НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ
Факультет інформаційних технологій

ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ
Завідувач кафедри
інформаційних систем і технологій
_____ Швиденко М. З.
(підпис) (ПІБ)
“___” _____ 20__р.

БАКАЛАВРСЬКА КВАЛІФІКАЦІЙНА РОБОТА
на тему

«Розробка вебзастосунку для формування комерційної пропозиції сонячної електростанції»

Спеціальність 122 – «Комп’ютерні науки»

Грант освітньої програми
д.е.н., професор _____ Руденський Р. А.
(науковий ступінь та вчене звання) (підпис) (ПІБ)

Керівник бакалаврської кваліфікаційної роботи
к.ек.н., доцент _____ Стариченко Є.М
(науковий ступінь та вчене звання) (підпис) (ПІБ)

Виконав _____ Глущенко В.О
(підпис) (ПІБ студента)

КИЇВ-2025

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ
Факультет інформаційних технологій

ЗАТВЕРДЖУЮ
Завідувач кафедри
інформаційних систем і технологій
_____к.ек. н, доцент __ _____ Швиденко М. З.
(науковий ступінь, вчене звання) (підпис) (ПІБ)
“___” _____ 20__р.

ЗАВДАННЯ
на виконання бакалаврської кваліфікаційної роботи студенту
Глущенко Валерія Олександрівна

Спеціальність 122 – «Комп’ютерні науки»

1. Тема бакалаврської кваліфікаційної роботи «Розробка вебзастосунку для формування комерційної пропозиції сонячної електростанції», затверджена наказом ректора НУБіП України від “18” березня 2025 р. № 394 “С”.

2. Термін подання завершеної роботи на кафедру __02.06.2025 р._
(рік, місяць, число)

3. Вихідні данні до бакалаврської кваліфікаційної роботи: створення системи, яка допомагатиме керувати медичними записами та документувати результати медичних обстежень пацієнтів за допомогою мовних технологій.

4. Перелік питань, які потрібно розробити:
1. Аналіз предметної області
2. Моделювання предметної області
3. Проєктування та реалізація програмного застосунку
4. Впровадження, експлуатація та оцінка якості системи

Дата видачі завдання “18” березня 2025 р.

Керівник бакалаврської кваліфікаційної роботи _____Стариченко Є.М__
(підпис) (прізвище та ініціали)

Завдання прийняв до виконання _____ Глущенко В.О.
(підпис) (прізвище та ініціали студента)

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

СЕС — Сонячна електростанція

КПЕ — Коефіцієнт продуктивності енергії

ККД — Коефіцієнт корисної дії

CO_2 — Вуглекислий газ (діоксид вуглецю)

NPV (Net Present Value) — Чиста приведена вартість

IRR (Internal Rate of Return) — Внутрішня норма прибутковості

LCOE (Levelized Cost of Energy) — Рівень приведеної собівартості електроенергії

CAPEX (Capital Expenditures) — Капітальні витрати

OPEX (Operational Expenditures) — Експлуатаційні витрати

Вт — Ват

кВт — Кіловат

кВт·год — Кіловат-година

€/кВт — Євро за кіловат

UI — Користувацький інтерфейс (User Interface)

UX — Досвід користувача (User Experience)

CRUD — Створення, читання, оновлення, видалення (Create, Read, Update, Delete)

PDF — Формат електронного документа Portable Document Format

PHP — Hypertext Preprocessor (мова програмування)

ORM — Об’єктно-реляційне відображення (Object-Relational Mapping)

LCA (Life Cycle Assessment) — Оцінка життєвого циклу

UPS — Джерело безперебійного живлення (Uninterruptible Power Supply)

ВСТУП

У сучасних умовах розвитку енергетичного сектору особливу актуальність набуває впровадження відновлюваних джерел енергії, зокрема сонячної енергетики, яка демонструє стабільне зростання як у промисловому, так і в приватному секторі. Розбудова малої генерації та децентралізованих систем споживання електроенергії створює потребу в оперативному та якісному інструменті для формування техніко-економічних обґрунтувань і комерційних пропозицій. Однак у більшості компаній ці процеси досі реалізуються вручну, що призводить до затримок, неточностей та обмежує гнучкість взаємодії з потенційними клієнтами. Саме тому створення вебзастосунку для автоматизованого формування комерційної пропозиції сонячної електростанції (СЕС) є актуальним і практично значущим завданням.

Метою даної роботи є розробка функціонального вебзастосунку, який дозволяє формувати персоналізовану пропозицію для клієнта на основі вхідних параметрів, таких як бажана потужність, бюджет, тариф на електроенергію та інші характеристики. Застосунок повинен враховувати обмеження бюджету, пропонувати найбільш відповідне обладнання та розраховувати ключові техніко-економічні показники: очікувану генерацію, економію, термін окупності та зменшення викидів CO_2. Реалізація такого інструменту дозволяє покращити якість взаємодії з клієнтами, підвищити точність інженерних розрахунків і скоротити час підготовки пропозицій.

Для досягнення поставленої мети використано низку сучасних технологій і методів розробки. Основною мовою програмування обрано PHP, зокрема в поєднанні з шаблонізатором Blade, бібліотеками Dompdf для генерації PDF-документів та Eloquent ORM для роботи з базою даних. Вебінтерфейс побудований з використанням Bootstrap 5, що забезпечує адаптивність і зручність користування. Логіка формування пропозицій реалізована у вигляді сервісного класу, який враховує доступне обладнання, обчислює економічні

показники та автоматично формує звіт для клієнта.

Структура дипломної роботи відображає послідовність розробки, тестування та аналізу системи. Робота складається зі 108 сторінок, включає 43 використані джерела літератури, додатки з прикладами коду, знімками екрана, схемами архітектури та інтерфейсу. Зміст структуровано таким чином:

– Розділ 1 — теоретичне обґрунтування теми, аналіз технічних та економічних аспектів сонячної енергетики, огляд вимог до системи;
– Розділ 2 — безпосередній опис проєктування архітектури, вибору технологій, реалізації алгоритмів та інтерфейсу;
– Розділ 3 — тестування ключових функцій, оцінка надійності та продуктивності системи;
– Висновки — узагальнення результатів, оцінка досягнення мети та перспектив подальшого розвитку.

Об'єктом дослідження є процес формування техніко-економічної пропозиції СЕС, а предметом — програмна реалізація відповідного вебзастосунку з урахуванням інженерної точності, зручності користування та відповідності вимогам ринку.

РОЗДІЛ 1. ТЕОРЕТИКО-АНАЛІТИЧНІ ЗАСАДИ ТА ПОСТАНОВКА ЗАДАЧІ

1.1. Стан і тенденції розвитку ринку сонячної енергетики

Сонячна енергетика є однією з найдинамічніше зростаючих галузей відновлюваної енергетики у світі. За даними Міжнародного агентства з відновлюваних джерел енергії (IRENA), встановлена потужність сонячних електростанцій (СЕС) у світі у 2023 році перевищила 1 200 ГВт, що свідчить про понад 25% щорічне зростання протягом останнього десятиліття. Таке стрімке збільшення обумовлено одночасним впливом кількох чинників: технологічного поступу, зниженням вартості обладнання та посиленням політики декарбонізації в країнах із високим рівнем промислового розвитку [1].

Лідерами у впровадженні сонячної генерації є Китай, Європейський Союз, США та Індія. Так, Китай щорічно встановлює понад 100 ГВт нових потужностей СЕС, суттєво випереджаючи інші регіони. У країнах ЄС розвиток фотогальванічної генерації стимулюється як субсидіями, так і регуляторними механізмами, зокрема обов’язковими квотами на частку ВДЕ у загальному енергобалансі. У США популярності набувають побутові сонячні установки у поєднанні з акумуляторними системами, зокрема на базі технологій Tesla Powerwall [2].

Окремої уваги заслуговує тенденція до поступового зниження вартості виробництва електроенергії з сонця. Згідно з аналітикою BloombergNEF, середній LCOE (Levelized Cost of Electricity) для СЕС у 2023 році становив менш ніж 0,05 $/кВт·год для великих станцій, що є нижчим за аналогічний показник для традиційних вугільних і газових електростанцій у багатьох країнах. За останнє десятиліття LCOE сонячної генерації знизився більш ніж на 80% [2], що зумовлено масштабуванням виробництва сонячних панелей, підвищенням їхньої ефективності, здешевленням інверторів і розвитком логістики.

Крім того, спостерігається зростання зацікавленості до інтелектуальних

систем керування СЕС, інтеграції систем накопичення енергії (BESS) і впровадження цифрових платформ для проєктування, моніторингу та комерційного обґрунтування інвестицій у ВДЕ. Застосування таких рішень дозволяє покращити планування та підвищити прозорість у процесі вибору конфігурації сонячної електростанції відповідно до індивідуальних потреб замовника.

Світові тенденції розвитку сонячної енергетики свідчать не лише про технологічну та економічну доцільність переходу до ВДЕ, а й про зростання ролі інформаційних технологій у цьому процесі.

В Україні сонячна енергетика відіграє ключову роль у процесі переходу до сталої та децентралізованої енергетичної системи. За даними Держенергоефективності, станом на кінець 2023 року загальна встановлена потужність СЕС сягнула понад 7 ГВт, з яких понад 30% припадає на приватні домогосподарства [3]. Починаючи з 2015 року, ринок характеризувався стрімким розвитком, зумовленим, зокрема, впровадженням привабливого "зеленого тарифу", який гарантував фіксовану викупну ціну на електроенергію, вироблену з відновлюваних джерел.

