_pages_need_to_be_shortened(paginator.num_pages):
        indexes          =          StartAndEndPagesIndexes(current_page_number,
paginator.num_pages)
        return indexes.get_pages_indexes()
    return paginator.page_range


@register.simple_tag
def next_invisible_page_number_if_exists_else_max_available_link(
        request: HttpRequest, page_number: int, num_pages: int) -> str:
    next_invisible_page_number = page_number + INVISIBLE_PAGE_DELTA
    if next_invisible_page_number < num_pages:
        page_to_redirect_to = next_invisible_page_number
    else:
        page_to_redirect_to = num_pages
    return url_replace(request, 'page', page_to_redirect_to)


@register.simple_tag
def previous_invisible_page_number_if_exists_else_min_available_link(
        request: HttpRequest, page_number: int) -> str:
    previous_invisible_page_number = page_number - INVISIBLE_PAGE_DELTA
    if previous_invisible_page_number < START_PAGE_NUMBER:
        page_to_redirect_to = START_PAGE_NUMBER
    else:
        page_to_redirect_to = previous_invisible_page_number
    return url_replace(request, 'page', page_to_redirect_to)


def __check_pages_need_to_be_shortened(num_pages: int) -> bool:
    return num_pages > MAX_NEIGHBOR_COUNT
```

Файл: utils/forms/crispy_forms_mixin.py

```
from typing import Iterable

from crispy_forms.bootstrap import StrictButton
from crispy_forms.helper import FormHelper
from django.forms import Form, ModelForm


class CrispyFormMixin:
    """
    provides unified way to make form crispy,
    relies on crispy forms so form used with it heed to be displayed
    by corresponding crispy tag
    """

    def __init__(self, *args, **kwargs):
        super().__init__(*args, **kwargs)
        self.__order_fields_if_present()
        self.helper = FormHelper(self)
        self.__add_buttons()

    def __add_buttons(self):
        try:
            # noinspection PyUnresolvedReferences
            buttons: Iterable[StrictButton] = self.Meta.buttons
```

```
        except AttributeError:
            return
        for button in buttons:
            self.helper.layout.append(button)

    def __order_fields_if_present(self: Form):
        try:
            # noinspection PyUnresolvedReferences
            self.order_fields(self.Meta.fields_order)
        except AttributeError:
            return

    def add_button(self, button: StrictButton):
        self.helper.layout.append(button)


class CrispyModelForm(CrispyFormMixin, ModelForm):
    pass
```

Файл: utils/forms/__init__.py

```
from .crispy_buttons import (
    create_button, create_danger_button, create_primary_button,
    UPDATE_DELETE_BUTTONS_SET,
)
from .crispy_forms_mixin import CrispyFormMixin, CrispyModelForm
from .secure_model_choice_field import INT_HIDER, INT_REVEALER,
SecureModelChoiceField
from .set_form_buttons_to_update_delete import set_form_buttons_to_update_delete
from .widgets import SelectWithFormControlClass
```

Файл: utils/forms/widgets.py

```
from django.forms import Select


class SelectWithFormControlClass(Select):
    """
    when rendering with {% crispy %} form select widget has no form-control,
    so this class adds it
    """

    def get_context(self, name, value, attrs):
        ctx = super().get_context(name, value, attrs)
        ctx['widget']['attrs']['class'] = 'form-control'
        return ctx
```

Файл: utils/forms/crispy_buttons.py

```
from crispy_forms.bootstrap import StrictButton
from django.utils.translation import gettext as _

from utils.types import StrTuple

COMMON_CSS_CLASSES = 'btn mt-4'
DEFAULT_BUTTON_VALUE = 'submit'
DEFAULT_BUTTON_NAME = 'submit'
DEFAULT_BUTTON_TYPE = 'submit'
DEFAULT_DELETE_VALUE = 'delete'
```

```
def create_button(
        text: str, *, value: str = DEFAULT_BUTTON_VALUE, name: str =
DEFAULT_BUTTON_NAME,
        button_type: str = DEFAULT_BUTTON_TYPE, additional_css_classes: StrTuple
= ()
) -> StrictButton:
    return StrictButton(
        text,
        value=value,
        name=name,
        type=button_type,
        css_class=' '.join(additional_css_classes + (COMMON_CSS_CLASSES,))
    )


def __add_classes_for_button(kwargs: dict, classes_to_add: StrTuple) -> dict:
    additional_css_classes = kwargs.get('additional_css_classes', None)
    if additional_css_classes:
        additional_css_classes += classes_to_add
    else:
        additional_css_classes = classes_to_add
    kwargs['additional_css_classes'] = additional_css_classes
    return kwargs


def create_primary_button(*args, **kwargs):
    kwargs = __add_classes_for_button(kwargs, ('btn-primary',))
    return create_button(*args, **kwargs)


def create_danger_button(*args, **kwargs):
    kwargs = __add_classes_for_button(kwargs, ('btn-danger',))
    return create_button(*args, **kwargs)


UPDATE_DELETE_BUTTONS_SET = (
    create_primary_button(_('Оновити')),
    create_danger_button(_('Видалити'), value=DEFAULT_DELETE_VALUE)
)
```

Файл: utils/forms/secure_model_choice_field.py

```
from typing import Literal

from django.db.models import Model
from django.forms import models

from utils.crypto import IntHasher
from utils.forms.widgets import SelectWithFormControlClass

INT_HIDER = IntHasher.hide_int
INT_REVEALER = IntHasher.reveal_int


class SecureModelChoiceField(models.ModelChoiceField):
    __int_hider = staticmethod(INT_HIDER)
    __int_revealer = staticmethod(INT_REVEALER)
    widget = SelectWithFormControlClass()

    def prepare_value(self, value: Model | None | Literal['']) -> str:
        if isinstance(value, Model):
```

```
            return self.__int_hider(value.pk)
        return super().prepare_value(value)

    def to_python(self, value: str | None) -> Model | None:
        if value:
            return super().to_python(self.__int_revealer(value))
        return super().to_python(value)

    def __init__(self, *args, **kwargs):
        label_from_instance_function                                    =
kwargs.pop('label_from_instance_function', None)
        super().__init__(*args, **kwargs)
        self.label_from_instance_function = label_from_instance_function

    def label_from_instance(self, obj):
        if self.label_from_instance_function:
            return self.label_from_instance_function(obj)
        return super().label_from_instance(obj)
```

Файл: utils/forms/set_form_buttons_to_update_delete.py

```
from django.forms import Form

from utils.forms.crispy_buttons import UPDATE_DELETE_BUTTONS_SET


def set_form_buttons_to_update_delete(form: type(Form)) -> type(Form):
    try:
        form.Meta.buttons = UPDATE_DELETE_BUTTONS_SET
    except AttributeError as e:
        raise ValueError('provided form has no Meta') from e
    return form
```

Файл: utils/common/db.py

```
from django.db import transaction


def delete_everything_created_in_transaction():
    transaction.set_rollback(True)
```

Файл: utils/common/decoration.py

```
import inspect
from typing import Any, TypeGuard

from django.utils.decorators import method_decorator
from django.views import View

from utils.types import ClassView, FuncView, ViewDecorator, ViewType


def decorate_class_or_function_view(decorator: ViewDecorator) -> ViewDecorator:
    def view_wrapper(view: ViewType) -> ViewType:
        if _check_object_is_function(view):
            view = _decorate_function(view, decorator)
        elif _check_object_is_class_view(view):
            view = _decorate_class_view(view, decorator)
        else:
            raise ValueError(
```

```
                f"passed view must be a class view or function view, got type
{type(view)}"
            )
        return view

    return view_wrapper


def _decorate_function(view: FuncView, decorator: ViewDecorator) -> FuncView:
    return decorator(view)


def _decorate_class_view(view: ClassView, decorator: ViewDecorator) -> ClassView:
    view.dispatch = method_decorator(decorator)(view.dispatch)
    return view


def _check_object_is_function(_object: Any) -> bool:
    return inspect.isfunction(_object)


def _check_object_is_class_view(_object: Any) -> TypeGuard[ClassView]:
    return issubclass(_object, View)
```

Файл: utils/common/admin_rights.py

```
import functools

from django.contrib.auth.decorators import login_required
from django.http import HttpRequest, HttpResponse

from users.models import User
from utils.common.decoration import decorate_class_or_function_view
from utils.types import FuncView, ViewType


def admin_rights_and_login_required(view: ViewType):
    return
decorate_class_or_function_view(_admin_rights_and_login_required)(view)


def _admin_rights_and_login_required(view: FuncView) -> FuncView:
    view = login_required(view)
    view = _admin_rights_required(view)
    return view


def _admin_rights_required(view: FuncView) -> FuncView:
    @functools.wraps(view)
    def wrapper(request: HttpRequest, *args, **kwargs):
        if is_admin_non_authenticated_safe(request.user):
            return view(request, *args, **kwargs)
        return HttpResponse(status=403)

    return wrapper


def is_admin_non_authenticated_safe(user: User) -> bool:
    return user.is_authenticated and user.is_admin
```

Файл: utils/common/__init__.py

```
from .admin_rights import admin_rights_and_login_required,
is_admin_non_authenticated_safe
from .ajax import redirect_to_object_pk_in_post
from .db import delete_everything_created_in_transaction
from .model_objects_crypto_related import (
    Choices, compose_secure_choices_for_queryset,
get_object_by_hashed_id_or_404, hash_id,
)
```

Файл: utils/common/model_objects_crypto_related.py

```
from typing import Any, Callable

from django.db.models import Manager, Model, QuerySet
from django.db.models.base import ModelBase
from django.http import Http404
from django.shortcuts import get_object_or_404

from utils.forms import INT_HIDER, INT_REVEALER, SecureModelChoiceField

Choices = list[tuple[str, str]]


def compose_secure_choices_for_queryset(
        qs: QuerySet,
        label_from_instance_function: Callable[[Any], str] = None
) -> Choices:
    choices = SecureModelChoiceField(
        queryset=qs, empty_label=None,
        label_from_instance_function=label_from_instance_function
    ).choices
    return [(str(value), label) for value, label in choices]


def get_object_by_hashed_id_or_404(_class: ModelBase | Manager | QuerySet,
hashed_id: str) -> Model:
    try:
        return get_object_or_404(_class, pk=INT_REVEALER(hashed_id))
    except ValueError as e:
        raise Http404 from e


def hash_id(model_object: Model) -> str:
    return INT_HIDER(model_object.pk)
```

Файл: utils/common/list_view_with_filtering.py

Файл: utils/common/email.py

```
import logging
from smtplib import SMTPException

from django.contrib import messages
from django.core.mail import EmailMultiAlternatives
from django.http import HttpRequest
```

```
from django.template.loader import render_to_string
from django.utils.safestring import mark_safe
from django.utils.translation import gettext_lazy as _


def try_send_email_add_warning_if_failed(
        request: HttpRequest, email: str, subject: str, message: str
):
    html = render_to_string(
        'email/regular_email.html',
        context={
            'header': subject,
            'text':   message
        }
    )
    if not send_html_email(email, subject, html):
        messages.warning(request, _(f'Не вдалося надіслати повідомлення на пошту
{email}'))


def send_html_email(email: str, subject: str, html: str) -> bool:
    msg = EmailMultiAlternatives(
        subject=subject,
        body=mark_safe(html),
        # django requires from_email to be explicitly set to None
        # to use settings.DEFAULT_FROM_EMAIL
        from_email=None,
        to=[email]
    )
    msg.content_subtype = 'html'
    try:
        msg.send()
    except SMTPException as e:
        logging.error(
            'failed to send email to %s with error: %s',
            email,  # pylint: disable=C0209
            e
        )
        return False
    return True
```

Файл: utils/common/ajax.py

```
from typing import Callable, Type

from django.db.models import Model
from django.http import HttpRequest, JsonResponse
from django.urls import reverse_lazy

from              utils.common.model_objects_crypto_related              import
get_object_by_hashed_id_or_404


def redirect_to_object_pk_in_post(_class: Type[Model], redirect_to_url: str) ->
Callable:
    def create_link(request: HttpRequest) -> JsonResponse:
        nonlocal _class
        nonlocal redirect_to_url
        obj = get_object_by_hashed_id_or_404(_class, request.POST.get('id'))
        return JsonResponse(
            {'url': reverse_lazy(redirect_to_url, kwargs={'pk': obj.pk})}
        )
```

```
    return create_link
```

Файл: utils/common/object_to_queryset.py

```
from django.db import models
from django.db.models import QuerySet


def object_to_queryset(obj: models.Model) -> QuerySet:
    return type(obj).objects.filter(pk=obj.pk)
```

Файл: utils/views/__init__.py

```
from .edit_object_update_view import (
    AdditionalSetupRequiredFormMixin,
    EditDeleteObjectUpdateView,
)
from .list_view_with_filtering import ListViewWithFiltering
```

Файл: utils/views/list_view_with_filtering.py

```
from functools import reduce
from typing import Iterable, TypeAlias, Union

from django.db.models import Q, QuerySet
from django.views.generic import ListView

from utils.types import StrTuple

_ListViewWithMixinType:           TypeAlias           =           Union[ListView,
'QuerySetFieldsIcontainsFilterPkOrderedMixin']
DEFAULT_PAGINATE_BY = 7


class QuerySetFieldsIcontainsFilterPkOrderedMixin:
    """this mixin is supposed to be used with ListViews"""
    filter_fields: StrTuple = None
    fields_order_by_before_pk: StrTuple = set()

    def get_queryset(self: _ListViewWithMixinType) -> QuerySet:
        self.__check_used_properly()
        if search_value := self.__get_search_value():
            qs = QuerySetFieldsIcontainsFilter(
                self.queryset,
                self.filter_fields,
            ).filter(search_value)
        else:
            qs = self.queryset
        return qs.order_by(*self.fields_order_by_before_pk, '-pk')

    def __check_used_properly(self: _ListViewWithMixinType):
        if not issubclass(self.__class__, ListView):
            raise ValueError('this mixin must be used with a ListView')
        if self.filter_fields is None:
            raise ValueError('you must set filter_fields for a model')
        return True

    def __get_search_value(self: ListView) -> str:
        return self.request.GET.get('search_value', None)
```

```
class QuerySetFieldsIcontainsFilter:
    def __init__(self, qs: QuerySet, fields_to_filter: Iterable[str]):
        self._qs = qs
        self._fields_to_filter = [f'{field}__icontains' for field in
fields_to_filter]

    def filter(self, value: str) -> QuerySet:
        _filter = reduce(
            lambda q, field: q | Q(**{field: value}),
            self._fields_to_filter,
            Q()
        )
        return self._qs.filter(_filter)


class ListViewWithFiltering(QuerySetFieldsIcontainsFilterPkOrderedMixin,
ListView):
    paginate_by = DEFAULT_PAGINATE_BY
    pass
```

Файл: utils/views/edit_object_update_view.py

```
from typing import TypeAlias, Union

from django.contrib import messages
from django.db.models import Model
from django.forms import Form
from django.http import HttpRequest
from django.shortcuts import redirect
from django.utils.translation import gettext as _
from django.views.generic import UpdateView

from utils.forms.crispy_buttons import DEFAULT_DELETE_VALUE


class AdditionalSetupRequiredFormMixin:
    def additionally_setup(self, obj: Model):
        raise NotImplementedError


_EditDeleteObjectViewWithMixinType: TypeAlias = Union[UpdateView,
'EditDeleteObjectUpdateViewMixin']


class EditDeleteObjectUpdateViewMixin:
    """
    must be used as a mixin for UpdateView, needs to be first in superclass list
    to correctly override methods
    """
    EDIT_SUCCESS_MESSAGE: str = _('Успішно відредаговано')
    DELETE_SUCCESS_MESSAGE: str = _('Успішно видалено')
    NO_CHANGES_MESSAGE: str = _('Ви не внесли жодних змін, тож були
перенаправлені')

    def post(self: _EditDeleteObjectViewWithMixinType, request: HttpRequest,
*args, **kwargs):
        if request.POST['submit'] == DEFAULT_DELETE_VALUE:
            return self.__delete_object()
        # noinspection PyUnresolvedReferences
        return super().post(request, *args, **kwargs)
```

```
    def get_context_data(self: _EditDeleteObjectViewWithMixinType, **kwargs):
        # noinspection PyUnresolvedReferences
        data = super().get_context_data(**kwargs)
        self.fill_from_object_if_needed(data)
        return data

    def   fill_from_object_if_needed(self:   _EditDeleteObjectViewWithMixinType,
data: dict):
        form: Form = data['form']
        if isinstance(form, AdditionalSetupRequiredFormMixin):
            form.additionally_setup(self.object)

    def form_valid(self: _EditDeleteObjectViewWithMixinType, form: Form):
        if form.has_changed():
            messages.success(
                self.request,
                self.EDIT_SUCCESS_MESSAGE
            )
        else:
            messages.info(
                self.request,
                self.NO_CHANGES_MESSAGE
            )
        return super().form_valid(form)

    def __delete_object(self: _EditDeleteObjectViewWithMixinType):
        self.get_object().delete()
        messages.warning(
            self.request,
            self.DELETE_SUCCESS_MESSAGE,
        )
        #  default  redirect  goes  to  self.get_success_url(),  but  is  uses
self.object,
        # which has just been deleted, so redirect to .success_url
        return redirect(self.success_url)


class EditDeleteObjectUpdateView(EditDeleteObjectUpdateViewMixin, UpdateView):
    pass
```

Файл: energokodros/asgi.py

```
"""
ASGI config for energokodros project.

It exposes the ASGI callable as a module-level variable named ``application``.

For more information on this file, see
https://docs.djangoproject.com/en/4.0/howto/deployment/asgi/
"""

import os

from django.core.asgi import get_asgi_application

os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'energokodros.settings')

application = get_asgi_application()
```

Файл: energokodros/__init__.py

Файл: energokodros/settings.py

```
import os.path
from pathlib import Path

import environ
from django.urls import reverse_lazy

env = environ.Env(
    DEBUG=(bool, False),
)

BASE_DIR = Path(__file__).resolve().parent.parent

DEBUG = env('DEBUG')
SECRET_KEY = env('SECRET_KEY')

ALLOWED_HOSTS = [] if DEBUG else env('ALLOWED_HOSTS').split(',')
CSRF_TRUSTED_ORIGINS = ['https://' + host for host in ALLOWED_HOSTS]

INSTALLED_APPS = [
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.staticfiles',
    'django.contrib.messages',

    'users',
    'energy',
    'institutions',
    'utils',

    'crispy_forms',
    'active_link',
    'treebeard',
    'formtools',
]

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]

ROOT_URLCONF = 'energokodros.urls'

AUTHENTICATION_BACKENDS = [
    'users.backends.AllowTryAuthenticateInactive',
]

TEMPLATES = [
    {
        'BACKEND':  'django.template.backends.django.DjangoTemplates',
        'DIRS':     [BASE_DIR.joinpath('templates')],
        'APP_DIRS': True,
        'OPTIONS':  {
            'context_processors': [
```

```
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',

'users.context_processors.user_roles_applications_to_review_count'
            ],
        },
    },
]

WSGI_APPLICATION = 'energokodros.wsgi.application'

DATABASES = {
    'default': {
        'ENGINE':   'django.db.backends.postgresql',
        'NAME':     env('DB_NAME'),
        'USER':     env('DB_USERNAME'),
        'PASSWORD': env('DB_PASSWORD'),
        'HOST':     env('DB_HOST'),
        'PORT':     env('DB_PORT'),
    }
}

AUTH_PASSWORD_VALIDATORS = [
    {
        'NAME':
'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
    },
    {
        'NAME':
'django.contrib.auth.password_validation.MinimumLengthValidator',
    },
    {
        'NAME':
'django.contrib.auth.password_validation.CommonPasswordValidator',
    },
    {
        'NAME':
'django.contrib.auth.password_validation.NumericPasswordValidator',
    },
]

AUTH_USER_MODEL = 'users.User'

LANGUAGE_CODE = 'uk'

TIME_ZONE = 'Europe/Kiev'

USE_I18N = True
# enable using timedelta objects to set session expiration time
SESSION_SERIALIZER = 'django.contrib.sessions.serializers.PickleSerializer'

STATIC_URL = 'static/'
STATIC_ROOT = BASE_DIR / 'staticfiles'

STATICFILES_DIRS = [os.path.join(BASE_DIR, STATIC_URL)]

DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'

LOGIN_URL = reverse_lazy('login')
LOGIN_REDIRECT_URL = reverse_lazy('home')
LOGOUT_REDIRECT_URL = LOGIN_URL
```

```
EMAIL_BACKEND = 'django.core.mail.backends.smtp.EmailBackend'
EMAIL_HOST = env('EMAIL_HOST')
EMAIL_HOST_USER = env('EMAIL_HOST_USER')
EMAIL_HOST_PASSWORD = env('EMAIL_HOST_PASSWORD')
EMAIL_PORT = env.int('EMAIL_PORT')
DEFAULT_FROM_EMAIL = env('DEFAULT_FROM_EMAIL')
EMAIL_USE_TLS = True

CRISPY_TEMPLATE_PACK = 'bootstrap4'

MIN_HASH_LENGTH = env.int('MIN_HASH_LENGTH')

SENSOR_COUNT_PER_BOX = env.int('SENSORS_COUNT_PER_BOX')

ACTIVE_LINK_STRICT = True

LOGGING = {
    "version":                  1,
    "disable_existing_loggers": False,
    "root":                     {"level": "INFO", "handlers": ["console"]},
    "handlers":                 {
        "console": {
            "level":     "INFO",
            "class":     "logging.StreamHandler",
            "formatter": "app",
        },
    },
    "loggers":                  {
        "django": {
            "handlers":  ["console"],
            "level":     "INFO",
            "propagate": True
        },
    },
    "formatters":               {
        "app": {
            "format":  (
                "%(asctime)s [%(levelname)s] "
                "(%(module)s.%(funcName)s) %(message)s"
            ),
            "datefmt": "%Y-%m-%d %H:%M:%S",
        },
    },
}
```

Файл: energokodros/urls.py

```
from django.urls import include, path, re_path, register_converter, reverse_lazy
from django.views.generic import RedirectView

from utils.crypto import IntHasher, StringHasher

register_converter(IntHasher, 'hashed_int')
register_converter(StringHasher, 'hashed_str')

urlpatterns = [
    re_path(
        r'^$|home',
        RedirectView.as_view(url=reverse_lazy('aggregated-consumption')),
        name='home'
    ),
```

```
    path('users/', include('users.urls')),
    path('energy/', include('energy.urls')),
    path('facilities/', include('institutions.urls'))
]
```

Файл: energokodros/wsgi.py

```
"""
WSGI config for energokodros project.

