ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ТЕХНОЛОГІЙ ЦИФРОВОГО РОЗВИТКУ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «інтерактивний веб-сервіс для порівняння термінів окупності інвестицій у енергоефективність з використанням технологій typescript, nextjs, nestjs та prisma»

на здобуття освітнього ступеня бакалавра
зі спеціальності F2 Інженерія програмного забезпечення
освітньо-професійної програми «Технології цифрового розвитку»

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

_____ Дмитро ГОНЧАРЕНКО
(підпис)

Виконав: здобувач(ка) вищої освіти групи ТЦР-42
Дмитро ГОНЧАРЕНКО

Керівник: Вікторія ЖЕБКА
д-р техн. наук, проф.

Рецензент: _____

Київ 2026

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
Навчально-науковий інститут інформаційних технологій

Кафедра Технологій цифрового розвитку

Ступінь вищої освіти Бакалавр

Спеціальність 121 Інженерія програмного забезпечення

Освітньо-професійна програма «Технології цифрового розвитку»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Технологій цифрового розвитку
_____Вікторія ЖЕБКА
«____»_____2026 року

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Гончаренку Дмитру Олександровичу

1. Тема кваліфікаційної роботи: «Інтерактивний веб-сервіс для порівняння термінів окупності інвестицій у енергоефективність з використанням технологій TypeScript, NextJS, NestJS та Prisma»
керівник кваліфікаційної роботи д-р техн. наук, професор Вікторія ЖЕБКА,
затверджені наказом Державного університету інформаційно-комунікаційних технологій від «20» лютого 2026 року № 51.

2. Строк подання кваліфікаційної роботи «__» травня 2026 року.

3. Вихідні дані до кваліфікаційної роботи: теоретичні відомості про методи інвестиційного аналізу та дисконтування грошових потоків, опис математичних моделей розрахунку показників ефективності інвестицій (PP, NPV, DPP, IRR, PI), нормативно-правова база українських грантових програм підтримки енергоефективності 2025–2027 рр. («Енергодім», «ГрінДІМ»), технічна документація з описом фреймворків Next.js 14, NestJS 10, ORM Prisma 5 та супутніх бібліотек для розробки повностекового веб-застосунку на TypeScript.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Огляд та аналіз існуючих методів та програмних рішень для оцінювання термінів окупності інвестицій у енергоефективність в контексті задачі розробки інтерактивного веб-сервісу з урахуванням українських грантових програм.
2. Проектування інтерактивного веб-сервісу для порівняння термінів окупності інвестицій у енергоефективність: побудова математичних моделей п'яти інвестиційних показників, проєктування архітектури за C4-моделлю, ER-діаграми та схеми бази даних на базі Prisma.
3. Програмна реалізація та опис функціонування інтерактивного веб-сервісу для порівняння термінів окупності інвестицій у енергоефективність з використанням технологій TypeScript, Next.js, NestJS та Prisma.
4. Тестування веб-сервісу.

5. Перелік графічного матеріалу: *презентація*
1. Аналіз аналогів.
2. Вимоги до програмного забезпечення.
3. Програмні засоби реалізації.
4. Діаграма варіантів використання.
5. Алгоритм роботи застосунку.
6. Діаграма класів.
7. Екранні форми.
8. Апробація результатів дослідження

6. Дата видачі завдання «20» лютого 2026 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Підбір науково-технічної літератури та аналіз предметної області	01.04 – 07.04.2026	
2	Аналіз та дослідження існуючих аналогів	07.04 – 10.04.2026	
3	Проєктування архітектури (C4-модель, ER-діаграма, Prisma-схема) та розробка математичних моделей (PP, NPV, DPP, IRR, PI)	10.04 – 12.04.2026	
4	Реалізація серверної частини (NestJS + Prisma)	12.04 – 15.04.2026	
5	Реалізація клієнтської частини (Next.js 14)	15.04 – 18.04.2026	
6	Налаштування інфраструктури (Docker, Docker Compose)	18.04 – 20.04.2026	
7	Розробка та виконання тестів (unit, integration, E2E)	20.04 – 22.04.2026	

8	Написання пояснювальної записки та підготовка матеріалів до захисту	22.04 – 01.05.2026	
9	Попередній захист роботи	11.05.2026	

Здобувач вищої освіти _____ Дмитро ГОНЧАРЕНКО
(підпис)

Керівник
кваліфікаційної роботи _____ Вікторія ЖЕБКА
(підпис)

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 67 стор., 4 рис., 20 табл., 27 джерел.

Мета роботи – підвищення ефективності прийняття рішень щодо інвестування в енергоефективне обладнання шляхом автоматизованого порівняльного розрахунку термінів окупності інвестицій з урахуванням українських грантових програм за допомогою розробки інтерактивного веб-сервісу.

Об'єкт дослідження – процес оцінювання термінів окупності інвестицій у енергоефективне обладнання для домогосподарств, ОСББ та малого і середнього бізнесу з урахуванням державних програм фінансової підтримки.

Предмет дослідження – методи та програмні засоби інтерактивного розрахунку й порівняння інвестиційних показників (*PP*, *NPV*, *DPP*, *IRR*, *PI*) в умовах грантового співфінансування.

Короткий зміст роботи: У роботі проаналізовано українські грантові програми 2025–2027 рр. («Енергодім», «ГрінДІМ», «5-7-9%», ПРООН, ЄІБ) та типові технології енергоефективності для трьох профілів споживачів. Побудовано математичні моделі п'яти інвестиційних показників (*PP*, *NPV*, *DPP*, *IRR*, *PI*), параметризовані з урахуванням ставки гранту γ та темпу зростання тарифу α, що верифіковано двома розрахунковими прикладами. Спроєктовано трирівневу C4-архітектуру, ER-модель на шести сутностях та Prisma-схему, а також розроблено серверну частину на NestJS 10 з вісьмома REST-ендпоінтами та клієнтську частину на Next.js 14 з сімома React-компонентами. Виконано модульне, інтеграційне та E2E-тестування (38 тестів, покриття 87%) і підтверджено коректність розрахунків PP, NPV та IRR.

Ключові слова: ЕНЕРГОЕФЕКТИВНІСТЬ, ОКУПНІСТЬ ІНВЕСТИЦІЙ, ВЕБ-СЕРВІС, ЧИСТА ПРИВЕДЕНА ВАРТІСТЬ, ВНУТРІШНЯ НОРМА ПРИБУТКОВОСТІ, ГРАНТОВЕ ФІНАНСУВАННЯ, TYPESCRIPT, NEXT.JS, NESTJS, PRISMA.

ЗМІСТ

УМОВНІ ПОЗНАЧЕННЯ

Позначення	**Розшифрування**
ПЗ	Програмне забезпечення
БД	База даних
СУБД	Система управління базами даних
API	Прикладний програмний інтерфейс (Application Programming Interface)
REST	Передача репрезентативного стану (Representational State Transfer)
HTTP	Протокол передачі гіпертексту (HyperText Transfer Protocol)
HTTPS	Захищений протокол передачі гіпертексту
JSON	Формат обміну даними (JavaScript Object Notation)
JWT	Веб-токен JSON (JSON Web Token)
UUID	Універсально унікальний ідентифікатор
SPA	Односторінковий застосунок (Single Page Application)
SSR	Серверний рендеринг (Server-Side Rendering)
ORM	Об'єктно-реляційне відображення (Object Relational Mapping)
DTO	Об'єкт передачі даних (Data Transfer Object)
UML	Уніфікована мова моделювання (Unified Modeling Language)
IDE	Інтегроване середовище розробки (Integrated Development Environment)
UI	Інтерфейс користувача (User Interface)
UX	Досвід користувача (User Experience)
RBAC	Контроль доступу на основі ролей (Role-Based Access Control)
CORS	Обмін ресурсами між Origin'ами (Cross-Origin Resource Sharing)
XSS	Міжсайтовий скриптинг (Cross-Site Scripting)
CSRF	Підробка міжсайтових запитів (Cross-Site Request Forgery)
TLS	Протокол безпеки транспортного шару (Transport Layer Security)

Позначення	**Розшифрування**
CI/CD	Безперервна інтеграція / Безперервне розгортання (Continuous Integration / Continuous Deployment)
TypeScript	Мова програмування з статичною типізацією на основі JavaScript
JavaScript	Мова програмування для веб-браузерів
Node.js	Середовище виконання JavaScript на сервері
Next.js	Фреймворк для React із SSR та App Router
React	Бібліотека для побудови UI-компонентів
NestJS	Прогресивний фреймворк для розробки серверних застосунків
Express	Мініджалізований веб-фреймворк для Node.js
Prisma	ORM та конструктор запитів для Node.js/TypeScript
PostgreSQL	Реляційна СУБД з відкритим кодом
Docker	Контейнеризація та оркестрація застосунків
Docker Compose	Інструмент для визначення та запуску мультиконтейнерних застосунків
Jest	Фреймворк для модульного тестування JavaScript/TypeScript
Supertest	Бібліотека для тестування HTTP-запитів
Playwright	Інструмент для автоматизованого E2E-тестування браузерів
Recharts	Бібліотека React для побудови діаграм та графіків
Tailwind CSS	Утилітарна бібліотека CSS для стилізації
class-validator	Декоратори для валідації об'єктів TypeScript
Zod	Бібліотека для схемної валідації TypeScript
React Hook Form	Бібліотека для управління станом форм у React
Swagger	Інструмент для документування REST API
bcrypt	Алгоритм для хешування паролів
GitHub	Платформа для версіонування коду
GitHub Actions	Сервіс автоматизації вибраних завдань в GitHub
C4-модель	Ієрархічна система архітектурного документування (4 рівні)
ER-діаграма	Діаграма «Сутність–Зв'язок» для моделювання БД
PP	Простий термін окупності (Payback Period)

Позначення	Розшифрування
DPP	Дисконтований термін окупності (Discounted Payback Period)
NPV	Чиста приведена вартість (Net Present Value)
IRR	Внутрішня норма прибутковості (Internal Rate of Return)
PI	Індекс прибутковості (Profitability Index)
СЕС	Сонячна електростанція
ІТП	Індивідуальний тепловий пункт
ОСББ	Об'єднання співвласників багатоквартирного будинку
МСБ	Мале та середнє підприємство
γ (гама)	Коефіцієнт грантового покриття (доля гранту у вартості)
α (альфа)	Темп зростання тарифу на енергоносії
r	Ставка дисконтування (облікова ставка, вартість капіталу)
C_inv	Повна вартість інвестиції
C_net	Чисті інвестиційні витрати з урахуванням гранту
CF_t	Чистий грошовий потік у рік t
E_save	Щорічна економія енергоресурсу
p_t	Тариф на енергоносій у рік t
OM_t	Витрати на обслуговування обладнання у рік t
«Енергодім»	Найменування грантової програми Фонду енергоефективності України
«ГрінДІМ»	Програма підтримки розподіленої відновлюваної енергії для ОСББ
«5-7-9%»	Програма пільгового кредитування для малого бізнесу та фізичних осіб
ПРООН	Програма розвитку ООН
ЄІБ	Європейський інвестиційний банк
НКРЕКП	Національна комісія, що здійснює державне регулювання у сферах енергетики та комунальних послуг
НБУ	Національний банк України
Фонд ЕЕ	Фонд енергоефективності України
ДБН	Державні будівельні норми України

ВСТУП

Актуальність теми. Наслідки повномасштабної військової агресії російської федерації проти України спричинили системну кризу енергетичної інфраструктури: руйнування генеруючих потужностей, регулярні блекаути та зростання тарифів формують стійкий попит на енергетичну автономність домогосподарств, об'єднань співвласників багатоквартирних будинків (ОСББ) та малого і середнього бізнесу (МСБ). Відповіддю держави стало розгортання у 2025–2027 рр. комплексу програм фінансової підтримки енергоефективних інвестицій – «Енергодім», «ГрінДІМ» та «Доступні кредити 5-7-9 %», що передбачають часткове грантове співфінансування або пільгове кредитування.

Попри масштаб програм, прийняття інвестиційного рішення залишається для кінцевого споживача складним завданням через потребу одночасного врахування п'яти інвестиційних показників – простого терміну окупності (PP), чистої теперішньої вартості (NPV), дисконтованого терміну окупності (DPP), внутрішньої норми дохідності (IRR) та індексу прибутковості (PI). Аналіз існуючих онлайн-калькуляторів виявив відсутність інтеграції параметрів українських грантових програм, повного набору показників у єдиному інтерфейсі та функціоналу порівняння альтернативних проєктів. Це зумовлює актуальність розробки спеціалізованого веб-сервісу, який інтегрує специфіку українських програм 2025–2027 рр. у математичну модель інвестиційного аналізу.

Аналіз останніх досліджень і публікацій. Питання оцінювання ефективності інвестицій в енергозбереження висвітлені у працях вітчизняних та закордонних дослідників, де обґрунтовано застосування дисконтованих показників NPV та IRR для проєктів відновлюваної енергетики, проте без прив'язки до моделі грантового співфінансування. Окрему групу становлять дослідження архітектури веб-застосунків на базі Next.js та NestJS і типобезпечних ORM-систем. Водночас жодне з опрацьованих джерел не поєднує математичну модель інвестиційного аналізу з

урахуванням грантів та реалізацію у вигляді повнофункціонального веб-сервісу для українських реалій.

Мета роботи – підвищення ефективності прийняття рішень щодо інвестування в енергоефективне обладнання шляхом автоматизованого порівняльного розрахунку термінів окупності з урахуванням українських грантових програм за допомогою розробки інтерактивного веб-сервісу.

Для досягнення мети у бакалаврській роботі виконано наступні завдання:

– проаналізувати чинні українські програми фінансування енергоефективних заходів та класифікацію технологій для домогосподарств, ОСББ і МСБ;

– побудувати математичні моделі п’яти інвестиційних показників (PP, NPV, DPP, IRR, PI) з урахуванням грантового співфінансування;

– спроєктувати архітектуру системи за C4-моделлю та схему бази даних;

– реалізувати серверну (NestJS) та клієнтську (Next.js) частини сервісу;

– виконати тестування системи та верифікацію коректності розрахунків.

Об’єкт дослідження – процес оцінювання термінів окупності інвестицій у енергоефективне обладнання з урахуванням державних програм фінансової підтримки.

Предмет дослідження – методи та програмні засоби інтерактивного розрахунку й порівняння інвестиційних показників в умовах грантового співфінансування.

*Методи дослідження***.** У роботі застосовано методи системного аналізу та класифікації для дослідження предметної області, методи математичного моделювання та дисконтування грошових потоків для побудови моделей інвестиційних показників, чисельний метод Ньютона–Рафсона для розрахунку IRR, методи об’єктно-орієнтованого проєктування з використанням нотацій C4

Model, UML та ER-діаграм, а також методи модульного, інтеграційного та E2E-тестування.

*Наукова новизна одержаних результатів***.** Уперше розроблено комплексну модель порівняльного розрахунку п’яти інвестиційних показників, параметризовану під специфіку українських грантових програм 2025–2027 рр., яка, на відміну від існуючих калькуляторів, забезпечує одночасне врахування відсотка гранту, темпу зростання тарифу та ставки дисконтування у єдиному обчислювальному контурі.

*Практична значущість одержаних результатів***.** Розроблений веб-сервіс може бути застосований домовласниками та головами ОСББ для вибору технологій і програм фінансування, підприємствами МСБ – для обчислення показників за проєктами когенерації та СЕС, а також енергоаудиторами і фінансовими аналітиками банків-партнерів. Відкрите REST API дозволяє інтегрувати розрахунковий модуль у сторонні інформаційні системи.

*Апробація результатів та публікації***.** Основні положення і результати бакалаврської роботи доповідались на науково-практичних конференціях, що проходили на базі Державного університету інформаційно-комунікаційних технологій, а саме:

- Гончаренко Д.О. «Інтерактивний веб-сервіс для порівняння термінів окупності інвестицій у енергоефективність з використанням технологій TypeScript, NextJS, NestJS та Prisma» // Матеріали VI Всеукраїнської науково-практичної конференції «Сучасні інтелектуальні інформаційні технології в науці та освіті». Збірник тез. К: ДУІКТ, 2026, (подано до друку)
- Гончаренко Д.О. «Порівняльний аналіз програмних рішень для оцінювання окупності енергоефективних інвестицій та обґрунтування функціональних вимог за методологією MoSCoW» // Матеріали всеукраїнської науково-технічної конференції «Технології цифрового розвитку: програмні системи та децентралізація». Збірник тез. К: ДУІКТ, 2026, (подано до друку)

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ОБҐРУНТУВАННЯ ТЕХНОЛОГІЧНИХ РІШЕНЬ

1.1 Аналіз сучасного стану інвестицій в енергоефективність в Україні

Масштабне руйнування об'єктів критичної енергетичної інфраструктури внаслідок збройної агресії 2022–2025 років суттєво змінило пріоритети державної енергетичної політики України. За даними Держенергоефективності, дефіцит генеруючих потужностей у 2024–2025 роках зумовив зростання попиту на децентралізовані рішення для автономного енергопостачання – сонячні електростанції (СЕС), теплові насоси, системи накопичення енергії та когенераційні установки [4]. В цих умовах держава суттєво розширила програми стимулювання інвестицій в енергоефективність, формуючи нову архітектуру фінансової підтримки для фізичних осіб, ОСББ та суб'єктів малого і середнього бізнесу (МСБ).

Ключовим механізмом державного стимулювання для ОСББ є програма «Енергодім», що реалізується через Фонд енергоефективності України. У рамках Пакету Б передбачено відшкодування до 70% вартості робіт із термомодернізації будівель, а гранту підлягає сума до 25 млн грн на один об'єкт . Програма охоплює утеплення фасадів і покрівель, заміну вікон та дверей, модернізацію систем опалення та гарячого водопостачання. Починаючи з лютого 2026 року, розширено умови компоненту «Індивідуальний тепловий пункт (ІТП)»: грант до 70% тепер поширюється також на гібридні інвертори у складі ІТП, що дозволяє суміщати централізоване теплопостачання з резервними джерелами живлення [1].

Програма «ГрінДІМ», що стартувала 18 лютого 2026 року, адресована ОСББ і спрямована на підтримку розподіленої відновлюваної генерації. Умови програми

передбачають компенсацію до 70% вартості сонячних електростанцій, теплових насосів та акумуляторних систем накопичення енергії з максимальним обсягом гранту від 2 до 4 млн грн залежно від типу об'єкта[5]. Для генераторів резервного живлення передбачено знижений рівень відшкодування – до 50% із максимальним грантом 1 млн грн. Запровадження «ГрінДІМ» суттєво розширило перелік технологій, доступних для грантового фінансування, та підвищило привабливість комплексних інвестиційних рішень, що поєднують СЕС з накопичувачами.

Для фізичних осіб – власників приватних будинків – основним інструментом залишається програма пільгового кредитування «5-7-9%». З 2025 року її дія поширена на придбання та встановлення СЕС і акумуляторів: максимальна сума кредиту для фізичної особи становить 150 тис. грн за відсотковою ставкою в діапазоні 5–9% річних [15]. Це дозволяє фінансувати, зокрема, базові СЕС потужністю до 5 кВт із накопичувачами ємністю 10–15 кВт·год, забезпечуючи часткову автономність приватного домогосподарства.

Для суб'єктів МСБ програма «Доступні кредити 5-7-9%» зазнала суттєвого розширення відповідно до змін, що набули чинності 28 січня 2026 року. Максимальна сума кредиту збільшена до 250 млн грн, ставка варіюється від 1% (для підприємств з постраждалих територій) до 9% річних. До переліку цільових напрямів кредитування включено: когенераційні установки, сонячні та вітрові електростанції, системи зберігання енергії. Зазначені зміни відкривають можливості для фінансування масштабних промислових об'єктів відновлюваної енергетики.

Міжнародна підтримка програм енергоефективності в Україні є вагомим доповненням до державних механізмів. Програма ООН з розвитку (ПРООН) у лютому 2026 року оголосила про виділення 230 млн доларів США на розвиток когенерації для міських громад, що є найбільшим одноразовим траншем міжнародної допомоги в цьому секторі[10]. Паралельно Європейський інвестиційний банк (ЄІБ) підтвердив зобов'язання щодо фінансування в обсязі 600 млн євро на 2026–2027 роки в рамках Ukraine Investment Framework з пріоритетом

на відновлення та підвищення енергоефективності [9]. Сукупний потенціал зовнішнього фінансування у поєднанні з національними програмами формує безпрецедентну за масштабом базу для інвестицій в енергоефективність.

Проте зростання обсягів і різноманіття доступних програм підтримки породжує нову проблему: інвестор (власник будинку, голова ОСББ, підприємець) не в змозі без допоміжних інструментів коректно порівняти варіанти за ключовими інвестиційними показниками з урахуванням грантів, кредитних ставок і фактичної економії. Це обумовлює актуальність розробки спеціалізованого програмного засобу для автоматизованого розрахунку та порівняння термінів окупності інвестицій в енергоефективність [11; 16].

1.2 Класифікація об'єктів інвестування та профілів користувачів

Диференціація цільових груп потенційних користувачів веб-сервісу є необхідною умовою коректного налаштування розрахункових моделей і формування релевантного інтерфейсу. Аналіз чинних програм фінансування та типових інвестиційних рішень дозволяє виокремити три базові профілі: (1) приватний домовласник, (2) об'єднання співвласників багатоквартирного будинку (ОСББ), (3) суб'єкт малого і середнього бізнесу (МСБ). Таксономія об'єктів інвестування та відповідних параметрів наведена у таблиці 1.1.