Однак у зв’язку з переглядом політики підтримки, починаючи з 2020 року, в Україні спостерігається поступовий перехід до системи Net Billing, що стимулює ефективніше споживання виробленої енергії на місці та впровадження систем накопичення. Така модель змінює підхід до проектування сонячних електростанцій, роблячи акцент на економічному обґрунтуванні інвестицій, а не лише на кількості виробленої електроенергії. Це, в свою чергу, формує потребу в інструментах для точного техніко-економічного аналізу ще на етапі формування комерційної пропозиції.

Повномасштабна війна з 2022 року істотно вплинула на енергетичну безпеку країни [4]. Пошкодження об’єктів централізованої генерації та ліній електропередач стимулювали зростання попиту на автономні, резервні та децентралізовані системи енергозабезпечення. У цих умовах приватні СЕС, зокрема ті, що поєднуються з акумуляторними батареями, стали ефективним

засобом забезпечення критичної інфраструктури та побутового споживання.

Також слід зазначити зростання інтересу з боку малого та середнього бізнесу до впровадження СЕС як елементу енергетичної незалежності та інструменту оптимізації витрат. Водночас, доступ до якісної інженерної підтримки, прогнозів окупності та інтегрованих рішень усе ще залишається обмеженим, що підкреслює актуальність цифрових сервісів для формування персоналізованих пропозицій [5].

Розвиток ринку СЕС в Україні супроводжується зміною моделей підтримки, адаптацією до кризових умов та підвищенням ролі інформаційних систем у процесі ухвалення рішень споживачами.

Попри стрімке зростання ринку сонячної енергетики, галузь стикається з низкою системних викликів, вирішення яких є необхідною умовою для забезпечення сталого розвитку. Насамперед, це обмеження інтеграції сонячної генерації до існуючих енергосистем через її варіабельність та залежність від погодних умов. У багатьох країнах це призводить до перевантаження мереж у години пікової генерації та потребує впровадження нових рішень у сфері диспетчеризації, балансування попиту та застосування накопичувачів енергії.

Одним із ключових технологічних трендів є розвиток систем накопичення енергії (BESS), що дозволяє згладжувати добові коливання генерації та підвищувати рівень самоспоживання електроенергії. Сучасні проєкти СЕС дедалі частіше проєктуються як гібридні комплекси, що включають сонячні модулі, акумулятори, інвертори з функцією автономної роботи, а також системи керування енергією (EMS). Це змінює підхід до формування комерційної пропозиції — від простої оцінки потужності та вартості до комплексного аналізу конфігурації, термінів окупності, LCOE та інших фінансових показників [6].

У контексті цифровізації енергетики зростає значення інформаційно-аналітичних платформ, що дозволяють автоматизувати проєктування, оптимізацію та презентацію комерційних пропозицій для замовників. Особливої актуальності це набуває у сфері малого бізнесу, інженерного консалтингу та дилерських мереж, де ефективне формування комерційної пропозиції значною

мірою впливає на рішення клієнта.

Крім технічних та економічних аспектів, важливу роль відіграють регуляторні фактори. У багатьох країнах світу, зокрема в Україні, нормативна база щодо підключення до мережі, компенсації надлишкової генерації та стандартизації компонентів постійно змінюється. Це створює потребу в системах, що можуть швидко адаптуватися до нових вимог — як з технічного, так і з аналітичного боку [7].

Майбутній розвиток сонячної енергетики буде визначатися здатністю галузі впроваджувати інновації, інтегрувати цифрові інструменти у процес прийняття рішень, а також забезпечувати економічну доцільність проєктів з урахуванням як локальних особливостей, так і глобальних викликів.

1.2. Технічні компоненти сучасної сонячної електростанції (СЕС)

Сонячні фотомодулі (сонячні панелі) є основним елементом сонячної електростанції, який перетворює енергію сонячного випромінювання у постійний електричний струм. На сьогодні найпоширенішими типами фотомодулів є монокристалічні, полікристалічні та тонкоплівкові.

Монокристалічні модулі, виготовлені на основі кремнієвих монокристалів, мають найвищий коефіцієнт перетворення енергії (до 22%) та найкращі показники стабільності при експлуатації. Полікристалічні модулі є дешевшими, проте мають дещо нижчу ефективність (приблизно 16–18%) та більшу площу на одиницю потужності. Тонкоплівкові модулі (CdTe, CIGS) використовуються рідше у комерційних СЕС через нижчу питому потужність та підвищені вимоги до площі розміщення [8].

Ключовими параметрами, що характеризують фотомодуль у контексті проєктування СЕС, є:

- Pmpp (Maximum Power Point, Вт) — номінальна пікова потужність при стандартних умовах (STC),

- Vmp (Voltage at Maximum Power Point, В) — робоча напруга при Pmpp,
- Isc, Voc — струм короткого замикання та напруга холостого ходу (використовуються для перевірки відповідності інвертору),
- Температурний коефіцієнт — впливає на продуктивність при підвищенні температури.

У таблиці 1.1 наведено приклади поширених моделей фотомодулів, які часто використовуються у комерційних і побутових СЕС.

Таблиця 1.1

Поширені моделі фотомодулів СЕС

Модель фотомодуля	**Тип**	**Pmpp, Вт**	**Vmp, В**	**Ефективність, %**
JA Solar JAM72S30-535/MR	Монокристалічний	535	41.5	21.3
Longi LR5-72HBD-540M	Монокристалічний	540	41.0	21.1
Canadian Solar CS6R-400MS	Монокристалічний	400	34.0	20.2
First Solar Series 6 FS-6440	Тонкоплівковий	440	50.0	18.1

Під час формування комерційної пропозиції вебзастосунок використовує параметри Pmpp та Vmp для розрахунку кількості модулів, потрібної для досягнення цільової встановленої потужності (у кВт), а також для попередньої перевірки відповідності інвертору. Ці дані автоматизовано витягуються з довідника фотомодулів у системі.

З огляду на значне різноманіття на ринку, актуальність довідника модулів в інформаційній системі є критично важливою. У запропонованому вебзастосунку реалізовано можливість гнучкого адміністрування бази фотомодулів для швидкої адаптації до реального комерційного каталогу постачальника.

Інвертор є ключовим елементом сонячної електростанції, який забезпечує перетворення постійного струму (DC), що виробляється фотомодулями, на змінний струм (AC), придатний для живлення електроприладів або передачі в електромережу. Вибір інвертора визначає не лише ефективність роботи всієї системи, а й її здатність взаємодіяти з мережею, витримувати навантаження та забезпечувати безперебійну роботу у визначених умовах [9].

Залежно від типу системи, інвертори поділяються на:

- Мережеві інвертори — працюють виключно при наявності напруги в мережі, є основними для більшості СЕС у комерційному секторі;
- Автономні інвертори — забезпечують живлення від батарей у разі відсутності мережі;
- Гібридні інвертори — поєднують можливості мережевих і автономних інверторів, підтримують системи накопичення енергії (BESS).

Основними технічними характеристиками інвертора, що враховуються при автоматизованому формуванні конфігурації СЕС, є:

- Максимальна DC-потужність (Pdc max, кВт) — максимально допустиме навантаження від панелей;
- Номінальна AC-потужність (Pac, кВт) — потужність, яку інвертор може стабільно видавати в мережу;
- Кількість MPPT-трекерів — визначає можливість підключення панелей з різною орієнтацією;
- Ефективність перетворення, зазвичай 97–99%;
- Можливість паралельної роботи (scalability) — важливо для конфігурацій з кількома інверторами.

У таблиці 1.2 подано приклади інверторів, які використовуються у реальних комерційних пропозиціях для систем середньої та великої потужності.

Таблиця 1.2

Інвертори для систем середньої та великої потужності

Модель інвертора	**Тип**	**Pdc max, кВт**	**Pac, кВт**	**ККД, %**
Huawei SUN2000-20KTL-M2	Мережевий	30	20	98.6
Sungrow SG110CX	Мережевий	150	110	98.5
SMA Sunny HighPower PEAK3 150	Мережевий	180	150	98.6
Solis-Hybrid 10K-5G	Гібридний	15	10	97.8

У створеному вебзастосунку параметр Pdc max є критичним при відборі інвертора. Якщо цільова потужність станції перевищує можливості одного

інвертора, система автоматично підбирає кілька однакових пристроїв для покриття DC-навантаження. Це забезпечує масштабованість рішень і точність розрахунку інвестиційних показників.

Під час створення комерційної пропозиції система використовує ці параметри для перевірки відповідності між обраними панелями та інвертором, що дозволяє запобігти технічним помилкам ще на етапі планування.

Крім фотомодулів та інверторів, кожна сонячна електростанція включає низку додаткових компонентів, що забезпечують механічну, електричну та інформаційну інтеграцію усієї системи. Сукупність цих елементів називається Balance of System (BOS), і саме вони часто становлять суттєву частину вартості проєкту — до 25–30% загального бюджету [10].

До основних складових BOS належать:

- Монтажні конструкції — алюмінієві або сталеві кронштейни, системи трекінгу, опори для дахових або наземних установок. Вони мають забезпечити надійне кріплення панелей під оптимальним кутом та з урахуванням вітрового навантаження.
- Кабельна інфраструктура — DC-кабелі між фотомодулями, з’єднання з інвертором, AC-кабелі до мережі, розподільчі щити, автоматика захисту та заземлення.
- Захисне обладнання — запобіжники, автоматичні вимикачі, пристрої захисту від імпульсних перенапруг (SPD), реле контролю напруги тощо.
- Системи моніторингу — контролери, лічильники, мережеві шлюзи для передачі даних у хмарні сервіси або локальні SCADA-системи.