It exposes the WSGI callable as a module-level variable named ``application``.

For more information on this file, see
https://docs.djangoproject.com/en/4.0/howto/deployment/wsgi/
"""

import os

from django.core.wsgi import get_wsgi_application

os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'energokodros.settings')

application = get_wsgi_application()
```

Файл: static/css/base.css

```
ul {
    list-style: none;
    padding: 0;
}

.asteriskField {
    display: none;
}

.active {
    opacity: 50%;
}
```

Файл: static/css/energy/consumption_table.css

```
table {
    border-collapse: collapse;
}

th, td {
    padding: 1rem;
}

thead, tfoot {
    background: #eee;
}

thead {
    position: sticky;
    top: 0;
    border-bottom: 2px solid #ccc;
}

tfoot {
    position: sticky;
```

```
    bottom: 0;
    border-top: 2px solid #ccc;
}
```

Файл: static/js/institutions/edit_facility.js

```
$(document).ready(function () {
    style_and_setup_selects();
})

function style_and_setup_selects() {
    // disabled is included in django field to ignore possible input
    // but user can move to facility or role page by clicking a descendant
    __get_descendants_select().removeAttr('disabled');
    __get_roles_for_facility_select().removeAttr('disabled');

    add_descendants_select_on_click_label();
    add_roles_on_click_label();

    __get_descendants_select().change(
        redirect_to_selected_object('edit-facility-pk-in-post')
    );
    __get_roles_for_facility_select().change(
        redirect_to_selected_object('edit-user-role-pk-in-post')
    );
}


function add_descendants_select_on_click_label() {
    add_on_click_redirects_text_after_div_first_label(__get_descendants_div());
}

function add_roles_on_click_label() {

add_on_click_redirects_text_after_div_first_label(__get_roles_for_facility_selec
t_div());
}

function __get_descendants_div() {
    return $('#div_id_descendants');
}


function __get_descendants_select() {
    return $('#id_descendants')
}

function __get_roles_for_facility_select_div() {
    return $('#div_id_roles_that_have_access_to_this_facility');
}

function __get_roles_for_facility_select() {
    return $('#id_roles_that_have_access_to_this_facility');
}
```

Файл: static/js/institutions/dynamic_facility_choices_for_institution.js

```
function update_facility_choices_for_chosen_institution_callback(
    institution_select_func,
    update_with_ids_and_labels_callback
) {
```

```
    return () => {
        $.ajax({
            url: reverse_url('get-institution-facilities'),
            type: 'POST',
            dataType: 'json',
            headers: get_headers_for_ajax_object(),
            data: {
                institution_id:
__get_selected_institution_id(institution_select_func)
            },
            success: (ids_and_labels) => {
                update_with_ids_and_labels_callback(ids_and_labels);
            },
            error: (error) => {
                add_error_alert(DEFAULT_UNEXPECTED_ERROR_MESSAGE);
            }
        });
    }
}

function __get_selected_institution_id(institution_select_func) {
    return get_selected_option_for_select(institution_select_func());
}
```

Файл: static/js/institutions/new_facility.js

```
$(document).ready(function () {
    set_width_for_all_selects();
    __get_institution_select().change(
        update_facility_choices_for_chosen_institution_callback(
            __get_institution_select,
            update_facility_select
        )
    );
    __get_institution_select().trigger('change');
})

function update_facility_select(ids_and_labels) {
    update_select_with_options(__get_parent_facility_select(), ids_and_labels);
}

function set_width_for_all_selects() {
    __get_parent_facility_select().addClass('w-100');
    __get_institution_select().addClass('w-100');
}


function __get_institution_select() {
    return $('#id_institution');
}

function __get_parent_facility_select() {
    return $('#id_parent_facility');
}
```

Файл: static/js/energy/edit_box_sensor_set.js

```
$(document).ready(() => {
    add_redirect_to_related_sensor_on_click();
    remove_disabled_from_related_objects_select();
    add_on_click_redirect_to_related_object_text();
    add_redirect_to_related_facility();
})
```

```
function add_redirect_to_related_facility() {
    let facility_label = $('label[for="id_facility"]');
    facility_label.append("| <a id='go_to_facility' class='link-primary'>Перейти
до об'єкту</a>");
    $('#go_to_facility').click(
        redirect_to_selected_object_with_get_id_callback(
            'edit-facility-pk-in-post',
            () => {
                return
get_selected_option_for_select(__get_related_facility_select());
            }
        )
    );
}

function add_redirect_to_related_sensor_on_click() {
    __get_related_sensor_select().click(
        redirect_to_selected_object('edit-sensor-pk-in-post')
    )
}

function remove_disabled_from_related_objects_select() {
    __get_related_sensor_select().removeAttr('disabled');
}

function add_on_click_redirect_to_related_object_text() {

add_on_click_redirects_text_after_div_first_label(__get_related_sensor_select_di
v())
}

function __get_related_sensor_select_div() {
    return $('#div_id_sensor_number')
}

function __get_related_sensor_select() {
    return $('#id_sensor_number')
}

function __get_related_facility_select() {
    return $('#id_facility')
}
```

Файл: static/js/energy/edit_box.js

```
$(document).ready(() => {
    add_on_click_redirects_texts();
    add_redirect_on_click_to_related_objects();
    remove_disabled_from_related_objects_selects();
})

function add_on_click_redirects_texts() {
    add_on_click_redirects_text_after_div_first_label(__get_sensor_sets_div());
}

function add_redirect_on_click_to_related_objects() {
    __get_sensor_sets_select().change(
        redirect_to_selected_object('edit-box-sensor-set-pk-in-post')
    );
}

function remove_disabled_from_related_objects_selects() {
```

```
    __get_sensor_sets_select().removeAttr('disabled');
}

function __get_sensor_sets_select() {
    return $('#id_sensor_sets');
}

function __get_sensor_sets_div() {
    return $('#div_id_sensor_sets');
}
```

Файл: static/js/energy/edit_sensor.js

```
$(document).ready(() => {
    add_on_click_redirects_texts();
    add_redirect_on_click_to_related_objects();
    remove_disabled_from_related_objects_selects();
})

function add_on_click_redirects_texts() {
    add_on_click_redirects_text_after_div_first_label(__get_box_div());
    add_on_click_redirects_text_after_div_first_label(__get_set_div());
}

function add_redirect_on_click_to_related_objects() {
    __get_box_select().click(
        redirect_to_selected_object('edit-box-pk-in-post')
    );
    __get_set_select().click(
        redirect_to_selected_object('edit-box-sensor-set-pk-in-post')
    );
}

function remove_disabled_from_related_objects_selects() {
    __get_box_select().removeAttr('disabled');
    __get_set_select().removeAttr('disabled');
}

function __get_box_select() {
    return $('#id_box')
}

function __get_set_select() {
    return $('#id_set')
}

function __get_box_div() {
    return $('#div_id_box');
}

function __get_set_div() {
    return $('#div_id_set');
}
```

Файл: static/js/energy/consumption/displaying.js

```
$(document).ready(function () {

__get_set_view_mode_to_table_button().click(__set_display_mode_to_table_and_upda
te_if_data_is_present);

__get_set_view_mode_to_chart_button().click(__set_display_mode_to_chart_and_upda
te_if_data_is_present);
```

```
})

const DRAW_TABLE = 'table';
const DRAW_CHART = 'chart';

const CHOSEN_BUTTON_CLASS = 'btn-outline-secondary';
const AVAILABLE_BUTTON_CLASS = 'btn-outline-primary';

let CHOSEN_DISPLAYING_OPTION = DRAW_TABLE;

let CONSUMPTION_WITH_OPTIONAL_FORECAST = null, TOTAL_CONSUMPTION = null;

let CONSUMPTION_INDEX_IN_RAW_DATA = 1,
    LABEL_INDEX_IN_RAW_DATA = 0,
    CONSUMPTION_FORECAST_INDEX_IN_RAW_DATA = 2;

function draw_content() {
    if (CONSUMPTION_WITH_OPTIONAL_FORECAST) {
        if (CHOSEN_DISPLAYING_OPTION === DRAW_TABLE) {
            _draw_table(CONSUMPTION_WITH_OPTIONAL_FORECAST);
            return;
        }
        if (CHOSEN_DISPLAYING_OPTION === DRAW_CHART) {
            _draw_chart(CONSUMPTION_WITH_OPTIONAL_FORECAST);
        }
    }
    throw new Error('consumption is not set')
}

function _draw_chart(data) {
    __get_aggregated_consumption_data_div().empty();
    __get_aggregated_consumption_data_div().append(
        '<canvas id="energy_chart"></canvas>'
    );
    let datasets = __generate_chart_data(data);
    const config = {
        type: 'bar',
        data: datasets,
        options: {
            plugins: {
                zoom: {
                    zoom: {
                        wheel: {
                            enabled: true,
                        },
                        pinch: {
                            enabled: true
                        },
                        mode: 'x',
                    }
                }
            },
            scales: {
                x: {
                    stacked: true,
                },
                y: {
                    stacked: false,
                    beginAtZero: true
                }
            }
        }
    }
```

```
    const myChart = new Chart(
        $('#energy_chart'),
        config
    );
}

function _draw_table(data) {
    __set_energy_html_content(__generate_table(data))
}

function __generate_chart_data(raw_consumption_data) {
    let labels = [];
    let consumption = [];
    let consumption_forecast = [];
    if (_check_consumption_forecast_is_present(raw_consumption_data)) {
        for (const line of raw_consumption_data) {
            labels.push(line[LABEL_INDEX_IN_RAW_DATA]);
            consumption.push(parseFloat(line[CONSUMPTION_INDEX_IN_RAW_DATA]));

consumption_forecast.push(parseFloat(line[CONSUMPTION_FORECAST_INDEX_IN_RAW_DATA
]))
        }
    } else {
        consumption_forecast = null;
        for (const line of raw_consumption_data) {
            labels.push(line[LABEL_INDEX_IN_RAW_DATA]);
            consumption.push(parseFloat(line[CONSUMPTION_INDEX_IN_RAW_DATA]));
        }
    }
    let data = {
        labels: labels,
        datasets: [
            {
                label: 'Споживання в кіловат-годинах',
                backgroundColor: 'rgb(255, 99, 132, 0.7)',
                data: consumption,
            }
        ]
    };
    if (consumption_forecast) {
        data.datasets.push({
            label: 'Прогнозоване споживання',
            backgroundColor: 'rgba(152,255,134,0.7)',
            data: consumption_forecast,
        })
    }
    return data;
}


function __set_energy_html_content(content) {
    __get_aggregated_consumption_data_div().html(content);
}

function __generate_table(raw_consumption_data) {
    let data_rows = '';
    for (const item of raw_consumption_data) {
        data_rows += `
            <tr>
            <td>${item[LABEL_INDEX_IN_RAW_DATA]}</td>
            <td>${item[CONSUMPTION_INDEX_IN_RAW_DATA]}</td>
            </tr>`
    }
```

```
    return '' +
        '<table class="rounded rounded-3 w-100">' +
        '<thead>' +
        '<tr>' +
        '<th scope="col">Час</th>' +
        '<th scope="col">Кіловат години</th>' +
        '</tr>' +
        '</thead>' +
        '<tfoot>' +
        '<tr>' +
        '<th scope="col">Загалом</th>' +
        `<th scope="col">${TOTAL_CONSUMPTION}</th>` +
        '</tr>' +
        '</tfoot>' +
        '<tbody>' +
        data_rows +
        '</tbody>'
}

function _check_consumption_forecast_is_present(raw_consumption_data) {
    return  raw_consumption_data[0][CONSUMPTION_FORECAST_INDEX_IN_RAW_DATA]  !==
undefined
}

function __set_display_mode_to_table_and_update_if_data_is_present() {
    __set_button_to_chosen(__get_set_view_mode_to_table_button());
    __set_button_to_can_be_chosen(__get_set_view_mode_to_chart_button());
    CHOSEN_DISPLAYING_OPTION = DRAW_TABLE;
    if (CONSUMPTION_WITH_OPTIONAL_FORECAST) {
        _draw_table(CONSUMPTION_WITH_OPTIONAL_FORECAST);
    }
}

function __set_display_mode_to_chart_and_update_if_data_is_present() {
    __set_button_to_chosen(__get_set_view_mode_to_chart_button());
    __set_button_to_can_be_chosen(__get_set_view_mode_to_table_button());
    CHOSEN_DISPLAYING_OPTION = DRAW_CHART;
    if (CONSUMPTION_WITH_OPTIONAL_FORECAST) {
        _draw_chart(CONSUMPTION_WITH_OPTIONAL_FORECAST);
    }
}

function __set_button_to_chosen(button) {
    button.removeClass(AVAILABLE_BUTTON_CLASS);
    button.addClass(CHOSEN_BUTTON_CLASS);
}

function __set_button_to_can_be_chosen(button) {
    button.removeClass(CHOSEN_BUTTON_CLASS);
    button.addClass(AVAILABLE_BUTTON_CLASS);
}

function __remove_chosen_class(button) {
    button.removeClass(CHOSEN_BUTTON_CLASS);
}
```

Файл: static/js/energy/consumption/aggregation.js

```
$(document).ready(function () {
    __get_run_aggregation_button().click(send_start_aggregation_request);
    __get_run_aggregation_button().click(clear_alerts);
```

```
    update_aggregation_status_and_last_time_run();
})

const FAILED_TO_RETRIEVE_DATA = 'не вдалося отримати дані';
const IDLE = 'idle', RUNNING = 'running';


function send_start_aggregation_request() {
    _set_run_aggregation_button_loading();
    $.ajax({
        url: reverse_url('run-aggregation'),
        type: 'POST',
        dataType: 'json',
        headers: get_headers_for_ajax_object(),
        success: (response) => {
            _set_run_aggregation_button_active();
            add_success_alert('Агрегацію було успішно запущено');
        },
        error: (response) => {
            _set_run_aggregation_button_active();
            add_error_alert('Виникла помилка при запуску агрегації');
        }
    });
}

function update_aggregation_status_and_last_time_run() {
    $.ajax({
        url: reverse_url('get-aggregation-status-and-last-time-run'),
        type: 'POST',
        dataType: 'json',
        headers: get_headers_for_ajax_object(),
        success: (response) => {
            _set_run_aggregation_button_active();
            set_aggregation_last_time_run(response.aggregation_last_rime_run);
            set_aggregation_status_adjust_start_aggregation_button_accordingly(
                response.aggregation_status
            );
        },
        error: (response) => {
            _set_run_aggregation_button_active();
            set_aggregation_last_time_run(null)

set_aggregation_status_adjust_start_aggregation_button_accordingly(null)
        }
    });
}

function
set_aggregation_status_adjust_start_aggregation_button_accordingly(aggregation_s
tatus) {
    if (aggregation_status === IDLE) {
        __get_aggregation_status_p().text('не активна')
        _set_start_aggregation_button_active();
    } else if (aggregation_status === RUNNING) {
        __get_aggregation_status_p().text('активна')
        _set_start_aggregation_button_disabled();
    } else {
        throw 'wrong aggregation status'
    }
}

function set_aggregation_last_time_run(value) {
    let display_value = value ? value : FAILED_TO_RETRIEVE_DATA
```

```
    __get_aggregation_last_time_run_p().text(display_value)
}

function _set_run_aggregation_button_active() {
    __get_start_aggregation_button_spinner_span().addClass('d-none');
    __get_start_aggregation_button_text_span().removeClass('d-none');
}

function _set_run_aggregation_button_loading() {
    __get_start_aggregation_button_text_span().addClass('d-none');
    __get_start_aggregation_button_spinner_span().removeClass('d-none');
}


function _set_start_aggregation_button_disabled() {
    __get_run_aggregation_button().removeClass('btn-primary');
    __get_run_aggregation_button().addClass('btn-secondary');
}

function _set_start_aggregation_button_active() {
    __get_run_aggregation_button().removeClass('btn-secondary');
    __get_run_aggregation_button().addClass('btn-primary');
}

function __get_start_aggregation_button_text_span() {
    return $('#run_aggregation_button_text')
}

function __get_start_aggregation_button_spinner_span() {
    return $('#start_aggregation_request_processing_spinner')
}

function __get_aggregation_last_time_run_p() {
    return $('#aggregation_last_time_run_id')
}

function __get_aggregation_status_p() {
    return $('#aggregation_status_id')
}

function __get_run_aggregation_button() {
    return $('#run_aggregation_button')
}
```

Файл: static/js/energy/consumption/querying.js

```
$(document).ready(function () {
    let get_data_button = $('#get_data_button');
    // order of handlers is important here
    get_data_button.on(
        'click',
        check_control_form_required_fields_are_filled
    )
    get_data_button.on(
        'click',
validate_hours_filters_if_one_hour_aggregation_interval_is_chosen
    )
    get_data_button.on(
        'click',
        get_consumption_data_and_update
    );
    get_data_button.on(
        'click',
```