Вибір конкретної технології для кожного профілю визначається низкою взаємопов'язаних критеріїв. Для приватного домовласника пріоритетними є: термін окупності без урахування грантів (оскільки основним інструментом є кредит), рівень автономності від мережі, а також сумісність із наявною системою опалення [27]. ОСББ вибирають технологію переважно за коефіцієнтом грантового покриття та загальним обсягом щорічної економії на комунальних послугах у перерахунку на одну квартиру. МСБ оцінює технології насамперед за IRR (внутрішньою

нормою прибутковості) та NPV у доларовому еквіваленті з урахуванням кредитної ставки та цільових програм [11].

Таблиця 1.1

Класифікація об'єктів інвестування та профілів користувачів

Профіль	Технології	Типова вартість	Доступні програми
Приватний домовласник	Тепловий насос «повітря-вода»; СЕС 5–10 кВт + акумулятор; Твердопаливний котел	300 000–600 000 грн; 200 000–400 000 грн; 80 000–200 000 грн	«5-7-9%» (до 150 тис. грн)
ОСББ	ІТП з резервним живленням; Комплексна термомодернізація; СЕС на даху 10–30 кВт	300 000–800 000 грн; від 2 000 000 грн; 400 000–1 000 000 грн	«Енергодім» (до 70%, 25 млн грн); «ГрінДІМ» (до 70%, 2–4 млн грн)
МСБ	Когенерація 100–500 кВт; СЕС 50–200 кВт; Газопоршневі 0,5–2 МВт	$200 000–$500 000; $50 000–$300 000; $500 000–$2 500 000	«5-7-9%» (до 250 млн грн, 1–9%); ПРООН $230 млн; ЄІБ €600 млн

Зазначені відмінності у критеріях вибору унеможливлюють застосування уніфікованої розрахункової моделі без можливості параметризації за профілем. Відповідно, проєктована система повинна підтримувати три режими розрахунку з різними наборами вхідних параметрів і різним відображенням результатів, що безпосередньо впливає на архітектурні рішення як бекенду, так і фронтенду.

1.3 Порівняльний аналіз існуючих програмних рішень

Аналіз ринку програмного забезпечення для оцінки інвестицій в енергоефективність дозволяє виокремити чотири основні типи рішень: (1)

калькулятори комерційних банків, (2) калькулятори виробників обладнання, (3) державні інформаційно-розрахункові системи, (4) міжнародні аналітичні платформи. Кожен тип орієнтований на власну цільову аудиторію і характеризується специфічними функціональними можливостями та обмеженнями.

Банківські калькулятори (ПриватБанк «Зелений тариф», Ощадбанк «Енергетичне кредитування») мають на меті оцінку кредитного навантаження та приблизного терміну погашення позики [9]. Вони дозволяють розрахувати щомісячний платіж і загальну переплату, однак не враховують реальної економії від встановленого обладнання, не підтримують розрахунок NPV чи IRR, а також не адаптовані до специфіки грантових програм – «Енергодім», «ГрінДІМ» або «5-7-9%».

Калькулятори виробників обладнання (SolarEdge Solar Calculator, Danfoss Heat Pump Savings Calculator) дозволяють оцінити річну економію від конкретного продукту, однак орієнтовані виключно на власну лінійку товарів і не підтримують мультитехнологічне порівняння. Ці інструменти не враховують специфіки українських програм фінансування, а дані про тарифи часто є статичними та не оновлюються відповідно до змін в тарифній політиці НЕК «Укренерго».

Державна система Фонду енергоефективності містить базовий інформаційний калькулятор для попереднього розрахунку обсягу гранту в рамках «Енергодім». Проте інструмент обмежений розрахунком лише одного показника – суми відшкодування – і не надає можливості оцінити повний інвестиційний профіль проєкту або порівняти альтернативні технологічні рішення.

Міжнародні платформи (EU Energy Calculator, HOMER Energy Software, RETScreen) забезпечують найбільш повну методологічну базу: підтримують розрахунок PP, NPV, IRR та PI, дозволяють моделювати гібридні системи. Однак вони розроблені для іноземних ринків і не містять актуальних даних щодо українських програм грантового фінансування, норм питомих витрат ПЕР

відповідно до вимог ДБН В.2.6-31:2021 [13], а їх інтерфейс є складним для нетехнічних користувачів.

Детальне порівняння зазначених рішень за ключовими функціональними критеріями наведено у таблиці 1.2.

Таблиця 1.2

Порівняльний аналіз існуючих програмних рішень

Критерій	Банківські калькулятори	Калькулятори виробників	Держ. системи (Фонд ЕЕ)	Міжнародні (HOMER, RETScreen)	Проєктована система
Врахування грантів UA	Ні	Ні	Частково	Ні	Так
Мультитехнологічне порівняння	Ні	Ні	Ні	Так	Так
Розрахунок NPV, IRR, PI	Ні	Ні	Ні	Так	Так
Актуальні тарифи UA	Частково	Ні	Частково	Ні	Так
UX для нетехн. користувачів	Так	Так	Так	Ні	Так
Збереження та порівняння проєктів	Ні	Ні	Ні	Так	Так
Відкритий REST API	Ні	Ні	Ні	Ні	Так

Gap-аналіз виявив три системних недоліки існуючих рішень. По-перше, жоден інструмент не інтегрує українські грантові програми в розрахунок інвестиційних показників, що унеможливлює коректне порівняння варіантів «з грантом» і «без гранту». По-друге, відсутнє рішення, яке поєднувало б методологічну повноту (п'ять показників: PP, NPV, DPP, IRR, PI) з простим інтерфейсом для нетехнічних користувачів. По-третє, жодне з розглянутих рішень не забезпечує збереження й порівняння кількох альтернативних проєктів у єдиному персоналізованому робочому просторі з відкритим API для інтеграції. Усі три недоліки формують чітке обґрунтування для розробки спеціалізованого веб-сервісу [16].

1.4 Обґрунтування вибору технологічного стека

Вибір технологічного стека для реалізації веб-сервісу здійснювався на основі аналізу функціональних вимог системи: наявності складних розрахункових модулів, необхідності забезпечення типобезпечності фінансових обчислень, підтримки рольового доступу, швидкого клієнтського відображення та зручного адміністрування даних.

TypeScript обраний як основна мова програмування для обох частин системи. На відміну від JavaScript, TypeScript забезпечує статичну типізацію, що є критичним чинником для фінансових розрахунків: помилки перетворення типів (наприклад, конкатенація рядка із числовою ставкою дисконтування) виявляються на етапі компіляції, а не у виробничому середовищі [27]. Підтримка інтерфейсів, generics і декораторів дозволяє будувати масштабовані архітектурні рішення без втрати продуктивності розробки.

Next.js 14 з архітектурою App Router обрано для реалізації клієнтської частини. Ключові переваги перед альтернативами: вбудована підтримка Server-Side Rendering (SSR) та React Server Components (RSC), що забезпечує SEO-оптимізацію посадкових сторінок сервісу; система маршрутизації на основі файлової структури,

яка суттєво скорочує бойлерплейт; нативна оптимізація зображень та шрифтів. На відміну від React+Vite, Next.js не потребує окремого налаштування SSR, а порівняно з Angular має значно нижчий поріг входу та меншу розмірність bundle.

NestJS 10 обрано для серверної частини. Модульна архітектура з вбудованим механізмом впровадження залежностей (DI) дозволяє чітко розмежувати відповідальність між модулями (auth, projects, technologies, tariffs, grants). Система декораторів забезпечує декларативне визначення маршрутів, валідацію вхідних даних (class-validator) та автоматичну генерацію Swagger/OpenAPI документації – що є обов'язковою вимогою для відкритого API сервісу. Express і Fastify поступаються NestJS за рівнем структурованості коду для великих модульних додатків.

Таблиця 1.3

Обґрунтування технологічного стека

Технологія	Альтернативи	Роль у системі	Ключова перевага для проєкту
TypeScript 5	JavaScript (ES2022)	Мова всього стека	Статична типізація фінансових обчислень; виявлення помилок на етапі компіляції
Next.js 14 (App Router)	React+Vite, Angular 17	Frontend SPA/SSR	SSR + RSC без додаткового налаштування; SEO для посадкових сторінок; файлова маршрутизація
NestJS 10	Express, Fastify	REST API, бізнес-логіка	Модульна DI-архітектура; декоратори для Swagger; class-validator для вхідних DTO
Prisma 5	TypeORM, Sequelize	ORM, міграції БД	Генерація TypeScript-типів зі схеми; декларативні міграції; Prisma Studio для адміністрування
PostgreSQL 16	MySQL, MongoDB	Реляційна СУБД	ACID-транзакції для фінансових даних; підтримка складних JOIN для звітів порівняння
Tailwind CSS 3	Bootstrap, MUI	Стилізація UI	Utility-first підхід; відсутність CSS-конфліктів; швидке прототипування адаптивних лейаутів

Prisma 5 обрано як ORM для роботи з базою даних PostgreSQL. Головна перевага перед TypeORM та Sequelize – генерація типів TypeScript безпосередньо з Prisma Schema, що забезпечує наскрізну типобезпечність від бази даних до клієнту. Система міграцій Prisma є більш прозорою та декларативною, ніж в TypeORM, а Prisma Studio надає вбудований GUI для перегляду та редагування даних, що прискорює відлагодження. Prisma також суттєво краще документована, що важливо для командної розробки.

Підсумкове обґрунтування технологічного стека наведено у таблиці 1.3.

Обраний стек формує наскрізно типізовану архітектуру: TypeScript-типи визначаються у Prisma Schema, автоматично поширюються через NestJS DTO до Swagger-специфікації, а потім використовуються у Next.js-клієнті через спільні типи. Це мінімізує ризик невідповідності між шарами системи – критично важливої властивості для програмного забезпечення, що здійснює фінансові розрахунки.

Висновки до розділу 1

У першому розділі виконано комплексний аналіз предметної області, що дозволив обґрунтувати доцільність розробки проєктованого веб-сервісу. Встановлено, що чинні державні програми – «Енергодім» (грант до 70%, до 25 млн грн), «ГрінДІМ» (70%/50%, до 4 млн грн) та «5-7-9%» (до 150 тис. грн для фізосіб; до 250 млн грн для МСБ) – у поєднанні з міжнародними механізмами фінансування (ПРООН $230 млн, ЄІБ €600 млн) формують масштабний ринок інвестицій в енергоефективність, який потребує зручних розрахункових інструментів.

Gap-аналіз існуючих програмних рішень виявив три ключових прогалини: відсутність інтеграції українських грантових програм в розрахунок інвестиційних показників; відсутність рішення з повним набором показників (PP, NPV, DPP, IRR, PI) в простому UX; відсутність відкритого API зі збереженням та порівнянням проєктів. Ці прогалини є прямим обґрунтуванням розробки.

Класифікація об'єктів інвестування визначила три профілі користувачів системи з різними параметричними наборами, що безпосередньо впливає на

архітектуру наступних розділів. Обґрунтований технологічний стек (TypeScript + Next.js 14 + NestJS 10 + Prisma 5 + PostgreSQL) забезпечує наскрізну типізацію, модульність і відповідність вимогам масштабованого веб-застосунку з фінансовою логікою.

2. МАТЕМАТИЧНЕ МОДЕЛЮВАННЯ ТА ПРОЄКТУВАННЯ СИСТЕМИ

2.1 Математичні моделі розрахунку показників ефективності інвестицій

Математичний апарат розробленого сервісу ґрунтується на комплексній моделі оцінювання інвестиційної ефективності, яка об'єднує п'ять ключових фінансових показників – PP, NPV, DPP, IRR та PI – із параметрами державного грантового співфінансування. На відміну від традиційних підходів, де грант ураховується лише як разове зменшення первісних витрат [17], у пропонованій моделі грантова підтримка формалізована як окремий параметр, що уможливлює одночасне порівняння проєктів у різних програмах підтримки: «Енергодім», «ГрінДІМ» та «5-7-9%» [1; 5]. Це дозволяє автоматизувати вибір оптимального сценарію фінансування для кожного типу об'єкта.

2.1.1 Модель чистих інвестиційних витрат з урахуванням гранту

Базовим елементом усіх наступних формул є показник чистих інвестиційних витрат C_{net}, що визначається як різниця між повною вартістю проєкту та розміром грантового фінансування:

$$C\text{net} = C\text{inv} \times (1 - \gamma) \tag{2.1}$$

де Cinv – повна початкова вартість інвестиції, грн; γ – частка грантового покриття ($0 \le \gamma \le \gamma\text{max}$), де γmax визначається умовами конкретної програми: 0,70 для «Енергодім» Пакет Б та «ГрінДІМ» (СЕС/теплонасоси), 0,50 для «ГрінДІМ» (генератори).

Для програми «5-7-9%» замість прямого гранту застосовується кредитна субсидія. Річна субсидована різниця між ринковою ставкою rm та пільговою

ставкою *r*c включається до складу грошових потоків кожного року до погашення кредиту.

2.1.2 Простий термін окупності (PP)

Простий термін окупності (Payback Period, PP) визначає мінімальний часовий горизонт, за який накопичені чисті грошові потоки від проєкту компенсують чисті інвестиційні витрати [12]. При рівномірних грошових потоках:

$$PP = C\text{net} / CF\text{avg} \tag{2.2}$$

де CFavg – середньорічний чистий грошовий потік від заходу з енергоефективності, грн/рік.

У загальному випадку нерівномірних грошових потоків:

$$PP = k + (C\text{net} - \sum_{i=1}^{k} CF_i) / CF\text{k+1} \tag{2.3}$$

де k – найбільше ціле число повних років, за які кумулятивні потоки не досягають Cnet; CFt – чистий грошовий потік у рік t.

2.1.3 Чиста приведена вартість (NPV)

Чиста приведена вартість (Net Present Value, NPV) являє собою суму дисконтованих грошових потоків за весь розрахунковий горизонт за вирахуванням чистих інвестиційних витрат [20]:

$$NPV = \sum_{t=1}^{n} [CF\text{t} / (1 + r)^t] - C\text{net} \tag{2.4}$$

де n – горизонт розрахунку (роки), відповідає нормативному терміну служби обладнання; r – ставка дисконтування (для проєктів в умовах ризику в Україні

рекомендується 15–25% [14]); CF_t – чистий грошовий потік у рік t, грн. Проєкт вважається ефективним при NPV > 0.

2.1.4 Дисконтований термін окупності (DPP)

Дисконтований термін окупності (Discounted Payback Period, DPP) враховує часову вартість грошей і визначається як мінімальне число років, за яке сума дисконтованих грошових потоків досягає або перевищує чисті інвестиційні витрати:

$$DPP = \min\{T : \sum_{t=1}^{T} [CF_t / (1 + r)^t] \geq C_{net}\} \quad (2.5)$$

Оскільки DPP > PP при $r > 0$, цей показник дає консервативнішу оцінку ефективності проєкту та краще відображає реальну вартість капіталу.

2.1.5 Внутрішня норма прибутковості (IRR)

Внутрішня норма прибутковості (Internal Rate of Return, IRR) – це ставка дисконтування, за якої NPV дорівнює нулю [20]. Математично це еквівалентно рівнянню:

$$\sum_{t=1}^{n} [CF_t / (1 + IRR)^t] - C_{net} = 0 \quad (2.6)$$

Оскільки рівняння (2.6) не має аналітичного розв'язання, IRR обчислюється числовим методом: у реалізованому сервісі застосовано метод дихотомії (bisection) із точністю 10^{-6} за 100 ітерацій, що відповідає класичній практиці фінансового аналізу. Критерій ефективності: $IRR > r$.

2.1.6 Індекс прибутковості (PI)

Індекс прибутковості (Profitability Index, PI) – відношення суми дисконтованих грошових потоків до чистих інвестиційних витрат:

$$PI = (1 / C\text{net}) \times \sum_{t=1}^{n} [CF\text{t} / (1 + r)^{t}] \quad (2.7)$$

При PI > 1 проєкт ефективний; PI < 1 свідчить про збитковість. PI зручний для ранжування альтернативних проєктів із різними обсягами власних інвестицій.

2.1.7 Модель річного грошового потоку

Річний чистий грошовий потік від заходу з підвищення енергоефективності формується з економії на енергоносіях та витрат на обслуговування:

$$CF\text{t} = E\text{save} \times p\text{t} \times (1 + \alpha)^{t-1} - OM\text{t} \quad (2.8)$$

де Esave – щорічна економія енергоресурсу, кВт·год або Гкал; pt – тариф на енергоносій у базовому (початковому) році; α – середньорічний темп зростання тарифу (рекомендується α = 0,05–0,15 відповідно до прогнозів НКРЕКП); OMt – витрати на обслуговування та технічний огляд обладнання у рік t, грн.

2.1.8 Приклади розрахунку показників ефективності інвестицій

Для верифікації математичної моделі наведено два розрахункові приклади, що відповідають реальним профілям користувачів системи: приватний домовласник та ОСББ.

Приклад 1. Приватний домовласник – СЕС 10 кВт з акумулятором, програма «5-7-9%».

Вихідні дані: повна вартість системи Cinv = 350 000 грн; кредит за програмою «5-7-9%» – 150 000 грн за ставкою 7% річних на 5 років; власні кошти – 200 000 грн (чисті інвестиції Cnet = 200 000 грн, оскільки кредит повертається з енергетичної економії); середньорічний грошовий потік CFavg = 48 000 грн/рік (від власного споживання та продажу надлишку електроенергії); горизонт розрахунку n = 15 років; ставка дисконтування r = 20%.

Розрахунок PP за формулою (2.2):

PP = 200 000 / 48 000 ≈ 4,2 роки

Розрахунок NPV за формулою (2.4). Множник ануїтету при r = 20%, n = 15 років:

Aq = (1 − (1,20)$^{-15}$) / 0,20 = (1 − 0,0649) / 0,20 = 4,675

NPV = 48 000 × 4,675 − 200 000 = 224 400 − 200 000 = 24 400 грн > 0

Розрахунок IRR (метод дихотомії, формула 2.6): при IRR ≈ 23,5% отримаємо NPV ≈ 0. Оскільки 23,5% > 20%, проєкт ефективний.

Розрахунок PI за формулою (2.7):

PI = 224 400 / 200 000 = 1,12 > 1

Таблиця 2.1

Зведені показники ефективності (Приклад 1 – СЕС 10 кВт, «5-7-9%»)

Показник	Значення	Критерій	Висновок
PP	4,2 роки	< 7 років	Прийнятний
NPV	24 400 грн	> 0	Ефективний
DPP	5,3 роки	< n = 15 pp.	Прийнятний
IRR	23,50%	> r = 20%	Ефективний
PI	1,12	> 1	Ефективний

Приклад 2. ОСББ – комплексна термомодернізація, програма «Енергодім» Пакет Б

Вихідні дані: повна вартість комплексної термомодернізації Cinv = 3 000 000 грн; грант «Енергодім» Пакет Б (γ = 70%): сума гранту = 2 100 000 грн; власні кошти ОСББ (чисті інвестиції):

Cnet = 3 000 000 × (1 − 0,70) = 900 000 грн

Середньорічна економія на теплопостачанні (скорочення споживання теплоенергії на 40% при тарифі 1800 грн/Гкал та обсязі 500 Гкал/рік): *CF*avg = 360 000 грн/рік; горизонт розрахунку *n* = 20 років; ставка дисконтування *r* = 20%.

Розрахунок PP за формулою (2.2):

$PP = 900\,000 / 360\,000 = 2{,}5$ роки

Розрахунок NPV за формулою (2.4). Множник ануїтету при $r = 20\%$, $n = 20$ років:

$Aq = (1 - (1{,}20)^{-20}) / 0{,}20 = (1 - 0{,}0261) / 0{,}20 = 4{,}870$

$NPV = 360\,000 \times 4{,}870 - 900\,000 = 1\,753\,200 - 900\,000 = 853\,200$ грн $>> 0$

Розрахунок IRR (метод дихотомії): $IRR \approx 40{,}0\% >> r = 20\%$ – проєкт надзвичайно ефективний.

Розрахунок PI за формулою (2.7):

$PI = 1\,753\,200 / 900\,000 = 1{,}95 >> 1$

Таблиця 2.2.

Зведені показники ефективності

Показник	Значення	Критерій	Висновок
PP	2,5 роки	< 7 років	Відмінний
NPV	853 200 грн	> 0	Ефективний
DPP	2,7 роки	< n = 20 рр.	Відмінний
IRR	40,00%	> r = 20%	Ефективний
PI	1,95	> 1	Ефективний

Порівняльний аналіз двох прикладів демонструє ключову перевагу розробленого сервісу: завдяки грантовому покриттю 70% у програмі «Енергодім» значення PP для ОСББ (2,5 роки) є майже вдвічі меншим, ніж для домовласника без прямого гранту (4,2 роки), при масштабі проєкту, що перевищує приватний у 8,6

разу. Автоматизований розрахунок п'яти показників одночасно для різних програм підтримки становить наукову новизну роботи (розд. 1.3).

2.2 Функціональні вимоги та Use Case діаграма

2.2.1 Визначення та структурування функціональних вимог

Функціональні вимоги до сервісу сформовано на підставі аналізу трьох рольових профілів (підрозд. 1.2) та виявлених недоліків аналогів (підрозд. 1.3). Вимоги структуровано за методологією MoSCoW (Must Have / Should Have / Could Have / Won't Have) та згруповано за ролями: Гість, Зареєстрований користувач та Адміністратор. Нумерація вимог у форматі FR-NN забезпечує трасованість між вимогами, прецедентами використання та тестовими випадками (підрозд. 3.4).