У сучасних СЕС дедалі частіше інтегруються системи накопичення енергії (акумулятори). Хоча вони не є обов’язковим компонентом для класичних мережевих станцій, їх наявність дозволяє забезпечити безперебійність живлення, оптимізувати самоспоживання та брати участь у ринках балансування потужності. Вартість BESS часто становить окрему статтю бюджету і може бути врахована опційно.

У таблиці 1.3 наведено орієнтовну структуру витрат на компоненти СЕС

середньої потужності (до 100 кВт), яка використовується у запропонованому вебзастосунку для формування прогнозної вартості проєкту (CAPEX).

Таблиця 1.3

Структура витрат на компоненти СЕС середньої потужності (до 100 кВт)

Компонент	**Частка у CAPEX, %**	**Коментар**
Фотомодулі	35–45 %	Залежить від типу та потужності
Інвертор(и)	15–25 %	Вища вартість у гібридних системах
Монтажні конструкції	10–15 %	Складність монтажу, матеріали
Кабелі, автоматика, BOS	10–15 %	Включає захист, вимикачі, SPD тощо
Роботи, проєктування, ПНР	10–20 %	Витрати на монтаж, дозвільні документи
Акумулятори (опційно)	15–30 %	Залежно від обсягу і типу BESS

У реалізованому веб застосунку при оцінці бюджету BOS враховується як пропорційна ставка до встановленої потужності (у середньому 150 €/кВт), що забезпечує узгодженість розрахунків із реальними комерційними оцінками. Такий підхід дозволяє швидко отримати прогнозну оцінку інвестицій без детального інженерного проєктування на ранньому етапі прийняття рішення [11].

1.3. Ключові показники ефективності СЕС та нормативно-правові вимоги

Розробка та впровадження сонячної електростанції є інвестиційним проєктом, що потребує оцінки не лише технічної доцільності, а й фінансової ефективності. У міжнародній та українській практиці для обґрунтування економічної доцільності встановлення СЕС використовують низку інтегральних техніко-економічних показників, серед яких ключовими є:

- LCOE (Levelized Cost of Electricity, грн/кВт·год або €/кВт·год) — середня приведена вартість виробленої електроенергії за весь життєвий цикл системи. Вона враховує як капітальні, так і операційні витрати. Цей показник дозволяє об’єктивно порівнювати СЕС з іншими джерелами генерації. Менший LCOE свідчить про вищу ефективність інвестиції [12].

- NPV (Net Present Value, грн або €) — чиста приведена вартість проєкту. Визначається як сума дисконтованих грошових потоків за весь період експлуатації мінус капітальні витрати. NPV > 0 означає прибутковість проєкту при заданій ставці дисконту [13].
- IRR (Internal Rate of Return, %) — внутрішня норма прибутковості. Це така ставка дисконту, при якій NPV дорівнює нулю. Якщо IRR перевищує вартість капіталу, проєкт вважається доцільним для реалізації [14].
- Строк окупності (Payback Period, років) — кількість років, необхідна для повного повернення інвестованих коштів за рахунок доходів від генерації електроенергії. Хоча цей показник не враховує часову вартість грошей, він широко використовується для початкової оцінки проєкту, особливо в умовах високої невизначеності.

У таблиці 1.4 наведено стислу інтерпретацію ключових показників.

Таблиця 1.4

Інтерпретація ключових показників

Показник	**Одиниця виміру**	**Інтерпретація для інвестора**
LCOE	€/кВт·год	Середня вартість 1 кВт·год протягом життєвого циклу
NPV	€	Додана вартість проєкту в порівнянні з нульовою інвестіцією
IRR	%	Орієнтовна прибутковість проєкту
Payback Period	років	Час до повернення вкладених коштів

У контексті формування комерційної пропозиції, яка реалізована у створеному веб застосунку, ці показники є основою для оцінки ефективності кожної конфігурації, що генерується на основі введених користувачем даних. Вони автоматично обчислюються та виводяться у PDF-документі, що підвищує інформативність та професійний рівень пропозиції.

Техніко-економічні метрики слугують не лише для внутрішніх розрахунків, а й виконують комунікаційну функцію — даючи замовнику можливість об’єктивно порівнювати варіанти та ухвалювати рішення на основі кількісних аргументів.

Розрахунок економічних показників ефективності СЕС базується на загальноприйнятих у фінансовій інженерії підходах до аналізу інвестицій. У контексті сонячної електростанції модель враховує:

- початкові капітальні вкладення (CAPEX) — вартість обладнання (модулів, інверторів, монтажних конструкцій), послуг з проєктування та встановлення;
- щорічні операційні витрати (OPEX) — планове технічне обслуговування, моніторинг, амортизація;
- прогнозовану генерацію електроенергії — залежно від потужності СЕС та рівня інсоляції в регіоні;
- ставку дисконтування — що відображає альтернативну вартість капіталу або ризик інвестування.

Основою для оцінки обсягу щорічного виробництва електроенергії є значення середньорічної інсоляції (кВт·год/кВт$_р$ік) — тобто кількість енергії, яку може згенерувати 1 кВт встановленої потужності впродовж року за середніх кліматичних умов. У створеному вебзастосунку це значення приймається як 1 200 кВт·год/кВт$_р$ік, що відповідає середнім умовам для центральних регіонів України [15].

Розрахунок LCOE виконується за формулою:

$$LCOE = \frac{C\ APEX + \sum_{t=1}^{T} \frac{OPEX_t}{(1+r)^t}}{\sum_{t=1}^{T} \frac{E_t}{(1+r)^t}} \quad (1.1)$$

де:

- *CAPEX* — загальні початкові витрати,
- $OPEX_t$ — річні експлуатаційні витрати у році *t*,
- E_t — генерація електроенергії у році *t*,
- *r* — ставка дисконту,
- *T* — тривалість життєвого циклу проєкту (20 років у нашій моделі).

Розрахунок NPV та IRR ґрунтується на фінансовому моделюванні грошових потоків:

$$NPV = -C\ APEX + \sum_{t=1}^{T} \frac{(Revenue_t - OPEX_t)}{(1+r)^t} \quad (1.2)$$

$$IRR : NPV = 0 \rightarrow \text{знайти } r, \text{при якому приведена вартість} = 0$$

Прибуток розраховується виходячи з тарифу на продаж електроенергії. У моделі застосунку це параметр feed-in tariff**,** що задається через .env-файл або форму користувача, наприклад:

$$Revenue_t = E_t \times \text{тариф} \quad (1.3)$$

Операційні витрати у моделі приймаються на рівні 2 % від початкової вартості щороку, що узгоджується з середніми ринковими оцінками для СЕС середньої потужності.

Розрахунки реалізовані у вигляді програмного сервісу CalculationService, і виконуються автоматично на серверній частині PHP-застосунку. Це гарантує однорідність підходів для всіх користувачів та відтворюваність результатів незалежно від масштабів системи.

Юридичні та регуляторні аспекти є невід'ємною складовою проєктування та впровадження сонячних електростанцій в Україні. Дотримання чинного законодавства забезпечує можливість підключення до електричних мереж, отримання дозвільної документації, а також використання механізмів компенсації витрат, передбачених державою [16].

Нормативне поле формують:

- Закон України «Про альтернативні джерела енергії» – встановлює загальні принципи використання відновлюваних джерел та правові засади стимулювання інвестицій [17].
- Закон України «Про ринок електричної енергії» – регулює підключення об’єктів генерації до мереж, взаємодію з операторами систем розподілу, продаж електроенергії [18].
- Постанови НКРЕКП – визначають технічні вимоги до обладнання, ліцензійні умови, порядок приєднання, тарифоутворення.
- Стандарти ДСТУ та ТУ – задають вимоги до якості, сертифікації фотомодулів, інверторів, монтажних конструкцій.

Важливим історичним чинником розвитку СЕС в Україні був механізм "зеленого тарифу", який гарантував фіксовану викупну ціну на електроенергію,

вироблену з ВДЕ, із прив’язкою до євро. Наразі для нових об'єктів поступово впроваджується система Net Billing, що базується на заліку між спожитою та переданою електроенергією. Це зумовлює зміну підходу до техніко-економічного обґрунтування: основний акцент переноситься на самоспоживання, а не на генерацію у мережу [19].

Регулювання передбачає й низку технічних обмежень:

- відповідність інверторів міжнародним стандартам (EN/IEC),
- гранична встановлена потужність для "домогосподарських" СЕС (до 30 кВт),
- вимоги до обладнання для паралельної роботи з мережею (антиострівна автоматика),
- обов’язковість лічильників з реле керування в певних конфігураціях.

У розроблюваному веб застосунку ці вимоги враховані на рівні логіки вибору обладнання: зокрема, система автоматично обирає конфігурації, які не перевищують Pdc-потужність інверторів, перевіряє відповідність потужності станції, та може адаптувати параметри під різні моделі тарифоутворення. Це дозволяє не лише формувати комерційні пропозиції, а й уникати помилок, які можуть призвести до порушення нормативів при впровадженні СЕС.

Нормативно-правова база визначає межі технічної реалізації проєкту та має бути безпосередньо інтегрована у процес автоматизованого проєктування через відповідні програмні механізми, як це буде реалізовано у веб застосунку.