```
        clear_alerts
    )
})

function get_consumption_data_and_update() {
    $.ajax({
        url: reverse_url('get-aggregated-consumption-for-facility'),
        type: 'POST',
        dataType: 'json',
        data: __compose_aggregation_query_parameters(),
        headers: get_headers_for_ajax_object(),
        success: (data) => {
            CONSUMPTION_WITH_OPTIONAL_FORECAST                                    =
data.consumption_with_optional_forecast;
            if (CONSUMPTION_WITH_OPTIONAL_FORECAST === null) {
                add_warning_alert('За заданими фільтрами дані відсутні');
                return;
            }
            TOTAL_CONSUMPTION = data.total_consumption;
            draw_content();
        },
        error: (error) => {
            if (error.responseJSON) {
                add_warning_alert(error.responseJSON);
                return
            }
            add_error_alert(DEFAULT_UNEXPECTED_ERROR_MESSAGE);
        }
    });
}


function __compose_aggregation_query_parameters() {
    let _ = get_selected_option_for_select;
    let facility_pk = _(__get_facility_select());
    let aggregation_interval_seconds = _(__get_aggregation_interval_select());
    let period_start_epoch_seconds = __get_period_start_epoch_seconds();
    let period_end_epoch_seconds = __get_period_end_epoch_seconds();
    let role_pk = _(__get_role_select());
    let include_forecast = __get_forecast_checkbox_value();
    let base_parameters = {
        role_pk: role_pk,
        facility_pk: facility_pk,
        period_start_epoch_seconds: period_start_epoch_seconds,
        period_end_epoch_seconds: period_end_epoch_seconds,
        aggregation_interval_seconds: aggregation_interval_seconds,
        include_forecast: include_forecast
    };
    if (INCLUDE_HOURS_FILTER) {
        base_parameters.hours_filtering_start_hour                                =
_(__get_hours_filtering_start_select());
        base_parameters.hours_filtering_end_hour                                  =
_(__get_hours_filtering_end_select());
        base_parameters.hour_filtering_method = CHOSEN_HOUR_FILTERING_METHOD;
    }
    return base_parameters;
}

function __get_aggregated_consumption_data_div() {
    return $('#energy_content');
}

function __get_period_start_epoch_seconds() {
```

```
    return
__get_data_input_epoch_value_seconds_by_id(__get_period_start_input());
}

function __get_data_input_epoch_value_seconds_by_id(date_input) {
    // converting to seconds from milliseconds
    return Date.parse(date_input.val()) / 1000;
}

function __get_period_end_epoch_seconds() {
    return __get_data_input_epoch_value_seconds_by_id(__get_period_end_input());
}

function __get_aggregation_interval_select() {
    return $('#id_aggregation_interval_seconds');
}

function __get_role_select() {
    return $('#id_role');
}
```

Файл: static/js/energy/consumption/helping.js

```
$(document).ready(function () {
    update_facilities_list_for_role();
    __get_roles_select().change(update_facilities_list_for_role);
    __get_aggregation_interval_select().change(
        add_or_remove_hours_filtering_for_aggregation_interval_select
    );
    set_default_hour_filtering_choices();

__get_hours_filtering_reset_button().click(set_default_hour_filtering_choices);

__get_hours_filtering_start_select().change(__remove_error_from_hours_select);
    __get_hours_filtering_end_select().change(__remove_error_from_hours_select);
    set_date_inputs_now();
    add_hour_filtering_filter_day_or_interval_buttons()
})

let INCLUDE_HOURS_FILTER = false;

let FILTER_EVERY_DAY_VALUE = 'filter-every-day', FILTER_WHOLE_INTERVAL_VALUE =
'filter-whole-interval';
let CHOSEN_HOUR_FILTERING_METHOD = FILTER_EVERY_DAY_VALUE;
let FILTER_EVERY_DAY_BUTTON = null, FILTER_WHOLE_INTERVAL_BUTTON = null;

const DEFAULT_START_HOUR_FILTER_OPTION = '0', DEFAULT_END_HOUR_FILTER_OPTION =
'23';

function update_facilities_list_for_role() {
    $.ajax({
        url: reverse_url('get-facilities-list-for-role'),
        type: 'POST',
        dataType: 'json',
        data: {
            role_id: __get_selected_role_id()
        },
        headers: get_headers_for_ajax_object(),
        success: (ids_and_labels) => {
            update_select_with_options(__get_facility_select(), ids_and_labels);
        },
        error: (error) => {
            let message = DEFAULT_UNEXPECTED_ERROR_MESSAGE;
```

```
            if (error.message) {
                message = error.message
            }
            add_error_alert(message);
        }
    });
}

function add_or_remove_hours_filtering_for_aggregation_interval_select() {
    let hours_filtering_div = __get_hours_filtering_div();
    if (_check_one_hour_interval_is_selected()) {
        hours_filtering_div.removeClass('d-none');
    } else {
        hours_filtering_div.addClass('d-none');
    }
}

function check_control_form_required_fields_are_filled(event) {
    let form = __get_control_form();
    let fields = form.find('input:required, select:required').serializeArray();
    $.each(fields, function (i, field) {
        if (!field.value) {
            add_error_alert('Не усі поля були заповнені');
            event.stopImmediatePropagation();
            return false
        }
    })
}

function set_default_hour_filtering_choices() {
    __get_hours_filtering_start_select()
        .val(DEFAULT_START_HOUR_FILTER_OPTION)
        .attr('selected', true);
    __get_hours_filtering_end_select()
        .val(DEFAULT_END_HOUR_FILTER_OPTION)
        .attr('selected', true);
}

function validate_hours_filters_if_one_hour_aggregation_interval_is_chosen() {
    INCLUDE_HOURS_FILTER = false;
    let                                start_value                                    =
get_selected_option_for_select(__get_hours_filtering_start_select());
    let                                 end_value                                     =
get_selected_option_for_select(__get_hours_filtering_end_select());
    // default values cover everything, so there is no point to pass them
    if (
        start_value === DEFAULT_START_HOUR_FILTER_OPTION
        && end_value === DEFAULT_END_HOUR_FILTER_OPTION
    ) {
        return
    }
    INCLUDE_HOURS_FILTER = true;
}


function add_hour_filtering_filter_day_or_interval_buttons() {
    let filter_every_day_button = `
        <a class="btn ${CHOSEN_BUTTON_CLASS}" id="filter_every_day_button_id">
            Кожен день
        </a>`;
    let filter_whole_interval_button = `
        <a                 class="btn                 ${AVAILABLE_BUTTON_CLASS}"
id="filter_whole_interval_button_id">
```

```
            Увесь інтервал
        </a>`;

__get_hour_filtering_options_buttons_container().append(filter_every_day_button)
;

__get_hour_filtering_options_buttons_container().append(filter_whole_interval_bu
tton);
    FILTER_EVERY_DAY_BUTTON = $('#filter_every_day_button_id');
    FILTER_WHOLE_INTERVAL_BUTTON = $('#filter_whole_interval_button_id');
    FILTER_EVERY_DAY_BUTTON.click(() => {
        __set_button_to_chosen(FILTER_EVERY_DAY_BUTTON);
        CHOSEN_HOUR_FILTERING_METHOD = FILTER_EVERY_DAY_VALUE;
        __set_button_to_can_be_chosen(FILTER_WHOLE_INTERVAL_BUTTON);
    });
    FILTER_WHOLE_INTERVAL_BUTTON.click(() => {
        __set_button_to_chosen(FILTER_WHOLE_INTERVAL_BUTTON);
        CHOSEN_HOUR_FILTERING_METHOD = FILTER_WHOLE_INTERVAL_VALUE;
        __set_button_to_can_be_chosen(FILTER_EVERY_DAY_BUTTON);
    })
}

function set_date_inputs_now() {
    let now = _get_current_date_for_date_input();
    __get_period_start_input().val(now);
    __get_period_end_input().val(now);
}
function _check_one_hour_interval_is_selected() {
    let                                  value                                    =
parseInt(get_selected_option_for_select(__get_aggregation_interval_select()));
    let seconds_in_hour = 60 * 60;
    return value === seconds_in_hour;
}

function __remove_error_from_hours_select() {
    $(this).removeClass('is-invalid');
}

function __add_error_for_hours_filtering_select(select) {
    select.addClass('is-invalid');
    add_error_alert('Обидва погодинних фільтри повинні бути обрані одночасно, або
жоден');
}

function _get_current_date_for_date_input() {
    let now = new Date();
    let month = (now.getMonth() + 1);
    let day = now.getDate();
    if (month < 10)
        month = "0" + month;
    if (day < 10)
        day = "0" + day;
    return now.getFullYear() + '-' + month + '-' + day;
}

function __get_hours_filtering_start_select() {
    return $('#id_hour_filter_start');
}

function __get_hours_filtering_end_select() {
    return $('#id_hour_filter_end');
}
```

```
function __get_hours_filtering_reset_button() {
    return $('#reset_hours_filters_btn');
}

function __get_hours_filtering_div() {
    return $('#id_hours_filtering');
}

function __get_facility_select() {
    return $('#id_facility_to_get_consumption_for');
}

function __get_selected_role_id() {
    return get_selected_option_for_select(__get_roles_select());
}

function __get_roles_select() {
    return $('#id_role');
}

function __get_control_form() {
    return $('#id_control_form');
}

function __get_set_view_mode_to_table_button() {
    return $('#set_table_button')
}

function __get_set_view_mode_to_chart_button() {
    return $('#set_chart_button')
}

function __get_period_start_input() {
    return $('#period_start')
}

function __get_period_end_input() {
    return $('#period_end')
}

function __get_include_forecast_checkbox() {
    return $('#id_include_forecast');
}

function __get_forecast_checkbox_value() {
    return __get_include_forecast_checkbox().is(':checked');
}

function __get_hour_filtering_options_buttons_container() {
    return $('#hour_filtering_option_buttons_container_id');
}
```

Файл: static/js/energy/box_with_sensors_creation/sensors.js

```
$(document).ready(function () {
    style_card();
    set_form_to_be_submitted_on_sensors_count_change();
})


function set_form_to_be_submitted_on_sensors_count_change() {
    __get_sensor_count_select().change(function () {
        add_input_to_indicate_that_sensor_count_was_changed();
```

```
        __get_sensors_form().submit();
    });
}

function add_input_to_indicate_that_sensor_count_was_changed() {
    __get_sensors_form().append($('<input                                type="hidden"
name="sensor_count_changed" value="true">'));
}


function style_card() {
    let body = $('#form-card .card-body');
    body.addClass('overflow-auto');
    body.css('max-height', '70vh');
}

function __get_sensor_count_select() {
    return $('#id_sensor_count');
}

function __get_sensors_form() {
    return $('#sensors_form');
}
```

Файл: static/js/energy/box_with_sensors_creation/box_sensor_set.js

```
$(document).ready(function () {
    __get_institution_select().change(
        update_facility_choices_for_chosen_institution_callback(
            __get_institution_select,
            update_facility_selects
        )
    );
    __get_institution_select().trigger('change');
});


function update_facility_selects(ids_and_labels) {
    __get_facilities_selects().forEach(
        (facility_select) => {
            update_select_with_options(facility_select, ids_and_labels);
        }
    );
}

function __get_institution_select() {
    return $('#id_institution');
}

function __get_facilities_selects() {
    let selects_count = $('.multiField').length;
    let selects = [];
    for (let i = 0; i < selects_count; i++) {
        selects.push($(`#id_box_sensor_set-${i}-facility`))
    }
    return selects;
}
```

Файл: static/js/common/search_form.js

```
$(document).ready(function () {
    let search_form = $('#searchForm');
    submit_on_enter(search_form);
```

```
    make_input_fit_placeholder();
    setup_clear_filter_button(search_form);
    fill_search_field_after_redirect_if_search_value_present();
})

function submit_on_enter(search_form) {
    search_form.keypress(function (e) {
        if (e.which === 13) {
            search_form.submit();
        }
    })
}

function fill_search_field_after_redirect_if_search_value_present() {
    let url = get_current_url();
    let search_value = url.searchParams.get('search_value');
    if (search_value) {
        __get_form_input().val(search_value);
    }
}

function make_input_fit_placeholder() {
    let input = __get_form_input();
    input.attr('size', input.attr('placeholder').length);
}

function setup_clear_filter_button(search_form) {
    $('#clearFilter').click(
        function () {
            __get_form_input().val('');
            search_form.submit();
        }
    )
}

function __get_form_input() {
    return $('#searchFormInput');
}
```

Файл: static/js/common/page_items.js

```
$(document).ready(function () {
    set_all_items_on_hover();
})

function set_all_items_on_hover() {
    $('#items > div').each(
        function () {
            bg_light_on_hover($(this));
        }
    )
}


function bg_light_on_hover(elem) {
    let bg_class = 'bg-light';
    elem.hover(
        function () {
            $(this).addClass(bg_class);
        },
        function () {
```

```
            $(this).removeClass(bg_class);
        },
    )
}
```

Файл: static/js/common/utils.js

```
function get_selected_option_for_select(select) {
    return select.find('option:selected').val();
}

function reverse_url(url_name) {
    // url name should be included in d-none block with id same as url name
    let url = $(`#${url_name}`).text();
    if (url) {
        return url;
    }
    throw new Error('url not found')
}

function update_select_with_options(select, options) {
    select.empty();
    let options_html = '';
    for (const id_label_object_index in (options)) {
        let value = options[id_label_object_index][0];
        let label = options[id_label_object_index][1];
        options_html += `<option value="${value}">${label}</option>`;
    }
    select.append(options_html);
}

function get_headers_for_ajax_object() {
    return {
        'X-CSRFToken': $.cookie('csrftoken')
    }
}

function add_on_click_redirects_text_after_div_first_label(div) {
    __add_muted_text_after_div_label(div, ON_CLICK_REDIRECTS_TEXT)
}

function __add_muted_text_after_div_label(div, text) {
    div.find('label').after(__generate_muted_p(text));
}

function clear_alerts() {
    __get_alerts_container().empty();
}

function __generate_muted_p(text) {
    return '<p class="fw-light text-muted m-0">' +
        `${text}` +
        '</p>'
}

function add_success_alert(message) {

__get_alerts_container_create_if_not_exists().append(__create_alert_div('success
', message));
}
```

```
function add_error_alert(message) {

__get_alerts_container_create_if_not_exists().append(__create_alert_div('danger'
, message));
}

function add_warning_alert(message) {

__get_alerts_container_create_if_not_exists().append(__create_alert_div('warning
', message));
}


function get_current_url() {
    return new URL(window.location.href);
}

function __create_alert_div(level, message) {
    return '' +
        `<div   class="alert   alert-${level}   alert-dismissible   fade   show"
role="alert">` +
        `${message}` +
        '<button  type="button"  class="btn-close"  data-bs-dismiss="alert"  aria-
label="Close">' +
        '</button>' +
        '</div>'
}

function __get_alerts_container_create_if_not_exists() {
    if (!__get_alerts_container().length) {
        $('header:first').after('' +
            '<div class="container">' +
            '   <div id="alerts">' +
            '   </div>' +
            '</div>');
    }
    return __get_alerts_container();
}

function __get_alerts_container() {
    return $("#alerts")
}

const DEFAULT_UNEXPECTED_ERROR_MESSAGE = '' +
    'Сталася непередбачена помилка, ' +
    'якщо після перезавантаження сторінки проблема не зникне, ' +
    'зверніться до адміністратора.'

const ON_CLICK_REDIRECTS_TEXT = 'При натисканні вас перенаправить на сторінку'

function redirect_to_object(url_name, id) {
    $.ajax({
        url: reverse_url(url_name),
        type: 'POST',
        dataType: 'json',
        headers: get_headers_for_ajax_object(),
        data: {id: id},
        success: (data) => {
            window.open(data.url, '_self');
        },
        error: (data) => {
            add_error_alert(DEFAULT_UNEXPECTED_ERROR_MESSAGE);
        }
```

```
    });
}

function redirect_to_selected_object(url_name) {
    return function () {
        let select = $(this);
        let id = get_selected_option_for_select(select);
        redirect_to_object(url_name, id);
    }
}

function              redirect_to_selected_object_with_get_id_callback(url_name,
get_id_callback) {
    return function () {
        let id = get_id_callback();
        redirect_to_object(url_name, id);
    }
}
```

Файл: static/js/common/form_in_center.js

```
$(document).ready(function () {
    add_spacing_between_form_group();
})

function add_spacing_between_form_group() {
    $('form > div.form-group').each(
        function () {
            $(this).addClass('pb-2');
        }
    )
}
```

Файл: static/js/users/edit_user.js

```
$(document).ready(function () {
    style_roles_select();
    add_redirect_on_roles_click();
    add_on_click_redirects_text_after_div_first_label(__get_roles_select_div());
})

function add_redirect_on_roles_click() {
    __get_roles_select().change(
        redirect_to_selected_object(
            'edit-user-role-pk-in-post'
        )
    );
}

function style_roles_select() {
    __get_roles_select().removeAttr('disabled');
}

function __get_roles_select_div() {
    return $('#div_id_roles');
}

function __get_roles_select() {
    return $('#id_roles');
}
```

Файл: static/js/users/edit_user_role.js

```
$(document).ready(function () {
    style_selects();

    __get_facility_role_has_access_to_select().click(
        redirect_to_selected_object(
            'edit-facility-pk-in-post'
        )
    );
    __get_user_role_owner_select().click(
        redirect_to_selected_object(
            'edit-user-pk-in-post'
        )
    )
})

function style_selects() {
    __get_user_role_owner_select().removeAttr('disabled');
    __get_facility_role_has_access_to_select().removeAttr('disabled');
    // removing arrows as select is used to provide id to redirect on click
    __get_user_role_owner_select().css('appearance', 'none');
    __get_facility_role_has_access_to_select().css('appearance', 'none');

add_on_click_redirects_text_after_div_first_label(__get_user_role_owner_div());

add_on_click_redirects_text_after_div_first_label(__get_facility_role_has_access
_to_div());
}


function __get_user_role_owner_select() {
    return $('#id_user_info');
}

function __get_user_role_owner_div() {
    return $('#div_id_user_info');
}

function __get_facility_role_has_access_to_select() {
    return $('#id_facility_has_access_to_info');
}

function __get_facility_role_has_access_to_div() {
    return $('#div_id_facility_has_access_to_info');
}
```

Файл: static/js/users/user_role_application_decision.js

```
$(document).ready(function () {
        // disable client-side decline button validation
        let button = $('button[type="submit"][value="decline"]');
        button.click(function () {
            let form = $(this).parents('form:first');
            form.attr('novalidate', '');
            form.submit();
        });
    }
)
```

Файл: users/models.py

```
from django.contrib.auth.base_user import AbstractBaseUser, BaseUserManager
from django.db import models
from django.urls import reverse, reverse_lazy
```

```
from django.utils.translation import gettext as _

from institutions.models import Facility
from users.utils import _full_name_validator


class UserManager(BaseUserManager):
    def create(self, full_name: str, email: str, password: str,
               is_admin: bool = False, is_active: bool = False) -> 'User':
        email = self.normalize_email(email)
        user  =  self.model(full_name=full_name,  email=email,  is_admin=is_admin,
is_active=is_active)
        user.set_password(password)
        user.save()
        return user


class User(AbstractBaseUser):
    full_name = models.CharField(
        _("повне ім'я"),
        max_length=150,
        validators=[_full_name_validator],
        null=False,
        blank=False
    )
    email = models.EmailField(
        _('електронна пошта'),
        unique=True,
        null=False,
        blank=False,
        error_messages={
            'unique': _('Користувач із такою поштою вже існує'),
        }
    )
    is_admin = models.BooleanField(
        _('чи є адміністратором'),
        default=False,
        null=False
    )
    is_active = models.BooleanField(
        _('чи підтвердив користувач пошту'),
        null=False,
        default=False
    )

    USERNAME_FIELD = 'email'

    objects = UserManager()

    def get_absolute_url(self):
        return reverse_lazy('edit-user', kwargs={'pk': self.pk})

    class Meta:
        db_table = 'users'
        verbose_name = _('Користувач')
        verbose_name_plural = _('Користувачі')

    def __str__(self):
        return _(self.full_name)


class UserRole(models.Model):
    user = models.ForeignKey(
```

```
        User,
        on_delete=models.CASCADE,
        null=False,
        blank=False,
        related_name='roles'
    )
    facility_has_access_to = models.ForeignKey(
        Facility,
        on_delete=models.CASCADE,
        null=False,
        blank=False,
        related_name='users_roles'
    )
    position_name = models.CharField(
        _('Назва посади'),
        max_length=255,
        null=False,
        blank=False
    )

    def get_absolute_url(self):
        return reverse_lazy('edit-user-role', kwargs={'pk': self.pk})

    class Meta:
        db_table = 'users_roles'
        verbose_name = _('Роль користувача')
        verbose_name_plural = _('Ролі користувача')

    def __str__(self):
        return      _(f'{self.position_name},      {self.facility_has_access_to},
{self.user}')


class UserRoleApplication(models.Model):
    user = models.ForeignKey(
        User,
        on_delete=models.CASCADE,
        null=False,
        blank=False,
        related_name='registration_requests'
    )
    institution = models.ForeignKey(
        Facility,
        on_delete=models.CASCADE,
        null=False,
        blank=False,
        related_name='+'
    )
    message = models.TextField(
        _('повідомлення від користувача'),
        blank=True,
        null=True
    )

    class Meta:
        db_table = 'users_roles_applications'
        verbose_name = _('Запит на отримання ролі користувачем')
        verbose_name_plural = _('Запити на отримання ролі користувачів')

    def __str__(self):
        return    _(f'Запит    на    реєстрацію    від    {self.user.full_name}    в
{self.institution}')
```