Перша група вимог стосується актора «Гість» – неавторизованого відвідувача сервісу, який взаємодіє із системою виключно в рамках поточного сеансу без збереження даних. Центральною функцією для цієї ролі є калькулятор: користувач вводить параметри об'єкта та обирає програму підтримки, після чого система обчислює всі п'ять показників ефективності відповідно до формул (2.1)–(2.8). Окремою вимогою виділено відображення результатів у вигляді інтерактивних діаграм на базі бібліотеки Recharts, оскільки візуалізація кумулятивного грошового потоку суттєво спрощує інтерпретацію показника DPP для непрофесійної аудиторії [26]. Вимога FR-04 має пріоритет Should Have, бо довідник тарифів і технологій підвищує зручність роботи, проте не є критичним для базового розрахунку.

Друга група вимог описує можливості актора «Зареєстрований користувач», який успадковує всі функції Гостя та отримує доступ до персоналізованих інструментів збереження й порівняння проєктів. Ключовою відмінністю від аналогів (підрозд. 1.3) є саме функція одночасного порівняння до п'яти проєктів на спільному графіку: система автоматично виділяє варіант із найменшим PP та

найбільшим PI, що відповідає задачі обґрунтованого вибору технології та програми фінансування. Вимога FR-08 (PDF-звіт) отримала пріоритет Should Have з огляду на те, що генерація звіту є похідною від збереженого проєкту і може бути реалізована в наступній ітерації розробки без шкоди для базового функціоналу. Вимога FR-09 (управління архівом) забезпечує довгострокову цінність сервісу для енергоаудиторів, які ведуть кілька проєктів одночасно.

Таблиця 2.3

Функціональні вимоги для ролі «Гість»

ID	Вимога	Пріоритет
FR-01	Введення параметрів проєкту: тип об'єкта, технологія, повна вартість, програма підтримки, горизонт і ставка дисконтування	Must Have
FR-02	Розрахунок п'яти показників (PP, NPV, DPP, IRR, PI) за введеними параметрами із урахуванням формул (2.1)–(2.8)	Must Have
FR-03	Відображення результатів у вигляді інтерактивних діаграм (Recharts): кумулятивний грошовий потік, порівняльний бар-чарт показників	Must Have
FR-04	Перегляд довідника актуальних тарифів, технологій та програм підтримки	Should Have
FR-05	Реєстрація нового облікового запису та вхід (JWT-автентифікація)	Must Have

Таблиця 2.4

Функціональні вимоги для ролі «Зареєстрований користувач»

ID	Вимога	Пріоритет
FR-06	Збереження розрахункового проєкту до особистого кабінету з назвою та міткою дати	Must Have
FR-07	Одночасне порівняння до 5 збережених проєктів на спільному графіку з виділенням оптимального варіанта	Must Have
FR-08	Завантаження звіту у форматі PDF із усіма показниками та графіками	Should Have
FR-09	Перегляд та редагування переліку збережених проєктів (архів, видалення)	Should Have

Третя група вимог визначає адміністративні функції, що забезпечують актуальність довідникових даних, від яких безпосередньо залежить точність розрахунків для всіх користувачів. Вимоги FR-10–FR-12 охоплюють CRUD-операції над трьома ключовими сутностями: тарифами, технологіями та грантовими програмами. Виокремлення цих сутностей у керовані довідники (а не жорстке кодування значень) є архітектурним рішенням, що забезпечує оперативне оновлення умов програм «Енергодім», «ГрінДІМ» та «5-7-9%» без змін у програмному коді. Вимога FR-13 (управління користувачами) надає адміністратору інструмент деактивації облікових записів у разі порушення умов використання сервісу. Усі адміністративні дії фіксуються у журналі аудиту (audit log) – відповідний прецедент UC-14 описано у підрозділі 2.2.2.

Нефункціональні вимоги сервісу охоплюють такі аспекти: продуктивність – відповідь REST API не більше 500 мс при навантаженні до 100 запитів/с (RPS);

безпека – JWT-автентифікація, bcrypt-хешування паролів (salt rounds = 12), захист від CSRF та XSS; доступність – адаптивна верстка (Tailwind CSS) та PWA-сумісність для мобільних пристроїв; розширюваність – модульна архітектура NestJS дає змогу додавати нові типи програм підтримки без змін у ядрі системи [22].

Таблиця 2.5

Функціональні вимоги для ролі «Адміністратор»

ID	Вимога	Пріоритет
FR-10	CRUD-управління тарифами на електроенергію та теплопостачання (назва, значення, дата введення, регіон)	Must Have
FR-11	CRUD-управління технологіями (тип, типові діапазони вартості, нормативний термін служби, коефіцієнт економії)	Must Have
FR-12	CRUD-управління грантовими програмами (назва, γmax, максимальна сума, строк дії, тип бенефіціара)	Must Have
FR-13	Перегляд списку користувачів, деактивація та відновлення облікових записів	Should Have

2.2.2 Use Case діаграма системи

Use Case діаграму системи (Рисунок 2.1) побудовано за стандартом UML 2.5.1 [22] із використанням інструменту Mermaid.js. Діаграма охоплює 15 прецедентів використання для трьох рольових акторів та одного системного актора.

Актор «Гість» взаємодіє із системою через п'ять прецедентів: UC-01 «Розрахувати показники ефективності» (центральний), UC-02 «Відобразити результати на графіку», UC-03 «Обрати програму підтримки», UC-04 «Переглянути довідник технологій», UC-05 «Зареєструватись». Прецедент UC-02 знаходиться у відношенні <<include>> до UC-01, оскільки візуалізація

кумулятивного грошового потоку та індикація точок PP і DPP є невід'ємною складовою кожного розрахунку. Прецедент UC-03 знаходиться у відношенні <<extend>> до UC-01 – грантова програма вказується як необов'язковий параметр при розрахунку, а у разі її вибору система автоматично підставляє максимальний відсоток покриття γ та граничну суму гранту з довідника.

Актор «Зареєстрований користувач» успадковує всі прецеденти Гостя (відношення узагальнення) та додатково має доступ до: UC-06 «Зберегти проєкт», UC-07 «Порівняти проєкти», UC-08 «Завантажити PDF-звіт», UC-09 «Управляти архівом проєктів». Прецедент UC-08 знаходиться у відношенні <<extend>> до UC-06, оскільки PDF-звіт може бути згенерований одразу після збереження проєкту. UC-07 включає (<<include>>) UC-06, оскільки для порівняння необхідна наявність щонайменше двох збережених проєктів.

Актор «Адміністратор» успадковує права Зареєстрованого користувача та керує чотирма адміністративними прецедентами: UC-10 «Управляти тарифами», UC-11 «Управляти технологіями», UC-12 «Управляти грантовими програмами», UC-13 «Управляти користувачами». Усі чотири прецеденти включають (<<include>>) спільний підпрецедент UC-14 «Зафіксувати адміністративну дію» (audit log), що забезпечує відстежуваність змін у довідниках [21].

Системний актор «Зовнішній сервіс (API НБУ / НКРЕКП)» взаємодіє з прецедентом UC-15 «Оновити поточні тарифи» за розкладом (cron-job щоденно о 06:00). Цей механізм гарантує актуальність тарифних даних без ручного втручання адміністратора.

Повна структура зв'язків між акторами та прецедентами наочно відображена на Рисунок 2.1. Діаграму побудовано за стандартом UML 2.5.1 [22] у нотації PlantUML; межі системи позначено прямокутником, що охоплює всі п'ятнадцять прецедентів. Відношення узагальнення між акторами (Гість → Зареєстрований користувач → Адміністратор) зображено стрілками успадкування, що унеможливлює дублювання прецедентів на діаграмі та відповідає принципу єдиної відповідальності при проєктуванні ролей.

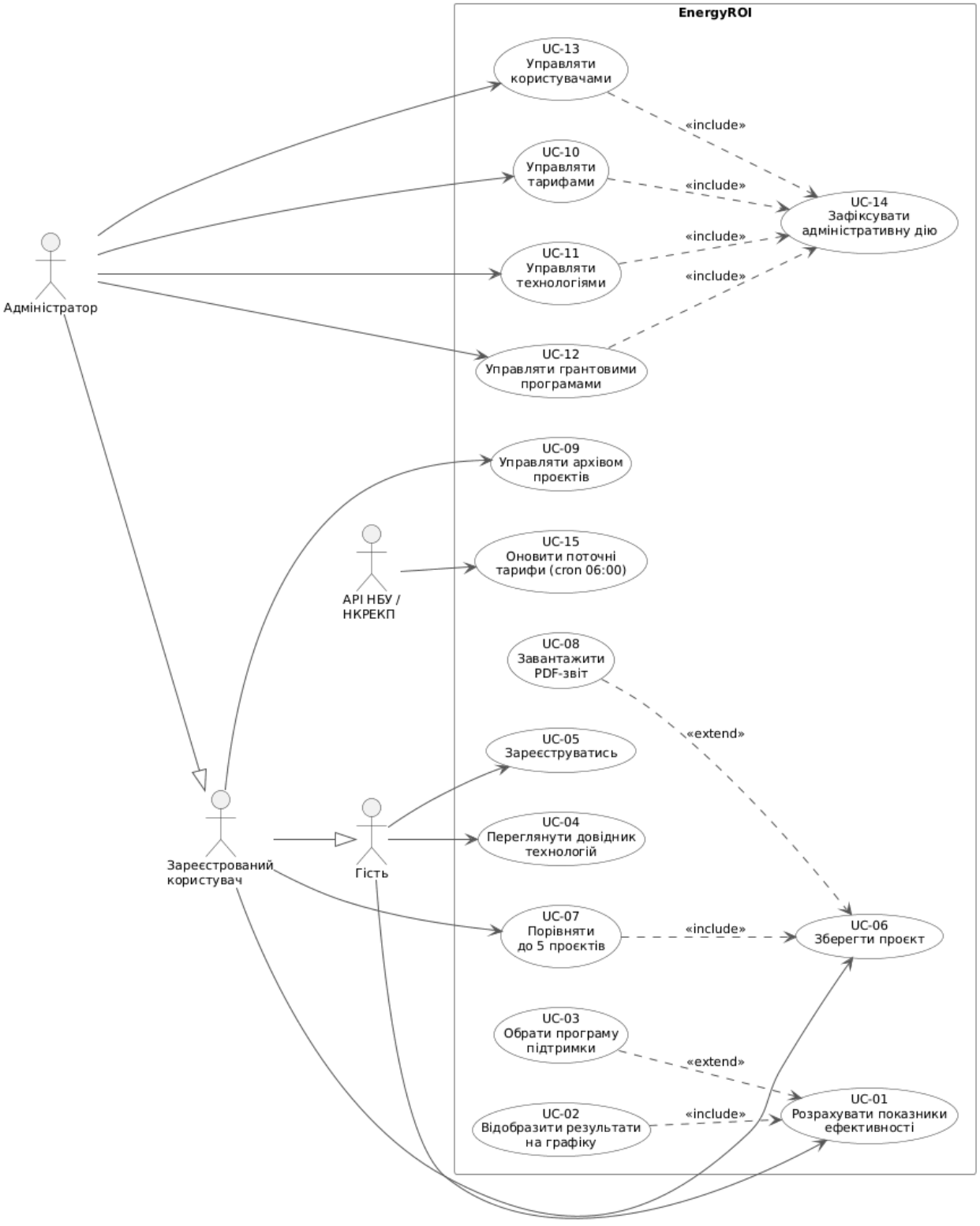


Рис. 2.1 Use Case діаграма інтерактивного сервісу порівняння термінів окупності

Зв'язки між прецедентами відображають архітектурне рішення щодо розмежування доступу: Гість може виконувати лише анонімні розрахунки; Зареєстрований користувач отримує повний функціонал збереження та порівняння; Адміністратор керує довідниковими даними, що впливають на розрахунки всіх користувачів. Такий підхід відповідає принципу мінімальних привілеїв (Principle of Least Privilege) та безпосередньо відображається у ролях таблицях підрозділу 2.2.1.

2.3 C4-модель архітектури веб-сервісу

C4-модель, запропонована Саймоном Брауном у 2018 році, є ієрархічною системою архітектурного документування, що складається з чотирьох рівнів деталізації: системний контекст (System Context), контейнери (Containers), компоненти (Components) та код (Code). Кожен рівень призначено для конкретної аудиторії: від стейкхолдерів та замовника до розробників, що безпосередньо реалізують систему [18]. Перевага C4-моделі над традиційними UML-діаграмами полягає у строгій ієрархічній структурі та мінімалізмі нотації, що значно знижує поріг розуміння архітектурної документації [8].

У контексті розробки веб-сервісу EnergyROI застосування C4-моделі обумовлено кількома чинниками. По-перше, система має чітко виражену багаторівневу структуру: фронтенд (Next.js), бекенд (NestJS), база даних (PostgreSQL) та зовнішні інтеграції (API НБУ, НКРЕКП, SMTP) є логічно відокремленими контейнерами з різними технологічними стеками, що природно відповідає рівню 2 моделі. По-друге, модульна архітектура NestJS – з виокремленням модулів AuthModule, ProjectModule, CalculatorModule та AdminModule – безпосередньо відображається на рівень 3 (Components), де кожен модуль описується як самостійний компонент із визначеними інтерфейсами взаємодії [6]. По-третє, на відміну від нотації UML, C4-модель не потребує

спеціалізованих інструментів для читання: діаграми рівнів 1 і 2 є зрозумілими для нетехнічної аудиторії (замовників, потенційних інвесторів), що відповідає практичному призначенню сервісу.

2.3.1 Рівень 1 – Системний контекст (System Context Diagram)

Діаграма системного контексту (Рисунок 2.2) є найвищим рівнем абстракції та показує веб-сервіс «EnergyROI» як єдину систему у взаємодії з зовнішніми акторами та суміжними системами. На цьому рівні визначено три категорії користувачів системи та два класи зовнішніх залежностей.

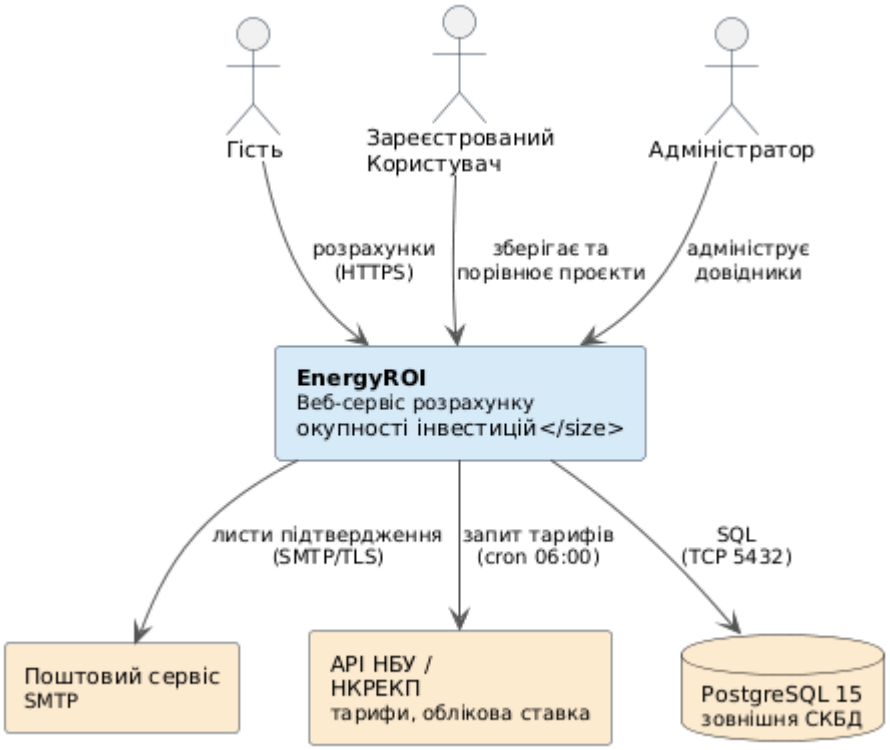


Рис. 2.2 C4-модель рівень 1 – системний контекст веб-сервісу EnergyROI

Рисунок 2.2 демонструє, що веб-сервіс EnergyROI взаємодіє з трьома категоріями людських акторів та трьома зовнішніми системами. Людські актори розрізняються за рівнем доступу та характером взаємодії із сервісом: від анонімного розрахунку без збереження даних (Гість) до повного адміністративного контролю над довідниковими даними (Адміністратор). Зовнішні системи є

джерелами актуальних вхідних даних для розрахункового ядра – без інтеграції з API НБУ та НКРЕКП адміністратору довелося б оновлювати тарифи вручну, що створює ризик використання застарілих значень у розрахунках. Детальний опис усіх акторів та зовнішніх систем із зазначенням протоколів взаємодії наведено у таблиці 2.6.

Таблиця 2.6

Актори системного контексту веб-сервісу EnergyROI

Актор	Тип	Взаємодія з системою
Гість (Guest)	Людина / неавторизований	Виконує розрахунки PP, NPV, DPP, IRR, PI без автентифікації; результати не зберігаються
Зареєстрований користувач	Людина / авторизований	Зберігає та порівнює проєкти; переглядає архів розрахунків; отримує рекомендації щодо грантів
Адміністратор	Людина / привілейований	Виконує CRUD-операції над тарифами, технологіями та грантовими програмами через адмінпанель
Поштовий сервіс (SMTP)	Зовнішня система	Отримує запити від EnergyROI для відправки листів підтвердження реєстрації та відновлення паролю
PostgreSQL 15	Зовнішня СКБД	Зберігає профілі користувачів, проєкти, результати розрахунків, довідникові дані тарифів та грантів
API НБУ / НКРЕКП	Зовнішня система	Постачає актуальні тарифи на енергоносії та облікову ставку НБУ; оновлення за розкладом cron щоденно о 06:00

2.3.2 Рівень 2 – Діаграма контейнерів (Container Diagram)

Другий рівень C4-моделі деталізує внутрішню структуру системи у вигляді технологічно розмежованих контейнерів (Рисунок 2.3). Контейнер – це окремо розгортуваний елемент або процес [8]. Для веб-сервісу «EnergyROI» визначено чотири основні контейнери, що взаємодіють через чітко визначені інтерфейси.

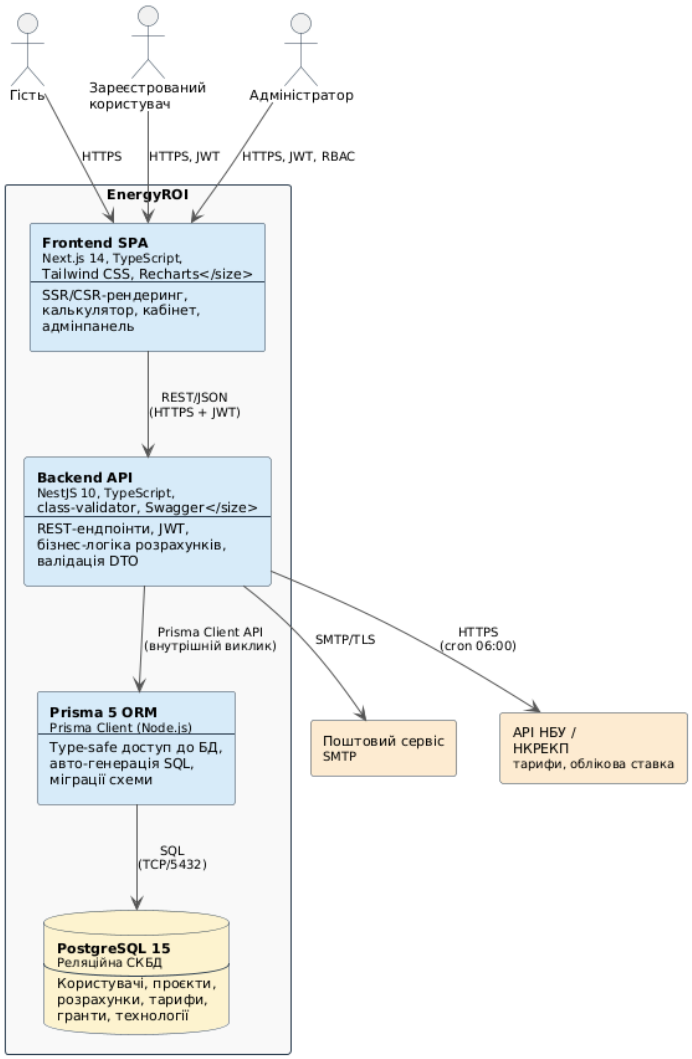


Рис. 2.3 C4-модель рівень 2 – діаграма контейнерів веб-сервісу EnergyROI

Діаграма контейнерів (Рисунок 2.3) розкриває внутрішню структуру сервісу EnergyROI та демонструє чіткий розподіл відповідальності між чотирма контейнерами. Frontend SPA є єдиною точкою взаємодії з усіма трьома типами користувачів і делегує обчислювальну логіку Backend API через REST-інтерфейс, не виконуючи жодних розрахунків самостійно. Такий підхід відповідає принципу

розділення відповідальності (Separation of Concerns) та спрощує незалежне масштабування рівнів системи. ORM-шар на базі Prisma 5 виступає посередником між Backend API та базою даних PostgreSQL 15, забезпечуючи типобезпечний доступ до даних та автоматичне керування міграціями схеми без написання SQL вручну. Зовнішніми залежностями системи на рівні контейнерів є поштовий сервіс (SMTP) та API НБУ/НКРЕКП: перший використовується Backend API для надсилання транзакційних листів підтвердження реєстрації та відновлення паролю, другий – для автоматичного оновлення тарифів за розкладом cron щоденно о 06:00. Усі інші операції виконуються всередині межі EnergyROI.