1.4. Потреби замовників щодо комерційних пропозицій СЕС

Сучасні споживачі сонячної енергетики, незалежно від того, йдеться про приватних осіб чи малий бізнес, дедалі частіше висувають до комерційної пропозиції не лише вимоги щодо вартості та термінів реалізації, а й очікування повної техніко-економічної аргументації. З розвитком ринку СЕС зміщується фокус уваги замовника з виключно технічних параметрів (кількість панелей, тип інвертора) до інтегральних показників ефективності — передусім таких, як

очікувана річна генерація, строк окупності, обсяг економії на рахунках за електроенергію та довгостроковий економічний ефект.

Це обумовлено як зростанням обізнаності споживачів, так і необхідністю зваженого прийняття інвестиційного рішення, яке передбачає витрати від кількох до десятків тисяч євро. Крім того, інтенсивна конкуренція серед постачальників СЕС стимулює останніх надавати пропозиції не у вигляді узагальнених прайсів, а як персоналізовані, адаптовані під конкретні потреби клієнта рішення, які враховують бажану потужність, доступний бюджет, обмеження на площу встановлення, режим споживання електроенергії, локальні кліматичні умови [20].

У зв’язку з цим комерційна пропозиція перетворюється на інструмент переддоговірного обґрунтування з детальним описом не лише конфігурації обладнання, а й фінансових переваг її впровадження. У типовому випадку потенційний клієнт прагне побачити відповідь не лише на запитання «Скільки це коштує?», але й «Що я отримаю за ці кошти, коли вкладення окупляться, яка економія буде у грошовому вираженні?». Тому саме такі параметри, як NPV (чиста приведена вартість), IRR (внутрішня норма прибутковості), строк окупності та LCOE (приведена вартість електроенергії), є найвагомішими у процесі вибору конфігурації.

Цифровізація процесу дозволяє формувати подібну пропозицію швидко, точно й без ризику людських прорахунків. Веб застосунок, який розроблено в межах цього дослідження, якраз і реалізує автоматизований підхід до формування персоналізованої комерційної пропозиції на основі вихідних даних клієнта — бажаної потужності та бюджету. Вбудована логіка розрахунку конфігурації та економічних метрик дозволяє створювати комерційні пропозиції, що не лише технічно обґрунтовані, а й фінансово переконливі [21].

Поряд із фінансовою аргументованістю не менш важливою є здатність комерційної пропозиції бути зрозумілою, зручною для сприйняття та візуально впорядкованою. Замовник, що стикається з проєктом вперше, часто не володіє спеціалізованими знаннями у галузі фотовольтаїки, тому перевантаження

пропозиції технічними термінами або складними таблицями без пояснень значно знижує її ефективність у спілкуванні з клієнтом. На практиці це призводить до втрати довіри, потреби в додатковому консультуванні або відмови від замовлення.

Комерційна пропозиція має виконувати функцію комунікаційного інструмента — бути містком між інженерною точністю та клієнтським розумінням. Структура документа має бути логічною та інтуїтивною: від загальної інформації про клієнта й базові параметри станції — до конфігурації обладнання, фінансових показників та короткого висновку з ключовими перевагами. Таблиці з технічними даними мають бути доповнені підписами, одиницями виміру, а фінансові метрики — короткими поясненнями (на кшталт «строк окупності — це кількість років до повного повернення вкладених коштів»). Такий підхід не лише підвищує довіру клієнта, а й зменшує потребу в особистому поясненні з боку менеджера чи інженера [22].

У запропонованому веб застосунку ця вимога реалізована через формування структурованого PDF-документа, який включає основні елементи: реквізити клієнта, потужність СЕС, вибране обладнання, загальний бюджет, графічне виділення фінансових показників (LCOE, NPV, IRR, строк окупності). Всі одиниці виміру вказані чітко, поля вирівняні, а мова документа стилістично витримана — як у презентації технічної комерційної пропозиції. Таке подання дозволяє легко використовувати цей документ у подальшій комунікації, наприклад, для погодження з керівництвом, інвестором або навіть банком, якщо мова йде про фінансування.

Дизайн документа враховує принципи візуальної ієрархії: заголовки, ключові цифри та висновки виділяються, а допоміжна інформація візуально другорядна. Така організація структури пропозиції безпосередньо впливає на те, як швидко замовник зможе зорієнтуватися у змісті та прийняти рішення.

Зрозумілість та якість структури комерційної пропозиції є не лише зручністю, а й фактором, що безпосередньо впливає на успішність угоди. Автоматизована система, що забезпечує формування чіткого, грамотно

структурованого документа, має істотну конкурентну перевагу в умовах сучасного ринку сонячної енергетики.

У сучасному динамічному середовищі енергетичних рішень важливу роль відіграє не лише якість, а й швидкість реагування на запити потенційних замовників. Особливо це стосується компаній, що займаються проєктуванням, продажем або монтажем СЕС у великому обсязі. У таких умовах підготовка індивідуальної комерційної пропозиції вручну — із добором обладнання, розрахунком потужності, фінансових показників і оформленням документа — перетворюється на трудомісткий процес, що вимагає часу, кваліфікації і постійного контролю [23].

Типова ситуація включає кілька ітерацій між замовником і компанією: первинний запит, збирання вихідних даних, підбір технічної конфігурації, перевірка бюджету, розрахунок генерації та окупності, фінальне погодження. У ручному режимі такий цикл може тривати від кількох днів до тижня, особливо за умови змін параметрів у процесі. У свою чергу, автоматизоване рішення, яке реалізовано у створеному веб застосунку, дозволяє суттєво скоротити цей час — до кількох хвилин від моменту введення даних до отримання повної пропозиції у PDF-форматі.

Крім того, автоматизація відкриває можливість до масштабованого застосування — зокрема, у мережах дилерів, інтеграторів, енергетичних консультантів. Один і той самий інструмент може бути використаний для генерації десятків пропозицій щодня, при цьому з гарантованою узгодженістю підходу, єдністю методики розрахунку та візуальної подачі. Це не лише економить ресурси, а й формує у клієнтів відчуття професійного рівня компанії.

Важливим є також те, що така система дозволяє зменшити людський фактор — зокрема, помилки в обрахунках, плутанину в потужностях обладнання чи несумісність компонентів. Застосунок виконує перевірку відповідності інверторів і панелей, оцінює бюджет, обирає найближчі до цільових параметрів конфігурації. Усе це відбувається відповідно до вбудованих логік і обмежень, що можуть бути адаптовані адміністратором без потреби в перепрограмуванні [24].

У результаті, автоматизоване формування пропозицій не лише задовольняє потреби клієнтів в оперативності, а й забезпечує внутрішню ефективність бізнесу, знижує витрати на підготовку та забезпечує масштабованість процесу. Реалізація такої системи, зокрема через веб застосунок, стає стратегічною перевагою в умовах конкурентного ринку.

1.5. Огляд існуючих рішень

На поточному етапі розвитку ринку сонячної енергетики в Україні й за кордоном значна частина компаній, що займаються проєктуванням та продажем сонячних електростанцій, продовжує використовувати переважно ручні або напівавтоматизовані методи формування комерційних пропозицій. Цей підхід передбачає використання шаблонів у форматах Excel, Word, іноді — заздалегідь підготовлених PDF-макетів, у які спеціалісти вручну вносять дані, обирають обладнання з локальних прайс-листів, розраховують потужність та орієнтовну вартість установки.

У типовому випадку менеджер або інженер-консультант отримує від клієнта базові вихідні дані — бажану потужність станції, доступний бюджет або площу даху — після чого виконує добір компонентів із власного переліку, здійснює розрахунок кількості панелей та підходящого інвертора. Фінансові показники (якщо взагалі враховуються) часто обмежуються лише строком окупності, без обрахунку таких інтегральних метрик, як LCOE чи NPV. Результати розрахунку подаються у вигляді документу з таблицями та логотипом компанії, який передається клієнту електронною поштою.

Попри певну гнучкість, ручні методи мають істотні обмеження. По-перше, вони є ресурсоємними: підготовка однієї пропозиції може займати від 1 до 4 годин, особливо якщо передбачено кілька варіантів конфігурацій. По-друге, висока залежність від людського чинника створює ризики помилок у підрахунках, несумісності обладнання або некоректного представлення інформації клієнту. По-третє, масштабування таких підходів у випадку

зростання кількості замовлень стає практично неможливим без пропорційного збільшення кадрового складу.

Особливої актуальності ці недоліки набувають у конкурентному середовищі, де швидкість реагування на запит і якість першої пропозиції нерідко визначають ймовірність укладання контракту. У таких умовах застосування ручних рішень ускладнює підтримку високої ділової репутації, а відсутність стандартизації й уніфікації оформлення знижує враження професійності з боку клієнта.

Хоча ручні методи можуть бути виправданими на початкових етапах діяльності або для невеликих компаній, у стратегічній перспективі вони не забезпечують необхідного рівня якості, швидкості та масштабованості. Це обґрунтовує потребу у створенні цифрових інструментів, які дозволяють формувати точні, наочні та уніфіковані пропозиції за лічені хвилини.

Інженерне проєктування сонячних електростанцій на професійному рівні часто виконується із застосуванням спеціалізованого програмного забезпечення, яке дозволяє моделювати складні конфігурації та проводити точний аналіз продуктивності систем. Найбільш поширеними продуктами цього класу є PVsyst, PV*Sol та HOMER Pro. Вони активно використовуються в проєктних організаціях, EPC-контракторами та інженерними консультантами для побудови докладних технічних моделей СЕС [24].