```
    def get_absolute_url(self):
        return reverse('user-role-application-decision', kwargs={'pk': self.pk})
```

Файл: users/__init__.py

Файл: users/apps.py

```
from django.apps import AppConfig


class UsersConfig(AppConfig):
    default_auto_field = 'django.db.models.BigAutoField'
    name = 'users'
```

Файл: users/backends.py

```
from django.contrib.auth import backends


class AllowTryAuthenticateInactive(backends.ModelBackend):
    """This backend allows every registered user to try to be authenticated
     in order to display custom login message for users who did not confirm
     their email address (is_active=False)"""

    def user_can_authenticate(self, user):
        return True
```

Файл: users/context_processors.py

```
from django.http import HttpRequest

from users.logic.simple import get_applications_from_users_who_confirmed_email
from utils.common import is_admin_non_authenticated_safe


def user_roles_applications_to_review_count(request: HttpRequest) -> dict:
    if is_admin_non_authenticated_safe(request.user):
        count = get_applications_from_users_who_confirmed_email().count()
        return {
            'users_roles_applications_count': count
        }
    return {}
```

Файл: users/utils.py

```
import re

from django.core.exceptions import ValidationError
from django.utils.translation import gettext_lazy as _


def _full_name_validator(full_name: str):
    valid_full_name_part_pattern = r'''[А-ЩЬЮЯҐЄІЇа-щьюяґєіїʼ`’’-]{2,40}'''   #
noqa
    full_name_pattern = '^' + r'\s'.join([valid_full_name_part_pattern] * 3) +
'$'
    if not bool(re.match(full_name_pattern, full_name)):
```

```
        raise ValidationError(_("Некоректне ім'я"))
```

Файл: users/urls.py

```
from django.urls import include, path

from users import views
from users.models import User, UserRole
from utils.common import redirect_to_object_pk_in_post

urlpatterns = [
    path('login/', views.LoginView.as_view()),
    path(
        'registration/',
        views.CreateUserRegistrationRequestView.as_view(),
        name='register'
    ),
    path(
        'successfully-created-registration-request/',
        views.successfully_created_registration_request,
        name='successfully-created-registration-request'
    ),
    path(
        'confirm-email/<hashed_int:user_id>/<hashed_str:user_email>/',
        views.confirm_email,
        name='confirm-email'
    ),
    path(
        'user-role-application/',
        views.UserRoleApplicationsListView.as_view(),
        name='user-role-applications'
    ),
    path(
        'user-role-application-decision/<hashed_int:pk>/',
        views.UserRoleApplicationDecisionView.as_view(),
        name='user-role-application-decision'
    ),
    path(
        'users-list/',
        views.UserListView.as_view(),
        name='users-list'
    ),
    path(
        'edit-user/<hashed_int:pk>/',
        views.simple.UserView.as_view(),
        name='edit-user'
    ),
    path(
        'user-role-list/',
        views.UserRoleListView.as_view(),
        name='user-role-list'
    ),
    path(
        'edit-user-role/<hashed_int:pk>/',
        views.UserRoleView.as_view(),
        name='edit-user-role'
    ),
    path(
        'my-profile/',
        views.ProfileView.as_view(),
        name='my-profile'
    ),
```

```
    path(
        'role-application/',
        views.RoleApplicationView.as_view(),
        name='role-application'
    ),
    path('', include('django.contrib.auth.urls')),
]

ajax_urls = [
    path(
        'edit-user-role-pk-in-post',
        redirect_to_object_pk_in_post(UserRole, 'edit-user-role'),
        name='edit-user-role-pk-in-post'
    ),
    path(
        'edit-user-pk-in-post',
        redirect_to_object_pk_in_post(User, 'edit-user'),
        name='edit-user-pk-in-post'
    )
]

urlpatterns += ajax_urls
```

Файл:
users/migrations/0002_rename_object_has_access_to_userrole_facility_has_access_to.py

```
# Generated by Django 4.1.1 on 2022-10-02 08:47

from django.db import migrations


class Migration(migrations.Migration):

    dependencies = [
        ('users', '0001_initial'),
    ]

    operations = [
        migrations.RenameField(
            model_name='userrole',
            old_name='object_has_access_to',
            new_name='facility_has_access_to',
        ),
    ]
```

Файл: users/migrations/0005_alter_userrole_position_name.py

```
# Generated by Django 4.1.3 on 2022-12-19 11:13

from django.db import migrations, models


class Migration(migrations.Migration):
    dependencies = [
        ('users', '0004_rename_position_userrole_position_name_and_more'),
    ]

    operations = [
        migrations.AlterField(
            model_name='userrole',
```

```
            name='position_name',
            field=models.CharField(max_length=255, verbose_name='Назва посади'),
        ),
    ]
```

Файл: users/migrations/__init__.py

Файл: users/migrations/0004_rename_position_userrole_position_name_and_more.py

```
# Generated by Django 4.1.1 on 2022-10-03 08:19

from django.db import migrations, models
import django.db.models.deletion


class Migration(migrations.Migration):

    dependencies = [
        ('institutions', '0003_rename_facilities_facility'),
        ('users', '0003_alter_userrole_facility_has_access_to'),
    ]

    operations = [
        migrations.RenameField(
            model_name='userrole',
            old_name='position',
            new_name='position_name',
        ),
        migrations.AlterField(
            model_name='userrole',
            name='facility_has_access_to',
            field=models.ForeignKey(on_delete=django.db.models.deletion.CASCADE,
related_name='users_roles', to='institutions.facility'),
        ),
    ]
```

Файл: users/migrations/0001_initial.py

```
# Generated by Django 4.0.5 on 2022-07-23 10:21

import django.db.models.deletion
from django.conf import settings
from django.db import migrations, models

import users.utils


class Migration(migrations.Migration):

    initial = True

    dependencies = [
        ('institutions', '0001_initial'),
    ]

    operations = [
        migrations.CreateModel(
            name='User',
            fields=[
```

```
                ('id', models.BigAutoField(auto_created=True, primary_key=True,
serialize=False, verbose_name='ID')),
                ('password',                    models.CharField(max_length=128,
verbose_name='password')),
                ('last_login',    models.DateTimeField(blank=True,    null=True,
verbose_name='last login')),
                ('full_name',                   models.CharField(max_length=150,
validators=[users.utils._full_name_validator], verbose_name="повне ім'я")),
                ('email',           models.EmailField(error_messages={'unique':
'Користувач із такою поштою вже існує'}, max_length=254, unique=True,
verbose_name='електронна пошта')),
                ('is_admin', models.BooleanField(default=False, verbose_name='чи
є адміністратором')),
                ('is_active',                models.BooleanField(default=False,
verbose_name='чи підтвердив користувач пошту')),
            ],
            options={
                'verbose_name': 'Користувач',
                'verbose_name_plural': 'Користувачі',
                'db_table': 'users',
            },
        ),
        migrations.CreateModel(
            name='UserRoleApplication',
            fields=[
                ('id', models.BigAutoField(auto_created=True, primary_key=True,
serialize=False, verbose_name='ID')),
                ('message',       models.TextField(blank=True,       null=True,
verbose_name='повідомлення від користувача')),
                ('institution',
models.ForeignKey(on_delete=django.db.models.deletion.CASCADE, related_name='+',
to='institutions.object')),
                ('user',
models.ForeignKey(on_delete=django.db.models.deletion.CASCADE,
related_name='registration_requests', to=settings.AUTH_USER_MODEL)),
            ],
            options={
                'verbose_name': 'Запит на отримання ролі користувачем',
                'verbose_name_plural': 'Запити на отримання ролі користувачів',
                'db_table': 'users_roles_applications',
            },
        ),
        migrations.CreateModel(
            name='UserRole',
            fields=[
                ('id', models.BigAutoField(auto_created=True, primary_key=True,
serialize=False, verbose_name='ID')),
                ('position',                    models.CharField(max_length=255,
verbose_name='посада')),
                ('object_has_access_to',
models.ForeignKey(on_delete=django.db.models.deletion.CASCADE,
related_name='users_roles', to='institutions.object')),
                ('user',
models.ForeignKey(on_delete=django.db.models.deletion.CASCADE,
related_name='roles', to=settings.AUTH_USER_MODEL)),
            ],
            options={
                'verbose_name': 'Роль користувача',
                'verbose_name_plural': 'Ролі користувача',
                'db_table': 'users_roles',
            },
        ),
    ]
```

Файл: users/migrations/0003_alter_userrole_facility_has_access_to.py

```
# Generated by Django 4.1.1 on 2022-10-02 08:49

from django.db import migrations, models
import django.db.models.deletion


class Migration(migrations.Migration):

    dependencies = [
        ('institutions', '0003_rename_facilities_facility'),
        ('users',
'0002_rename_object_has_access_to_userrole_facility_has_access_to'),
    ]

    operations = [
        migrations.AlterField(
            model_name='userrole',
            name='facility_has_access_to',
            field=models.ForeignKey(db_column='facility_has_access_to_id',
on_delete=django.db.models.deletion.CASCADE,         related_name='users_roles',
to='institutions.facility'),
        ),
    ]
```

Файл: users/forms/user_role_application_request_decision_form.py

```
from django import forms
from django.utils.decorators import classonlymethod
from django.utils.translation import gettext_lazy as _

from institutions.logic import (
    get_all_descendants_of_facility_with_self,
get_facility_name_space_padded_according_to_nesting,
)
from institutions.models import Facility
from users.models import UserRole, UserRoleApplication
from utils.forms import (
    create_danger_button, create_primary_button, CrispyModelForm,
    SecureModelChoiceField,
    SelectWithFormControlClass,
)


class UserRoleApplicationRequestsDecisionForm(CrispyModelForm):
    """This form must be created from a user role application"""
    # qs set to all to allow any facility to be chosen
    # correct qs for institution is set in custom creating method
    facility_has_access_to = SecureModelChoiceField(
        label=_("Оберіть об'єкт до якого користувач матиме доступ"),
        queryset=Facility.objects.all(),
        required=True,
        empty_label=None,

label_from_instance_function=get_facility_name_space_padded_according_to_nesting
,
        widget=SelectWithFormControlClass({'size': 6})
    )
    position_name = forms.CharField(
```

```
        label=_('Уведіть назву позиції'),
        max_length=255,
        required=True,
    )
    message_for_user = forms.CharField(
        label=_('Залиште опціональне повідомлення для користувача'),
        max_length=255,
        required=False,
        widget=forms.Textarea({'rows': 2})
    )
    # info readonly fields
    user_with_email = forms.CharField(
        label=_('Користувач та пошта'),
        max_length=255,
        widget=forms.TextInput({'readonly': 'readonly'}),
        required=False
    )
    institution_verbose = forms.CharField(
        label=_('Установа'),
        max_length=255,
        widget=forms.TextInput({'readonly': 'readonly'}),
        required=False
    )
    message_from_user = forms.CharField(
        label=_('Повідомлення від користувача'),
        max_length=255,
        widget=forms.Textarea({'rows': 2, 'readonly': 'readonly'}),
        required=False
    )

    class Meta:
        model = UserRole
        fields = ('position_name', 'facility_has_access_to')
        info_readonly_fields = (
            'user_with_email',
            'institution_verbose',
            'message_from_user',
        )
        fields_order = info_readonly_fields + fields + ('message_for_user',)
        buttons = (
            create_primary_button(
                _('Підтвердити'),
                name='decision',
                value='accept'
            ),
            create_danger_button(
                _('Відхилити'),
                name='decision',
                value='decline'
            )
        )

    @classonlymethod
    def          create_from_role_application(cls,          application_request:
UserRoleApplication):
        obj = cls()
        obj.__additionally_setup_form(application_request)
        return obj

    # pylint bug so have to be disabled pylint: disable-next=W0238
    def            __additionally_setup_form(self,           application_request:
UserRoleApplication):
```

```
self.__set_correct_queryset_for_facility_has_access_to(application_request.insti
tution)
        self.__populate_info_fields(application_request)

    def   __set_correct_queryset_for_facility_has_access_to(self,   institution:
Facility):
        self.fields['facility_has_access_to'].queryset                          =
get_all_descendants_of_facility_with_self(
            institution
        )

    def __populate_info_fields(self, application_request: UserRoleApplication):
        self.fields['institution_verbose'].initial                              =
_(str(application_request.institution))
        self.fields['user_with_email'].initial = \
            _(f'{application_request.user} {application_request.user.email}')
        self.fields['message_from_user'].initial                                =
_(application_request.message)

    def get_message_for_user(self) -> str:
        return self.data['message_for_user']
```

Файл: users/forms/__init__.py

```
from .simple import (
    LoginForm, NewUserForm, ProfileForm, UserForm, UserRoleApplicationForm,
    UserRoleApplicationFormForRegistration, UserRoleForm,
)
from             .user_role_application_request_decision_form             import
UserRoleApplicationRequestsDecisionForm
```

Файл: users/forms/simple.py

```
from django import forms
from django.contrib.auth import forms as auth_forms
from django.db.models import Model
from django.forms import CharField, Form
from django.utils.translation import gettext_lazy as _

from institutions.models import Facility
from users.models import User, UserRole, UserRoleApplication
from utils.common.object_to_queryset import object_to_queryset
from utils.forms import (
    create_primary_button,           CrispyFormMixin,           CrispyModelForm,
SecureModelChoiceField,
    SelectWithFormControlClass, UPDATE_DELETE_BUTTONS_SET,
)
from utils.views import AdditionalSetupRequiredFormMixin


class LoginForm(CrispyFormMixin, auth_forms.AuthenticationForm):
    remember_me = forms.BooleanField(
        label=_("Запам'ятати мене на два тижні"),
        required=False
    )

    error_messages = {
        'invalid_login':  _('Уведіть правильну електронну пошту та пароль або
зареєструйтесь'),
        'inactive':      _('Електронна пошта ще не підтверджена'),
```

```
    }

    class Meta:
        buttons = (create_primary_button(_('Увійти')),)


class NewUserForm(CrispyFormMixin, auth_forms.UserCreationForm):
    class Meta:
        model = User
        fields = ('full_name', 'email')


class UserRoleApplicationForm(CrispyModelForm):
    institution = SecureModelChoiceField(
        label=_('Оберіть заклад'),
        queryset=Facility.objects.get_institutions(),
        widget=SelectWithFormControlClass({'size': 4}),
        required=True
    )
    message = forms.CharField(
        label=_('Залиште опціональне повідомлення'),
        max_length=200,
        required=False,
        widget=forms.Textarea({'rows': 2})
    )

    class Meta:
        model = UserRoleApplication
        fields = ('institution', 'message')
        buttons = (create_primary_button(_('Відправити заявку на роль')),)

    def set_application_request_user(self, user: User):
        self.instance.user = user


class UserRoleApplicationFormForRegistration(UserRoleApplicationForm):
    class Meta(UserRoleApplicationForm.Meta):
        # buttons are set here to be displayed at the bottom of registration form
        buttons = (create_primary_button(_('Відправити заявку на реєстрацію')),)


class UserForm(CrispyModelForm, AdditionalSetupRequiredFormMixin):
    # info only field, qs is filled in custom method
    roles = SecureModelChoiceField(
        queryset=UserRole.objects.none(),
        label=_("Ролі"),
        empty_label=None,
        required=False,
        disabled=True,
        widget=SelectWithFormControlClass(attrs={'size': 4}),
        label_from_instance_function=lambda user_role:
        _(f"{user_role.position_name},                                    об'єкт
{user_role.facility_has_access_to.name}")
    )
    full_name = forms.CharField(
        label=_("Повне ім'я"),
        required=False,
        disabled=True,
    )
    email = forms.CharField(
        label=_('Електронна пошта'),
        required=False,
        disabled=True,
```

```
    )

    class Meta:
        model = User
        fields = ('full_name', 'email', 'is_admin')
        buttons = UPDATE_DELETE_BUTTONS_SET

    def additionally_setup(self, obj: Model):
        # noinspection PyTypeChecker
        user: User = obj
        self.fields['roles'].queryset = user.roles


class UserRoleForm(CrispyModelForm, AdditionalSetupRequiredFormMixin):
    user_info = SecureModelChoiceField(
        queryset=User.objects.none(),
        label=_("Користувач"),
        disabled=True,
        empty_label=None
    )
    facility_has_access_to_info = SecureModelChoiceField(
        queryset=Facility.objects.none(),
        label=_("Об'єкт"),
        disabled=True,
        empty_label=None,
    )

    class Meta:
        model = UserRole
        fields = ('position_name',)
        buttons = UPDATE_DELETE_BUTTONS_SET

    def additionally_setup(self, obj: Model):
        # noinspection PyTypeChecker
        role: UserRole = obj
        self.fields['user_info'].queryset = object_to_queryset(role.user)
        self.fields['facility_has_access_to_info'].queryset = \
            object_to_queryset(role.facility_has_access_to)


class ProfileForm(Form):
    email = CharField(
        label=_('Моя пошта'),
        disabled=True
    )
    full_name = CharField(
        label=_("Повне ім'я"),
        disabled=True
    )
    # must be set dynamically for user
    roles = SecureModelChoiceField(
        label=_('Мої ролі'),
        queryset=UserRole.objects.none(),
        widget=SelectWithFormControlClass({'size': 6}),
        empty_label=None
    )

    def __init__(self, *args, user: User, **kwargs):
        super().__init__(*args, **kwargs)
        self.__init_dynamic_fields(user)

    def __init_dynamic_fields(self, user: User):
        self.fields['roles'].queryset = user.roles.all()
```

```
        self.fields['email'].initial = user.email
        self.fields['full_name'].initial = user.full_name
```

Файл: users/tests/__init__.py

Файл: users/tests/factories.py

```
import random

import factory
from django.contrib.auth import get_user_model
from django.test import Client
from factory import django

from institutions.tests.factories import InstitutionFactory
from users.models import UserRoleApplication

User = get_user_model()


class UserFactory(django.DjangoModelFactory):
    password = factory.Sequence(lambda n: f'generated_password{n}')
    email = factory.Sequence(lambda n: f'generated_email{n}@email.com')

    @factory.lazy_attribute
    def full_name(self):
        first_names = [
            'Яснолик-Гузь',
            'Шарль',
            'Златоус',
            'Милована',
        ]
        second_names = [
            'Оробко',
            'Білоножко',
            'Даценко',
            'Гасенко',
        ]
        patronims = [
            'Адріанович',
            'Романович',
            'Мстиславович',
            "В'ячеславович"
        ]
        return " ".join([
            random.choice(first_names),
            random.choice(second_names),
            random.choice(patronims)
        ])

    class Meta:
        model = User


class UserRoleApplicationFactory(django.DjangoModelFactory):
    class Meta:
        model = UserRoleApplication

    user = factory.SubFactory(UserFactory, is_active=True)
    institution = factory.SubFactory(InstitutionFactory)
```

```
    message = factory.Faker('text', max_nb_chars=50)


def create_admin_client() -> Client:
    client = Client()
    client.force_login(UserFactory(is_admin=True))
    return client
```

Файл: users/tests/test_user_role_application_decision.py