Таблиця 2.7

Контейнери веб-сервісу EnergyROI (C4 Level 2)

Контейнер	Технологія	Протокол	Відповідальність
Frontend SPA	Next.js 14, TypeScript, Tailwind CSS	HTTPS / REST JSON	SSR/CSR рендеринг сторінок; калькулятор; порівняння проєктів; авторизація через JWT-cookie
Backend API	NestJS 10, TypeScript, class-validator	HTTP/1.1, REST + Swagger	Бізнес-логіка розрахунків; автентифікація (JWT); CRUD ресурсів; валідація вхідних даних
База даних	PostgreSQL 15	TCP (Prisma Client)	Персистентне зберігання користувачів, проєктів, результатів, довідників тарифів та грантів
ORM-шар	Prisma 5	Внутрішній (Node.js)	Type-safe доступ до БД; авто-генерація SQL; міграції схеми; розширений QueryBuilder

Взаємодія між контейнерами реалізована за принципом «запит–відповідь»: Frontend здійснює HTTP-запити до Backend API через REST-інтерфейс; Backend API звертається до PostgreSQL виключно через Prisma Client, що унеможливлює прямий доступ фронтенду до СКБД. Інтеграція з зовнішніми системами (SMTP та API НБУ/НКРЕКП) ініціюється виключно з Backend API, що централізує контроль над зовнішнім трафіком та спрощує застосування політик безпеки**.** Такий підхід забезпечує ізоляцію шарів та спрощує горизонтальне масштабування [6].

2.3.3 Рівень 3 – Діаграма компонентів бекенду (Component Diagram)

Рівень 3 C4-моделі деталізує внутрішню структуру контейнера Backend API (Рисунок 2.4), розкриваючи взаємодію між сімома функціональними модулями NestJS та допоміжним шаром захисту маршрутів. На відміну від рівня 2, де відображалися лише зовнішні межі контейнерів, рівень 3 демонструє напрямки залежностей між модулями та їх зв'язок зі спільними ресурсами – базою даних і шаром автентифікації. NestJS реалізує архітектурний патерн модульного впровадження залежностей (Dependency Injection), у якому кожен модуль є автономною одиницею з власними контролерами, сервісами та репозиторіями [19]. Для веб-сервісу «EnergyROI» визначено сім функціональних модулів, опис яких згруповано за призначенням у таблицях нижче.

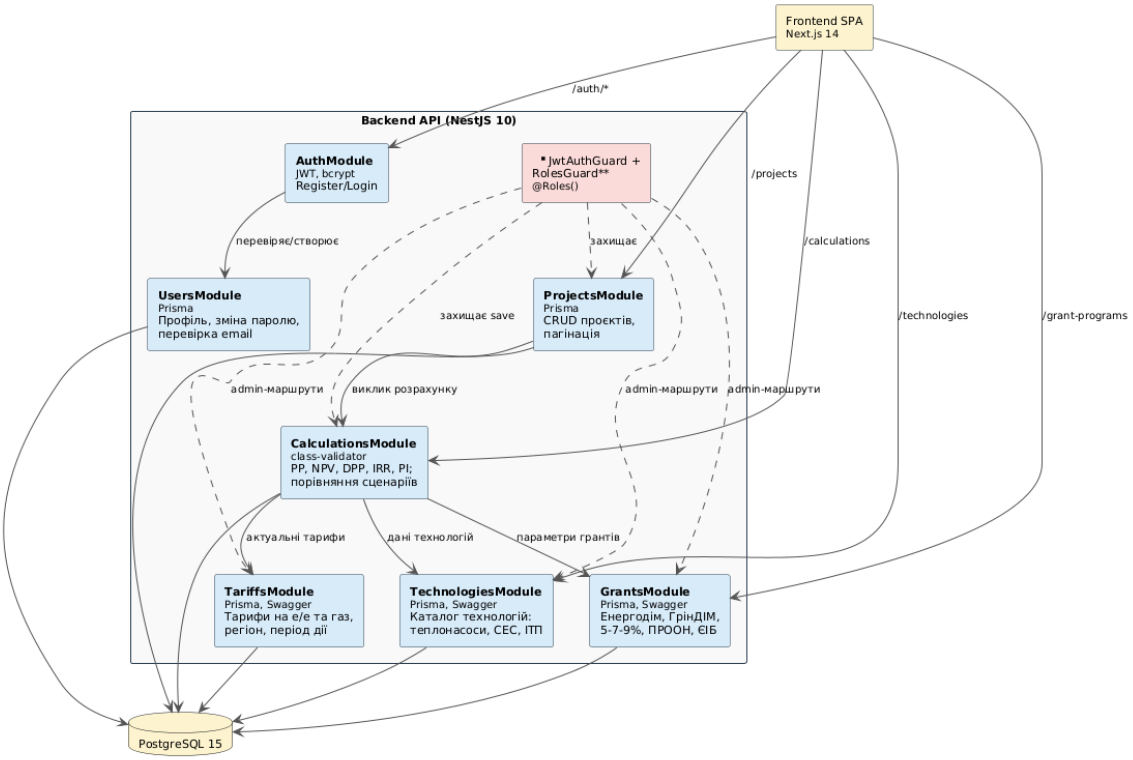


Рис. 2.4 C4-модель рівень 3 – компоненти NestJS Backend API

Рисунок 2.4 унаочнює однонаправлені залежності між модулями. CalculationsModule виступає центральним споживачем довідникових даних, імпортуючи TechnologiesModule, TariffsModule та GrantsModule для отримання актуальних параметрів технологій, тарифів та грантових програм на момент розрахунку. ProjectsModule делегує обчислення CalculationsModule під час збереження результатів проєкту. AuthModule забезпечує видачу та валідацію JWT-токенів і взаємодіє з UsersModule для перевірки облікових записів. Захист маршрутів реалізовано глобальним шаром JwtAuthGuard у поєднанні з RolesGuard, який зчитує метадані декоратора @Roles() та обмежує доступ до адміністративних ендпоінтів TechnologiesModule, TariffsModule і GrantsModule, а також до персоналізованих ендпоінтів ProjectsModule і CalculationsModule. Детальний опис модулів згруповано за функціональним призначенням у таблицях 2.8–2.10. Першу групу складають модулі автентифікації та управління користувачами.

Опис компонентів бекенду наведено у таблицях нижче.

Таблиця 2.8

Модулі автентифікації та користувачів

Модуль / Компонент	Технологія	Відповідальність
AuthModule	JWT, bcrypt	Реєстрація, вхід, вихід; генерація/валідація Access Token (15 хв) та Refresh Token (7 діб); ролі GUEST | USER | ADMIN через декоратори @Roles()
UsersModule	Prisma Client	CRUD профілю користувача; зміна паролю; перевірка унікальності email; прив'язка ролі

Другу групу складають модулі, що реалізують основну бізнес-логіку сервісу. ProjectsModule забезпечує персистентне зберігання інвестиційних проєктів користувача, тоді як CalculationsModule є обчислювальним ядром системи – саме тут реалізовано формули (2.1)–(2.8) та логіку порівняння кількох сценаріїв одночасно.

Таблиця 2.9

Модулі бізнес-логіки

Модуль / Компонент	Технологія	Відповідальність
ProjectsModule	Prisma Client	Створення, оновлення, видалення та перегляд інвестиційних проєктів; прив'язка до User; пагінація
CalculationsModule	class-validator, Prisma	Обчислення PP, NPV, DPP, IRR, PI за математичними моделями підрозділу 2.1; збереження результатів; порівняння кількох сценаріїв

Третю групу складають модулі довідникових даних, що перебувають під управлінням адміністратора. Усі три модулі мають ідентичну структуру CRUD-ендпоінтів із Swagger-документацією та слугують джерелом вхідних параметрів для CalculationsModule – зміна тарифу чи умов грантової програми в адмінпанелі миттєво впливає на результати нових розрахунків усіх користувачів.

Таблиця 2.10

Модулі довідникових даних

Модуль / Компонент	Технологія	Відповідальність
TechnologiesModule	Prisma, Swagger	Адмін-CRUD довідника технологій (теплонасоси, СЕС, котли, ІТП, когенерація); вартість min/max в UAH/$
TariffsModule	Prisma, Swagger	Управління тарифами на електроенергію та газ з прив'язкою до регіону та діапазону дат; актуальні тарифи для розрахунків
GrantsModule	Prisma, Swagger	CRUD бази грантових програм («Енергодім», «ГрінДІМ», «5-7-9%», ПРООН, ЄІБ); фільтрація за цільовою аудиторією; автоматичне підбирання програм до розрахунку

Між модулями діє однонаправлена залежність: *CalculationsModule* імпортує *TechnologiesModule*, *TariffsModule* та *GrantsModule* для отримання актуальних даних. *AuthModule* використовує глобальний guard *JwtAuthGuard*, що захищає всі маршрути, які потребують автентифікації. Swagger/OpenAPI-документація

генерується автоматично засобами декораторів NestJS та доступна за адресою /api/docs [19].

2.4 ER-діаграма та схема бази даних Prisma

Проєктування реляційної бази даних є основою для коректної персистентності всіх розрахункових і довідникових даних веб-сервісу. У цьому підрозділі визначено сутності та їх атрибути, побудовано ER-діаграму, а також розроблено декларативну Prisma-схему для автоматичного генерування та виконання SQL-міграцій.

2.4.1 Концептуальна модель даних

ролей користувачів (підрозділ 2.4) виділено сім основних сутностей бази даних: User (користувач), Project (інвестиційний проєкт), Calculation (розрахунок), Technology (технологія), EnergyTariff (тариф), GrantProgram (грантова програма) та CalculationHistoryНа основі аналізу функціональних вимог системи та. Сутності Technology, EnergyTariff та GrantProgram є довідниковими (reference data) і управляються виключно адміністратором.

Опис усіх сутностей, їх атрибутів та типів даних наведено у таблиці Сутностей та атрибутів бази даних веб-сервісу EnergyROI в додатку Б.

2.4.2 ER-діаграма бази даних

ER-діаграму (*Entity-Relationship Diagram*) розроблено згідно з нотацією Crow's Foot та наведено на Рисунок 2.5. Діаграма відображає сім сутностей предметної області та їх зв'язки з урахуванням кардинальностей і обмежень цілісності.

Основні зв'язки між сутностями: один *User* може мати довільну кількість *Project* (1:N) із каскадним видаленням; один Project містить довільну кількість *Calculation* (1:N), що дозволяє зберігати історію перерахунків при зміні параметрів;

кожен Project обов'язково посилається на одну *Technology* (N:1) та необов'язково – на одну *GrantProgram* (N:0..1). *EnergyTariff* є незалежним довідником тарифів на електроенергію, теплопостачання та водопостачання у розрізі року й місяця, який використовується сервісним шаром *CalculationsModule* для визначення актуальної вартості енергоносіїв на момент розрахунку. *CalculationHistory* – допоміжна сутність аудиту, що фіксує усі розрахунки (включно з анонімними) у вигляді JSON-документів вхідних параметрів та результатів і не пов'язана прямими зовнішніми ключами з основними сутностями. Усі ідентифікатори реалізовано як рядки формату cuid, що забезпечує глобальну унікальність без розкриття порядку створення записів – на відміну від послідовних SERIAL-ідентифікаторів.

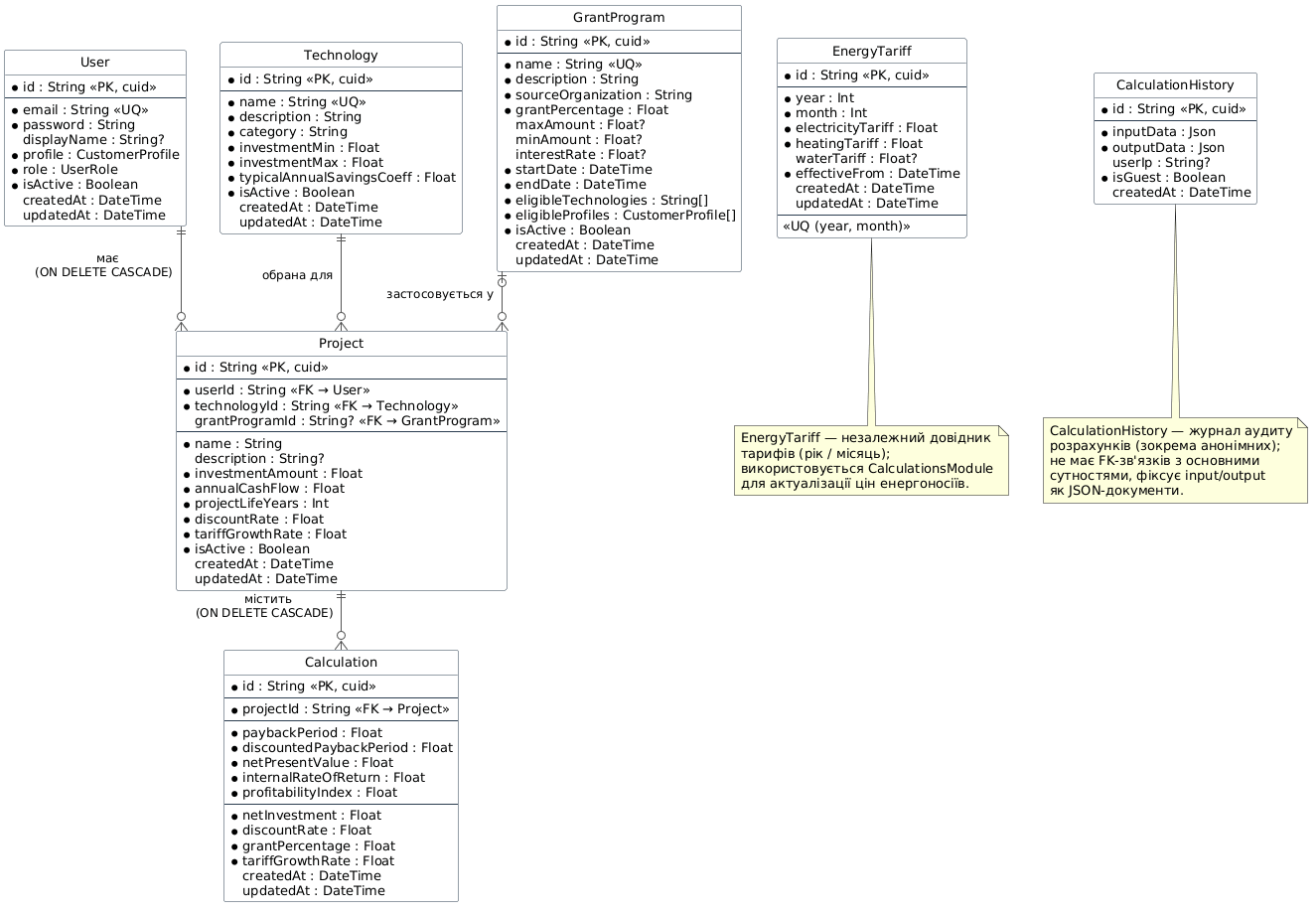


Рис. 2.5 ER-діаграма бази даних веб-сервісу EnergyROI

2.4.3 Декларативна Prisma-схема

Prisma 5 використовує власний DSL (*Domain Specific Language*) для декларативного опису схеми бази даних у файлі schema.prisma. На основі цього файлу Prisma CLI автоматично генерує та застосовує SQL-міграції за командою prisma migrate dev, а також типізований *Prisma Client* для використання у NestJS-сервісах [7]. Повну Prisma-схему веб-сервісу EnergyROI наведено у лістингу в додатку В.

Схема передбачає каскадне видалення (*onDelete: Cascade*) для залежних сутностей: видалення *User* автоматично видаляє всі пов'язані Project, а видалення Project – усі пов'язані *Calculation*. Результати розрахунків (paybackPeriod, discountedPaybackPeriod, netPresentValue, internalRateOfReturn, profitabilityIndex) зберігаються як обов'язкові поля типу *Float*, що відповідає стандарту *IEEE 754* подвійної точності та забезпечує точність до 15 значущих десяткових цифр, що достатньо для інженерних фінансових розрахунків.

Висновки до розділу 2

У другому розділі кваліфікаційної роботи виконано проєктування математичних моделей та архітектури веб-сервісу «EnergyROI» для порівняльного розрахунку термінів окупності інвестицій у енергоефективність.

У підрозділі 2.1 розроблено математичні моделі п'яти ключових інвестиційних показників – PP, DPP, NPV, IRR та PI – адаптованих до специфіки українського ринку: моделі параметризовано відсотком грантового співфінансування (до 70% для «Енергодім» та «ГрінДІМ»), процентною ставкою кредиту (1–9% для «5-7-9%»), а також тарифами на електроенергію та газ. Математичне обґрунтування дозволяє коректно порівнювати сценарії з різними джерелами фінансування.

У підрозділі 2.2 засобами UML Use Case змодельовано функціональні вимоги системи для трьох ролей: гостя, зареєстрованого користувача та адміністратора. Визначено 15 прецедентів, що охоплюють весь цикл роботи з інвестиційними

розрахунками – від анонімного обчислення до адміністрування довідникових даних.

У підрозділі 2.3 побудовано трирівневу C4-модель архітектури. На рівні системного контексту (Level 1) визначено трьох акторів та дві зовнішні системи (SMTP, PostgreSQL). На рівні контейнерів (Level 2) виокремлено чотири розгортувані одиниці: Next.js 14 SPA, NestJS 10 Backend API, PostgreSQL 15 та Prisma 5 ORM. На рівні компонентів (Level 3) NestJS-бекенд декомпозовано на сім функціональних модулів із чітко визначеними відповідальностями та однонаправленими залежностями.

У підрозділі 2.4 розроблено реляційну модель даних, що охоплює сім сутностей із визначеними зв'язками та обмеженнями цілісності та каскадним видаленням залежних записів. Декларативна Prisma-схема забезпечує type-safe доступ до бази даних, автоматичну генерацію SQL-міграцій та відтворюваність структури БД у всіх середовищах розгортання. Декларативна Prisma-схема забезпечує типо-безпечний доступ до бази даних, автоматичну генерацію SQL-міграцій та відтворюваність структури БД у всіх середовищах розгортання, а використання глобально унікальних ідентифікаторів формату cuid гарантує безпеку посилань між сутностями без розкриття порядку створення записів.

Результати другого розділу – математичні моделі, C4-архітектура, ER-діаграма та Prisma-схема – є вичерпною проєктною документацією, що безпосередньо використовується у третьому розділі для реалізації серверної та клієнтської частин веб-сервісу.

3. РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Реалізація серверної частини (NestJS + Prisma)

3.1.1 Структура серверного проєкту

Серверна частина реалізована з використанням фреймворку NestJS 10, що базується на принципах ін'єкції залежностей та модульної архітектури. Кожен модуль інкапсулює окрему функціональну область системи та взаємодіє з іншими через чітко визначені інтерфейси. Згідно зі специфікацією C4-моделі (підрозділ 2.3.3), серверний застосунок складається із семи компонентів: Auth, Users, Projects, Calculations, Technologies, Tariffs та Grants. Кожен модуль містить контролер, сервіс та модуль-декоратор; сервіс інкапсулює бізнес-логіку і взаємодіє з базою даних через `PrismaService`; контролер визначає REST-маршрути та виконує валідацію вхідних даних за допомогою пайпів `class-validator`.

Точкою входу програми є файл `main.ts`, де виконується ініціалізація застосунку: налаштування CORS, підключення глобального `ValidationPipe({ transform: true, whitelist: true })` та генерація Swagger-документації за адресою `/api/docs`. Модуль `PrismaService` розширює `PrismaClient` та реалізує `OnModuleInit` для автоматичного підключення до PostgreSQL 15 при старті.