Програма PVsyst розроблена для глибокого моделювання енергетичних характеристик фотоелектричних систем у різних кліматичних умовах. Вона дає змогу враховувати широкий спектр параметрів: орієнтацію модулів, тіні від навколишніх об’єктів, втрати в кабельній мережі, ефекти розсіювання, деградацію панелей тощо. PV*Sol орієнтований на візуальне 3D-моделювання даху або майданчика, що дозволяє інженеру створити просторову модель установки з урахуванням розміщення панелей і впливу затінення. HOMER Pro, своєю чергою, спеціалізується на аналізі гібридних енергетичних систем і дозволяє моделювати взаємодію між генерацією з ВДЕ, накопичувачами та дизель-генераторами.

Перевагами таких програм є висока точність моделювання, підтримка міжнародних стандартів, наявність баз кліматичних даних та сертифікованих компонентів, а також можливість побудови докладних звітів для технічної експертизи чи отримання фінансування. Однак ці інструменти мають також низку обмежень у контексті використання для підготовки масових комерційних пропозицій. Передусім, вони орієнтовані на кваліфікованих інженерів і вимагають спеціального навчання. Крім того, робота з такими програмами є часоємкою — навіть базове моделювання для невеликої СЕС може зайняти кілька годин, що є неприйнятним у процесі швидкого реагування на запити клієнтів [25].

Важливо також відзначити, що більшість подібних програм не надає готового виводу у вигляді комерційної пропозиції, зрозумілої для замовника. Звіти зазвичай містять суто інженерні таблиці та графіки, які не пристосовані для використання в маркетингових чи комунікаційних цілях. Додатковим фактором є вартість ліцензій, що для малого та середнього бізнесу може бути бар’єром до впровадження.

У підсумку інженерне програмне забезпечення високого класу залишається важливим інструментом для детального технічного проєктування, його використання є недоцільним у завданнях попереднього підбору конфігурації та автоматизованого формування пропозицій, які потребують оперативності, простоти та інтуїтивного інтерфейсу. Це підкреслює доцільність створення окремого, легшого веб застосунку, спеціально орієнтованого на цільову задачу комерційного формування пропозицій СЕС.

На хвилі цифровізації у сфері відновлюваної енергетики останніми роками з’явилась низка онлайн-сервісів, які дозволяють потенційним клієнтам отримати попередню оцінку вартості сонячної електростанції без необхідності звертатися до інженера чи консультанта. Такі веб-платформи розміщуються переважно на сайтах постачальників обладнання, інтеграторів або енергетичних компаній і виконують функцію «калькулятора СЕС».

Користувачеві зазвичай пропонується ввести кілька ключових параметрів:

бажану потужність станції, площу даху, орієнтацію, регіон встановлення або щомісячне енергоспоживання. Система на основі попередньо закладених сценаріїв розраховує приблизну вартість проєкту, термін окупності, а іноді навіть річну генерацію [25]. Такий підхід має беззаперечні переваги: доступність, простота у використанні, відсутність потреби в реєстрації чи спеціальній підготовці. Водночас, за своєю суттю більшість подібних сервісів виконує лише маркетингову функцію — тобто залучення уваги до бренду чи продукту — і не призначена для інженерно точного проєктування.

Першим важливим обмеженням таких платформ є використання фіксованих шаблонів або середніх значень без врахування конкретного обладнання. У результаті отримана вартість або генерація не мають інженерної обґрунтованості, що може вводити користувача в оману. По-друге, такі платформи майже ніколи не враховують фінансові метрики — LCOE, NPV, IRR — які є визначальними для інвестора. Навіть якщо подається строк окупності, він розрахований за спрощеною формулою без урахування вартості капіталу чи інфляції.

Ще одним недоліком є відсутність персоналізованого добору обладнання. Результати ґрунтуються на типовій конфігурації, без перевірки технічної сумісності компонентів або відповідності бюджету. Користувач не має змоги змінити моделі панелей чи інверторів, порівняти варіанти або отримати документацію, придатну для комерційного використання. Крім того, такі сервіси зазвичай не генерують фінальний документ, який можна передати клієнтові або додати до заявки на фінансування.

Веб-калькулятори СЕС є зручним інструментом попередньої оцінки, їхня функціональність суттєво обмежена і не відповідає професійним вимогам щодо формування повноцінної, аргументованої, адаптованої під клієнта комерційної пропозиції. Це створює нішу для вебзастосунку нового покоління — такого, що поєднує простоту інтерфейсу з глибиною розрахунків і прозорою логікою конфігурації обладнання, як реалізовано у цьому проєкті [25].

Окрім відкритих програмних продуктів і загальнодоступних

веб-калькуляторів, на ринку сонячної енергетики дедалі частіше трапляються приклади власних внутрішніх систем, які розробляються великими інтеграторами, дистриб'юторами обладнання або інженерними компаніями з метою автоматизації бізнес-процесів. Такі рішення зазвичай реалізуються у вигляді вбудованих модулів у CRM-системи, внутрішніх порталів для менеджерів або спеціалізованих веб-інтерфейсів, доступних лише авторизованим користувачам компанії.

Ці системи дозволяють автоматизувати добір обладнання, здійснювати миттєвий розрахунок вартості проєкту, генерувати стандартизовані пропозиції в PDF-форматі, а іноді навіть інтегруватися з базами постачальників для актуалізації цін. Найчастіше логіка конфігурації прив’язана до певного обмеженого переліку товарів, які реалізує компанія, що робить процес швидким і стандартизованим для внутрішнього використання. Деякі з таких систем підтримують мультиваріантність пропозицій, ведення історії запитів, контроль маржі, автоматичне оновлення курсів валют тощо.

Незважаючи на високу ефективність у межах однієї компанії, такі рішення є закритими за своєю природою. Їх створення зазвичай ініціюється з урахуванням специфіки внутрішніх процесів, складу продуктового каталогу, цінової політики та структури продажів. Унаслідок цього їх неможливо або вкрай складно адаптувати до умов іншого бізнесу чи зробити загальнодоступними для ширшого кола користувачів. Крім того, підтримка таких рішень вимагає окремої команди розробників або IT-відділу, що підвищує експлуатаційні витрати [26].

Ще одним бар’єром є високий поріг входу для нових користувачів: вивчення інтерфейсу, специфіки налаштувань, обмеження в можливостях кастомізації під нові проєкти [26]. Часто ці платформи розробляються як однокористувацькі або для обмеженого числа регіональних дилерів, без урахування вимог широкої інтеграції або доступу ззовні.

Внутрішні корпоративні рішення демонструють високу функціональність у своїй ніші, вони не покривають потреби незалежних інсталяторів,

консультантів або невеликих компаній, які потребують гнучкого, доступного, універсального інструменту з можливістю налаштування, розгортання на власному хостингу або інтеграції з іншими сервісами. Саме на цю цільову аудиторію орієнтовано вебзастосунок, розроблений у рамках цього проєкту, що поєднує відкриту архітектуру, простоту використання і можливість адаптації під конкретні бізнес-потреби.

1.6. Формулювання функціональних і нефункціональних вимог до вебзастосунку

Функціональні вимоги визначають, яку саме поведінку повинна реалізовувати інформаційна система, та описують основні сценарії взаємодії користувача з вебзастосунком. У межах цього проєкту основним призначенням системи є автоматизоване формування комерційної пропозиції на встановлення сонячної електростанції відповідно до введених параметрів. Така пропозиція повинна бути сформована на основі реальних моделей обладнання, враховувати технічну сумісність компонентів та містити розрахунки економічних метрик, що відображають доцільність інвестицій.

У системі реалізовано механізм введення вихідних даних замовника через веб-форму. Зокрема, передбачається введення таких полів: ім’я (назва) клієнта, бажана встановлена потужність СЕС у кВт, граничний бюджет у євро, а також додаткові налаштування тарифу, якщо застосовується диференційована модель розрахунку. Після отримання вхідних даних вебзастосунок виконує автоматизований підбір фотомодулів та інверторів на основі внутрішніх довідників. Система перевіряє технічну відповідність: наприклад, чи покриває обраний інвертор заявлену потужність, чи допустимі витрати на закупівлю компонентів у межах вказаного бюджету.

Після визначення конфігурації обладнання програма розраховує техніко-економічні показники, які включають: прогнозовану річну генерацію

електроенергії, строк окупності (Payback Period), чисту приведену вартість (NPV), рівень прибутковості (IRR) та середню приведену вартість електроенергії (LCOE). Ці метрики обчислюються з урахуванням інсоляції, дисконтування грошових потоків та операційних витрат, заданих у параметрах системи. Обробка відбувається на серверній частині — на базі окремого сервісного класу з алгоритмічною логікою.

Важливою функціональною можливістю є генерація PDF-документа, що містить всю релевантну інформацію у зручному для клієнта форматі. Документ включає реквізити, перелік обладнання, технічні параметри, фінансові метрики та короткий підсумок. Він може бути завантажений або відкритий безпосередньо в браузері. Задля забезпечення повторного доступу до результатів кожна пропозиція зберігається в базі даних, а користувач має змогу переглядати історію попередніх запитів.