```
from typing import TypedDict

from django.core.exceptions import ObjectDoesNotExist
from django.http import HttpResponse
from django.test import TestCase
from django.urls import reverse, reverse_lazy
from faker import Faker

from institutions.models import Facility
from users.models import UserRole, UserRoleApplication
from users.tests.factories import create_admin_client, UserRoleApplicationFactory
from users.views.user_role_application_decision import DECISIONS
from utils.common import hash_id


class UserRoleApplicationDetailTest(TestCase):
    _user_role_application_decision_data_dict = TypedDict(
        '_user_role_application_decision_data_dict',
        {
            'decision':               str,
            'message_for_user':       str,
            'position_name':          str,
            'user':                   str,
            'institution':            str,
            'facility_has_access_to': str
        }
    )

    def setUp(self):
        self.admin_client = create_admin_client()
        self.user_role_application = UserRoleApplicationFactory()
        self.facility_has_access_to = self.user_role_application.institution
        fake = Faker()
        self.position_to_grant_user: str = fake.text(max_nb_chars=20)

    def test_accepting_application(self):
        resp = self.send_decision(
            decision=DECISIONS.ACCEPT,
            facility_has_access_to=self.user_role_application.institution,
            position_name=self.position_to_grant_user,
        )
        self.assertEqual(
            200,
            resp.status_code,
            'error while accepting user application'
        )
        self.assertRedirects(
            resp,
            reverse_lazy('user-role-applications'),
        )
        try:
            UserRole.objects.get(
```

```
                user=self.user_role_application.user,
                facility_has_access_to=self.user_role_application.institution,
                position_name=self.position_to_grant_user
            )
        except ObjectDoesNotExist:
            assert False, 'user role is not created even though response status
code is 200'
        self.__check_user_application_is_deleted()

    def test_declining_application(self):
        resp = self.send_decision(
            DECISIONS.DECLINE
        )
        self.assertEqual(
            200,
            resp.status_code,
            'error while declining user application'
        )
        self.assertRedirects(
            resp,
            reverse_lazy('user-role-applications'),
        )
        self.__check_user_application_is_deleted()

    def send_decision(
            self, decision: DECISIONS,
            facility_has_access_to: Facility = None, position_name: str = None) -
> HttpResponse:
        return self.admin_client.post(
            reverse(
                'user-role-application-decision',
                kwargs={'pk': self.user_role_application.pk}
            ),
            {
                **self.__complete_data(
                    decision,
                    position_name,
                    facility_has_access_to
                )
            },
            follow=True
        )

    def __complete_data(
            self, decision: DECISIONS,
            position_name: str | None, facility_has_access_to: Facility | None):
        complete_data = self.__get_form_data()
        # due to using enum pycharm thinks that .value is a method,
        # but at runtime it is a property
        complete_data['decision'] = decision.value  # noqa
        # when declining application neither position nor facility_has_access_to
        # have to be filled in form, so they are just ''
        complete_data['position_name'] = position_name if position_name else ''
        hashed_id_or_empty_string = (
            hash_id(facility_has_access_to)
            if facility_has_access_to
            else ''
        )
        complete_data['facility_has_access_to'] = hashed_id_or_empty_string
        return complete_data

    def __get_form_data(self) -> _user_role_application_decision_data_dict:
        # here we set data as if it was pre-populated by form
```

```
        user_id_hashed = hash_id(self.user_role_application.user)
        data = {
            'message_for_user': '',
            'user':             user_id_hashed
        }
        data = self._user_role_application_decision_data_dict(**data)
        return data

    def __check_user_application_is_deleted(self):
        self.assertFalse(
            UserRoleApplication.objects.filter(
                user=self.user_role_application.user,
                institution=self.user_role_application.institution,
            ).exists(),
            'user role application is not deleted'
        )
```

Файл: users/tests/test_user_registration.py

```
from typing import TypedDict

from django.contrib.auth import get_user_model
from django.core.exceptions import ObjectDoesNotExist
from django.http import HttpResponse
from django.test import Client, RequestFactory, TestCase
from django.urls import reverse

from institutions.models import Facility
from institutions.tests.factories import InstitutionFactory
from users.logic.user_registration_controller import _EmailConfirmationController
# noqa
from users.models import UserRoleApplication
from users.tests.factories import UserFactory
from utils.common import hash_id

User = get_user_model()


class UserRegistrationTest(TestCase):
    _user_registration_data_dict = TypedDict(
        '_user_registration_data_dict',
        {
            'full_name':   str,
            'email':       str,
            'password1':   str,
            'password2':   str,
            'institution': str,
            'message':     str,
        }
    )

    def setUp(self):
        self.client = Client()
        self.raw_password = '%tv{,,E)36'
        self.message = 'some message'
        self.user: User = UserFactory.build(password=self.raw_password)
        self.institution = InstitutionFactory()

    def test_with_correct_data_set(self):
        resp = self.send_correct_registration_data()
        self.assertEqual(
            resp.status_code,
```

```
            200,
            'form validation failed with correct data'
        )
        try:
            user = User.objects.get(
                full_name=self.user.full_name,
                email=self.user.email
            )
        except ObjectDoesNotExist:
            assert False, 'user is not created'
        try:
            UserRoleApplication.objects.get(
                user=user
            )
        except ObjectDoesNotExist:
            assert False, 'user registration request is not created'

    def test_user_and_role_application_is_created_correctly(self):
        self.send_correct_registration_data()
        user = User.objects.get(
            full_name=self.user.full_name,
            email=self.user.email
        )
        self.assertFalse(
            user.is_active,
            'user must me inactive on registration (email unconfirmed)'
        )
        user_role_application = UserRoleApplication.objects.get(
            user=user
        )
        self.assertEqual(
            user_role_application.institution,
            self.institution,
            'registration request is set for wrong institution'
        )

    def test_email_confirmation(self):
        self.send_correct_registration_data()
        # here we mock request just to use it`s is_secure and get_host methods to
generate link
        request = RequestFactory().get('')
        # here we get just first user, as it should be created and be the only
one
        user = User.objects.all().first()
        _ = _EmailConfirmationController
        # pylint: disable-next=W0212
        link_for_user  =  _._EmailConfirmationController__generate_link_for_user(
# noqa
            user,
            request
        )
        resp = self.client.get(
            link_for_user
        )
        self.assertTemplateUsed(
            resp,
            'registration/successfully_confirmed_email.html',
            'needed template is not used'
        )
        user.refresh_from_db()
        self.assertTrue(
            user.is_active,
            'is_active is not set when user successfully confirmed email'
```

```
        )

    def send_correct_registration_data(self) -> HttpResponse:
        return self.send_registration_request(
            self.user,
            self.institution,
            self.raw_password
        )

    def __complete_data(self, user: User, institution: Facility, raw_password:
str):
        data = self.__get_form_data()
        data['full_name'] = user.full_name
        data['email'] = user.email
        data['password1'] = raw_password
        data['password2'] = raw_password
        data['institution'] = hash_id(institution)
        return data

    def __get_form_data(self) -> _user_registration_data_dict:
        # when field is not filled it is sent with empy data anyway
        data = {
            'message': ''
        }
        data = self._user_registration_data_dict(
            **data
        )
        return data

    def send_registration_request(
            self, user: User, institution: Facility, raw_password: str) ->
HttpResponse:
        return self.client.post(
            reverse('register'),
            {
                **self.__complete_data(user, institution, raw_password)
            },
            follow=True
        )
```

Файл: users/tests/test_models.py

```
from django.contrib.auth import get_user_model
from django.core.exceptions import ValidationError
from django.test import TestCase

User = get_user_model()


class UserTest(TestCase):
    def setUp(self):
        self.valid_email = 'valid@email.com'
        self.some_password = 'some_password'
        self.valid_full_name = "Тестове Валідне Ім'я"

    def test_creating_user(self):
        email = self.valid_email
        raw_password = self.some_password
        name = self.valid_full_name
        user    =    User.objects.create(email=email,    password=raw_password,
full_name=name)
        self.assertEqual(email, user.email, 'user email was set incorrectly')
```

```
        self.assertTrue(user.check_password(raw_password), 'user password was set
incorrectly')
        self.assertEqual(name,  user.full_name,  'user  full  name  was  set
incorrectly')
        self.assertEqual(email,  user.get_username(),  'email  is  not  used  as
username')

    def test_creating_with_incorrect_fields_sets(self):
        with self.assertRaises(TypeError, msg='User was created with wrong fields
set'):
            User.objects.create()
        with self.assertRaises(TypeError, msg='User was created with wrong fields
set'):
            User.objects.create(full_name=self.valid_full_name)
        with self.assertRaises(TypeError, msg='User was created with wrong fields
set'):
            User.objects.create(email=self.valid_email)
        with self.assertRaises(TypeError, msg='User was created with wrong fields
set'):
            User.objects.create(full_name=self.valid_full_name,
email=self.valid_email)

    def test_full_name_check(self):
        def test_for_names_list(names: list[str], should_throw: bool):
            for name in names:
                try:
                    User.objects.create(
                        full_name=name,
                        email=self.valid_email,
                        password=self.some_password
                    ).delete()
                    # instantly deleting to avoid email duplication exception
                except ValidationError:
                    if not should_throw:
                        assert False, 'validating full name throws exception for
a valid name'

        correct_names_list = [
            """П'ятницький Хотибор Денисович""",
            """Никоненко Лаврентій-Арсен Полянович""",
            """Слюсар Наслав Денисович""",
        ]
        incorrect_names_list = [
            'Некоректне',
            "Некоректне Ім'я",
            'Incorrect Name Example',
            'Некорректный Пример Имени',
        ]
        test_for_names_list(correct_names_list, should_throw=False)
        test_for_names_list(incorrect_names_list, should_throw=True)

    def test_creating_admin(self):
        admin = User.objects.create(
            full_name=self.valid_full_name,
            email=self.valid_email,
            password=self.some_password,
            is_admin=True
        )
        self.assertTrue(admin.is_admin)
```

Файл: users/logic/__init__.py

```
from .simple import (
    check_role_belongs_to_user,              check_role_has_access_for_facility,
check_user_has_no_roles,
    remember_user_for_two_week,
)
from .user_registration_controller import UserRegistrationController
from               .user_role_application_review_controller               import
UserRoleApplicationReviewController
```

Файл: users/logic/user_role_application_review_controller.py

```
from django.contrib import messages
from django.db import transaction
from django.http import HttpRequest
from django.utils.decorators import method_decorator
from django.utils.safestring import mark_safe
from django.utils.translation import gettext_lazy as _

from users.forms import UserRoleApplicationRequestsDecisionForm
from users.models import UserRole, UserRoleApplication
from utils.common.email import try_send_email_add_warning_if_failed


class UserRoleApplicationReviewController:
    def                  __init__(self,                 application_decision_form:
UserRoleApplicationRequestsDecisionForm,
                 application:    UserRoleApplication,    request:    HttpRequest,
is_approved: bool):
        self.application = application
        self.application_decision_form = application_decision_form
        self.request = request
        self.is_approved = is_approved

    @method_decorator(transaction.atomic)
    def process_depending_on_decision(self):
        if self.is_approved:
            self.__create_user_role()
        self.__delete_user_role_application()
        self.__notify_on_role_application_decision()
        self.__add_message_for_decision()

    def __delete_user_role_application(self):
        self.application.delete()

    def __add_message_for_decision(self):
        if self.is_approved:
            message = _('Успішно додано роль для користувача')
        else:
            message = _('Запит користувача було успішно відхилено')
        messages.success(self.request, message)

    def __create_user_role(self):
        user_role_without_user:                    UserRole                    =
self.application_decision_form.instance
        user_role_without_user.user = self.application.user
        user_role_without_user.save()

    def __notify_on_role_application_decision(self):
        if self.is_approved:
            subject = 'Ваш запит на роль підтверджено'
            message = 'Нова роль на energokodros вже доступна вам на сайті ' \
                      'і ви можете починати використовувати її.'
```

```
        else:
            subject = 'Ваш запит на роль відхилено'
            message = 'На жаль, адміністратор не схвалив ваш запит.'
        if                          message_from_admin                          :=
self.application_decision_form.get_message_for_user():
            message  +=  mark_safe(f'<br>Повідомлення  від  адміністратора:<br>
{message_from_admin}')

        try_send_email_add_warning_if_failed(
            self.request,
            self.application.user.email,
            subject,
            message
        )
```

Файл: users/logic/user_registration_controller.py

```
from django.http import HttpRequest
from django.shortcuts import get_object_or_404
from django.template.loader import render_to_string
from django.urls import reverse

from users.forms import NewUserForm, UserRoleApplicationFormForRegistration
from users.models import User
from utils.common.email import send_html_email


class UserRegistrationController:
    __user_form: NewUserForm
    __role_application_without_user_form: UserRoleApplicationFormForRegistration
    __user: User

    def __init__(self, user_form: NewUserForm,
                 role_application_without_user_form:
UserRoleApplicationFormForRegistration):
        self.__user_form = user_form
        self.__role_application_without_user_form                                  =
role_application_without_user_form

    def save_user_along_with_registration_request_return_user(self) -> User:
        self.__user = self.__user_form.save(commit=False)
        self.__user.save()

self.__role_application_without_user_form.set_application_request_user(self.__us
er)
        self.__role_application_without_user_form.save()
        return self.__user

    def send_email_confirmation_message(self, request: HttpRequest):
        return _EmailConfirmationController.send_email_confirmation_message(
            request, self.__user
        )

    @staticmethod
    def confirm_email_if_user_exists(user_id: int, email: str):

_EmailConfirmationController.confirm_email_if_user_exists_or_404(user_id, email)


class _EmailConfirmationController:
    @classmethod
```

```
    def send_email_confirmation_message(cls, request: HttpRequest, user: User) -
> bool:
        ctx = cls.__generate_context(request, user)
        html = render_to_string(
            template_name='email/email_confirmation.html',
            context=ctx
        )
        return send_html_email(user.email, 'Підтвердження реєстрації', html)

    @staticmethod
    def confirm_email_if_user_exists_or_404(user_id: int, email: str):
        user = get_object_or_404(User, pk=user_id, email=email)
        user.is_active = True
        user.save()

    @classmethod
    def __generate_context(cls, request: HttpRequest, user: User) -> dict[str,
str]:
        return {
            'link': cls.__generate_link_for_user(user, request),
        }

    @classmethod
    def __generate_link_for_user(cls, user: User, request: HttpRequest) -> str:
        protocol = 'https' if request.is_secure() else 'http'
        link_without_host                                                    =
cls.__generate_path_for_email_confirmation_for_user(user)
        return protocol + '://' + request.get_host() + link_without_host

    @classmethod
    def __generate_path_for_email_confirmation_for_user(cls, user: User) -> str:
        return reverse(
            'confirm-email',
            kwargs={
                'user_id':    user.pk,
                'user_email': user.email,
            }
        )
```

Файл: users/logic/simple.py

```
from datetime import timedelta

from django.core.exceptions import PermissionDenied
from django.db.models import QuerySet
from django.http import HttpRequest

from institutions.models import Facility
from users.models import User, UserRole, UserRoleApplication


def check_role_belongs_to_user(user: User, user_role: UserRole):
    if user_role.user != user:
        raise PermissionDenied
    return True


def check_role_has_access_for_facility(user_role: UserRole, facility: Facility):
    if user_role.facility_has_access_to == facility:
        return
    if facility.is_descendant_of(user_role.facility_has_access_to):
        return
```

```
    raise PermissionDenied


def remember_user_for_two_week(request: HttpRequest):
    __remember_user_for_timedelta(request, timedelta(weeks=2))


def check_user_has_no_roles(user: User) -> bool:
    return not user.roles.exists()


def get_applications_from_users_who_confirmed_email() -> QuerySet:
    return UserRoleApplication.objects.filter(user__is_active=True)


def get_users_with_confirmed_email() -> QuerySet:
    return User.objects.filter(is_active=True)


def __remember_user_for_timedelta(request: HttpRequest, td: timedelta):
    request.session.set_expiry(td)
```

Файл: users/views/list_views.py

```
from users.logic.simple import (
    get_applications_from_users_who_confirmed_email,
    get_users_with_confirmed_email,
)
from users.models import UserRole
from utils.common import admin_rights_and_login_required
from utils.views import ListViewWithFiltering


@admin_rights_and_login_required
class UserRoleApplicationsListView(ListViewWithFiltering):
    queryset = get_applications_from_users_who_confirmed_email()
    filter_fields = ('user__full_name', 'user__email', 'institution__name')
    template_name = 'users/users_roles_applications.html'


@admin_rights_and_login_required
class UserListView(ListViewWithFiltering):
    queryset = get_users_with_confirmed_email()
    filter_fields = ('full_name', 'email')
    template_name = 'users/users_list.html'


@admin_rights_and_login_required
class UserRoleListView(ListViewWithFiltering):
    queryset = UserRole.objects.all()
    filter_fields    =    ('facility_has_access_to__name',    'user__full_name',
'position_name')
    template_name = 'users/users_roles_list.html'
```

Файл: users/views/create_user_registration_request.py

```
from django.contrib import messages
from django.db import transaction
from django.shortcuts import redirect
from django.urls import reverse
from django.utils.decorators import method_decorator
```

```
from django.utils.translation import gettext_lazy as _
from django.views.generic import CreateView

from users.forms import NewUserForm, UserRoleApplicationFormForRegistration
from users.logic import UserRegistrationController
from utils.common import delete_everything_created_in_transaction


class CreateUserRegistrationRequestView(CreateView):
    form_class = NewUserForm
    template_name = 'registration/register.html'

    def get_context_data(self, **kwargs):
        ctx = super().get_context_data(**kwargs)
        ctx['role_application_form'] = UserRoleApplicationFormForRegistration()
        return ctx

    def post(self, request, *args, **kwargs):
        # noinspection PyAttributeOutsideInit
        self.object = None
        user_form = self.form_class(self.request.POST or None)
        role_application_without_user = UserRoleApplicationFormForRegistration(
            data=self.request.POST or None
        )
        if user_form.is_valid() and role_application_without_user.is_valid():
            return self.form_valid(user_form, role_application_without_user)
        return self.form_invalid(user_form, role_application_without_user)

    @method_decorator(transaction.atomic)
    def form_valid(self, user_form: NewUserForm,  # noqa pylint: disable=W0221
                   role_application_without_user_form:
UserRoleApplicationFormForRegistration):
        controller            =            UserRegistrationController(user_form,
role_application_without_user_form)
        # noinspection PyAttributeOutsideInit
        self.object                                                              =
controller.save_user_along_with_registration_request_return_user()
        if controller.send_email_confirmation_message(self.request):
            return          redirect(reverse('successfully-created-registration-
request'))
        messages.error(
            self.request,
            _('Не вдалося відправити повідомлення на пошту, перевірте дані та
спробуйте ще раз')
        )
        delete_everything_created_in_transaction()
        return redirect(reverse('register'))

    def form_invalid(  # noqa pylint: disable=W0221
            self, user_form: NewUserForm,
            role_application_form:
            UserRoleApplicationFormForRegistration):
        return self.render_to_response(
            self.get_context_data(
                user_form=user_form,
                role_application_form=
                role_application_form,
            ),
            status=400
        )
```

Файл: users/views/__init__.py

```
from .create_user_registration_request import CreateUserRegistrationRequestView
from    .list_views    import    UserListView,    UserRoleApplicationsListView,
UserRoleListView
from .simple import (
    confirm_email, LoginView, ProfileView, RoleApplicationView,
    successfully_created_registration_request, UserRoleView, UserView,
)
from .user_role_application_decision import UserRoleApplicationDecisionView
```

Файл: users/views/user_role_application_decision.py

```
from enum import Enum

from django.http import HttpRequest
from django.shortcuts import get_object_or_404
from django.urls import reverse_lazy
from django.views.generic import FormView

from users.forms import UserRoleApplicationRequestsDecisionForm
from           users.logic           import           check_user_has_no_roles,
UserRoleApplicationReviewController
from users.models import UserRoleApplication
from utils.common import admin_rights_and_login_required


class DECISIONS(str, Enum):
    DECISION_ARGUMENT = 'decision'
    ACCEPT = 'accept'
    DECLINE = 'decline'