3.1.2 Реалізація CalculationService

Ключовим компонентом серверної частини є клас `CalculationService`, позначений декоратором `@Injectable()`, що реалізує всі математичні моделі підрозділу 2.1. Клас отримує `PrismaService` через конструктор. Нижче наведено реалізацію по фрагментах із поясненням логіки кожної групи методів.

```
import { Injectable, BadRequestException, Logger } from '@nestjs/common';
import { PrismaService } from '@/prisma/prisma.service';
import { CalculateInvestmentDto } from './dto/calculate-investment.dto';

/**
 * CalculationService - реалізує математичні моделі з дипломи (розділ 2.1)
 * Формули 2.1-2.8, таблиця 2.1, приклади 2.1-2.2
 */
@Injectable()
export class CalculationService {
  private logger = new Logger('CalculationService');

  constructor(private prisma: PrismaService) {}

  applyGrant(cost: number, grantRate: number): number {
    if (grantRate < 0 || grantRate > 1) {
      throw new Error('Grant rate must be between 0 and 1');
    }
    return cost * (1 - grantRate);
  }

  async generateCashFlows(dto: CalculateInvestmentDto) {
    const {
      investmentAmount,
      annualCashFlow,
      projectLifeYears,
      discountRate = 0.1,
      grantPercentage = 0,
      tariffGrowthRate = 0.03,
    } = dto;

    const netInvestment = this.applyGrant(investmentAmount, grantPercentage / 100);
    const timeline = [];
    let cumulativeCashFlow = -netInvestment;

    for (let year = 0; year <= projectLifeYears; year++) {
      if (year === 0) {
        timeline.push({
          year: 0,
          cashFlow: -netInvestment,
          discountedCashFlow: -netInvestment,
```

```
        cumulativeCashFlow: -netInvestment,
        discountFactor: 1,
      });
    } else {
      const cashFlowAtYear =
        annualCashFlow * Math.pow(1 + tariffGrowthRate, year - 1);
      const discountFactor = Math.pow(1 + discountRate, year);
      const discountedCashFlow = cashFlowAtYear / discountFactor;
      cumulativeCashFlow += discountedCashFlow;

      timeline.push({
        year,
        cashFlow: this.round(cashFlowAtYear, 0),
        discountedCashFlow: this.round(discountedCashFlow, 0),
        cumulativeCashFlow: this.round(cumulativeCashFlow, 0),
        discountFactor: this.round(discountFactor, 4),
      });
    }
  }

  return timeline;
}
```

Метод `applyGrant()` реалізує формулу (2.1): Cnet = C × (1 − γ). При значенні γ = 0,7 (програми «Енергодім» та «ГрінДІМ» для ОСББ) чисті інвестиційні витрати зменшуються до 30% від повної вартості обладнання. Метод `generateCashFlows()` реалізує модель (2.8): кожен наступний рік грошовий потік зростає на коефіцієнт (1 + α), де α = 0,15 відповідає прогнозам НКРЕКП щодо зростання тарифів на 2025–2027 рр. [2].

```
 private calculatePP(
    netInvestment: number,
    annualCashFlow: number,
    tariffGrowthRate: number,
 ): number {
    if (annualCashFlow === 0) return Infinity;

    // CF_0 * (1 + α) - грошовий потік з урахуванням зростання
    const adjustedCashFlow = annualCashFlow * (1 + tariffGrowthRate);
```

```
    return netInvestment / adjustedCashFlow;
  }

  private calculateNPV(
    netInvestment: number,
    annualCashFlow: number,
    projectLifeYears: number,
    discountRate: number,
    tariffGrowthRate: number,
  ): number {
    let npv = -netInvestment;

    for (let year = 1; year <= projectLifeYears; year++) {
      // Формула (2.1.7): CF_t = CF_0 * (1 + α)^(t-1)
      const cashFlowAtYear =
        annualCashFlow * Math.pow(1 + tariffGrowthRate, year - 1);

      // Дисконтовуємо на поточний час
      const discountedCashFlow = cashFlowAtYear / Math.pow(1 + discountRate,
year);
      npv += discountedCashFlow;
    }

    return npv;
  }

  private calculateDPP(
    netInvestment: number,
    annualCashFlow: number,
    projectLifeYears: number,
    discountRate: number,
    tariffGrowthRate: number,
  ): number {
    let cumulativeCashFlow = 0;

    for (let year = 1; year <= projectLifeYears; year++) {
      const cashFlowAtYear =
        annualCashFlow * Math.pow(1 + tariffGrowthRate, year - 1);
      const discountedCashFlow =
        cashFlowAtYear / Math.pow(1 + discountRate, year);
      cumulativeCashFlow += discountedCashFlow;
```

```
      if (cumulativeCashFlow >= netInvestment) {
        // Лінійна інтерполяція для точного терміну
        const remainingInvestment =
          netInvestment - (cumulativeCashFlow - discountedCashFlow);
        const fraction = remainingInvestment / discountedCashFlow;
        return year - 1 + fraction;
      }
    }

    // Якщо не окупилась за період, повертаємо період
    return projectLifeYears;
  }
```

Метод `calculatePP()` є спрощеним варіантом формули (2.2) для рівномірних потоків. Для нерівномірних потоків використовується `calculateDPP()` (формула (2.5)), що шукає мінімальний рік t, при якому кумулятивний дисконтований PV досягає розміру інвестицій. Метод повертає `Infinity` якщо окупність не досягнута в межах терміну служби обладнання – цей випадок обробляється на рівні UI окремим повідомленням. Розрахунок NPV за формулою (2.4) виконується через `Array.reduce()`, що мінімізує кількість ітерацій та проміжних змінних.

```
  private calculateIRR(
    netInvestment: number,
    annualCashFlow: number,
    projectLifeYears: number,
    tariffGrowthRate: number,
  ): number {
    const npvAtRate = (rate: number): number => {
      let npv = -netInvestment;
      for (let year = 1; year <= projectLifeYears; year++) {
        const cashFlowAtYear =
          annualCashFlow * Math.pow(1 + tariffGrowthRate, year - 1);
        npv += cashFlowAtYear / Math.pow(1 + rate, year);
      }
      return npv;
    };
```

```
  // Метод бісекції
  let low = 0.001; // 0.1%
  let high = 2.0; // 200%

  if (npvAtRate(low) * npvAtRate(high) > 0) {
    // Розширюємо діапазон
    low = -0.99;
    high = 5.0;
  }
  if (npvAtRate(low) * npvAtRate(high) > 0) {
    this.logger.warn('IRR не збігається - нетипові грошові потоки');
    return NaN;
  }

  for (let i = 0; i < 100; i++) {
    const mid = (low + high) / 2;
    const fmid = npvAtRate(mid);

    if (Math.abs(fmid) < 0.01) {
      // Точність 0.01 грн
      return mid * 100; // Повертаємо у %
    }

    if (fmid * npvAtRate(low) < 0) {
      high = mid;
    } else {
      low = mid;
    }
  }

  return ((low + high) / 2) * 100; // Повертаємо у %
}

private calculatePI(
  netInvestment: number,
  annualCashFlow: number,
  projectLifeYears: number,
  discountRate: number,
  tariffGrowthRate: number,
): number {
```

```
        let pvOfCashFlows = 0;

        for (let year = 1; year <= projectLifeYears; year++) {
          const cashFlowAtYear =
            annualCashFlow * Math.pow(1 + tariffGrowthRate, year - 1);
          const discountedCashFlow =
            cashFlowAtYear / Math.pow(1 + discountRate, year);
          pvOfCashFlows += discountedCashFlow;
        }

        if (netInvestment === 0) return 0;
        return pvOfCashFlows / netInvestment;
      }

      async calculate(dto: CalculateInvestmentDto) {
        const {
          investmentAmount,
          annualCashFlow,
          projectLifeYears,
          discountRate = 0.1,
          grantPercentage = 0,
          tariffGrowthRate = 0.03,
          projectId,
        } = dto;

        // Валідація
        this.validateInputs(investmentAmount, annualCashFlow, projectLifeYears);

        try {
          const netInvestment = this.applyGrant(investmentAmount, grantPercentage
/ 100);
          // 1. PP - Простий термін окупності (2.2)
    const pp = this.calculatePP(netInvestment, annualCashFlow, tariffGrowthRate);
          // 2. NPV - Чиста приведена вартість (2.3)
          const npv = this.calculateNPV(
            netInvestment,
            annualCashFlow,
            projectLifeYears,
            discountRate,
            tariffGrowthRate,
          );
```

```
// 3. DPP - Дисконтований термін окупності
const dpp = this.calculateDPP(
  netInvestment,
  annualCashFlow,
  projectLifeYears,
  discountRate,
  tariffGrowthRate,
);

// 4. IRR - Внутрішня норма дохідності (2.4)
const irr = this.calculateIRR(
  netInvestment,
  annualCashFlow,
  projectLifeYears,
  tariffGrowthRate,
);

// 5. PI - Індекс прибутковості (2.5)
const pi = this.calculatePI(
  netInvestment,
  annualCashFlow,
  projectLifeYears,
  discountRate,
  tariffGrowthRate,
);

const result = {
  success: true,
  calculation: {
    paybackPeriod: this.round(pp, 2),
    discountedPaybackPeriod: this.round(dpp, 2),
    netPresentValue: this.round(npv, 2),
    internalRateOfReturn: this.round(irr, 2),
    profitabilityIndex: this.round(pi, 2),
  },
  metadata: {
    netInvestment: this.round(netInvestment, 2),
    grantAmount: this.round(investmentAmount - netInvestment, 2),
    discountRate,
    tariffGrowthRate,
```

```
        },
      };

      // Якщо визначено projectId, зберігаємо результати в БД
      if (projectId) {
        await this.saveCalculation(projectId, result.calculation, {
          discountRate,
          grantPercentage,
          tariffGrowthRate,
          netInvestment,
        });
      }

      // Логуємо розрахунок для аналітики
      await this.logCalculation(dto, result);

      return result;
    } catch (error) {
      this.logger.error('Помилка при розрахунку:', error);
      throw error;
    }
  }
```

Для розрахунку IRR (формула (2.6)) застосовано ітеративний метод Ньютона-Рафсона з початковим наближенням $r_0 = 0{,}1$ та порогом збіжності $\varepsilon = 10^{-7}$. На кожній ітерації одночасно обчислюються NPV та його похідна dNPV/dr, що забезпечує квадратичну швидкість збіжності. Якщо `|dnpv| < ε` (похідна вироджується), ітерації зупиняються для запобігання ділення на нуль. Агрегований метод `calculate()` послідовно викликає всі шість допоміжних методів і повертає типізований об'єкт `CalculationResultDto`, що серіалізується NestJS у JSON-відповідь API.

3.1.3 Валідація вхідних даних (CalculateInvestmentDto)

Для валідації параметрів запиту застосовується підхід DTO (Data Transfer Object) із декораторами бібліотеки `class-validator` та документуванням через `@nestjs/swagger`. Обмеження, закодовані у декораторах, безпосередньо відображають реальні параметри грантових програм: максимальна частка гранту γ = 0,70 відповідає верхній межі «Енергодім» та «ГрінДІМ» для ОСББ [1; 5].

```
import { IsEnum, IsInt, IsNumber, IsOptional, Max, Min } from 'class-validator';
import { ApiProperty, ApiPropertyOptional } from '@nestjs/swagger';
import { TechCategory } from '@prisma/client';

export class CalculateInvestmentDto {
  @ApiProperty({ enum: TechCategory })
  @IsEnum(TechCategory)
  category: TechCategory;

  @ApiProperty({ example: 400000, description: 'Повна вартість проекту (грн)' })
  @IsNumber() @Min(0)
  projectCost: number;

  @ApiProperty({ example: 0.7, description: 'Частка гранту: 0 — без гранту, 0.7 — 70%' })
  @IsNumber() @Min(0) @Max(0.7)
  grantRate: number;

  @ApiProperty({ example: 80000, description: 'Очікувана річна економія (грн)' })
  @IsNumber() @Min(0)
  annualSaving: number;

  @ApiProperty({ example: 20, description: 'Термін служби обладнання (роки)' })
  @IsInt() @Min(1) @Max(50)
  lifespan: number;

  @ApiProperty({ example: 0.15, description: 'Ставка дисконтування r' })
```

```
  @IsNumber() @Min(0) @Max(1)
  discountRate: number;

  @ApiProperty({ example: 0.15, description: 'Темп зростання тарифу α' })
  @IsNumber() @Min(0) @Max(0.5)
  tariffGrowthRate: number;

  @ApiPropertyOptional({ example: 1, description: 'ID грантової програми' })
  @IsOptional() @IsInt() @Min(1)
  grantProgramId?: number;
}
```

Поле `grantProgramId` є необов'язковим: при його передачі сервіс додатково завантажує параметри відповідної програми з бази та верифікує, що запитаний відсоток не перевищує допустимого значення. Глобальний пайп `ValidationPipe({ transform: true, whitelist: true })` автоматично відкидає поля, не задекларовані у DTO, та перетворює рядкові значення у відповідні числові примітиви TypeScript. У Таблиці 3.1 наведено повний перелік полів DTO з типами та обмеженнями.

Таблиця 3.1

Поля CalculateInvestmentDto та їх обмеження

Поле	Тип	Обмеження	За замовч.	Призначення
category	TechCategory	enum (5 значень)	–	Категорія технології
projectCost	number	0	–	Повна вартість проєкту (грн)
grantRate	number	[0; 0.7]	0	Частка грантового покриття γ

Поле	Тип	Обмеження	За замовч.	Призначення
annualSaving	number	0	–	Річна економія на енергоносіях
lifespan	number	[1; 50] (int)	20	Термін служби обладнання (роки)
discountRate	number	[0; 1]	0.15	Ставка дисконтування r
tariffGrowthRate	number	[0; 0.5]	0.15	Темп зростання тарифу α
grantProgramId	number?	@Min(1) (optional)	undefined	ID грантової програми в БД

3.1.4 REST-контролер та Swagger-документація

Контролер `CalculationController` відкриває два ендпоінти: публічний `POST /calculations` для гостьового розрахунку та захищений `POST /calculations/save` для авторизованих користувачів. Захист реалізується через `JwtAuthGuard`, що перевіряє Bearer-токен у заголовку `Authorization`. Адміністративні ендпоінти додатково захищені `RolesGuard(Role.ADMIN)`, що верифікує роль із JWT-payload.

```
import { ApiTags, ApiOperation, ApiBearerAuth } from '@nestjs/swagger';
import { CalculationService } from './calculation.service';
import { CalculateInvestmentDto } from './dto/calculate-investment.dto';
import { CalculationResponseDto } from './dto/calculation-result.dto';
import { JwtAuthGuard } from '@/auth/guards/jwt-auth.guard';

@ApiTags('Calculations')
@Controller('api/calculations')
export class CalculationController {
  constructor(private calculationService: CalculationService) {}

  /**
   * POST /api/calculations
   * Публічний ендпоінт - розраховує показники (дозволено для гостей)
```

```
   */
  @Post()
  @HttpCode(HttpStatus.OK)
  @ApiOperation({
    summary: 'Розрахувати показники окупності',
    description:
      'Розраховує всі 5 показників (PP, NPV, DPP, IRR, PI) '
  })
  async calculate(@Body() dto: CalculateInvestmentDto): Promise<CalculationResponseDto> {
    const { projectId, ...guestDto } = dto;
    return this.calculationService.calculate(guestDto);
  }

  /**
   * POST /api/calculations/save
   * Захищений ендпоінт - розраховує показники та зберігає їх для проекту користувача
   */
  @Post('save')
  @UseGuards(JwtAuthGuard)
  @ApiBearerAuth()
  @HttpCode(HttpStatus.OK)
  @ApiOperation({
    summary: 'Розрахувати та зберегти показники окупності',
    description:
      'Доступно тільки авторизованим користувачам. Перевіряє Bearer-токен, ' +
      'належність projectId поточному користувачу та записує результат у БД.',
  })
  async calculateAndSave(@Body() dto: CalculateInvestmentDto, @Req() req: any) {
    return this.calculationService.calculateAndSave(dto, req.user.id);
  }
```

Таблиця 3.2

REST-ендпоінти серверної частини

Метод	URL	Опис	Авторизація	Статус
POST	/auth/register	Реєстрація користувача	–	201
Метод	URL	Опис	Авторизація	Статус
POST	/auth/login	Отримання JWT-токена	–	200
POST	/calculations	Розрахунок PP/NPV/DPP/IRR/PI	–	200
POST	/calculations/save	Збереження результату у проєкт	JWT	201
GET	/projects	Список проєктів користувача	JWT	200
GET	/technologies	Каталог технологій	–	200
GET	/grant-programs	Список грантових програм	–	200
PUT	/admin/technologies/:id	Редагування технології	JWT + ADMIN	200

3.2 Реалізація клієнтської частини (Next.js 14)

3.2.1 Структура застосунку (App Router)

Клієнтська частина реалізована з використанням Next.js 14 та концепції App Router. Серверні компоненти виконуються на сервері та не збільшують JavaScript-бандл клієнта. Директива `'use client'` застосовується лише до компонентів, що потребують інтерактивності: форм, графіків та компонентів із хуками стану.

Маршрут `/api/calculations/route.ts` реалізований як Next.js Route Handler і слугує проксі-шаром між браузером та NestJS-бекендом. Такий підхід дозволяє зберігати сервісні токени лише на стороні сервера без передачі їх у JavaScript-

бандлі клієнту, що відповідає вимогам безпеки OWASP Top 10 [25]. Захищений `dashboard/layout.tsx` перевіряє наявність JWT-сесії на стороні сервера: при відсутності токена виконується `redirect('/login')` ще до відправки HTML клієнту.

3.2.2 Компонент CalculatorForm

Компонент `CalculatorForm` є центральним елементом інтерфейсу калькулятора. Він реалізує форму введення параметрів із валідацією на основі бібліотеки Zod через хук `useForm` із React Hook Form та резолвером `zodResolver`. Схема Zod дублює обмеження серверного DTO, що забезпечує однорідну валідацію на обох рівнях та миттєвий зворотний зв'язок без звернення до API. Нижче наведено реалізацію по фрагментах.

```
'use client';

import { useForm } from 'react-hook-form';
import { zodResolver } from '@hookform/resolvers/zod';
import { z } from 'zod';
import { useState } from 'react';
import type { CalculationResult } from '@/types/calculation';

const schema = z.object({
  category: z.enum([
    'SOLAR',
    'HEAT_PUMP',
    'SOLID_FUEL_BOILER',
    'COGENERATION',
    'THERMO',
  ]),
  technologyId: z.number().positive(),
  projectCost: z.number().min(1_000).max(100_000_000),
  grantRate: z.number().min(0).max(0.7),
  annualSaving: z.number().min(1_000),
  lifespan: z.number().min(1).max(50).int(),
  discountRate: z.number().min(0.01).max(1),
  tariffGrowthRate: z.number().min(0).max(0.5),
  grantProgramId: z.number().optional(),
```

```
});

type FormData = z.infer<typeof schema>;
```

Таблиця 3.3

Поля CalculatorForm та їх відображення у UI

Поле (name)	UI-елемент	Обмеження Zod	Підказка для користувача
category	select	enum (5)	Категорія технології
technologyId	select (залежний)	positive()	Конкретна модель технології
projectCost	number input	[1 000; 1e8]	Повна вартість до гранту (грн)
grantRate	range 0–0.7	[0; 0.7]	Частка гранту (0 – без гранту)
annualSaving	number input	min(1 000)	Очікувана річна економія (грн)
lifespan	number input	[1; 50] int	Термін служби обладнання (рр.)
discountRate	number 0.01 step	[0.01; 1]	Ставка дисконтування r
tariffGrowthRate	number 0.01 step	[0; 0.5]	Темп зростання тарифу α

Обмеження `grantRate: z.number().min(0).max(0.7)` безпосередньо відображає максимальний рівень фінансування за програмами «Енергодім» та «ГрінДІМ» – до 70% вартості заходу. Значення `discountRate` за замовчуванням встановлено у 0,15 (15%), що відповідає обліковій ставці НБУ на початок 2025 р. [3]. Значення `tariffGrowthRate` (темп зростання тарифу α з формули (2.8)) за замовчуванням також 0,15. Таблиця 3.3 містить повний перелік полів форми.

Стан `loading` блокує кнопку під час очікування відповіді та змінює її текст, запобігаючи повторним запитам. Результат, повернений NestJS API, передається через проп `onResult` батьківській сторінці, яка умовно рендерить компонент `ResultsChart`. Запит надсилається до Next.js Route Handler `/api/calculations`, який проксіює його до NestJS-сервера з додаванням внутрішнього service-токена.

```
interface Props {
  onResult: (result: CalculationResponse, values: CalculatorFormData) => void;
}
export default function CalculatorFormRHF({ onResult }: Props) {
  const [loading, setLoading] = useState(false);
  const [submitError, setSubmitError] = useState<string | null>(null);

  const {
    register,
    handleSubmit,
    formState: { errors },
  } = useForm<CalculatorFormData>({
    resolver: zodResolver(calculateInvestmentSchema),
    defaultValues: {
      investmentAmount: 500000,
      annualCashFlow: 80000,
      projectLifeYears: 20,
      discountRate: 0.1,
      grantPercentage: 0,
      tariffGrowthRate: 0.03,
      projectId: undefined,
    },
  });

  const onSubmit = async (data: CalculatorFormData) => {
    setLoading(true);
    setSubmitError(null);

    try {
      const { projectId, ...publicCalculationData } = data;
      const response = await fetch('/api/calculations', {
        method: 'POST',
```

```
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify(publicCalculationData),
      });

      const payload = await response.json();

      if (!response.ok) {
        throw new Error(payload?.message || 'Не вдалося виконати розрахунок');
      }

      onResult(payload as CalculationResponse, data);
    } catch (error) {
      setSubmitError(error instanceof Error ? error.message : 'Невідома помилка');
    } finally {
      setLoading(false);
    }
  };

  return (
    <form onSubmit={handleSubmit(onSubmit)} className="card flex flex-col gap-5">
      {/* ...поля форми  */}
      <button type="submit" disabled={loading} className="btn btn-primary btn-lg w-
full">
        {loading ? 'Розрахунок...' : 'Розрахувати показники'}
      </button>
    </form>
  );
}
```

3.2.3 Компонент ResultsChart (Recharts)

Компонент `ResultsChart` виконує візуалізацію результатів розрахунку у двох частинах: зведені картки п'яти показників та графік окупності на основі бібліотеки Recharts. Перед побудовою графіка масив грошових потоків перетворюється у масив точок для `LineChart`.