Окремий функціональний блок передбачає панель адміністратора, де користувач із відповідними правами може додавати, редагувати або видаляти записи у довідниках фотомодулів та інверторів. Це забезпечує адаптивність системи до змін на ринку, оновлення цін і введення нових моделей обладнання без потреби втручання у код.

Загалом сформульовані функціональні вимоги відображають не лише технічну реалізацію, а й очікування користувача щодо швидкості, точності та повноти пропозиції. Вони стали основою для подальшого проєктування модулів системи, структури бази даних та маршрутизації, які докладно розкриваються у другому розділі цієї дипломної роботи.

Нефункціональні вимоги визначають не те, *що* саме має робити система, а *якою* вона повинна бути з точки зору продуктивності, надійності, безпеки, зручності користування та адаптивності до умов розгортання. У рамках створення вебзастосунку для формування комерційних пропозицій СЕС ці вимоги відіграють критично важливу роль, оскільки визначають успішність впровадження рішення в реальних умовах, зокрема для малого бізнесу та консультантів, які не мають складної серверної інфраструктури або штатного

IT-персоналу.

Однією з основних нефункціональних вимог є простота та доступність розгортання. Система має працювати на стандартному стеку технологій (LAMP або LEMP) із мінімальними залежностями. Використання PHP-мови версії 8 і вище дозволяє забезпечити сумісність із більшістю хостингів. Наявність шаблонізатора Blade та композера як менеджера залежностей мінімізує вимоги до навчання нового розробника. Важливо, щоб застосунок не потребував встановлення складних фреймворків або окремих серверів (як-от Laravel Forge чи Docker), що дозволяє швидко розгортати його в освітніх, тестових чи комерційних середовищах.

Другим ключовим аспектом є зручність користування та інтуїтивний інтерфейс (UI/UX). Інтерфейс вебзастосунку має бути доступним для користувача без технічної освіти: форми повинні мати зрозумілі підписи, одиниці виміру, підказки. Особливу увагу приділено респонсивності інтерфейсу — він повинен коректно працювати як на ПК, так і на мобільних пристроях. Сторінки мають завантажуватися швидко, без затримок у відгуку, а помилки (наприклад, при введенні даних) повинні коректно оброблятися без перезавантаження сторінки.

Також важливо забезпечити збереження та безпеку персональних даних. Зокрема, дані клієнтів, збережені у базі, повинні бути недоступні для сторонніх осіб. Всі запити з форм повинні проходити перевірку на валідність і бути захищеними від XSS та CSRF-атак. У разі розширення системи до багато користувацької — повинна бути реалізована система авторизації з розмежуванням прав доступу. З міркувань безпеки також передбачено, що згенеровані PDF-документи мають зберігатися у недоступній для прямого перегляду директорії, з посиланням лише для зареєстрованого користувача або адміністратора.

Окремо формулюється вимога щодо локалізації та масштабованості інтерфейсу. Оскільки цільовою аудиторією є українські користувачі, основна мова інтерфейсу — українська, проте структура системи дозволяє реалізувати

багатомовність у майбутньому без зміни архітектури. Масштабованість у даному контексті передбачає не лише зростання кількості користувачів, а й можливість розширення функціоналу (наприклад, додавання нових типів обладнання або типів станцій).

Усі вищезазначені нефункціональні вимоги закладено у технічну архітектуру системи з урахуванням реалістичних обмежень ресурсу, що забезпечує не лише працездатність у дослідницькому чи навчальному середовищі, а й адаптивність до реального впровадження у практиці малих і середніх енергетичних компаній.

У процесі розробки вебзастосунку для автоматизованого формування комерційної пропозиції сонячної електростанції було свідомо обрано архітектуру, яка поєднує мінімалізм реалізації з достатньою гнучкістю для майбутнього масштабування. Основою програмної частини став стек технологій на базі мови PHP, яка є стабільною, поширеною, легко підтримуваною і не потребує складного середовища виконання. Версія PHP 8+ обрана з огляду на її підтримку строгої типізації, сучасного об’єктно-орієнтованого синтаксису та сумісності з актуальними бібліотеками.

У якості шаблонізатора інтерфейсу використано Blade — компонент, запозичений із фреймворку Laravel, але інтегрований у легку кастомну архітектуру без використання повного фреймворку. Це дозволяє розробляти сторінки з дотриманням принципів MVC, із чітким розділенням логіки, подання і даних, але без надлишкових залежностей. Усі маршрути обробляються за допомогою власного класу Router, який забезпечує підтримку базового REST-підходу та зручне делегування контролерам. Автоматичне завантаження класів організоване через Composer — індустріальний стандарт для PHP-екосистеми.

Зовнішні залежності зведені до мінімуму. Для генерації PDF-документів використано бібліотеку Dompdf, що дозволяє конвертувати HTML-шаблони в повноформатні документи із підтримкою українських шрифтів. Для збереження пропозицій у базі даних застосовано легку реалізацію ORM-шару, засновану на PDO, без складних міграційних механізмів, але з підтримкою зворотної

серіалізації об’єктів.

Серед свідомих технічних обмежень — відсутність клієнтської частини у вигляді SPA або JS-фреймворків. Інтерфейс реалізовано повністю на основі серверного рендерингу з використанням Bootstrap 5, що забезпечує адаптивність і простоту підтримки. Це дозволяє розгорнути систему навіть на бюджетному хостингу або локальному середовищі без залежності від Node.js чи Vue/React. Базові можливості перевірки форм реалізовано за допомогою вбудованих HTML-атрибутів та невеликої кількості JS-скриптів.

Окремо слід зазначити, що для адміністрування конфігурацій обладнання (інверторів, панелей тощо) реалізовано автономну панель, доступ до якої обмежується системою автентифікації. Усі змінні конфігурації (тарифи, коефіцієнти BOS, авторські підписи) задаються через .env-файл, що дозволяє швидко адаптувати застосунок до різних умов без внесення змін у код.

У сукупності технічна реалізація враховує вимоги до наочності, повторюваності, розширюваності та відповідності типовому стеку для студентських, навчальних і малих комерційних проєктів, уникаючи при цьому надмірної складності. Така модель дозволяє не лише формально задовольнити потреби користувача, а й закласти основу для подальшого розвитку функціоналу в рамках другого розділу цієї роботи.

РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ ВЕБЗАСТОСУНКУ

2.1. Вибір технологічного стеку

Вибір мови програмування для серверної частини вебзастосунку є критичним рішенням, яке визначає архітектурну модель, набір доступних бібліотек, швидкість розробки, а також ресурси, необхідні для підтримки системи в експлуатації. У межах цього проєкту як основну мову обрано PHP — зріле, стабільне та спеціалізоване рішення для створення динамічних вебресурсів.

PHP, або «Hypertext Preprocessor», є інтерпретованою мовою сценаріїв, яка понад два десятиліття використовується в сфері веброзробки. Її перевага полягає у глибокій інтеграції з вебсерверами (зокрема Apache, Nginx) [27], широкій підтримці хостинг-платформ, великій кількості вбудованих функцій для обробки HTTP-запитів, роботи з базами даних, обробки файлів і форматування виводу. Завдяки цьому PHP залишається одним із найпоширеніших рішень для створення як простих сайтів, так і повнофункціональних вебсистем.

У контексті реалізації системи для формування комерційних пропозицій CEC PHP виявився особливо доцільним з таких причин. По-перше, він дозволяє швидко реалізувати архітектуру MVC (Model–View–Controller) [28], не вимагаючи використання важкого фреймворку, що важливо для навчальних, експериментальних або автономних застосунків. По-друге, PHP має низький поріг входу, що є перевагою в умовах, коли система розробляється студентом або невеликою командою без окремого DevOps-підрозділу [28]. По-третє, сучасна версія PHP 8+ підтримує об’єктно-орієнтовану парадигму, строгу типізацію, анотації та механізми автозавантаження класів через Composer, що дозволяє підтримувати чисту та масштабовану архітектуру.

Окрім того, PHP забезпечує повну сумісність із базами даних MySQL, що дозволяє легко зберігати пропозиції, конфігурації обладнання та інші

довідникові дані. Також мова дозволяє реалізувати інтеграцію з шаблонізаторами, такими як Blade, а також із бібліотеками для PDF-генерації (Dompdf), без необхідності встановлювати зовнішні інтерпретатори або середовища виконання.

Вибір PHP є цілком виправданим у межах цілей дипломного проєкту: створити стабільний, доступний для розгортання, функціональний вебзастосунок, який дозволяє користувачам без зайвих технічних навичок швидко отримувати індивідуалізовані комерційні пропозиції. Простота екосистеми, підтримка спільноти та відсутність критичних бар'єрів для обслуговування роблять PHP оптимальним рішенням для серверної частини даної системи.

Одним із ключових завдань при проєктуванні вебзастосунку є правильне організаційне розділення логіки програми та її візуального представлення. У рамках даного проєкту цю функцію виконує Blade — сучасний шаблонізатор, який походить із фреймворку Laravel, але у реалізації даної системи інтегрується як самостійний компонент. Такий підхід дозволяє реалізувати структуровану і гнучку модель подання без необхідності повної інтеграції з Laravel, що відповідає вимогам мінімалізму і контрольованості в архітектурі дипломного проєкту [29].

Blade є потужним і водночас простим у використанні шаблонізатором, який дозволяє створювати динамічні HTML-сторінки із вбудованими директивами для умовних конструкцій, циклів, включення інших шаблонів (через @include) та спадкування макетів (@extends, @section, @yield). Завдяки цьому розробник може зберігати єдину логіку подання на всіх сторінках, винести спільні елементи (навігація, футер, шапка) в окремий layout-файл і централізовано їх редагувати. Це значно підвищує зручність підтримки та розширення інтерфейсу, особливо у багатосторінковому застосунку.