@admin_rights_and_login_required
class UserRoleApplicationDecisionView(FormView):
    template_name = 'users/user_role_application_decision.html'
    form_class = UserRoleApplicationRequestsDecisionForm
    success_url = reverse_lazy('user-role-applications')

    def get_context_data(self, **kwargs):
        ctx = super().get_context_data(**kwargs)
        role_application = self.__get_application_request()
        ctx['user_has_no_roles']                                                =
check_user_has_no_roles(role_application.user)
        ctx['form']                                                             =
self.form_class.create_from_role_application(role_application)
        return ctx

    def post(self, request: HttpRequest, *args, **kwargs):
        form = self.form_class(request.POST or None)
        if self.__application_approved_but_form_invalid(form):
            return self.form_invalid(form)
        # noinspection PyAttributeOutsideInit
        self.controller = UserRoleApplicationReviewController(
            form,
            self.__get_application_request(),
            self.request,
            self.is_application_approved
        )
        return self.form_valid(form)

    def form_valid(self, form):
        self.controller.process_depending_on_decision()
```

```
        return super().form_valid(form)

    def __get_application_request(self):
        return get_object_or_404(
            UserRoleApplication.objects.select_related('user'),
            pk=self.kwargs['pk']
        )

    def __application_approved_but_form_invalid(
            self, form: UserRoleApplicationRequestsDecisionForm):
        return self.is_application_approved and not form.is_valid()

    @property
    def is_application_approved(self):
        return self.request.POST[DECISIONS.DECISION_ARGUMENT] == DECISIONS.ACCEPT

    @property
    def is_application_declined(self):
        return not self.is_application_approved
```

Файл: users/views/simple.py

```
from django.contrib import messages
from django.contrib.auth.decorators import login_required
from django.contrib.auth.views import LoginView as LogView
from django.http import HttpRequest
from django.shortcuts import render
from django.urls import reverse_lazy
from django.utils.translation import gettext_lazy as _
from django.views.generic import FormView

from users.forms import (
    LoginForm, ProfileForm, UserForm,
    UserRoleApplicationForm, UserRoleForm,
)
from users.logic import (
    remember_user_for_two_week,
    UserRegistrationController,
)
from users.models import User, UserRole
from utils.common import admin_rights_and_login_required
from utils.common.decoration import decorate_class_or_function_view
from utils.views.edit_object_update_view import EditDeleteObjectUpdateView


def successfully_created_registration_request(request: HttpRequest):
    return                                                    render(request,
'registration/successfully_created_registration_request.html')


def confirm_email(request: HttpRequest, user_id: int, user_email: str):
    UserRegistrationController.confirm_email_if_user_exists(user_id, user_email)
    return render(request, 'registration/successfully_confirmed_email.html')


class LoginView(LogView):
    authentication_form = LoginForm

    def form_valid(self, form):
        if form.data.get('remember_me'):
            remember_user_for_two_week(self.request)
        return super().form_valid(form)
```

```
@decorate_class_or_function_view(login_required)
class ProfileView(FormView):
    form_class = ProfileForm
    template_name = 'users/profile.html'

    def get_form(self, form_class=None):
        return self.form_class(user=self.request.user)


@decorate_class_or_function_view(login_required)
class RoleApplicationView(FormView):
    form_class = UserRoleApplicationForm
    template_name = 'users/role_application.html'
    success_url = reverse_lazy('my-profile')

    def form_valid(self, form):
        messages.success(
            self.request,
            _('Успішно подано заяву на роль')
        )
        form.set_application_request_user(self.request.user)
        form.save()
        return super().form_valid(form)


@admin_rights_and_login_required
class UserRoleView(EditDeleteObjectUpdateView):
    model = UserRole
    form_class = UserRoleForm
    success_url = reverse_lazy('user-role-list')
    template_name = 'users/edit_user_role.html'
    EDIT_SUCCESS_MESSAGE = _('Роль користувача успішно відредаговано')


@admin_rights_and_login_required
class UserView(EditDeleteObjectUpdateView):
    model = User
    form_class = UserForm
    success_url = reverse_lazy('users-list')
    template_name = 'users/edit_user.html'
    # overriden due to 'user' taken by client user variable
    context_object_name = 'user_obj'
    EDIT_SUCCESS_MESSAGE = _('Користувача успішно відредаговано')
```

Файл: templates/institutions/facility_list.html

```
{% extends 'common/__items_list_with_paginator_base.html' %}
{% block title %}
    Заклади та об'єкти
{% endblock %}
{% block heading_text %}
    Заклади та об'єкти
{% endblock %}
{% block heading_buttons %}
    <a class="btn btn-primary d-block ms-3" href="{% url 'new-institution'
%}">Додати заклад</a>
    <a class="btn btn-primary d-block ms-3" href="{% url 'new-facility' %}">Додати
об'єкт закладу</a>
{% endblock %}
{% block search_form %}
```

```
    {% include 'common/search_form.html' with filter_fields_verbose='назва чи
опис' %}
{% endblock %}
```

Файл: templates/institutions/edit_facility.html

```
{% extends 'common/form_in_center_with_default_padding_and_width.html' %}
{% load js_reverse %}
{% load static %}
{% load crispy_forms_tags %}
{% block title %}Редагування об’єкту{% endblock %}
{% block form_header %}
    Редагування об’єкту
{% endblock %}
{% block form %}
    <form method="post">
        {% crispy form %}
    </form>
{% endblock %}
{% block scripts %}
    <script src="{% static 'js/institutions/edit_facility.js' %}"></script>
{% endblock %}
{% block display_none %}
    {% create_url_name_id_p_tag_with_link_content 'edit-facility-pk-in-post' %}
    {% create_url_name_id_p_tag_with_link_content 'edit-user-role-pk-in-post' %}
{% endblock %}
```

Файл: templates/institutions/new_facility.html

```
{% extends 'common/form_in_center_with_default_padding_and_width.html' %}
{% load js_reverse %}
{% load static %}
{% load crispy_forms_tags %}
{% block title %}Новий об'єкт{% endblock %}
{% block form_header %}
    Уведіть дані об'єкту
{% endblock %}
{% block form %}
    <form method="post">
        {% crispy form %}
    </form>
{% endblock %}
{% block scripts %}
    <script src="{% static 'js/institutions/new_facility.js' %}"></script>
    <script                              src="{%                              static
'js/institutions/dynamic_facility_choices_for_institution.js' %}"></script>
{% endblock %}
{% block display_none %}
    {% create_url_name_id_p_tag_with_link_content 'get-institution-facilities' %}
{% endblock %}
```

Файл: templates/institutions/new_institution.html

```
{% extends 'common/form_in_center_with_default_padding_and_width.html' %}
{% load crispy_forms_tags %}
{% block title %}Новий заклад{% endblock %}
{% block form_header %}
    Уведіть дані закладу
{% endblock %}
{% block form %}
    <form method="post">
```

```
        {% crispy form %}
    </form>
{% endblock %}
```

Файл: templates/registration/successfully_created_registration_request.html

```
{% extends 'common/some_text_in_center_base.html' %}
{% block title %}
    Заявка на реєстрацію успішна
{% endblock %}
{% block header %}
    Ви успішно зареєструвалися
{% endblock %}
{% block text %}
    Тепер вам необхідно підтвердити вашу електронну пошту, слідуючи вказівкам у
листі.
{% endblock %}
```

Файл: templates/registration/register.html

```
{% extends 'common/form_in_center_needs_padding_and_width.html' %}
{% load crispy_forms_tags %}
{% load crispy_forms_filters %}
{% block form_header %}
    Реєстрація
{% endblock %}
{% block padding_top_class %}
    pt-1
{% endblock %}
{% block width_class %}
    w-50
{% endblock %}
{% block form %}
    <form method="post">
        {{ form | crispy }}
        {% crispy role_application_form %}
    </form>
{% endblock %}
```

Файл: templates/registration/login.html

```
{% extends 'common/form_in_center_needs_padding_and_width.html' %}
{% load crispy_forms_tags %}
{% load crispy_forms_filters %}
{% block title %}Логін{% endblock %}
{% block form_header %}
    Уведіть дані авторизації | <a href="{% url 'register' %}">Реєстрація</a>
{% endblock %}
{% block padding_top_class %}
    pt-5
{% endblock %}
{% block width_class %}
    w-25
{% endblock %}
{% block form %}
    <form method="post">
        {% crispy form %}
    </form>
{% endblock %}
```

Файл: templates/registration/successfully_confirmed_email.html

```
{% extends 'common/some_text_in_center_base.html' %}
{% block title %}
    Пошту підтверджено
{% endblock %}
{% block header %}
    Ви успішно підтвердили свою пошту
{% endblock %}
{% block text %}
    Очікуйте розгляду вашого запиту адміністратором.
    Ви отримаєте повідомлення про його результати.
{% endblock %}
```

Файл: templates/registration/password_change.html

```
{% extends 'common/form_in_center_with_default_padding_and_width.html' %}
{% load crispy_forms_tags %}
{% block title %}
    Зміна паролю
{% endblock %}
{% block form_header %}
    Зміна паролю
{% endblock %}
{% block form %}
    <form method="post">
        {% crispy form %}
    </form>
{% endblock %}
```

Файл: templates/energy/edit_sensor.html

```
{% extends 'common/form_in_center_with_default_padding_and_width.html' %}
{% load js_reverse %}
{% load i18n %}
{% load static %}
{% load crispy_forms_tags %}
{% block title %}{% trans "Редагування сенсору" %}{% endblock %}
{% block form_header %}
    {% trans "Редагування сенсору" %}
{% endblock %}
{% block form %}
    {% crispy form %}
    <script src="{% static 'js/energy/edit_sensor.js' %}"></script>
{% endblock %}
{% block display_none %}
    {% create_url_name_id_p_tag_with_link_content 'edit-box-pk-in-post' %}
    {%   create_url_name_id_p_tag_with_link_content   'edit-box-sensor-set-pk-in-
post' %}
{% endblock %}
```

Файл: templates/energy/edit_box_sensor_set.html

```
{% extends 'common/form_in_center_with_default_padding_and_width.html' %}
{% load i18n %}
{% load static %}
{% load crispy_forms_tags %}
{% load js_reverse %}
{% block title %}{% trans "Редагування набору" %}{% endblock %}
{% block form_header %}
    {% trans "Редагування набору" %}
{% endblock %}
{% block form %}
```

```
    {% crispy form %}
    <script src="{% static 'js/energy/edit_box_sensor_set.js' %}"></script>
{% endblock %}
{% block display_none %}
    {% create_url_name_id_p_tag_with_link_content 'edit-sensor-pk-in-post' %}
    {% create_url_name_id_p_tag_with_link_content 'edit-facility-pk-in-post' %}
{% endblock %}
```

Файл: templates/energy/box_sensor_set_list.html

```
{% extends 'common/__items_list_with_paginator_base.html' %}
{% load i18n %}
{% block title %}
    {% trans "Набори сенсорів" %}
{% endblock %}
{% block heading_text %}
    {% trans "Набори сенсорів" %}
{% endblock %}
{% block search_form %}
    {%             include             'common/search_form.html'             with
filter_fields_verbose='ідентифікатор ящику, назва установи чи лінії' %}
{% endblock %}
```

Файл: templates/energy/edit_box.html

```
{% extends 'common/form_in_center_with_default_padding_and_width.html' %}
{% load js_reverse %}
{% load i18n %}
{% load static %}
{% load crispy_forms_tags %}
{% block title %}{% trans "Редагування ящику" %}{% endblock %}
{% block form_header %}
    {% trans "Редагування ящику" %}
{% endblock %}
{% block form %}
    {% crispy form %}
    <script src="{% static 'js/energy/edit_box.js' %}"></script>
{% endblock %}
{% block display_none %}
    {%   create_url_name_id_p_tag_with_link_content   'edit-box-sensor-set-pk-in-
post' %}
{% endblock %}
```

Файл: templates/energy/sensor_list.html

```
{% extends 'common/__items_list_with_paginator_base.html' %}
{% load i18n %}
{% block title %}
    {% trans "Сенсори" %}
{% endblock %}
{% block heading_text %}
    {% trans "Сенсори" %}
{% endblock %}
{% block search_form %}
    {%  include  'common/search_form.html'  with  filter_fields_verbose='номер  чи
опис' %}
{% endblock %}
```

Файл: templates/energy/box_list.html

```
{% extends 'common/__items_list_with_paginator_base.html' %}
{% load i18n %}
{% block title %}
```

```
    {% trans "Ящики" %}
{% endblock %}
{% block heading_text %}
    {% trans "Ящики" %}
{% endblock %}
{% block heading_buttons %}
    <a class="btn btn-primary d-block ms-3" href="{% url 'create-box-with-sensors'
%}">{% trans "Новий ящик" %}</a>
{% endblock %}
{% block search_form %}
    {%             include             'common/search_form.html'             with
filter_fields_verbose='ідентифікатор чи опис' %}
{% endblock %}
```

Файл: templates/energy/consumption/control_form.html

```
{% load i18n %}
{% load crispy_forms_filters %}
<div class="container d-inline-block p-0">
    <form id="id_control_form" method="post">
        {{ form | crispy }}
        <div id="id_hours_filtering">
            <div  class="d-inline">{%  trans  "Фільтрація  інтервалу  агрегації"
%}</div>
            <div class="d-flex align-items-center py-2">
                <div                                                  class="d-inline"
id="hour_filtering_option_buttons_container_id"></div>
            </div>
            <div class="d-flex align-items-center">
                <div class="d-inline">
                    {%   include   'energy/consumption/__hour_filter_select.html'
with start_or_end='start' text='Від' default_value='00' %}
                    {%   include   'energy/consumption/__hour_filter_select.html'
with start_or_end='end' text='До' default_value='23' %}
                </div>
                <div class="d-inline ms-2">
                    <a           class="btn           btn           btn-primary"
id="reset_hours_filters_btn">{% trans "Скинути" %}</a>
                </div>
            </div>
        </div>
        <div id="div_id_period_start" class="form-group">
            <label for="period_start">{% trans "Початкова дата" %}</label>
            <div>
                <input    type="date"    name="period_start"    id="period_start"
required>
            </div>
        </div>
        <div id="div_id_period_end" class="form-group">
            <label for="period_end">{% trans "Кінцева дата" %}</label>
            <div>
                <input type="date" name="period_end" id="period_end" required>
            </div>
        </div>
        <a  class="btn  btn  btn-primary  mt-3"  id="get_data_button">{%  trans
"Отримати дані" %}</a>
        <p class="fw-light text-muted mt-3">{% trans "Усі параметри застосовуються
включно" %}</p>
    </form>
    <hr>
    <p class="fw-bold">{% trans "Агрегація" %}</p>
    <div>
        <p id="aggregation_status_p_id" class="d-inline">
```

```
            {% trans "Поточний статус:" %}
        </p>
        <p class="d-inline" id="aggregation_status_id">
        </p>
    </div>
    <div>
        <p id="aggregation_status_p_id" class="d-inline">
            {% trans "Останній запуск:" %}
        </p>
        <p class="d-inline" id="aggregation_last_time_run_id">
        </p>
    </div>
    <a class="btn mt-0" id="run_aggregation_button">
        <span class="spinner-border spinner-border-sm d-none" role="status" aria-
hidden="true"
              id="start_aggregation_request_processing_spinner"></span>
        <span id="run_aggregation_button_text">{% trans "Запустити" %}</span>
    </a>
</div>
```

Файл: templates/energy/consumption/main_page.html

```
{% extends 'common/base.html' %}
{% load js_reverse %}
{% load i18n %}
{% load crispy_forms_filters %}
{% load static %}
{% load crispy_forms_tags %}
{% block title %}
    {% trans "Енергоспоживання" %}
{% endblock %}
{% block content %}
    <div class="container" style="height: 80vh;">
        <div class="row">
            <div class="" style="width: 30%;">
                {% include 'energy/consumption/control_form.html' %}
            </div>
            <div class="d-inline w-auto flex-grow-1" style="max-height: 80vh;">
                {% include 'energy/consumption/consumption_data.html' %}
            </div>
        </div>
    </div>
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/hammerjs@2.0.8"></script>
    <script          src="https://cdnjs.cloudflare.com/ajax/libs/chartjs-plugin-
zoom/2.0.0/chartjs-plugin-zoom.min.js"
            integrity="sha512-
B6F98QATBNaDHSE7uANGo5h0mU6fhKCUD+SPAY7KZDxE8QgZw9rewDtNiu3mbbutYDWOKT3SPYD8qDBp
G2QnEg=="
            crossorigin="anonymous" referrerpolicy="no-referrer"></script>
    <script            type="text/javascript"            src="{%            static
"js/energy/consumption/querying.js" %}"></script>
    <script            type="text/javascript"            src="{%            static
"js/energy/consumption/helping.js" %}"></script>
    <script            type="text/javascript"            src="{%            static
"js/energy/consumption/displaying.js" %}"></script>
    <script            type="text/javascript"            src="{%            static
"js/energy/consumption/aggregation.js" %}"></script>
    <link  rel="stylesheet"  href="{%  static  "css/energy/consumption_table.css"
%}">
{% endblock %}
{% block display_none %}
```

```
    {% create_url_name_id_p_tag_with_link_content 'get-facilities-list-for-role'
%}
    {%  create_url_name_id_p_tag_with_link_content  'get-aggregated-consumption-
for-facility' %}
    {% create_url_name_id_p_tag_with_link_content 'run-aggregation' %}
    {%  create_url_name_id_p_tag_with_link_content  'get-aggregation-status-and-
last-time-run' %}
{% endblock %}
```

Файл: templates/energy/consumption/__hour_filter_select.html

```
{% load i18n %}
<div id="div_id_hour_filter_{{ start_or_end }}" class="d-inline form-group w-
auto">
    <label for="id_hour_filter_{{ start_or_end }}">
        {% trans text %}
        <span class="asteriskField">*</span>
    </label>
    <div class="d-inline-block">
        <select name="hour_filter_{{ start_or_end }}" class="select form-control
custom-select w-auto"
                id="id_hour_filter_{{ start_or_end }}">
            <option value="0">00</option>
            <option value="1">01</option>
            <option value="2">02</option>
            <option value="3">03</option>
            <option value="4">04</option>
            <option value="5">05</option>
            <option value="6">06</option>
            <option value="7">07</option>
            <option value="8">08</option>
            <option value="9">09</option>
            <option value="10">10</option>
            <option value="11">11</option>
            <option value="12">12</option>
            <option value="13">13</option>
            <option value="14">14</option>
            <option value="15">15</option>
            <option value="16">16</option>
            <option value="17">17</option>
            <option value="18">18</option>
            <option value="19">19</option>
            <option value="20">20</option>
            <option value="21">21</option>
            <option value="22">22</option>
            <option value="23">23</option>
        </select>
    </div>
</div>
```

Файл: templates/energy/consumption/you_have_no_roles.html

```
{% extends 'common/some_text_in_center_base.html' %}
{% load i18n %}
{% block title %}
    {% trans "У вас ще немає жодної ролі" %}
{% endblock %}
{% block header %}
    {% trans "У вас ще немає жодної ролі" %}
{% endblock %}
{% block text %}
```

```
    {% trans "Адміністратор ще не встиг підтвердити ваш запит на роль, чи відхилив
його. " %}
    {% trans "Ви отримаете листа про будь-яку з цих дій." %}
{% endblock %}
```

Файл: templates/energy/consumption/consumption_data.html

```
<div>
    <div class="d-flex justify-content-center pb-2">
        <a            class="btn            btn            btn-outline-secondary"
id="set_table_button">Таблиця</a>
        <a class="btn btn-outline-primary ms-5" id="set_chart_button">Графік</a>
    </div>
    <div id="energy_content" class="overflow-auto" style="max-height: 70vh;">
        {#table or chart is placed here via js#}
    </div>
</div>
```

Файл: templates/energy/box_with_sensors_creation/sensors.html

```
{% extends 'common/form_in_center_with_default_padding_and_width.html' %}
{% load crispy_forms_filters %}
{% load i18n %}
{% load static %}
{% load crispy_forms_tags %}
{% block title %}{% trans "Сенсори" %}{% endblock %}
{% block form_header %}
    {% trans "Уведіть дані сенсорів" %}
{% endblock %}
{% block form %}
    <form      method="post"      id="sensors_form">{%      csrf_token      %}{{
wizard.management_form }}
        {{ sensors_count_form | crispy }}
        {{ wizard.form | crispy }}
        {% include 'energy/box_with_sensors_creation/wizard_buttons.html' %}
    </form>
{% endblock %}
{% block scripts %}
    <script   src="{%   static   'js/energy/box_with_sensors_creation/sensors.js'
%}"></script>
{% endblock %}
```

Файл: templates/energy/box_with_sensors_creation/box_sensors_set.html