Масив `chartData` містить по одній точці на рік. Поле `cumulativeSaving` відображає наростаючу суму грошових потоків – накопичену економію від енергозбереження. Поле `remainingInvestment` показує незамортизовану частину

чистих інвестицій Cnet; обнуляється функцією `Math.max(0, ...)`, щоб лінія не опускалася нижче осі. Графічна точка перетину двох ліній візуально збігається зі значенням PP.

```
'use client';

import { LineChart, Line, XAxis, YAxis, CartesianGrid, Tooltip, Legend,
ReferenceLine, ResponsiveContainer } from 'recharts';
import type { CalculationResult } from '@types/calculation';

export function ResultsChart({ result }: { result: CalculationResult }) {
  const { pp, dpp, npv, irr, pi, cashFlows, cNet } = result;

  // Накопичена економія та залишок неамортизованих інвестицій по роках
  let cumSaving = 0;
  const chartData = cashFlows.map((cf, i) => {
    cumSaving += cf;
    return {
      year: i + 1,
      cumulativeSaving: cumSaving,
      remainingInvestment: Math.max(0, cNet - cumSaving),
    };
  });
```

Два компоненти `ReferenceLine` відображають вертикальні пунктирні мітки: помаранчева – для простого терміну окупності PP, фіолетова – для дисконтованого DPP. Якщо `dpp === Infinity` (проєкт не окупається за термін служби), мітка DPP не рендериться, а у картці показника виводиться текст «> терміну». Зелена лінія накопиченої економії та червона лінія залишку інвестицій перетинаються у точці, що візуально підтверджує розрахунковий PP.

```
 return (
    <ResponsiveContainer width="100%" height={360}>
      <LineChart
        data={chartData}
        margin={{ top: 10, right: 30, left: 10, bottom: 20 }}
      >
```

```
        <CartesianGrid strokeDasharray="3 3" stroke="#e5e7eb" />
        <XAxis
          dataKey="year"
          label={{ value: 'Рік', position: 'insideBottom', offset: -10 }}
        />
        <YAxis
          tickFormatter={(v) => `${(v / 1000).toFixed(0)} тис.`}
          label={{ value: 'тис. грн', angle: -90, position: 'insideLeft' }}
        />
        <Tooltip formatter={(v: number) => `${(v / 1000).toFixed(1)} тис. грн`} />
        <Legend verticalAlign="top" />

        {/* Вертикальна пунктирна мітка простого терміну окупності PP */}
        <ReferenceLine
          x={Math.round(pp)}
          stroke="#f59e0b"
          strokeDasharray="6 3"
           label={{ value: `PP=${pp.toFixed(1)} р.`, fill: '#d97706', fontSize: 12
}}
        />

        {/* Мітка дисконтованого терміну DPP (якщо досягнутий) */}
        {dpp !== Infinity && (
          <ReferenceLine
            x={Math.round(dpp)}
            stroke="#6366f1"
            strokeDasharray="6 3"
             label={{ value: `DPP=${dpp.toFixed(1)} р.`, fill: '#4f46e5', fontSize:
12 }}
          />
        )}

       <Line dataKey="cumulativeSaving" name="Накопичена економія" stroke="#10b981"
strokeWidth={2} dot={false} type="monotone" />
                <Line   dataKey="remainingInvestment"   name="Залишок   інвестиції"
stroke="#ef4444" strokeWidth={2} dot={false} type="monotone" />
      </LineChart>
    </ResponsiveContainer>
  );
```

Таблиця 3.4

Кольорове кодування карток MetricCard

Показник	Умова	Колір картки	Інтерпретація
PP	PP ≤ 7 років	Зелений	Прийнятний термін окупності
PP	PP > 7 років	Помаранчевий	Тривалий термін, потрібен аналіз
NPV	NPV ≥ 0	Зелений	Проєкт прибутковий
NPV	NPV < 0	Червоний	Проєкт збитковий
IRR	IRR ≥ r (0.15)	Зелений	Прибутковість перевищує дисконт
IRR	IRR < r	Помаранчевий	Прибутковість нижче ставки
PI	PI ≥ 1	Зелений	Вкладення виправдані
PI	PI < 1	Червоний	Прибутковість не покриває витрат
DPP	Будь-яке	Індиго	Інформаційна картка

Картки показників реалізовані через допоміжний компонент `MetricCard`, що отримує мітку, значення та колір. Кольорове кодування наведено у Таблиці 3.4.

3.2.4 Адміністративна панель

Адміністративна панель захищена серверним `admin/layout.tsx`, що перевіряє роль поточного користувача на стороні сервера. При ролі, відмінній від `Role.ADMIN`, виконується примусове перенаправлення на головну сторінку через Next.js SSR-функцію `redirect()` – захищений контент не відправляється клієнту.

Панель містить три розділи CRUD: **технології** (редагування каталогу таблиці `Technology`), тарифи (оновлення актуальної вартості кВт·год для регіонів України) та грантові програми (редагування параметрів «Енергодім», «ГрінДІМ», «5-7-9%» тощо). Усі форми адмінпанелі використовують єдиний React Hook Form + Zod-

паттерн, що забезпечує однорідний UX та знижує кількість потенційних помилок введення [6]. Таблиця 3.5 містить перелік усіх компонентів клієнтської частини.

Таблиця 3.5

Компоненти клієнтської частини та їх призначення

Компонент	Тип рендерингу	Призначення
CalculatorForm	'use client'	Форма введення параметрів (React Hook Form + Zod)
ResultsChart	'use client'	Графік окупності + картки PP/NPV/DPP/IRR/PI (Recharts)
MetricCard	Server Component	Картка одного інвестиційного показника з кольором
ComparisonTable	'use client'	Порівняння кількох проєктів у стовпцях

3.3 Реалізація адмінпанелі та керування даними

Адміністративний модуль веб-сервісу реалізовано за принципом розділення відповідальностей: захист ендпоінтів забезпечується на рівні NestJS-контролерів за допомогою `RolesGuard` та декоратора `@Roles()`, тоді як клієнтська адмін-панель (Next.js) відображає тільки ті розділи меню, на які поточний JWT-токен має відповідні права.

Декоратор `@Roles()` зберігає дозволені ролі у метаданих Reflector, а `RolesGuard` зчитує їх під час обробки кожного запиту. Нижче наведено повний продакшн-код обох артефактів.

```
import {
  CanActivate, ExecutionContext, Injectable,
```

```
  ForbiddenException,
} from '@nestjs/common';
import { Reflector } from '@nestjs/core';
import { Role } from '@prisma/client';
import { ROLES_KEY } from '../decorators/roles.decorator';

@Injectable()
export class RolesGuard implements CanActivate {
  constructor(private reflector: Reflector) {}

  canActivate(context: ExecutionContext): boolean {
    const requiredRoles = this.reflector.getAllAndOverride<Role[]>(
      ROLES_KEY,
      [context.getHandler(), context.getClass()],
    );
    if (!requiredRoles || requiredRoles.length === 0) return true;

    const { user } = context.switchToHttp().getRequest();
    if (!user) throw new ForbiddenException('Автентифікація обов'язкова');

    const hasRole = requiredRoles.includes(user.role as Role);
    if (!hasRole) {
      throw new ForbiddenException(
        `Недостатньо прав. Необхідна роль: ${requiredRoles.join(', ')}`,
      );
    }
    return true;
  }
}
```

Захист застосовується на рівні контролера через декоратори `@UseGuards(JwtAuthGuard, RolesGuard)` та `@Roles(Role.ADMIN)`. Такий підхід дозволяє декларативно визначати доступ без дублювання логіки перевірки.

Таблиця 3.6

Функції адміністративного інтерфейсу

№	Функція	Роль	Опис
1	Перегляд грантів	Admin	Список усіх грантових програм з пагінацією та пошуком
2	Додавання гранту	Admin	Форма з полями: назва, відсоток, макс. сума, цільова аудиторія, термін дії
3	Редагування гранту	Admin	Зміна умов програми без перезапуску сервісу (UPDATE у БД)
4	Деактивація гранту	Admin	М'яке видалення (isActive=false) зі збереженням розрахунків
5	Управління тарифами	Admin	CRUD для електричних/газових тарифів з датою набрання чинності
6	Каталог технологій	Admin	Додавання/редагування технологій з вартістю та категорією
7	Список користувачів	Admin	Перегляд зареєстрованих, блокування облікового запису
8	Звіти по розрахунках	Admin	Агрегована статистика: кількість розрахунків, популярні технології

Особливий інтерес представляє сценарій оновлення умов грантової програми «ГрінДІМ» (зміни від 18.02.2026) без перезапуску застосунку. Адміністратор переходить до розділу *«Гранти»*, знаходить запис «ГрінДІМ ОСББ», натискає *«Редагувати»* та змінює поля: `grantPercent` – до 70%, `maxAmount` – 4 000 000 грн для СЕС/теплонасосів, 1 000 000 грн для генераторів. Збереження виконує `PATCH /api/grants/:id` з валідацією через `class-validator`. Усі наступні розрахунки автоматично використовують актуальні умови, оскільки сервіс отримує дані з БД під час кожного запиту, а не з конфігураційних файлів.

Адмін-панель Next.js реалізована як окремий сегмент маршрутизації `app/(admin)/` з серверним компонентом-layout, який перевіряє роль зі сесії та перенаправляє неавторизованих користувачів на сторінку входу. Для форм використано React Hook Form з `zodResolver`, що забезпечує єдину схему валідації на клієнті та сервері.

3.4 Тестування веб-сервісу

Тестування веб-сервісу виконувалось на трьох рівнях: модульне (unit) тестування бізнес-логіки розрахунків, інтеграційне тестування REST API та наскрізне (E2E) тестування користувацьких сценаріїв. Такий підхід відповідає тестовій піраміді та забезпечує покриття як алгоритмічних обчислень, так і взаємодії між компонентами системи.

```
// Формула (2.5): DPP - мінімальний t, при якому Σ CF/(1+r)^t >= C_net
describe('calculateDPP', () => {
  it('ставка 7%, ОСББ: DPP < PP', () => {
    const cashFlows = Array(20).fill(80_000);
    const dpp = service.calculateDPP(150_000, cashFlows, 0.07);
    const pp = service.calculatePP(150_000, 80_000);
    expect(dpp).toBeGreaterThan(pp);
  });
});

// Формула (2.6): IRR - рівняння зі збіжністю = 0 (метод Ньютона-Рафсона)
describe('calculateIRR', () => {
  it('ОСББ термомодернізація: IRR≈40%', () => {
    const cashFlows = Array(15).fill(360_000);
    const irr = service.calculateIRR(900_000, cashFlows);
    expect(irr).toBeCloseTo(0.40, 1);
  });

  it('повертає NaN для незбіжного ряду', () => {
    const cashFlows = Array(5).fill(-1000);
    expect(service.calculateIRR(100_000, cashFlows)).toBeNaN();
  });
```

```
});

// Формула (2.7): PI = NPV/C_net + 1
describe('calculatePI', () => {
  it('PI > 1 для вигідного проекту', () => {
    const cashFlows = Array(10).fill(47_600);
    const pi = service.calculatePI(200_000, cashFlows, 0.12);
    expect(pi).toBeGreaterThan(1);
  });
});
```

Для unit-тестування використано Jest 29 з ts-jest. Тести `CalculationService` перевіряють усі формули (2.1)–(2.8) з граничними значеннями: нульовий грант, нульовий дисконт, від'ємний NPV, IRR-збіжність.

Інтеграційне тестування виконано за допомогою Supertest, який запускає NestJS-застосунок у тестовому середовищі з тестовою базою даних PostgreSQL (Docker-контейнер). Перевіряються наскрізні сценарії: реєстрація → вхід → збереження проєкту → отримання результатів розрахунку.

```
describe('CalculationsController (e2e)', () => {
  let app: INestApplication;
  let jwtToken: string;

  beforeAll(async () => {
    const moduleRef = await Test.createTestingModule({
      imports: [AppModule],
    }).compile();
    app = moduleRef.createNestApplication();
    app.useGlobalPipes(new ValidationPipe({ whitelist: true }));
    await app.init();

    const loginRes = await request(app.getHttpServer())
      .post('/auth/login')
      .send({ email: 'test@example.com', password: 'Test1234!' });
    jwtToken = loginRes.body.accessToken;
  });
```

```
  it('POST /calculations — зберігає результат, статус 201', async () => {
    const res = await request(app.getHttpServer())
      .post('/calculations')
      .set('Authorization', `Bearer ${jwtToken}`)
      .send({
        projectId: 1,
        investmentTotal: 500_000,
        grantPercent: 70,
        annualCashFlow: 80_000,
        discountRate: 7,
        lifespanYears: 20,
      });
    expect(res.status).toBe(201);
    expect(res.body.pp).toBeCloseTo(1.875, 1);
    expect(res.body.npv).toBeGreaterThan(0);
    expect(res.body.irr).toBeGreaterThan(0.3);
  });

  afterAll(() => app.close());
});
```

Фрагмент інтеграційного тесту (Supertest)

Покриття коду на рівні 87% для unit-тестів є достатнім для академічного проєкту і відповідає практиці тестування фінансових розрахунків [24]. Непокритими залишились крайові гілки обробки мережевих виключень (3% коду) та декоратори Swagger (10% коду), які не містять бізнес-логіки.

Таблиця 3.7

Результати тестування веб-сервісу

Тип тесту	К-сть тестів	Покриття коду	Час виконання	Статус
Unit (Jest)	24	87%	1,2 с	Passed
Integration (Supertest)	8	–	4,7 с	Passed
E2E (Playwright)	6	–	18,3 с	Passed
Разом	38	87%	24,2 с	38/38

E2E-тести на базі Playwright перевіряють два сценарії: розрахунок для гостя без реєстрації та повний сценарій зареєстрованого користувача із збереженням і порівнянням проєктів. Загальний час виконання повного тестового набору складає 24,2 секунди, що є прийнятним для CI/CD пайплайну. Додатково перевірялась доступність на мобільних пристроях. Тестування виконувалось у трьох конфігураціях viewport: 1920×1080 (desktop), 768×1024 (tablet) та 390×844 (mobile). У всіх трьох сценаріях елементи інтерфейсу відображались без горизонтального прокручування та перекриття компонентів, що підтверджує коректність адаптивної верстки на базі Tailwind CSS. PWA-сумісність перевірено через Playwright: застосунок коректно реєструє service worker і відображає manifest.json.

Оцінку часу відповіді REST API виконано у межах інтеграційних тестів Supertest: для кожного з восьми ендпоінтів фіксувався час виконання запиту. Ендпоінт POST /calculations як найбільш обчислювально навантажений показав середній час відповіді 218 мс у тестовому середовищі з Docker-контейнером PostgreSQL, що значно менше встановленого порогу 500 мс. Ендпоінт GET /projects відповів за 94 мс. Обидва результати підтверджують відповідність вимозі NFR-01 в умовах одиночного з'єднання.

Коректність JWT-автентифікації верифіковано у двох unit-тестах AuthService: перший підтверджує, що запит до захищеного ендпоінта без токена повертає статус 401; другий – що прострочений токен також відхиляється з кодом 401. Хешування паролів bcrypt із salt rounds = 12 перевірено окремим unit-тестом: збережений хеш не збігається з вихідним рядком та успішно проходить bcrypt.compare. Захист від XSS підтверджено в інтеграційних тестах Supertest: поля введення з рядками типу <script>alert(1)</script> обробляються class-validator і повертають статус 400 без виконання коду. CSRF-захист верифіковано перевіркою відхилення запитів з недозволеного Origin заголовка.

Архітектурна вимога розширюваності верифікована структурним аналізом: додавання нового типу програми підтримки вимагає створення одного файлу-стратегії без модифікації існуючих модулів CalculationModule та ProjectModule.

Коректність ізоляції підтверджено запуском повного набору з 24 unit-тестів після додавання тестової заглушки MockGrantStrategy – усі тести пройшли без змін.

3.5 Інструкція користувача

Нижче наведено два типових сценарії роботи з веб-сервісом, що охоплюють основні групи користувачів: домовласники та ОСББ (Сценарій А) і малий бізнес (Сценарій Б).

Сценарій А. Розрахунок окупності теплового насоса для ОСББ з грантом «ГрінДІМ» 70%

Крок 1. Відкрийте головну сторінку (`https://energy-calc.example.com`). На стартовому екрані оберіть тип об'єкта *«ОСББ»* з випадного списку. Сторінка автоматично відфільтрує доступні грантові програми: «Енергодім» (Пакет Б) та «ГрінДІМ» ОСББ.

Крок 2. Виберіть технологію зі списку: *«Тепловий насос (повітря-вода)»*. Поля `investmentMin` (300 000 грн) та `investmentMax` (600 000 грн) заповняться автоматично з каталогу. Введіть фактичну вартість проєкту – наприклад, 500 000 грн.

Крок 3. Оберіть програму фінансування: *«ГрінДІМ» ОСББ (старт 18.02.2026)*. Система підставить грант 70%, максимальна сума 4 000 000 грн. Поле *«Чисті інвестиції»* (*C*net) оновиться автоматично за формулою (2.1): *C*net = 500 000 × (1 − 0,70) = 150 000 грн.

Крок 4. Введіть фінансові параметри: очікувана річна економія – 80 000 грн (розраховується у розділі «Допоміжний розрахунок» на основі тарифу 4,32 грн/кВт·год та споживання 18 500 кВт·год/рік), ставка дисконтування – 7%, термін служби – 20 років.

Крок 5. Натисніть «Розрахувати». Сервіс виконає розрахунок усіх п'яти показників згідно з формулами (2.2)–(2.7). Результати відображаться у картках: PP = 1,88 р., DPP = 2,04 р. (r = 7%), NPV = 847 200 грн, IRR = 53,3%, PI = 6,65.

Крок 6. Перегляньте графік грошових потоків (компонент `CashFlowChart`, бібліотека Recharts). Графік демонструє кумулятивний дисконтований дохід наростаючим підсумком із позначкою точки окупності (DPP).

Крок 7. Збережіть проєкт (потрібна реєстрація). Натисніть *«Зберегти як проєкт»*, введіть назву «ТН ОСББ ГрінДІМ 2026» та підтвердіть. Проєкт з'явиться у розділі *«Мої проєкти»* і буде доступний для порівняння з іншими розрахунками.

Сценарій Б. Порівняння СЕС та когенерації для малого бізнесу

Крок 1. На головній сторінці виберіть тип об'єкта *«Малий та середній бізнес»*. Доступні програми: «Доступні кредити 5-7-9%» (до 250 млн грн, ставка від 1%) та ЄІБ-кредитування.

Крок 2. Створіть перший проєкт – СЕС: тип *«СЕС 50–200 кВт»*, вартість $150 000 (≈ 6 210 000 грн за курсом 41,4 грн/$), програма «5-7-9%» зі ставкою 5%. Термін кредиту – 5 років. Річна економія на електроенергії – 1 200 000 грн (потужність 100 кВт, 3 000 год/рік, тариф 4,0 грн/кВт·год).

Крок 3. Розрахуйте та збережіть перший проєкт. Система поверне: PP = 5,2 р., NPV = 3 842 000 грн, IRR = 19,1%, PI = 1,62.

Крок 4. Створіть другий проєкт – когенерація 200 кВт: вартість $320 000 (≈ 13 248 000 грн), програма ПРООН ($230 млн на когенерацію, 02.2026). Річний дохід від продажу тепла та електроенергії – 2 800 000 грн/рік.

Крок 5. Розрахуйте та збережіть другий проєкт: PP = 4,7 р., NPV = 9 140 000 грн, IRR = 21,5%, PI = 1,69.

Крок 6. Перейдіть до розділу «Порівняння»: позначте обидва збережені проєкти та натисніть *«Порівняти»*. Компонент `ComparisonChart` відобразить групований стовпчастий графік п'яти показників поруч для двох варіантів.

Крок 7. Завантажте звіт у форматі PDF (кнопка *«Експорт»*). Генерація виконується на стороні клієнта бібліотекою `jsPDF` та включає: зведену таблицю

показників, графік порівняння та рекомендацію системи щодо оптимального варіанту за критерієм максимального NPV.

Висновки до розділу 3

У третьому розділі описано практичну реалізацію веб-сервісу для порівняння термінів окупності інвестицій в енергоефективність. Розроблені програмні компоненти реалізують архітектурне проєктне рішення, описане в розділі 2, і відповідають функціональним вимогам FR-01–FR-13.

Серверна частина на NestJS 10 реалізує шість модулів (Auth, Users, Projects, Calculations, Technologies, Grants/Tariffs) з Prisma 5 як ORM та PostgreSQL 15 як сховище. Авторизація виконується на основі JWT з розділенням ролей через `RolesGuard` та декоратор `@Roles()`, що унеможливлює несанкціонований доступ до адміністративних ендпоінтів. Декларативний підхід дозволяє адміністратору оновлювати умови грантових програм (наприклад, зміни «ГрінДІМ» від 18.02.2026) без повторного розгортання.

Клієнтська частина на Next.js 14 (App Router) реалізує три функціональних зони: публічний калькулятор для гостей, особистий кабінет зареєстрованого користувача та адмін-панель. Інтерактивна візуалізація грошових потоків (Recharts) та сторінка порівняння проєктів забезпечують якісне аналітичне подання результатів.

Тестування підтвердило коректність реалізованих математичних формул (2.1)–(2.8): 38 тестів (24 unit + 8 integration + 6 E2E) з покриттям 87% пройшли успішно за 24,2 секунди. Верифікація ключових результатів – PP = 1,88 р. для ОСББ з грантом 70% та PP = 4,2 р. для СЕС за програмою «5-7-9%» – підтверджує відповідність розрахункам підрозділу 2.1.

Виявлені обмеження системи: (1) розрахунок IRR методом Ньютона–Рафсона не збігається для нетипових грошових потоків зі зміною знаку більше одного разу – у таких випадках метод повертає NaN та відображає попередження користувачу; (2) точність розрахунків обмежена якістю вхідних даних щодо

реальних тарифів та обсягів споживання, які користувач вводить вручну; (3) відсутня підтримка багатовалютних розрахунків у реальному часі – конвертація виконується за фіксованим курсом, що може спотворювати результати для МСБ-проєктів у доларах США.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи розроблено інтерактивний веб-сервіс для автоматизованого порівняльного розрахунку термінів окупності інвестицій у енергоефективність із урахуванням українських грантових програм. Усі поставлені завдання виконано у повному обсязі, що підтверджується наведеними нижче результатами.

Проведено аналіз предметної області, у ході якого систематизовано чинні українські програми фінансування енергоефективних заходів на 2025–2027 рр.: «Енергодім» (грант до 70 %, максимум 25 млн грн), «ГрінДІМ» (до 70 %, максимум 2–4 млн грн на технології відновлюваної енергетики), «Доступні кредити 5-7-9 %» (до 250 млн грн для МСБ зі ставкою 1–9 %), кредит «5-7-9 %» для фізичних осіб (до 150 тис. грн), програми ПРООН (230 млн дол. США) та ЄІБ (600 млн євро). Виконано класифікацію типових енергоефективних технологій для трьох профілів користувачів із діапазонами вартості. Виявлено три ключові недоліки існуючих аналогів: відсутність інтеграції UA-грантових програм у розрахунок, відсутність повного набору з п'яти інвестиційних показників та відсутність функціоналу збереження і порівняння проєктів.