Інтеграція Blade у проєкт реалізована через встановлення пакета jenssegers/blade, що дозволяє підключити шаблонізатор без необхідності використання повного Laravel-ядра. Компіляція шаблонів виконується

автоматично, а результати кешуються, що сприяє підвищенню продуктивності. Важливо й те, що Blade підтримує передавання змінних із контролерів у шаблони через асоціативні масиви, що забезпечує чітку межу між логікою обробки даних і їх відображенням.

Окрім зручності структурування коду, Blade дає змогу створювати лінгвістично чисті шаблони, що легко читаються навіть нефахівцем. Це особливо актуально у студентських проєктах, де важливо забезпечити прозору логіку та зрозумілу архітектуру для перевіряючого. Відсутність складних структур JavaScript або JSX також знижує вимоги до фронтенд-підготовки розробника [30].

Використання Blade у проєкті є стратегічним рішенням, що дозволяє досягти балансу між функціональністю, продуктивністю і простотою підтримки інтерфейсу. Завдяки йому забезпечено гнучке розділення структури сторінок, повторне використання компонентів, а також централізовану зміну вигляду, що є особливо важливим у проєктах, де інтерфейс має бути водночас зрозумілим для користувача і логічно оформленим для розробника.

У структурі будь-якого вебзастосунку критичною компонентою є механізм маршрутизації (routing), який відповідає за обробку HTTP-запитів і передавання керування до відповідного функціонального модуля. У класичних фреймворках маршрутизація реалізована через конфігураційні файли або анотації, але у розробленому застосунку використано власну полегшену реалізацію маршрутизатора, яка забезпечує необхідну гнучкість без перевантаження архітектури.

Основу маршрутизації становить спеціалізований клас Router, що підтримує базові HTTP-методи (GET, POST) [31], дозволяє реєструвати маршрути та асоціювати їх із викликами контролерів або анонімних функцій. Цей підхід зберігає відповідність REST-принципам і забезпечує зрозуміле логічне розділення запитів за їх призначенням: наприклад, GET / — для відображення головної сторінки, POST /calculate — для обробки форми з розрахунком, GET /admin — для панелі адміністрування. Додатково реалізовано

підтримку динамічних параметрів у маршрутах, наприклад GET /proposal/{id}, що дозволяє передавати ідентифікатор пропозиції для її перегляду.

Для кожного маршруту визначено делегування у контролер — окремий клас, відповідальний за обробку логіки певного сегмента застосунку. Контролери організовані у відповідному просторі імен (App\Controllers) та структуровані за функціональною ознакою: ProposalController — для роботи з комерційними пропозиціями, AdminController — для обробки адміністративного інтерфейсу, AuthController — для авторизації користувачів. Такий підхід дозволяє досягти модульності, логічної ізоляції відповідальності та спрощення підтримки коду.

У реалізації маршрутизатора передбачено механізм обробки маршруту за методом і URI, де відповідність здійснюється за допомогою регулярних виразів, а у випадку невідповідності повертається стандартна помилка 404. Також передбачено виклик контролера у форматі [Клас::class, 'метод'], що забезпечує сумісність з автозавантаженням класів через Composer та підтримку PHP-типізації.

У випадках, коли певні маршрути потребують авторизації (guarding), реалізовано перевірку доступу до них перед викликом dispatch-методу. Наприклад, доступ до /admin або /admin/inverters перевіряється через сервіс аутентифікації, який визначає, чи користувач авторизований. У разі порушення правил доступу користувач автоматично перенаправляється на форму входу.

У результаті реалізована система маршрутизації поєднує простоту і гнучкість. Вона не прив’язана до складного фреймворку, зберігає логічну структуру програми, дозволяє чітко делегувати обробку до контролерів і забезпечує передбачувану поведінку для всіх типових запитів. Це дає змогу ефективно організувати як користувацьку логіку взаємодії з формами, так і адміністративний інтерфейс із контролем доступу.

Однією з ключових функціональних вимог до розробленого вебзастосунку є здатність формувати структуровану, візуально привабливу комерційну пропозицію у форматі PDF — документ, який може бути роздрукований,

надісланий електронною поштою або збережений для подальшого використання клієнтом. З огляду на цю вимогу, було обрано бібліотеку Dompdf як основний інструмент для реалізації механізму генерації друкованої версії пропозиції.

Dompdf є бібліотекою на мові PHP, що дозволяє конвертувати HTML-шаблони в PDF-документи із застосуванням CSS-стилів. Перевагами даного рішення є його простота використання, нативна підтримка HTML5, відсутність необхідності в зовнішніх бінарних залежностях, а також достатньо висока якість відтворення верстки [32]. Dompdf здатен працювати автономно без встановлення додаткових системних компонентів, що особливо актуально для студентських і освітніх проєктів, де середовище обмежене лише PHP-інтерпретатором.

У межах реалізації даної системи шаблон PDF-документа створюється як Blade-шаблон, що забезпечує уніфікацію логіки інтерфейсу. Верстка пропозиції (включно з таблицями обладнання, техніко-економічними показниками та контактними даними) відтворюється у звичному для фронтенду HTML-середовищі, а потім передається в Dompdf для рендерингу. Це значно спрощує внесення змін у вигляд пропозиції, оскільки не потребує знання специфікацій PDF-формату [33].

Особливу увагу приділено підтримці українських шрифтів. У проєкті задіяно шрифт Roboto, який додатково зареєстрований у налаштуваннях Dompdf та включає кириличну підтримку. Це дозволяє уникнути типових помилок з відображенням знаків питання або некоректного кодування. Бібліотека також дозволяє виводити документ як у вигляді inline-перегляду у браузері, так і зберігати його у файловій системі сервера — з унікальним ім’ям, що містить ідентифікатор пропозиції.

З технічної точки зору генерація документа реалізована у вигляді окремого сервісного класу PdfService, що приймає на вхід об’єкт пропозиції (Proposal) та формує PDF шляхом передачі його в Blade-шаблон. Результат зберігається в директорії storage/proposals, а шлях до файлу повертається для подальшого завантаження або перегляду.

Використання Dompdf дозволяє реалізувати професійний вигляд комерційної пропозиції, забезпечити її адаптивність під зміни в контенті, підтримку локалізації та контрольований процес виводу. Це рішення повністю відповідає вимогам до простоти, стабільності та точності відтворення фінансово-технічної інформації, необхідної для прийняття клієнтом інвестиційного рішення [33].

Для забезпечення стабільного зберігання, обробки та доступу до інформації про комерційні пропозиції, клієнтів, фотомодулі та інвертори в межах реалізованого вебзастосунку використано реляційну базу даних MySQL, що працює в парі з серверною частиною на PHP через інтерфейс PDO (PHP Data Objects). Такий підхід дозволяє ефективно взаємодіяти з базою даних, використовуючи захищені підготовлені запити, об’єктну обгортку над даними та гнучку модель роботи з таблицями без потреби у повноцінному фреймворковому ORM.

MySQL обрано з міркувань сумісності, надійності, широкої підтримки хостинг-провайдерами, а також з огляду на її продуктивність і зручність інтеграції з PHP-застосунками. Вона дозволяє реалізувати логічно структуровану схему бази, де кожен об’єкт системи (пропозиція, панель, інвертор) відповідає окремій таблиці з чітко визначеними атрибутами.

У проєкті реалізовано базову ORM-абстракцію, яка не використовує важкі зовнішні бібліотеки, проте забезпечує повноцінну обробку об'єктів моделей через відповідні класи (Proposal, SolarPanel, Inverter). Кожна модель містить логіку зчитування, збереження та серіалізації у масив, що дозволяє швидко конвертувати дані для відображення у шаблоні або збереження в таблицю. Зв'язки між таблицями реалізовано через зовнішні ключі (наприклад, proposal.panel_id → solar_panels.id), що забезпечує цілісність і відслідковуваність конфігурації обладнання у межах кожної пропозиції [34].

З технічного погляду, таблиця proposals містить такі поля, як ідентифікатор пропозиції, ім’я клієнта, обрану потужність станції, ідентифікатори компонентів, кількість панелей, розраховані фінансові метрики (CAPEX, LCOE, NPV, IRR,

строк окупності) та часову мітку створення. Таблиці solar_panels та inverters містять відповідно характеристики обладнання — потужність у Вт, ціну в євро, допустимі параметри навантаження, тощо. Усі дані вводяться адміністратором через окрему частину застосунку (панель керування), що дозволяє гнучко оновлювати каталог без втручання в програмний код.

Зберігання даних реалізоване у вигляді персистентного шару, до якого звертається логіка розрахунку через сервіс CalculationService, отримуючи доступ до переліку компонентів, перевіряючи їхню сумісність і здійснюючи вибір оптимальної конфігурації. Усі зміни в даних здійснюються через класи моделей, що унеможливлює пряме порушення цілісності таблиць.

Реалізована модель зберігання забезпечує баланс між простотою, гнучкістю і надійністю. Вона легко адаптується до змін у структурі обладнання, дає змогу зберігати історію заявок, підтримує подальшу аналітику, а також залишається придатною до масштабування у випадку розширення функціоналу або переходу на складніший ORM-рівень у майбутньому.