```
{% extends 'common/form_in_center_with_default_padding_and_width.html' %}
{% load js_reverse %}
{% load crispy_forms_filters %}
{% load i18n %}
{% load static %}
{% load crispy_forms_tags %}
{% block title %}{% trans "Прив'язка сенсорів до об'єктів" %}{% endblock %}
{% block form_header %}
    {% trans "Співвіднесіть сенсори та об'єкти" %}
{% endblock %}
{% block form %}
    <form      method="post"      id="sensors_form">{%      csrf_token      %}{{
wizard.management_form }}
        {{ choose_institution_form | crispy }}
        {{ wizard.form | crispy }}
        {% include 'energy/box_with_sensors_creation/wizard_buttons.html' %}
    </form>
{% endblock %}
{% block scripts %}
```

```
    <script                              src="{%                              static
'js/energy/box_with_sensors_creation/box_sensor_set.js' %}"></script>
    <script                              src="{%                              static
'js/institutions/dynamic_facility_choices_for_institution.js' %}"></script>
{% endblock %}
{% block display_none %}
    {% create_url_name_id_p_tag_with_link_content 'get-institution-facilities' %}
{% endblock %}
```

Файл: templates/energy/box_with_sensors_creation/wizard_buttons.html

```
{% load i18n %}
{% if wizard.steps.prev %}
    <button   class="btn   btn   btn-primary   mt-3"   name="wizard_goto_step"
type="submit" value="{{ wizard.steps.prev }}">
        {% trans "Назад" %}
    </button>
{% endif %}
<button class="btn btn btn-primary mt-3" type="submit">
    {% if wizard.next %}
        {% trans "Створити" %}
    {% else %}
        {% trans "Далі" %}
    {% endif %}
</button>
```

Файл: templates/energy/box_with_sensors_creation/box.html

```
{% extends 'common/form_in_center_with_default_padding_and_width.html' %}
{% load crispy_forms_filters %}
{% load i18n %}
{% load static %}
{% load crispy_forms_tags %}
{% block title %}Ящик{% endblock %}
{% block form_header %}
    Уведіть дані ящику
{% endblock %}
{% block form %}
    {{ wizard.management_form }}
    <form method="post">{% csrf_token %}
        {{ wizard.management_form }}
        {{ wizard.form | crispy }}
        {% include 'energy/box_with_sensors_creation/wizard_buttons.html' %}
    </form>
{% endblock %}
```

Файл: templates/common/message.html

```
<div id="alerts">
    <div  class="alert  alert-{{  message_class  }}  alert-dismissible  fade  show"
role="alert">
        {{ message_text }}
        <button  type="button"  class="btn-close"  data-bs-dismiss="alert"  aria-
label="Close"></button>
    </div>
</div>
```

Файл: templates/common/some_text_in_center_base.html

```
{% extends 'common/__bare_base.html' %}
{% block body %}
    <div  class="d-flex  justify-content-center  align-items-center  flex-column"
style="height: 100vh;">
        <div>
            <h1>
                {% block header %}
                {% endblock %}
            </h1>
        </div>
        <div class="w-25 text-center pt-5">
            <p>
                {% block text %}
                {% endblock %}
            </p>
        </div>
    </div>
{% endblock %}
```

Файл: templates/common/base.html

```
{% extends 'common/__bare_base.html' %}
{% load show_messages_if_any %}
{% block body %}
    {% include 'header_and_footer/header.html' %}
    {% if messages %}
        <div class="container">
            {% show_messages %}
        </div>
    {% endif %}
    {% block content %}
    {% endblock %}
    <div class="d-none">
        {% block display_none %}
        {% endblock %}
    </div>
    {% include 'header_and_footer/footer.html' %}
{% endblock %}
```

Файл: templates/common/__bare_base.html

```
<!doctype html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
          content="width=device-width, user-scalable=no, initial-scale=1.0,
           maximum-scale=1.0, minimum-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link                    href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0-
beta1/dist/css/bootstrap.min.css" rel="stylesheet"
          integrity="sha384-
0evHe/X+R7YkIZDRvuzKMRqM+OrBnVFBL6DOitfPri4tjfHxaWutUpFmBp4vmVor"
crossorigin="anonymous">
    <script                    src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0-
beta1/dist/js/bootstrap.bundle.min.js"
            integrity="sha384-
pprn3073KE6tl6bjs2QrFaJGz5/SUsLqktiwsUTF55Jfv3qYSDhgCecCxMW52nD2"
            crossorigin="anonymous"></script>
    <link     rel="stylesheet"     href="https://cdn.jsdelivr.net/npm/bootstrap-
icons@1.8.3/font/bootstrap-icons.css">
```

```
    <!--bootstrap-->
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
    <script                  src="https://cdnjs.cloudflare.com/ajax/libs/jquery-
cookie/1.4.1/jquery.cookie.min.js"
            integrity="sha512-
3j3VU6WC5rPQB4Ld1jnLV7Kd5xr+cq9avvhwqzbH/taCRNURoeEpoPBK9pDyeukwSxwRPJ8fDgvYXd6S
kaZ2TA=="
            crossorigin="anonymous" referrerpolicy="no-referrer"></script>
    <!--jquery-->
    {% load static %}
    <link rel="stylesheet" href="{% static "css/base.css" %}">
    <script src="{% static 'js/common/utils.js' %}"></script>
    <title>{% block title %}{% endblock %}</title>
    <link rel="shortcut icon" type="image/svg" href="{% static 'icons/logo.svg'
%}"/>
</head>
<body>
{% block body %}
{% endblock %}
</body>
</html>
```

Файл: templates/common/paginator.html

```
{% load static %}
{% if paginator and page %}
    {% load pagination %}
    {% load url_replace %}
    <div    class="btn-group"    role="group"    aria-label="Item    pagination"
id="pagination_buttons">
        {% if page_obj.number == 1 %}
            <a class="btn btn-outline-secondary">
                ⇤
            </a>
        {% else %}
            <a class="btn btn-outline-primary"
               href="?{%
previous_invisible_page_number_if_exists_else_min_available_link         request
page.number %}">
                ⇤
            </a>
        {% endif %}
        {% if page_obj.has_previous %}
            <a  class="btn  btn-outline-primary"  href="?{%  url_replace  request
'page' page_obj.previous_page_number %}">
                &laquo;
            </a>
        {% else %}
            <a class="btn btn-outline-secondary">
                &laquo;
            </a>
        {% endif %}

        {% for i in paginator|paginate_with_items_per_page_limit:page_obj.number
%}
            {% if page_obj.number == i %}
                <a class="btn btn-outline-secondary">{{ i }}</a>
            {% else %}
                <a class="btn btn-outline-primary" href="?{% url_replace request
'page' i %}">{{ i }}</a>
            {% endif %}
        {% endfor %}
```

```
        {% if page_obj.has_next %}
            <a class="btn btn-outline-primary"
               href="?{% url_replace request 'page' page_obj.next_page_number
%}">&raquo;</a>
        {% else %}
            <a class="btn btn-outline-secondary">&raquo;</a>
        {% endif %}
        {% if page_obj.number == paginator.num_pages %}
            <a class="btn btn-outline-secondary">⇥</a>
        {% else %}
            <a class="btn btn-outline-primary"
               href="?{%
next_invisible_page_number_if_exists_else_max_available_link request
page_obj.number paginator.num_pages %}">
                ⇥
            </a>
        {% endif %}
    </div>
{% endif %}
```

Файл: templates/common/__items_list_with_paginator_base.html

```
{% extends 'common/base.html' %}
{% block content %}
    <div class="container" id="id_main_container">
        <div class="container px-0 d-flex align-items-center flex-wrap">
            <h4 class="d-block">
                {% block heading_text %}
                {% endblock %}
            </h4>
            {% block heading_buttons %}
            {% endblock %}
            <div class="ms-auto">
                {% block search_form %}
                {% endblock %}
            </div>
        </div>
        <hr>
        {% include 'common/page_items_with_paginator.html' %}
    </div>
{% endblock %}
```

Файл: templates/common/form_in_center_with_default_padding_and_width.html

```
{% extends 'common/form_in_center_needs_padding_and_width.html' %}
{% block padding_top_class %}
    pt-2
{% endblock %}
{% block width_class %}
    w-50
{% endblock %}
```

Файл: templates/common/search_form.html

```
{% load static %}
<form method="get" role="search" id="searchForm" class="container px-0">
    <div class="row g-1">
        <div class="col col-md-auto">
            <input type="search" class="form-control d-inline" placeholder="Пошук
({{ filter_fields_verbose }})"
                   aria-label="Search" name="search_value" id="searchFormInput">
        </div>
```

```
        <div class="col col-md-auto">
            <button    type="button"    class="form-control    btn    btn-primary"
id="clearFilter">Очистити пошук</button>
        </div>
    </div>
</form>
<script src="{% static 'js/common/search_form.js' %}"></script>
```

Файл: templates/common/page_items_with_paginator.html

```
{% load static %}
<div class="container px-0" id="items">
    {% for item in page_obj.object_list %}
        <div class="card mt-2">
            <div class="card-body">
                <p class="card-text m-0">
                    {{ item }}
                </p>
                <a href="{{ item.get_absolute_url }}"
                   class="stretched-link">
                </a>
            </div>
        </div>
    {% endfor %}
</div>
{% if is_paginated %}
    <div class="container d-flex pt-4 px-0 justify-content-center">
        {% include 'common/paginator.html' with page=page_obj %}
    </div>
{% endif %}
<script src="{% static 'js/common/page_items.js' %}"></script>
```

Файл: templates/common/form_in_center_needs_padding_and_width.html

```
{% extends 'common/base.html' %}
{% load static %}
{% block content %}
    <div  class="d-flex  align-items-center  justify-content-center  {%  block
padding_top_class %}{% endblock %}"
         style="height: 100%;" id="form-container">
        <div class="card p-0 {% block width_class %}{% endblock %}" id="form-
card">
            <div class="card-header">
                {% block form_header %}
                {% endblock %}
            </div>
            <div class="card-body">
                {% block form %}
                {% endblock %}
            </div>
        </div>
    </div>
    {% block scripts %}
    {% endblock %}
    <script src="{% static 'js/common/form_in_center.js' %}"></script>
{% endblock %}
```

Файл: templates/header_and_footer/admin_items.html

```
{% load i18n %}
{% load active_link_tags %}
<li>
    <a href="{% url 'user-role-applications' %}"
```

```
       class="nav-link px-2 text-dark {% active_link 'user-role-applications'
%}">
        {% trans "Запити на ролі" %}
        {% if users_roles_applications_count %}
            <span  class="badge  bg-success">{{  users_roles_applications_count
}}</span>
        {% endif %}
    </a>
</li>
<li>
    <a href="{% url 'facilities-list' %}"
       class="nav-link px-2 text-dark {% active_link 'facilities-list' %}">
        {% trans "Заклади та об'єкти" %}
    </a>
</li>
<li>
    <a href="{% url 'users-list' %}"
       class="nav-link px-2 text-dark {% active_link 'users-list' %}">
        {% trans "Користувачі" %}
    </a>
</li>
<li>
    <a href="{% url 'user-role-list' %}"
       class="nav-link px-2 text-dark {% active_link 'user-role-list' %}">
        {% trans "Ролі користувачі" %}
    </a>
</li>
<li>
    <a href="{% url 'box-list' %}"
       class="nav-link px-2 text-dark {% active_link 'box-list' %}">
        {% trans "Ящики" %}
    </a>
</li>
<li>
    <a href="{% url 'sensor-list' %}"
       class="nav-link px-2 text-dark {% active_link 'sensor-list' %}">
        {% trans "Сенсори" %}
    </a>
</li>
<li>
    <a href="{% url 'box-sensor-set-list' %}"
       class="nav-link px-2 text-dark {% active_link 'box-sensor-set-list' %}">
        {% trans "Набори сенсорів" %}
    </a>
</li>
```

Файл: templates/header_and_footer/footer.html

Файл: templates/header_and_footer/header.html

```
{% load static %}
{% load active_link_tags %}
{% load i18n %}
{% if user.is_authenticated %}
    <header class="p-3 text-white">
        <div class="container px-1">
            <div  class="d-flex  flex-wrap  align-items-center  justify-content-
center justify-content-lg-start">
                <ul  class="nav  col-12  col-lg-auto  me-lg-auto  mb-2  justify-
content-center mb-md-0 align-items-center">
                    <li>
```

```
                        <img src="{% static "icons/logo.svg" %}" alt="Logo"
                             style="height: 6vh;">
                    </li>
                    <li>
                        <a href="{% url 'aggregated-consumption' %}"
                           class="nav-link   px-2   text-dark   {%   active_link
'aggregated-consumption' %}">
                            {% trans "Енергоспоживання" %}
                        </a>
                    </li>
                    <li>
                        <a href="{% url 'my-profile' %}"
                           class="nav-link  px-2  text-dark  {%  active_link  'my-
profile' %}">
                            {% trans "Мій профіль" %}
                        </a>
                    </li>
                    {% if user.is_admin %}
                        {% include 'header_and_footer/admin_items.html' %}
                    {% endif %}
                </ul>
                <div class="text-end ml-lg-auto">
                    <a href="{% url 'logout' %}"
                       class="py-1  btn  btn-outline-secondary  rounded-lg  text-
dark">
                        {% trans "Вихід" %}
                    </a>
                </div>
            </div>
        </div>
    </header>
{% endif %}
```

Файл: templates/users/users_roles_list.html

```
{% extends 'common/__items_list_with_paginator_base.html' %}
{% block title %}
    Ролі
{% endblock %}
{% block heading_text %}
    Список ролей
{% endblock %}
{% block search_form %}
    {%   include   'common/search_form.html'   with   filter_fields_verbose="назва
установи, ім'я користувача чи назва ролі" %}
{% endblock %}
```

Файл: templates/users/profile.html

```
{% extends 'common/form_in_center_with_default_padding_and_width.html' %}
{% load i18n %}
{% load static %}
{% load crispy_forms_tags %}
{% block title %}Мій профіль{% endblock %}
{% block form_header %}
    Мій профіль
{% endblock %}
{% block form %}
    <form method="post">
        {% crispy form %}
        <a href="{% url 'role-application' %}" class="btn btn-primary mt-3">
            {% trans "Подати запит на роль" %}
        </a>
```

```
    </form>
{% endblock %}
```

Файл: templates/users/user_role_application_decision.html

```
{% extends 'common/base.html' %}
{% load static %}
{% load crispy_forms_tags %}
{% load crispy_forms_filters %}
{% block title %}
    Розгляд запиту на отримання ролі
{% endblock %}
{% block content %}
    <div class="container">
        {% if user_has_no_roles %}
            <div class="card bg-warning">
                <h3 class="card-body">
                    Цей користувач ще не має жодної ролі, уважно перевірте
пошту&amp;#8201;&amp;#8211;&amp;#8201;це може бути
                    стороння людина, що подала заявку на реєстрацію
                </h3>
            </div>
        {% endif %}
        <form method="post">
            {% crispy form %}
        </form>
    </div>
    <script            type="text/javascript"            src="{%            static
"js/users/user_role_application_decision.js" %}"></script>
{% endblock %}
```

Файл: templates/users/edit_user.html

```
{% extends 'common/form_in_center_with_default_padding_and_width.html' %}
{% load js_reverse %}
{% load static %}
{% load crispy_forms_tags %}
{% block title %}Користувач{% endblock %}
{% block form_header %}
    Користувач
{% endblock %}
{% block form %}
    <form method="post">
        {% crispy form %}
    </form>
{% endblock %}
{% block scripts %}
    <script src="{% static 'js/users/edit_user.js' %}"></script>
{% endblock %}
{% block display_none %}
    {% create_url_name_id_p_tag_with_link_content 'edit-user-role-pk-in-post' %}
{% endblock %}
```

Файл: templates/users/users_roles_applications.html

```
{% extends 'common/__items_list_with_paginator_base.html' %}
{% load i18n %}
{% block title %}
    {% trans "Запити на ролі" %}
{% endblock %}
{% block heading_text %}
```

```
    {% trans "Запити на отримання ролі від користувачів" %}
{% endblock %}
{% block search_form %}
    {% include 'common/search_form.html' with filter_fields_verbose="ім'я, пошта користувача чи назва закладу" %}
{% endblock %}
```

Файл: templates/users/users_list.html

```
{% extends 'common/__items_list_with_paginator_base.html' %}
{% block title %}
    Користувачі
{% endblock %}
{% block heading_text %}
    Список користувачів
{% endblock %}
{% block search_form %}
    {% include 'common/search_form.html' with filter_fields_verbose="ім'я чи пошта користувача" %}
{% endblock %}
```

Файл: templates/users/role_application.html

```
{% extends 'common/form_in_center_with_default_padding_and_width.html' %}
{% load i18n %}
{% load static %}
{% load crispy_forms_tags %}
{% block title %}Запит на роль{% endblock %}
{% block form_header %}
    Запит на роль
{% endblock %}
{% block form %}
    <form method="post">
        {% crispy form %}
    </form>
{% endblock %}
```

Файл: templates/users/edit_user_role.html

```
{% extends 'common/form_in_center_with_default_padding_and_width.html' %}
{% load js_reverse %}
{% load static %}
{% load crispy_forms_tags %}
{% block title %}Роль користувача{% endblock %}
{% block form_header %}
    Роль користувача
{% endblock %}
{% block form %}
    {% crispy form %}
{% endblock %}
{% block scripts %}
    <script src="{% static 'js/users/edit_user_role.js' %}"></script>
{% endblock %}
{% block display_none %}
    {% create_url_name_id_p_tag_with_link_content 'edit-user-pk-in-post' %}
    {% create_url_name_id_p_tag_with_link_content 'edit-facility-pk-in-post' %}
{% endblock %}
```

Файл: templates/email/__regular_email_not_rendered.html