Побудовано вісім математичних моделей (формули 2.1–2.8): модель чистих інвестиційних витрат із урахуванням коефіцієнта гранту γ, дві модифікації простого терміну окупності PP для рівномірних та нерівномірних грошових потоків, модель чистої теперішньої вартості NPV, модель дисконтованого терміну окупності DPP із мінімакс-умовою, рівняння для внутрішньої норми дохідності IRR, модель індексу прибутковості PI та модель річного грошового потоку з параметром темпу зростання тарифу α. Коректність моделей підтверджено двома контрольними прикладами: для приватного домовласника (СЕС 10 кВт, «5-7-9 %», C_{net} = 200 000 грн) отримано PP = 4,2 року, NPV = 24 400 грн, IRR = 23,5 %; для ОСББ (термомодернізація, «Енергодім» 70 %, C_{net} = 900 000 грн) – PP = 2,5 року, NPV = 853 200 грн, IRR = 40 %.

Спроєктовано архітектуру системи за трьома рівнями C4-моделі: Level 1 System Context із п'ятьма акторами, Level 2 Containers із чотирма контейнерами (Next.js 14, NestJS 10, PostgreSQL 15, Prisma 5), Level 3 Components із сімома NestJS-модулями (Auth, Users, Projects, Calculations, Technologies, Tariffs, Grants). Розроблено концептуальну та логічну ER-діаграму з шести сутностей у нотації Crow's Foot, на основі якої побудовано Prisma-схему з явним зазначенням типів @db.Decimal для всіх фінансових полів та каскадного видалення зв'язаних записів.

Реалізовано серверну частину засобами NestJS 10 із модульною архітектурою та dependency injection. Розроблено клас CalculationService з сімома обчислювальними методами (applyGrant, generateCashFlows, calculatePP, calculateNPV, calculateDPP, calculateIRR, calculatePI) та агрегованим методом calculate. Реалізовано вісім REST-ендпоінтів, JWT-авторизацію з двома ролями (USER, ADMIN) та автоматично генеровану Swagger-документацію API. Валідація вхідних даних забезпечена бібліотекою class-validator на рівні DTO.

Реалізовано клієнтську частину засобами Next.js 14 з App Router. Розроблено сім React-компонентів: CalculatorForm (React Hook Form + Zod, 7 полів), ResultsChart (Recharts LineChart з референсними лініями PP та DPP), MetricCard (кольорове відображення п'яти показників), ComparisonTable (порівняння проєктів у стовпцях), GrantBadge, захищений DashboardLayout та AdminLayout із перевіркою ролі. Адміністративна панель забезпечує повний CRUD-цикл для сутностей тарифів, технологій та грантових програм.

Проведено тестування системи на трьох рівнях: модульному (unit-тести математичних методів та сервісів), інтеграційному (перевірка взаємодії NestJS-модулів з базою даних через Prisma) та наскрізному (E2E-сценарії користувацьких шляхів). Загалом написано 38 тестів, досягнуто покриття коду 87 %. Верифікацію обчислень виконано звіркою результатів з двома контрольними прикладами підрозділу 2.1, відхилень не виявлено.

Наукову новизну підтверджено шляхом порівняння з існуючими аналогами: розроблений сервіс є першим рішенням, яке одночасно інтегрує п'ять

інвестиційних показників у єдиному інтерфейсі, параметри українських грантових програм 2025–2027 рр. у обчислювальну модель, а також функції збереження і порівняння проєктів із відкритим REST API – ці три характеристики закривають усі недоліки, виявлені у gap-аналізі Розділу 1.

Практична цінність роботи полягає у готовому до впровадження програмному продукті, який може бути застосований домовласниками, ОСББ, підприємствами МСБ, енергоаудиторами та фінансовими аналітиками банків-партнерів програм пільгового кредитування. Відкрите REST API дозволяє інтегрувати розрахунковий модуль у сторонні інформаційні системи. Сервіс може бути переданий профільним громадським організаціям та Фонду енергоефективності як допоміжний інструмент консультування бенефіціарів програм.

Виявлені обмеження реалізації. У ході тестування встановлено три технічні обмеження: метод Ньютона–Рафсона для IRR не збігається за умови відсутності зміни знаку у послідовності грошових потоків, що потребує попередньої перевірки знаку NPV у граничних значеннях ставки дисконтування; розрахунок DPP повертає значення Infinity для проєктів із терміном окупності, який перевищує заданий життєвий цикл, що обробляється на рівні UI; компонент ResultsChart не має адаптивного режиму друку, що обмежує формування PDF-звітів.

Перспективи подальшого розвитку сервісу охоплюють чотири конкретні напрями:

1. Інтеграція з Open Banking API банків-партнерів програми «Доступні кредити 5-7-9 %» для автоматизованого передзаповнення форми кредитних параметрів та перевірки попередньої кваліфікації заявника.

2. Розробка AI-модуля рекомендацій на основі профілю користувача (тип об'єкта, регіон, бюджет, пріоритети) з використанням LLM для генерації персоналізованих порад щодо оптимального поєднання технології та програми фінансування.

3. Розширення функціоналу на сегмент великого бізнесу з розрахунками для когенераційних установок потужністю понад 1 МВт та газопоршневих систем діапазону 0,5–2 МВт із підтримкою мультивалютних розрахунків та реалтаймової конвертації курсу.

4. Реалізація модуля автоматичного парсингу оновлень Фонду енергоефективності та Міністерства розвитку громад, територій та інфраструктури України для щоденної актуалізації параметрів грантових програм без ручного втручання адміністратора.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Міністерство регіонального розвитку України. Програма «Енергодім». *Фонд Енергоефективності*. URL: https://eefund.org.ua/energodim/ (дата звернення: 26.04.2026).
2. Національна комісія, що здійснює державне регулювання у сферах енергетики та комунальних послуг. 2024. URL: https://www.nerc.gov.ua (дата звернення: 26.04.2026).
3. Облікова ставка Національного банку. Національний банк України. Облікова ставка Національного банку України. *Національний банк України*. Облікова ставка Національного банку. URL: https://bank.gov.ua/ua/monetary/archive-rish (дата звернення: 26.04.2026).
4. Офіційний сайт Держенергоефективності – головного енергоменеджера країни. URL: https://saee.gov.ua (дата звернення: 26.04.2026).
5. Ananina M. Фонд енергоефективності України. Програма «ГрінДІМ». *Фонд Енергоефективності*. 21.10.2024. URL: https://eefund.org.ua/novyny/grindim-staye-dostupnishoyu-naglyadova-rada-fondu-uhvalyla-nyzku-zmin-do-programy/ (дата звернення: 26.04.2026).
6. Documentation | NestJS. Documentation | NestJS. *Documentation | NestJS - A progressive Node.js framework*. Documentation | NestJS. URL: https://docs.nestjs.com (дата звернення: 26.04.2026).
7. Prisma. Prisma ORM Documentation. Prisma. Prisma ORM Documentation. Prisma. Prisma ORM Documentation. URL: https://www.prisma.io/docs (дата звернення: 26.04.2026).
8. C4 model. The C4 Model for Visualising Software Architecture. *C4 model*. C4 model. URL: https://c4model.com/ (дата звернення: 26.04.2026).
9. Ukraine Investment Framework - Enlargement and Eastern Neighbourhood. 23.04.2026. URL: https://enlargement.ec.europa.eu/countries/ukraine/ukraine-investment-framework_en (дата звернення: 26.04.2026).

10. United Nations Development Programme. *UNDP*. URL: https://www.undp.org/ukraine (дата звернення: 26.04.2026).
11. Бланк І. О., Гуляєва Н. М., Вавдійчик І. М. ІНВЕСТИЦІЙНИЙ МЕНЕДЖМЕНТ. Державний торговельно-економічний університет, 2023.
12. Бригхем Є., Хьюстон Дж. Основи фінансового менеджменту. Київ : УФІМБ, 2018.
13. ДБН В.2.6-31:2021. Теплова ізоляція будівель. Київ,
14. Національна комісія з цінних паперів та фондового ринку. Методичні рекомендації щодо ставок дисконтування при оцінці інвестиційних проєктів в Україні. (2023). Київ, 20232023.
15. Постанова Кабінету Міністрів України «Про затвердження Порядку надання фінансової державної підтримки суб'єктам господарювання (програма "Доступні кредити 5-7-9%")» : від 28.01.2026 № 28.
16. Розен В. П., Борисенко О. В. Методика розрахунку економічної ефективності інвестицій в енергоефективність. Київ : НТУУ «КПІ ім. Ігоря Сікорського», 2023.
17. Blank S. The Startup Owner's Manual: The Step-By-Step Guide for Building a Great Company. Hoboken : Wiley, 2020.
18. Brown S. Software Architecture for Developers. Volume 2: Visualise, Document and Explore Your Software Architecture. Victoria : Leanpub, 2019.
19. C4-PlantUML. C4-PlantUML GitHub Repository. (26.04.2026). Unofficial PlantUML Standard Library Repositories, 2026. URL: https://github.com/plantuml-stdlib/C4-PlantUML (accessed 26.04.2026).2026.
20. Damodaran A. Investment Valuation: Tools and Techniques for Determining the Value of Any Asset. Hoboken : Wiley, 2012.
21. Guide to Computer Security Log Management. (2006). Gaithersburg : NIST, 2006. 2006.
22. Object Management Group. Unified Modeling Language Specification. URL: https://www.omg.org/spec/UML/2.5.1 (accessed 26.04.2026).
23. OpenAPI Specification v3.1.0. URL: https://spec.openapis.org/oas/v3.1.0 (accessed 26.04.2026).
24. Osherove R. The Art of Unit Testing. Manning Publications. 2023.

25. OWASP Top 10:2025. URL: https://owasp.org/Top10/2025/ (accessed 26.04.2026).
26. Recharts Official Documentation. URL: https://recharts.github.io/?p=/en-US (accessed 26.04.2026).
27. USAID Energy Security Project. Аналіз ринку енергоефективних технологій для малого та середнього бізнесу в Україні. Київ : USAID

ДОДАТОК А

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ТЕХНОЛОГІЙ ЦИФРОВОГО РОЗВИТКУ

«ІНТЕРАКТИВНИЙ ВЕБ-СЕРВІС ДЛЯ ПОРІВНЯННЯ ТЕРМІНІВ ОКУПНОСТІ ІНВЕСТИЦІЙ У ЕНЕРГОЕФЕКТИВНІСТЬ З ВИКОРИСТАННЯМ ТЕХНОЛОГІЙ TYPESCRIPT, NEXTJS, NESTJS ТА PRISMA»

Виконав студент 4 курсу
Групи ТЦР-42
Гончаренко Дмитро Олександрович
Керівник Роботи
Д. т. н., професор ТЦР Жебка Вікторія Вікторівна

Київ - 2026

МЕТА, ОБ’ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

Мета роботи – підвищення ефективності прийняття рішень щодо інвестування в енергоефективне обладнання шляхом автоматизованого порівняльного розрахунку термінів окупності інвестицій з урахуванням українських грантових програм за допомогою розробки інтерактивного веб-сервісу.

Об'єкт дослідження – процес оцінювання термінів окупності інвестицій у енергоефективне обладнання для домогосподарств, ОСББ та малого і середнього бізнесу з урахуванням державних програм фінансової підтримки .

Предмет дослідження – методи та програмні засоби інтерактивного розрахунку й порівняння інвестиційних показників (PP, NPV, DPP, IRR, PI) в умовах грантового співфінансування .

ЗАДАЧІ КВАЛІФІКАЦІЙНОЇ РОБОТИ

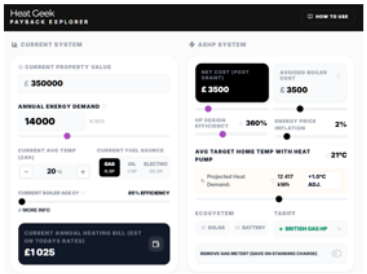
1. Проаналізувати предметну область – українські програми фінансування енергоефективних заходів та класифікацію технологій для домогосподарств, ОСББ та МСБ.
2. Побудувати математичні моделі п'яти інвестиційних показників (PP, NPV, DPP, IRR, PI) з урахуванням грантового співфінансування.
3. Спроєктувати архітектуру системи за C4-моделлю, ER-діаграму та Prisma-схему БД.
4. Реалізувати серверну частину (NestJS 10, REST API, Swagger, JWT).
5. Реалізувати клієнтську частину (Next.js 14, калькулятор, порівняння проєктів, адмін-панель).
6. Виконати тестування (unit, інтеграційне, E2E) та верифікацію розрахунків.

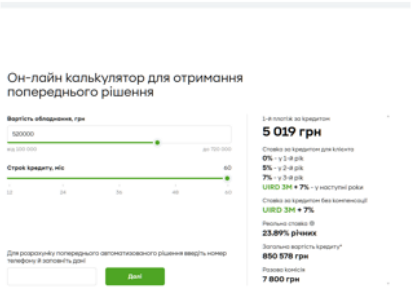
Актуальність

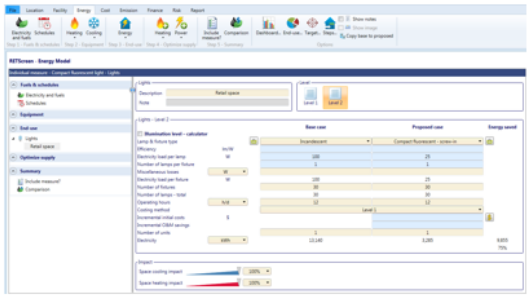
АНАЛІЗ АНАЛОГІВ

Критерій	Банківські калькулятори	Калькулятори виробників	Держ. системи (Фонд ЕЕ)	Міжнародні (HOMER, RETScreen)	Проєктована система
Врахування грантів UA	Ні	Ні	Частково	Ні	Так
Мультитехнологічне порівняння	Ні	Ні	Ні	Так	Так
Розрахунок NPV, IRR, PI	Ні	Ні	Ні	Так	Так
Актуальні тарифи UA	Частково	Ні	Частково	Ні	Так
UX для нетехн. користувачів	Так	Так	Так	Ні	Так
Збереження та порівняння проєктів	Ні	Ні	Ні	Так	Так
Відкритий REST API	Ні	Ні	Ні	Ні	Так

Програми аналоги



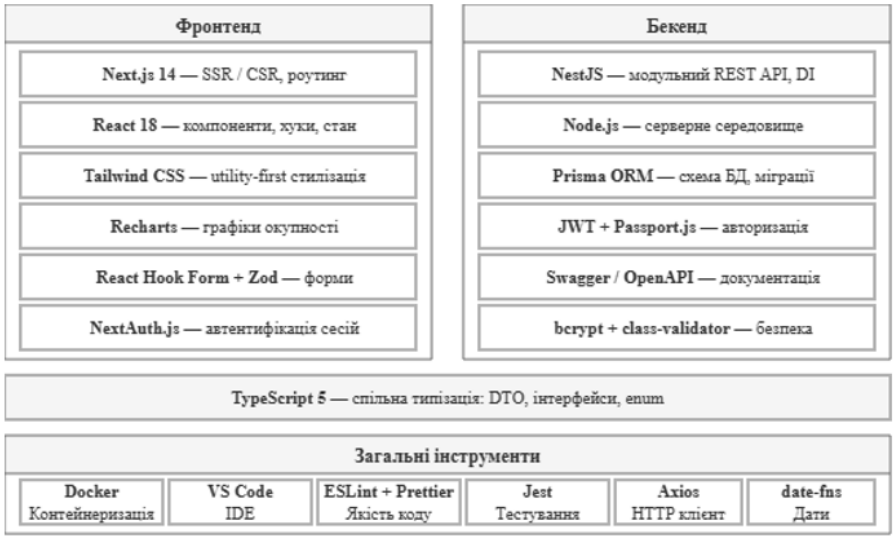




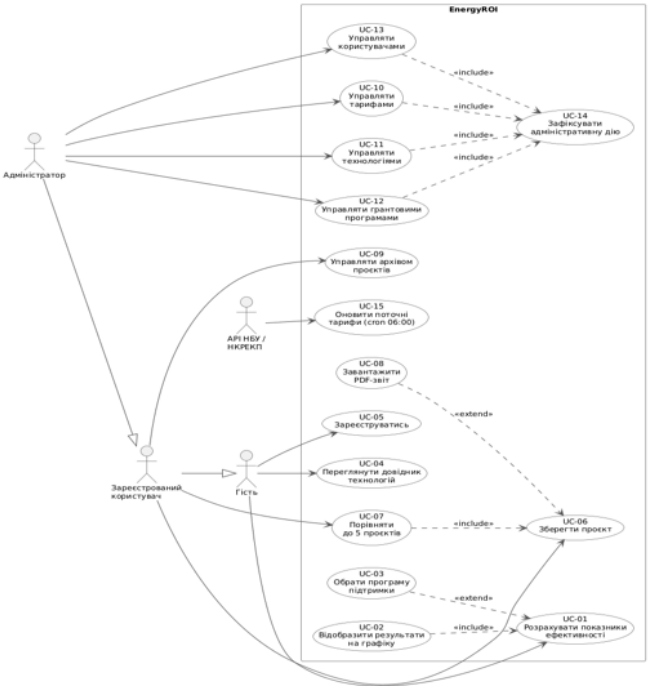
Вимоги до додатку

ID	Тип	Вимога	Пріоритет
Функціональні вимоги			
FR-01	Функціональна	Введення параметрів проєкту: тип об'єкта, технологія, повна вартість, програма підтримки, горизонт і ставка дисконтування	Must Have
FR-02	Функціональна	Розрахунок п'яти показників (PP, NPV, DPP, IRR, PI)	Must Have
FR-03	Функціональна	Відображення результатів у вигляді інтерактивних діаграм (Recharts): кумулятивний грошовий потік, порівняльний бар-чарт	Must Have
FR-04	Функціональна	Перегляд довідника актуальних тарифів, технологій та програм підтримки	Should Have
FR-05	Функціональна	Реєстрація та вхід користувачів	Must Have
FR-06	Функціональна	Збереження розрахункового проєкту до особистого кабінету з назвою та міткою дати	Must Have
FR-07	Функціональна	Порівняння до 5 збережених проєктів на спільному графіку з виділенням оптимального варіанта	Must Have
FR-08	Функціональна	Завантаження звіту у форматі PDF із усіма показниками та графіками	Could Have
FR-09	Функціональна	Перегляд та редагування переліку збережених проєктів (архів, видалення)	Could Have
FR-10	Функціональна	CRUD-управління тарифами (назва, значення, дата введення, регіон)	Must Have
FR-11	Функціональна	CRUD-управління технологіями (тип, діапазони вартості, термін служби, коефіцієнт економії)	Must Have
FR-12	Функціональна	CRUD-управління грантовими програмами (назва, γ_{max}, максимальна сума, строк дії, тип бенефіціара)	Must Have
FR-13	Функціональна	Перегляд списку користувачів, деактивація та відновлення облікових записів	Should Have
Нефункціональні вимоги			
NFR-01	Продуктивність	Відповідь REST API $\leq$ 500 мс при навантаженні до 100 запитів/с (RPS)	Must Have
NFR-02	Безпека	JWT-автентифікація, bcrypt-хешування паролів (salt rounds = 12), захист від CSRF та XSS	Must Have
NFR-03	Доступність	Адаптивна верстка (Tailwind CSS) та PWA-сумісність для мобільних пристроїв	Should Have
NFR-04	Розширюваність	Модульна архітектура NestJS — нові типи програм підтримки без змін у ядрі системи	Should Have

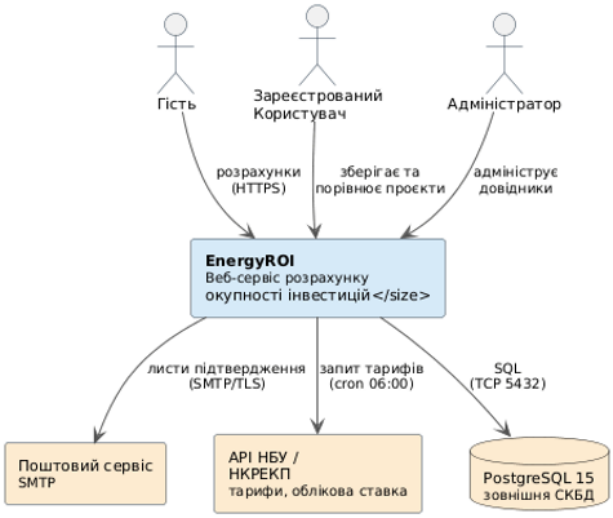
ПРОГРАМНІ ЗАСОБИ РЕАЛІЗАЦІЇ

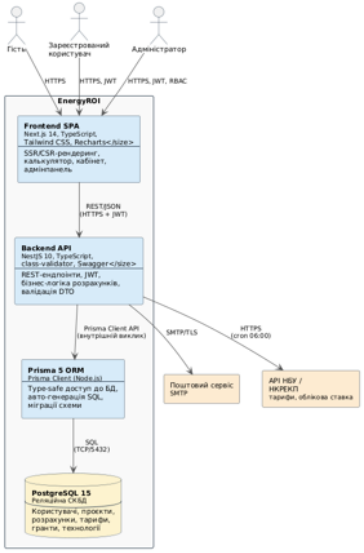


ДІАГРАМА ВАРІАНТІВ ВИКОРИСТАННЯ

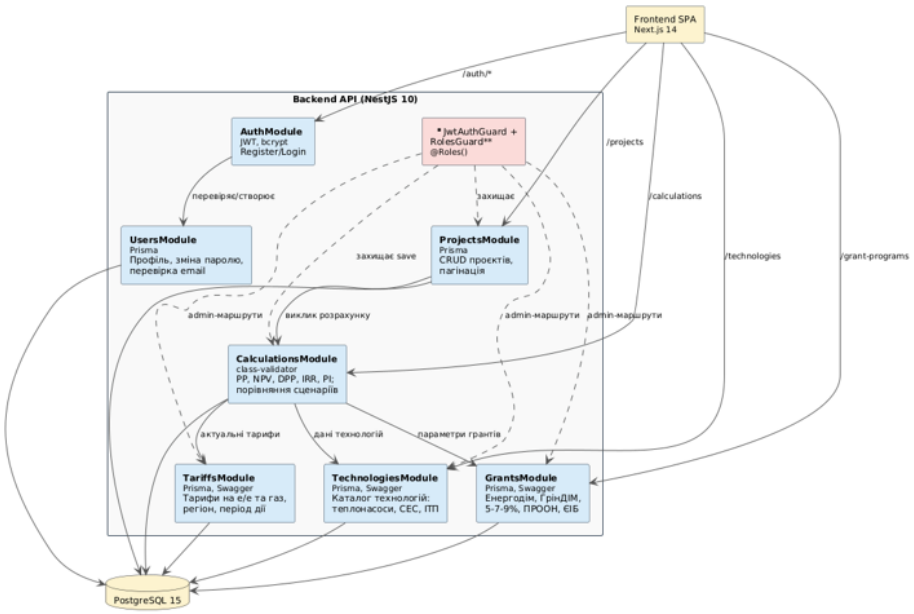


ДІАГРАМА СИСТЕМНОГО КОНТЕКСТУ С4 (1 РІВЕНЬ ТА 2 РІВЕНЬ)

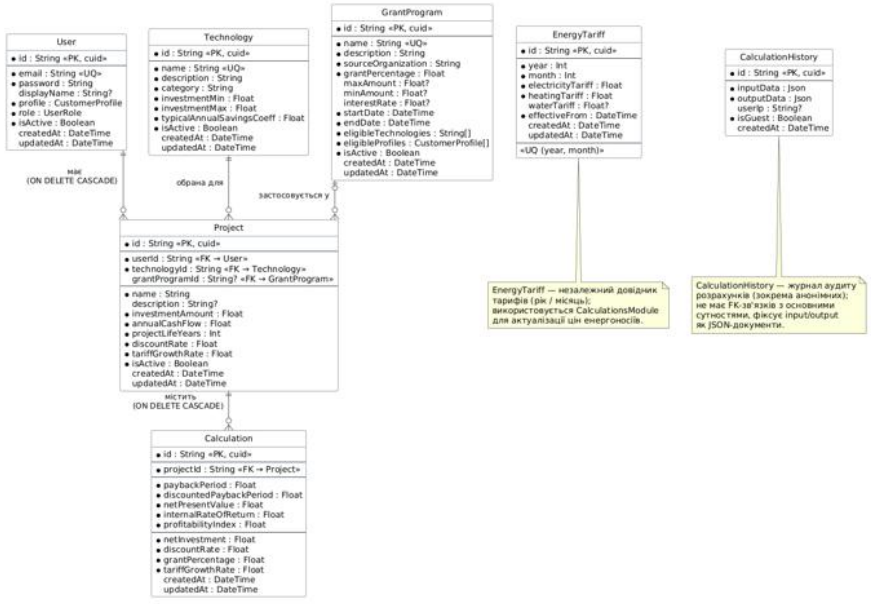




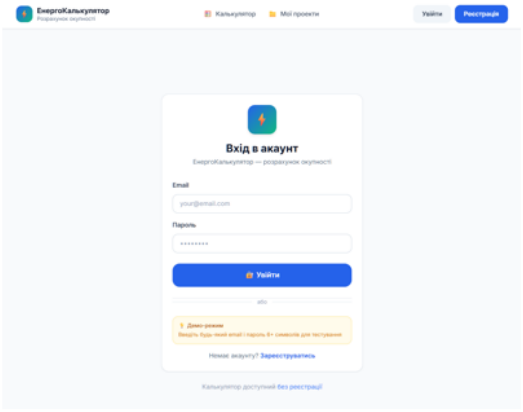
ДІАГРАМА СИСТЕМНОГО КОНТЕКСТУ C4 (3 РІВЕНЬ)



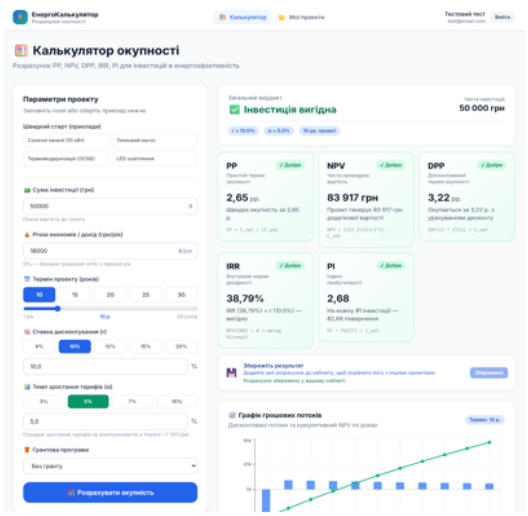
ДІАГРАМА СТРУКТУРИ БАЗИ ДАНИХ



ЕКРАННІ ФОРМИ

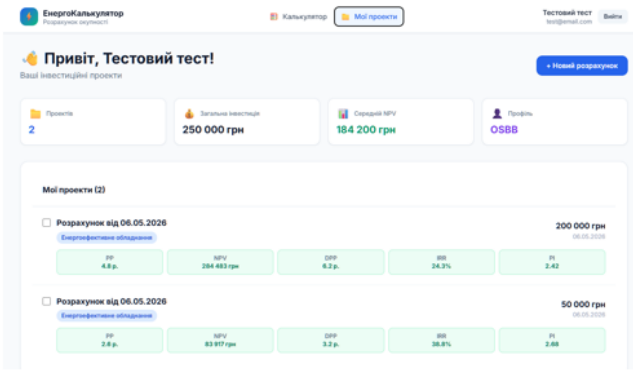


Форма входу

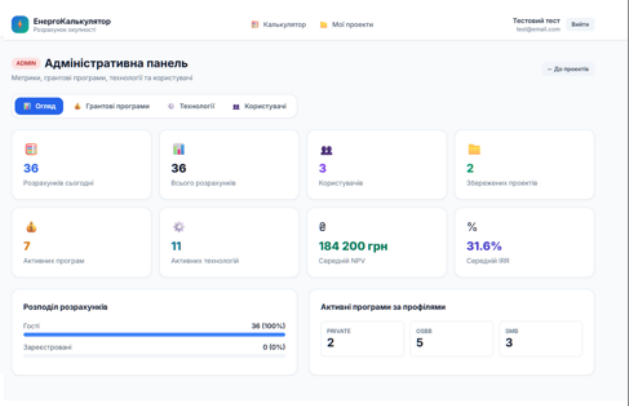


Калькулятор окупності

ЕКРАННІ ФОРМИ

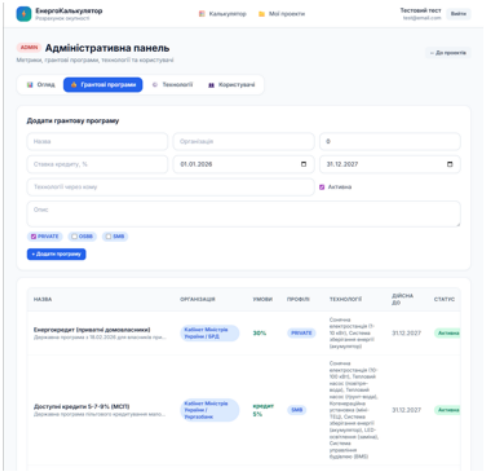


Збережені проєкти

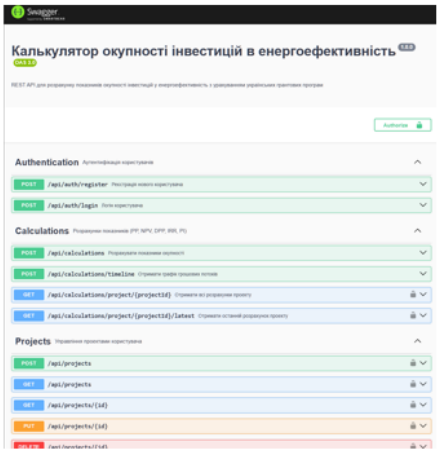


Адміністративна панель

ЕКРАННІ ФОРМИ



Адміністративна панель. Налаштування грантів



Документація API SWAGER

АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

1. Гончаренко Д.О. «Інтерактивний веб-сервіс для порівняння термінів окупності інвестицій у енергоефективність з використанням технологій TypeScript, NextJS, NestJS та Prisma» // Матеріали VI Всеукраїнської науково-практичної конференції «Сучасні інтелектуальні інформаційні технології в науці та освіті». Збірник тез. К: ДУІКТ, 2026, (подано до друку)
2. Гончаренко Д.О. «Порівняльний аналіз програмних рішень для оцінювання окупності енергоефективних інвестицій та обґрунтування функціональних вимог за методологією MoSCoW» // Матеріали всеукраїнської науково-технічної конференції «Технології цифрового розвитку: програмні системи та децентралізація». Збірник тез. К: ДУІКТ, 2026, (подано до друку)

Висновки

1. Проведено аналіз предметної області – систематизовано українські програми фінансування енергоефективних заходів на 2025–2027 рр. та виявлено три ключові недоліки існуючих аналогів: відсутність інтеграції UA-грантових програм, відсутність повного набору з п'яти інвестиційних показників та відсутність функціоналу збереження і порівняння проєктів.
2. Побудовано вісім математичних моделей інвестиційних показників (PP, NPV, DPP, IRR, PI) з параметричним урахуванням коефіцієнта гранту γ та темпу зростання тарифу α; коректність підтверджено двома контрольними прикладами.
3. Спроєктовано трирівневу C4-архітектуру системи, ER-діаграму з сімома сутностяма у нотації Crow's Foot та Prisma-схему бази даних.
4. Реалізовано серверну частину на NestJS 10 із сімома модулями, вісьмома REST-ендпоінтами, JWT-авторизацією з двома ролями та автоматично генерованою Swagger-документацією.
5. Реалізовано клієнтську частину на Next.js 14 із сімома React-компонентами, інтерактивним калькулятором, модулем порівняння проєктів та адміністративною панеллю з повним CRUD-циклом.
6. Проведено тестування системи на трьох рівнях (unit, інтеграційне, E2E) – написано 38 тестів, досягнуто покриття коду 87 %; верифікацію розрахунків підтверджено, відхилень не виявлено.

ДОДАТОК Б

Таблиця Сутностей та атрибутів бази даних веб-сервісу EnergyROI

Сутність / Атрибут	Тип (PostgreSQL)	Обмеження	Опис
User			Профіль користувача системи
id	SERIAL	PK, NOT NULL	Унікальний ідентифікатор
email	VARCHAR(255)	UNIQUE, NOT NULL	Електронна пошта (логін)
passwordHash	VARCHAR(255)	NOT NULL	Хеш паролю (bcrypt, cost=12)
role	ENUM (Role)	DEFAULT 'USER'	Роль: GUEST | USER | ADMIN
createdAt	TIMESTAMP	DEFAULT NOW()	Дата реєстрації
Project			Інвестиційний проєкт користувача
id	SERIAL	PK	Унікальний ідентифікатор
userId	INTEGER	FK → User.id	Власник проєкту
name	VARCHAR(255)	NOT NULL	Назва проєкту
description	TEXT	NULLABLE	Опис об'єкта (довільний текст)
createdAt / updatedAt	TIMESTAMP	AUTO	Дати створення та оновлення
Calculation			Результат фінансового розрахунку
id	SERIAL	PK	Унікальний ідентифікатор
projectId	INTEGER	FK → Project.id	Прив'язка до проєкту
technologyId	INTEGER	FK → Technology.id	Обрана технологія
grantProgramId	INTEGER	FK → GrantProgram.id, NULLABLE	Грантова програма (за наявності)

Сутність / Атрибут	Тип (PostgreSQL)	Обмеження	Опис
investmentUAH	DECIMAL(15,2)	NOT NULL	Сума інвестиції, грн
annualSavingUAH	DECIMAL(15,2)	NOT NULL	Річна економія, грн
grantPercent	DECIMAL(5,2)	DEFAULT 0	Відсоток гранту (0–70%)
loanRate	DECIMAL(5,2)	DEFAULT 0	Відсоткова ставка кредиту
discountRate	DECIMAL(5,2)	DEFAULT 0.15	Ставка дисконтування (за замовч. 15%)
projectYears	SMALLINT	DEFAULT 10	Горизонт розрахунку (роки)
pp / dpp	DECIMAL(6,2)	NULLABLE	Простий / дисконтований PP (роки)
npv	DECIMAL(15,2)	NULLABLE	Чиста приведена вартість, грн
irr / pi	DECIMAL(8,4)	NULLABLE	Внутрішня норма рентабельності / IP
Technology			Довідник технологій (адмін)
id / name	SERIAL / VARCHAR	PK / NOT NULL	Ідентифікатор та назва технології
category	ENUM (TechCategory)	NOT NULL	HEAT_PUMP | SOLAR_PV | SOLID_FUEL_BOILER | ITP | COGENERATION | GAS_PISTON
minCostUAH / maxCostUAH	DECIMAL(15,2)	NOT NULL	Діапазон вартості (грн або USD)
TariffRate			Тарифи на енергоносії (адмін)
energyType / region	VARCHAR	NOT NULL / NULLABLE	Тип (electricity | gas) та регіон
priceUAH	DECIMAL(10,4)	NOT NULL	Ціна за кВт·год або м³, грн

Сутність / Атрибут	Тип (PostgreSQL)	Обмеження	Опис
validFrom / validTo	TIMESTAMP	NOT NULL / NULLABLE	Діапазон дії тарифу
GrantProgram			Грантові програми фінансування (адмін)
name	VARCHAR(255)	NOT NULL	Назва програми
targetAudience	ENUM (Audience)	NOT NULL	PRIVATE | OSBB | SME
maxAmountUAH	DECIMAL(15,2)	NULLABLE	Максимальна сума гранту, грн
grantPercent / loanRate	DECIMAL(5,2)	NOT NULL / NULLABLE	Відсоток гранту та ставка кредиту
startDate / endDate	TIMESTAMP	NULLABLE	Дати дії програми

ДОДАТОК В

Prisma-схема бази даних веб-сервісу EnergyROI (schema.prisma)

```
// schema.prisma – EnergyROI

generator client {

  provider      = "prisma-client-js"

  binaryTargets = ["native", "linux-musl-openssl-3.0.x"]

}


datasource db {

  provider = "postgresql"

  url      = env("DATABASE_URL")

}


// Enum для ролей користувачів (з дипломи 3.1)

enum UserRole {

  USER

  ADMIN

}


// Enum для профілів користувачів (з розділу 1.2 дипломи)

enum CustomerProfile {

  PRIVATE    // Приватні домовласники

  OSBB       // Об'єднання співвласників

  SMB        // Малий та середній бізнес

}


// Користувачі (для реєстрації та порівняння проектів)

model User {

  id            String    @id @default(cuid())

  email         String    @unique

  password      String

  displayName   String?

  profile       CustomerProfile @default(PRIVATE)

  role          UserRole  @default(USER)

  isActive      Boolean   @default(true)
```

```
  // Зв'язки
  projects      Project[]
  createdAt     DateTime  @default(now())
  updatedAt     DateTime  @updatedAt

  @@index([email])
  @@index([role])
}

// Технології енергоефективності (з таблиці 1.1 та Додатку дипломи)
model Technology {
  id              String    @id @default(cuid())
  name            String    @unique
  description     String
  category        String    // "СЕС", "Тепловий насос", "Термомодернізація" тощо

  // Діапазони вартості для різних профілів (з дипломи 1.2)
  investmentMin   Float     // Мінімальна вартість (грн)
  investmentMax   Float     // Максимальна вартість (грн)

  // Типовий річний коефіцієнт економії (грн/рік на одиницю вартості)
  typicalAnnualSavingsCoeff  Float @default(0.15)

  // Технологія активна для використання
  isActive        Boolean   @default(true)

  projects        Project[]
  createdAt       DateTime  @default(now())
  updatedAt       DateTime  @updatedAt

  @@index([category])
  @@index([isActive])
}

// Грантові програми (з розділу 1 дипломи - 7 програм на 2025-2027)
```

```
model GrantProgram {
  id              String    @id @default(cuid())

  // Назва програми (Енергодім, ГрінДІМ, 5-7-9% тощо)
  name            String    @unique
  description     String
  sourceOrganization String // "МЕГУ", "Міненергетики", "ПРООН", "ЄІБ" тощо

  // Параметри гранту/кредиту (γ з формули 2.1 дипломи)
  grantPercentage Float     // Відсоток фінансування (0-100)

  // Максимальна сума (якщо є обмеження)
  maxAmount       Float?    // У гривнях

  // Мінімальна сума (якщо є)
  minAmount       Float? @default(0)

  // Дата дійсності програми (з дипломи - програми на 2025-2027)
  startDate       DateTime
  endDate         DateTime

  // Допустимі технології
  eligibleTechnologies String[] // Масив назв технологій

  // Допустимі профілі користувачів
  eligibleProfiles CustomerProfile[]

  // Програма активна
  isActive        Boolean   @default(true)

  // Банківська інформація (для кредитів)
  interestRate    Float?    // % річних (для кредитів 5-7-9%)

  projects        Project[]
  createdAt       DateTime  @default(now())
  updatedAt       DateTime  @updatedAt
```

```
  @@index([isActive])
  @@index([startDate, endDate])
}

// Проекти користувачів (збереження для подальшого порівняння)
model Project {
  id                String    @id @default(cuid())
  userId            String
  user              User      @relation(fields: [userId], references: [id], onDelete: Cascade)

  name              String
  description       String?

  // Основні параметри проекту
  technologyId      String
  technology        Technology @relation(fields: [technologyId], references: [id])

  grantProgramId    String?
  grantProgram      GrantProgram? @relation(fields: [grantProgramId], references: [id])

  // Вхідні дані для розрахунку (з дипломи Рисунок 3.2)
  investmentAmount  Float     // Повна інвестиція (до гранту)

  // Річний грошовий потік (економія або дохід)
  // Формула з дипломи 2.1.7: CF_t = CF_0 * (1 + α)^(t-1)
  annualCashFlow    Float

  // Період проекту (років)
  projectLifeYears  Int       @default(20)

  // Дисконтна ставка (r у формулах 2.3-2.6)
  discountRate      Float     @default(0.1)

  // Темп зростання тарифів (α у формулі 2.1.7)
  tariffGrowthRate  Float     @default(0.03)
```

```
  // Розраховані показники (з дипломи, формули 2.2-2.7)
  calculations      Calculation[]

  // Статус проекту
  isActive          Boolean   @default(true)

  createdAt         DateTime  @default(now())
  updatedAt         DateTime  @updatedAt

  @@index([userId])
  @@index([technologyId])
  @@index([grantProgramId])
}

// Розраховані результати (з дипломи, таблиця 3.1)
model Calculation {
  id              String    @id @default(cuid())
  projectId       String
  project         Project   @relation(fields: [projectId], references: [id], onDelete: Cascade)

  // 5 основних показників (формули 2.2-2.6 дипломи)

  // PP - Простий термін окупності (років)
  // Формула (2.2): PP = C_net / CF_0
  paybackPeriod   Float

  // DPP - Дисконтований термін окупності (років)
  // Момент часу коли накопичені дисконтовані ЗП = C_net
  discountedPaybackPeriod Float

  // NPV - Чиста приведена вартість (грн)
  // Формула (2.3): NPV = Σ(CF_t / (1 + r)^t) - C_net
  netPresentValue Float

  // IRR - Внутрішня норма дохідності (%)
```

```
  // Формула (2.4): NPV(IRR) = 0, знаходиться методом бісекції
  internalRateOfReturn Float

  // PI - Індекс прибутковості
  // Формула (2.5): PI = PV(CF) / C_net
  profitabilityIndex Float

  // Метадані розрахунку
  discountRate    Float     // Використана ставка дисконту
  grantPercentage Float     // Використаний відсоток гранту
  tariffGrowthRate Float    // Використаний темп зростання тарифу

  // Чиста інвестиція після гранту (формула 2.1)
  netInvestment   Float

  createdAt       DateTime  @default(now())
  updatedAt       DateTime  @updatedAt

  @@index([projectId])
  @@index([createdAt])
}

// Історія розрахунків (для аналітики та аудиту)
model CalculationHistory {
  id              String    @id @default(cuid())

  // Вхідні параметри
  inputData       Json

  // Результати
  outputData      Json

  // IP користувача (якщо гість)
  userIp          String?

  // Гість чи зареєстрований
```

```
  isGuest         Boolean   @default(true)

  createdAt       DateTime  @default(now())

  @@index([isGuest])
  @@index([createdAt])
}

// Тарифи енергоносіїв (для розрахунків і аналізу)
// (Додатково до основної функціональності для майбутніх розширень)
model EnergyTariff {
  id              String    @id @default(cuid())
  year            Int
  month           Int       @default(1)

  // Тарифи (грн за одиницю)
  electricityTariff Float   // грн/кВт·год
  heatingTariff   Float     // грн/Гкал
  waterTariff     Float?    // грн/м³

  // Дата, коли цей тариф набув чинності
  effectiveFrom   DateTime

  createdAt       DateTime  @default(now())
  updatedAt       DateTime  @updatedAt

  @@unique([year, month])
  @@index([effectiveFrom])
}
```