```
{% extends 'email/__email_base.html' %}
```

```
{% block content %}
    <div class="card my-10">
        <div class="card-body">
            <h1 class="h1 mb-2">HEADER</h1>
            <hr>
            <div class="space-y-3">
                <p class="text-gray-700">
                    TEXT
                </p>
            </div>
        </div>
    </div>
{% endblock %}
```

Файл: templates/email/__regular_email_rendered_base.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html lang="uk">
<head>
    <!-- Compiled with Bootstrap Email version: 1.3.0 -->
    <meta http-equiv="x-ua-compatible" content="ie=edge">
    <meta name="x-apple-disable-message-reformatting">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta name="format-detection" content="telephone=no, date=no, address=no,
email=no">
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <meta charset="UTF-8">
    <style type="text/css">
        body, table, td {
            font-family: Helvetica, Arial, sans-serif !important
        }

        .ExternalClass {
            width: 100%
        }

        .ExternalClass, .ExternalClass p, .ExternalClass span, .ExternalClass
font, .ExternalClass td, .ExternalClass div {
            line-height: 150%
        }

        a {
            text-decoration: none
        }

        * {
            color: inherit
        }

        a[x-apple-data-detectors], u + #body a, #MessageViewBody a {
            color: inherit;
            text-decoration: none;
            font-size: inherit;
            font-family: inherit;
            font-weight: inherit;
            line-height: inherit
        }

        img {
            -ms-interpolation-mode: bicubic
        }
```

```
        table:not([class^=s-]) {
            font-family: Helvetica, Arial, sans-serif;
            mso-table-lspace: 0pt;
            mso-table-rspace: 0pt;
            border-spacing: 0px;
            border-collapse: collapse
        }

        table:not([class^=s-]) td {
            border-spacing: 0px;
            border-collapse: collapse
        }

        @media screen and (max-width: 600px) {
            .w-full, .w-full > tbody > tr > td {
                width: 100% !important
            }

            *[class*=s-lg-] > tbody > tr > td {
                font-size: 0 !important;
                line-height: 0 !important;
                height: 0 !important
            }

            .s-2 > tbody > tr > td {
                font-size: 8px !important;
                line-height: 8px !important;
                height: 8px !important
            }

            .s-5 > tbody > tr > td {
                font-size: 20px !important;
                line-height: 20px !important;
                height: 20px !important
            }

            .s-10 > tbody > tr > td {
                font-size: 40px !important;
                line-height: 40px !important;
                height: 40px !important
            }
        }
    </style>
</head>
<body class="bg-light"
      style="outline: 0; width: 100%; min-width: 100%; height: 100%; -webkit-
text-size-adjust: 100%; -ms-text-size-adjust: 100%; font-family: Helvetica,
Arial, sans-serif; line-height: 24px; font-weight: normal; font-size: 16px; -moz-
box-sizing: border-box; -webkit-box-sizing: border-box; box-sizing: border-box;
color: #000000; margin: 0; padding: 0; border-width: 0;"
      bgcolor="#f7fafc">
<table  class="bg-light  body"  valign="top"  role="presentation"  border="0"
cellpadding="0" cellspacing="0"
       style="outline: 0; width: 100%; min-width: 100%; height: 100%; -webkit-
text-size-adjust: 100%; -ms-text-size-adjust: 100%; font-family: Helvetica,
Arial, sans-serif; line-height: 24px; font-weight: normal; font-size: 16px; -moz-
box-sizing: border-box; -webkit-box-sizing: border-box; box-sizing: border-box;
color: #000000; margin: 0; padding: 0; border-width: 0;"
       bgcolor="#f7fafc">
    <tbody>
    <tr>
        <td valign="top" style="line-height: 24px; font-size: 16px; margin: 0;"
align="left" bgcolor="#f7fafc">
```

```
            <table     class="container"     role="presentation"     border="0"
cellpadding="0" cellspacing="0"
                   style="width: 100%;">
                <tbody>
                <tr>
                    <td align="center" style="line-height: 24px; font-size: 16px;
margin: 0; padding: 0 16px;">
                        <!--[if (gte mso 9)|(IE)]>
                        <table align="center" role="presentation">
                            <tbody>
                            <tr>
                                <td width="600">
                        <![endif]-->
                        <table  align="center"  role="presentation"  border="0"
cellpadding="0" cellspacing="0"
                               style="width: 100%; max-width: 600px; margin: 0
auto;">
                            <tbody>
                            <tr>
                                <td  style="line-height:  24px;  font-size:  16px;
margin: 0;" align="left">
                                    <table          class="s-10          w-full"
role="presentation" border="0" cellpadding="0"
                                           cellspacing="0"  style="width:  100%;"
width="100%">
                                        <tbody>
                                        <tr>
                                            <td style="line-height: 40px; font-
size: 40px; width: 100%; height: 40px; margin: 0;"
                                                align="left"        width="100%"
height="40">
                                                 
                                            </td>
                                        </tr>
                                        </tbody>
                                    </table>
                                    <table    class="card"    role="presentation"
border="0" cellpadding="0" cellspacing="0"
                                           style="border-radius:   6px;   border-
collapse: separate !important; width: 100%; overflow: hidden; border: 1px solid
#e2e8f0;"
                                           bgcolor="#ffffff">
                                        <tbody>
                                        <tr>
                                            <td style="line-height: 24px; font-
size: 16px; width: 100%; margin: 0;"
                                                align="left" bgcolor="#ffffff">
                                                <table         class="card-body"
role="presentation" border="0" cellpadding="0"
                                                       cellspacing="0"
style="width: 100%;">
                                                    <tbody>
                                                    <tr>
                                                        <td  style="line-height:
24px; font-size: 16px; width: 100%; margin: 0; padding: 20px;"
                                                            align="left">
                                                            <h1 class="h1"
                                                                style="padding-
top: 0; padding-bottom: 0; font-weight: 500; vertical-align: baseline; font-size:
36px; line-height: 43.2px; margin: 0;"
                                                                align="left">{%
block header %}{% endblock %}</h1>
```

```
                                                                <table class="s-2 w-
full" role="presentation" border="0"

cellpadding="0" cellspacing="0" style="width: 100%;"
                                                                       width="100%">
                                                                    <tbody>
                                                                    <tr>
                                                                        <td
style="line-height: 8px; font-size: 8px; width: 100%; height: 8px; margin: 0;"

align="left" width="100%" height="8">
                                                                             
                                                                        </td>
                                                                    </tr>
                                                                    </tbody>
                                                                </table>
                                                                <table class="s-5 w-
full" role="presentation" border="0"

cellpadding="0" cellspacing="0" style="width: 100%;"
                                                                       width="100%">
                                                                    <tbody>
                                                                    <tr>
                                                                        <td
style="line-height: 20px; font-size: 20px; width: 100%; height: 20px; margin: 0;"

align="left" width="100%" height="20">
                                                                             
                                                                        </td>
                                                                    </tr>
                                                                    </tbody>
                                                                </table>
                                                                <table    class="hr"
role="presentation" border="0"

cellpadding="0" cellspacing="0" style="width: 100%;">
                                                                    <tbody>
                                                                    <tr>
                                                                        <td
style="line-height: 24px; font-size: 16px; border-top-width: 1px; border-top-
color: #e2e8f0; border-top-style: solid; height: 1px; width: 100%; margin: 0;"

align="left">
                                                                        </td>
                                                                    </tr>
                                                                    </tbody>
                                                                </table>
                                                                <table class="s-5 w-
full" role="presentation" border="0"

cellpadding="0" cellspacing="0" style="width: 100%;"
                                                                       width="100%">
                                                                    <tbody>
                                                                    <tr>
                                                                        <td
style="line-height: 20px; font-size: 20px; width: 100%; height: 20px; margin: 0;"

align="left" width="100%" height="20">
                                                                             
                                                                        </td>
                                                                    </tr>
                                                                    </tbody>
                                                                </table>
```

```
                                                                    <div class="space-y-
3">
                                                                        <p  class="text-
gray-700"
                                                                           style="line-
height: 24px; font-size: 16px; color: #4a5568; width: 100%; margin: 0;"
                                                                           align="left">
                                                                            {%     block
text %} {% endblock %}
                                                                        </p>
                                                                    </div>
                                                                </td>
                                                            </tr>
                                                            </tbody>
                                                        </table>
                                                    </td>
                                                </tr>
                                                </tbody>
                                            </table>
                                            <table          class="s-10          w-full"
role="presentation" border="0" cellpadding="0"
                                                   cellspacing="0"  style="width:  100%;"
width="100%">
                                                <tbody>
                                                <tr>
                                                    <td style="line-height:  40px;  font-
size: 40px; width: 100%; height: 40px; margin: 0;"
                                                        align="left"        width="100%"
height="40">
                                                         
                                                    </td>
                                                </tr>
                                                </tbody>
                                            </table>
                                        </td>
                                    </tr>
                                    </tbody>
                                </table>
                                <!--[if (gte mso 9)|(IE)]>
                                </td>
                                </tr>
                                </tbody>
                                </table>
                                <![endif]-->
                            </td>
                        </tr>
                        </tbody>
                    </table>
                </td>
            </tr>
            </tbody>
        </table>
        </body>
        </html>
```

Файл: templates/email/__email_confirmation_base_not_rendered.html

```
{% extends 'email/__email_base.html' %}
{% block content %}
    <div class="card my-10">
        <div class="card-body">
            <h1 class="h1 mb-2">HEADER</h1>
```

```
            <hr>
            <div class="space-y-3">
                <p class="text-gray-700">
                    TEXT
                </p>
                <hr>
                <a          class="btn          btn-primary"          href="#"
target="_blank">Підтвердити</a>
            </div>
        </div>
    </div>
{% endblock %}
```

Файл: templates/email/__email_base.html

```
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    {#title is not needed for email#}
</head>
<body class="bg-light">
<div class="container">
    {% block content %}
        {# VITALY IMPORTANT: due to the fact that email html only supports internal
styles #}
        {# email`s are compliled using boostrap-email ruby and all auxiliary are
__*.html #}
    {% endblock %}
</div>
</body>
</html>
```

Файл: templates/email/regular_email.html

```
{% extends 'email/__regular_email_rendered_base.html' %}
{% block header %}
    {{ header }}
{% endblock %}
{% block text %}
    {{ text }}
{% endblock %}
```

Файл: templates/email/__email_confirmation_base_rendered.html

```
<!DOCTYPE     html     PUBLIC     "-//W3C//DTD     XHTML     1.0     Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html lang="uk">
<head>
    <!-- Compiled with Bootstrap Email version: 1.3.0 -->
    <meta http-equiv="x-ua-compatible" content="ie=edge">
    <meta name="x-apple-disable-message-reformatting">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta  name="format-detection"  content="telephone=no,  date=no,  address=no,
email=no">
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <meta charset="UTF-8">
    <style type="text/css">
        body, table, td {
            font-family: Helvetica, Arial, sans-serif !important
        }

        .ExternalClass {
```

```
            width: 100%
        }

        .ExternalClass, .ExternalClass p, .ExternalClass span, .ExternalClass
font, .ExternalClass td, .ExternalClass div {
            line-height: 150%
        }

        a {
            text-decoration: none
        }

        * {
            color: inherit
        }

        a[x-apple-data-detectors], u + #body a, #MessageViewBody a {
            color: inherit;
            text-decoration: none;
            font-size: inherit;
            font-family: inherit;
            font-weight: inherit;
            line-height: inherit
        }

        img {
            -ms-interpolation-mode: bicubic
        }

        table:not([class^=s-]) {
            font-family: Helvetica, Arial, sans-serif;
            mso-table-lspace: 0pt;
            mso-table-rspace: 0pt;
            border-spacing: 0px;
            border-collapse: collapse
        }

        table:not([class^=s-]) td {
            border-spacing: 0px;
            border-collapse: collapse
        }

        @media screen and (max-width: 600px) {
            .w-full, .w-full > tbody > tr > td {
                width: 100% !important
            }

            *[class*=s-lg-] > tbody > tr > td {
                font-size: 0 !important;
                line-height: 0 !important;
                height: 0 !important
            }

            .s-2 > tbody > tr > td {
                font-size: 8px !important;
                line-height: 8px !important;
                height: 8px !important
            }

            .s-3 > tbody > tr > td {
                font-size: 12px !important;
                line-height: 12px !important;
                height: 12px !important
```

```
            }

            .s-5 > tbody > tr > td {
                font-size: 20px !important;
                line-height: 20px !important;
                height: 20px !important
            }

            .s-10 > tbody > tr > td {
                font-size: 40px !important;
                line-height: 40px !important;
                height: 40px !important
            }
        }
    </style>
</head>
<body class="bg-light"
      style="outline: 0; width: 100%; min-width: 100%; height: 100%; -webkit-
text-size-adjust: 100%; -ms-text-size-adjust: 100%; font-family: Helvetica,
Arial, sans-serif; line-height: 24px; font-weight: normal; font-size: 16px; -moz-
box-sizing: border-box; -webkit-box-sizing: border-box; box-sizing: border-box;
color: #000000; margin: 0; padding: 0; border-width: 0;"
      bgcolor="#f7fafc">
<table class="bg-light body" valign="top" role="presentation" border="0"
cellpadding="0" cellspacing="0"
       style="outline: 0; width: 100%; min-width: 100%; height: 100%; -webkit-
text-size-adjust: 100%; -ms-text-size-adjust: 100%; font-family: Helvetica,
Arial, sans-serif; line-height: 24px; font-weight: normal; font-size: 16px; -moz-
box-sizing: border-box; -webkit-box-sizing: border-box; box-sizing: border-box;
color: #000000; margin: 0; padding: 0; border-width: 0;"
       bgcolor="#f7fafc">
    <tbody>
    <tr>
        <td valign="top" style="line-height: 24px; font-size: 16px; margin: 0;"
align="left" bgcolor="#f7fafc">
            <table class="container" role="presentation" border="0"
cellpadding="0" cellspacing="0"
                   style="width: 100%;">
                <tbody>
                <tr>
                    <td align="center" style="line-height: 24px; font-size: 16px;
margin: 0; padding: 0 16px;">
                        <!--[if (gte mso 9)|(IE)]>
                        <table align="center" role="presentation">
                            <tbody>
                            <tr>
                                <td width="600">
                        <![endif]-->
                        <table align="center" role="presentation" border="0"
cellpadding="0" cellspacing="0"
                               style="width: 100%; max-width: 600px; margin: 0
auto;">
                            <tbody>
                            <tr>
                                <td style="line-height: 24px; font-size: 16px;
margin: 0;" align="left">
                                    <table class="s-10 w-full"
role="presentation" border="0" cellpadding="0"
                                           cellspacing="0" style="width: 100%;"
width="100%">
                                        <tbody>
                                        <tr>
```

```
                                        <td style="line-height: 40px; font-
size: 40px; width: 100%; height: 40px; margin: 0;"
                                            align="left"        width="100%"
height="40">
                                             
                                        </td>
                                    </tr>
                                    </tbody>
                                </table>
                                <table    class="card"    role="presentation"
border="0" cellpadding="0" cellspacing="0"
                                       style="border-radius:   6px;   border-
collapse: separate !important; width: 100%; overflow: hidden; border: 1px solid
#e2e8f0;"
                                       bgcolor="#ffffff">
                                    <tbody>
                                    <tr>
                                        <td style="line-height: 24px; font-
size: 16px; width: 100%; margin: 0;"
                                            align="left" bgcolor="#ffffff">
                                            <table         class="card-body"
role="presentation" border="0" cellpadding="0"
                                                   cellspacing="0"
style="width: 100%;">
                                                <tbody>
                                                <tr>
                                                    <td  style="line-height:
24px; font-size: 16px; width: 100%; margin: 0; padding: 20px;"
                                                        align="left">
                                                        <h1 class="h1"
                                                            style="padding-
top: 0; padding-bottom: 0; font-weight: 500; vertical-align: baseline; font-size:
36px; line-height: 43.2px; margin: 0;"
                                                            align="left">{%
block header %}{% endblock %}</h1>
                                                        <table class="s-2 w-
full" role="presentation" border="0"
cellpadding="0" cellspacing="0" style="width: 100%;"
                                                           width="100%">
                                                            <tbody>
                                                            <tr>
                                                                <td
style="line-height: 8px; font-size: 8px; width: 100%; height: 8px; margin: 0;"
align="left" width="100%" height="8">
                                                                     
                                                                </td>
                                                            </tr>
                                                            </tbody>
                                                        </table>
                                                        <table class="s-5 w-
full" role="presentation" border="0"
cellpadding="0" cellspacing="0" style="width: 100%;"
                                                           width="100%">
                                                            <tbody>
                                                            <tr>
                                                                <td
style="line-height: 20px; font-size: 20px; width: 100%; height: 20px; margin: 0;"
align="left" width="100%" height="20">
                                                                     
```

```
                                                                    </td>
                                                                </tr>
                                                                </tbody>
                                                            </table>
                                                            <table    class="hr"
role="presentation" border="0"

cellpadding="0" cellspacing="0" style="width: 100%;">
                                                                <tbody>
                                                                <tr>
                                                                    <td
style="line-height: 24px; font-size: 16px; border-top-width: 1px; border-top-
color: #e2e8f0; border-top-style: solid; height: 1px; width: 100%; margin: 0;"

align="left">
                                                                    </td>
                                                                </tr>
                                                                </tbody>
                                                            </table>
                                                            <table class="s-5 w-
full" role="presentation" border="0"

cellpadding="0" cellspacing="0" style="width: 100%;"
                                                                   width="100%">
                                                                <tbody>
                                                                <tr>
                                                                    <td
style="line-height: 20px; font-size: 20px; width: 100%; height: 20px; margin: 0;"

align="left" width="100%" height="20">
                                                                         
                                                                    </td>
                                                                </tr>
                                                                </tbody>
                                                            </table>
                                                            <div class="space-y-
3">
                                                                <p  class="text-
gray-700"
                                                                   style="line-
height: 24px; font-size: 16px; color: #4a5568; width: 100%; margin: 0;"
                                                                   align="left">
                                                                    {%     block
text %}{% endblock %}
                                                                </p>
                                                                <table class="s-
3 w-full" role="presentation" border="0"

cellpadding="0" cellspacing="0"

style="width: 100%;" width="100%">
                                                                    <tbody>
                                                                    <tr>
                                                                        <td
style="line-height: 12px; font-size: 12px; width: 100%; height: 12px; margin: 0;"

align="left" width="100%" height="12">

 
                                                                        </td>
                                                                    </tr>
                                                                    </tbody>
                                                                </table>
```

```
                                                                    <table class="s-
5 w-full" role="presentation" border="0"

cellpadding="0" cellspacing="0"

style="width: 100%;" width="100%">
                                                                        <tbody>
                                                                        <tr>
                                                                            <td
style="line-height: 20px; font-size: 20px; width: 100%; height: 20px; margin: 0;"

align="left" width="100%" height="20">

 
                                                                            </td>
                                                                        </tr>
                                                                        </tbody>
                                                                    </table>
                                                                    <table
class="hr" role="presentation" border="0"

cellpadding="0" cellspacing="0"

style="width: 100%;">
                                                                        <tbody>
                                                                        <tr>
                                                                            <td
style="line-height: 24px; font-size: 16px; border-top-width: 1px; border-top-
color: #e2e8f0; border-top-style: solid; height: 1px; width: 100%; margin: 0;"

align="left">
                                                                            </td>
                                                                        </tr>
                                                                        </tbody>
                                                                    </table>
                                                                    <table class="s-
5 w-full" role="presentation" border="0"

cellpadding="0" cellspacing="0"

style="width: 100%;" width="100%">
                                                                        <tbody>
                                                                        <tr>
                                                                            <td
style="line-height: 20px; font-size: 20px; width: 100%; height: 20px; margin: 0;"

align="left" width="100%" height="20">

 
                                                                            </td>
                                                                        </tr>
                                                                        </tbody>
                                                                    </table>
                                                                    <table
class="btn btn-primary" role="presentation"

border="0" cellpadding="0" cellspacing="0"

style="border-radius: 6px; border-collapse: separate !important;">
                                                                        <tbody>
                                                                        <tr>
                                                                            <td
style="line-height: 24px; font-size: 16px; border-radius: 6px; margin: 0;"
```

```
align="center" bgcolor="#0d6efd">
                                                                                <a
href="{% block link %}{% endblock %}"

target="_blank"

style="color: #ffffff; font-size: 16px; font-family: Helvetica, Arial, sans-serif;
text-decoration: none; border-radius: 6px; line-height: 20px; display: block;
font-weight: normal; white-space: nowrap; background-color: #0d6efd; padding: 8px
12px; border: 1px solid #0d6efd;">

{% block button_text %} {% endblock %}</a>
                                                                                    </td>
                                                                                </tr>
                                                                                </tbody>
                                                                            </table>
                                                                        </div>
                                                                    </td>
                                                                </tr>
                                                                </tbody>
                                                            </table>
                                                        </td>
                                                    </tr>
                                                    </tbody>
                                                </table>
                                                <table          class="s-10          w-full"
role="presentation" border="0" cellpadding="0"
                                                       cellspacing="0"  style="width:  100%;"
width="100%">
                                                    <tbody>
                                                    <tr>
                                                        <td style="line-height:  40px;  font-
size: 40px; width: 100%; height: 40px; margin: 0;"
                                                                align="left"        width="100%"
height="40">
                                                                 
                                                        </td>
                                                    </tr>
                                                    </tbody>
                                                </table>
                                            </td>
                                        </tr>
                                        </tbody>
                                    </table>
                                    <!--[if (gte mso 9)|(IE)]>
                                    </td>
                                    </tr>
                                    </tbody>
                                    </table>
                                    <![endif]-->
                                </td>
                            </tr>
                            </tbody>
                        </table>
                    </td>
                </tr>
                </tbody>
            </table>
            </body>
            </html>
```

Файл: templates/email/email_confirmation.html

```
{% extends 'email/__email_confirmation_base_rendered.html' %}
{% block header %}
    Підтвердьте реєстрацію
{% endblock %}
{% block text %}
    Натисніть на кнопку знизу, щоб підтвердити реєстрацію на energokodros.
    Якщо ви не надсилали запит, можете сміливо видаляти цей лист.
    Не натискається кнопка? Спробуйте скопіювати посилання вручну та перейти по
ньому
    {{ link }}
{% endblock %}
{% block link %}
    {{ link }}
{% endblock %}
{% block button_text %}
    Підтвердити
{% endblock %}
```