2.2. Архітектура, ключові модулі системи

Розроблений вебзастосунок реалізовано відповідно до трирівневої архітектурної моделі, яка забезпечує чітке розділення відповідальностей між рівнями подання (інтерфейс користувача), бізнес-логіки (контролери, сервіси) та доступу до даних (моделі, база даних). Такий підхід покращує структурованість системи, полегшує її тестування, розширення й підтримку, а також дозволяє масштабувати окремі компоненти незалежно один від одного.

На рівні presentation (view) функціонують шаблони Blade, які відповідають за відображення вебсторінок, форм введення, повідомлень і таблиць даних. Вони отримують інформацію від контролерів, які виступають посередниками між представленням і бізнес-логікою. Самі контролери (рівень logic) реалізують маршрутизацію, обробку HTTP-запитів, виклик сервісів, перевірку авторизації

тощо. Наприклад, ProposalController обробляє запити / та /calculate, а AdminController — керує інтерфейсом адміністрування.

Бізнес-логіка зосереджена в сервісних класах, таких як CalculationService і PdfService, які відповідають відповідно за вибір конфігурації СЕС на основі введених параметрів та генерацію PDF-документа. На рівні data access реалізовано моделі Proposal, Inverter, SolarPanel, що оперують даними в базі MySQL через інтерфейс PDO. Кожна модель містить функціональність для читання, запису, оновлення та серіалізації інформації. Загальну структуру архітектури показано на діаграмі нижче (рис. 2.1).

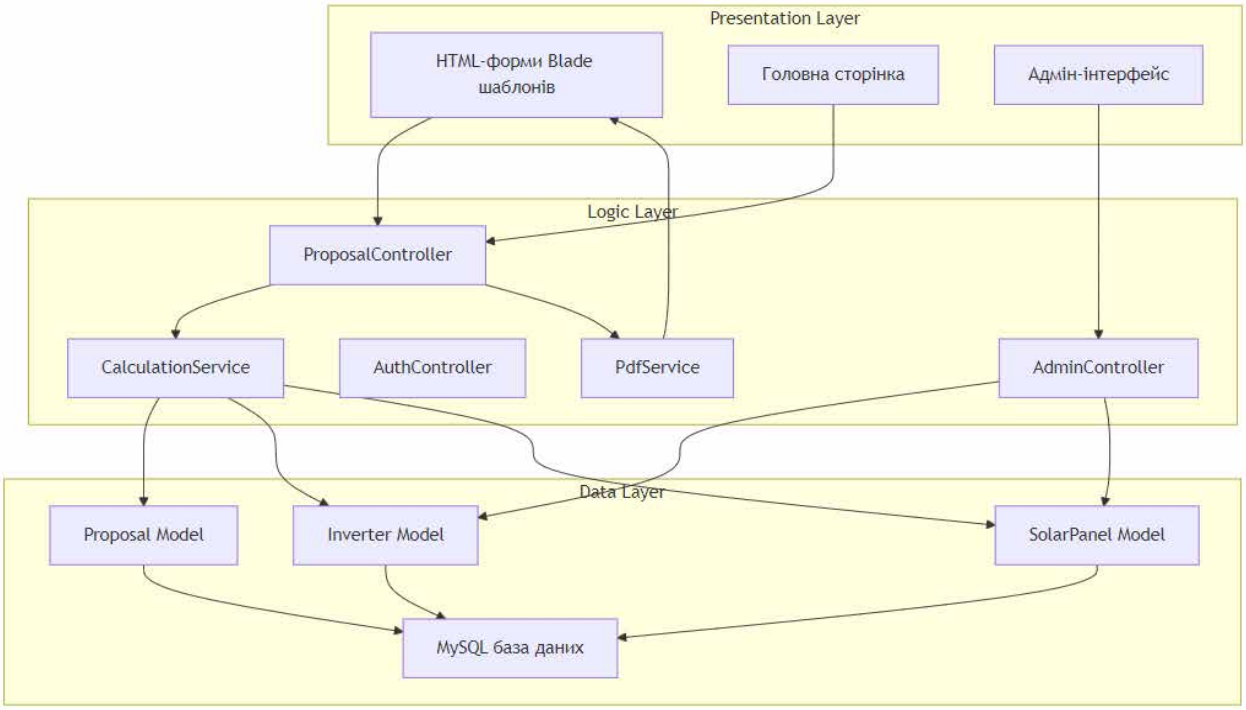


Рис. 2.1 – Загальна архітектура вебзастосунку

У результаті побудовано гнучку, модульну архітектуру, в якій кожен компонент має чітко визначену відповідальність. Це дає змогу незалежно змінювати логіку розрахунків, структуру інтерфейсу або типи збережених даних без впливу на інші частини системи. Така модель відповідає сучасним вимогам до розробки вебзастосунків у галузі інженерного проєктування та є зручною як у навчальних, так і в комерційних умовах.

Серцем обробки запитів у вебзастосунку є модуль маршрутизації, який визначає, як система реагує на конкретні HTTP-запити користувача. Його

основна функція — зв’язати вхідний URI (Uniform Resource Identifier) та метод запиту (GET або POST) з відповідною логікою, реалізованою в контролерах. В умовах кастомної архітектури, що не використовує повнофреймворкове рішення на зразок Laravel, реалізовано власний клас Router, який дозволяє реєструвати маршрути за допомогою статичних методів get() та post() [35].

Кожен маршрут прив’язаний до конкретного контролера — класу, відповідального за обробку запиту, взаємодію з моделями та сервісами, а також повернення результатів користувачу у вигляді сторінки або JSON-відповіді.

Структура контролерів реалізована у просторі імен App\Controllers і включає:

- ProposalController — обробка запитів на головну сторінку, розрахунок комерційної пропозиції, перегляд результатів;
- AdminController — управління довідниками обладнання (CRUD);
- AuthController — забезпечення автентифікації адміністратора (вхід, вихід, перевірка сесії).

Усі маршрути розділено логічно на публічні (для користувача) та адміністративні. Доступ до маршрутів, що починаються з /admin, контролюється через guard-перевірку авторизації. Якщо користувач не автентифікований, його перенаправляють на сторінку входу.

Цю логіку маршрутизації відображає схема нижче (рис. 2.2), яка показує ключові переходи між сторінками застосунку.

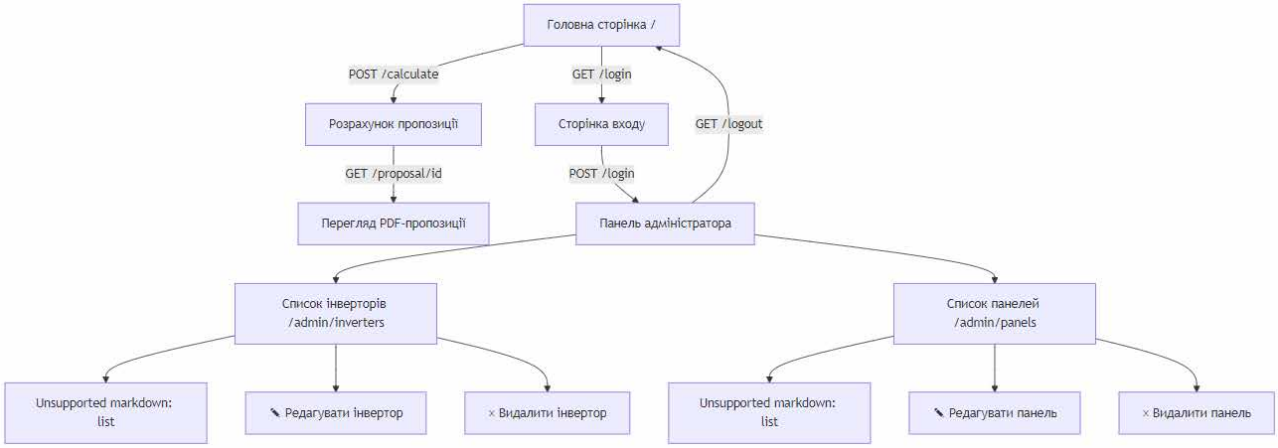


Рис. 2.2 – Схема маршрутизації вебзастосунку

Кожен маршрут пов’язаний із певним контролером. Наприклад:

- GET / обробляється методом index() класу ProposalController,
- POST /calculate — методом calculate(), який викликає сервіс обчислень,
- GET /admin/inverters — InverterController::index(), і т.д.

Уся маршрутизація централізовано реєструється в єдиному вході системи — index.php, що розташований у каталозі public/, а сам маршрутизатор здійснює делегування виклику з урахуванням HTTP-методу та URI-шаблону.

Такий підхід до маршрутизації поєднує простоту реалізації, зручність підтримки та логічну ізоляцію логіки кожного функціонального модуля, що дозволяє безболісно розширювати застосунок у майбутньому.

У розробленому вебзастосунку обчислення параметрів сонячної електростанції, підбір обладнання та формування фінального документа реалізовано у вигляді сервісного шару (service layer). Це окрема логічна частина системи, що інкапсулює бізнес-логіку й дозволяє контролерам залишатися «тонкими», делегуючи складні операції в автономні класи [36].

Головну роль у процесі формування комерційної пропозиції відіграє клас CalculationService. Саме він, отримуючи параметри з вхідної форми (бажана потужність, бюджет, тариф), здійснює пошук оптимальної конфігурації сонячних панелей і відповідного інвертора з урахуванням цін, продуктивності й бюджетного обмеження. У випадку успішного підбору система додатково обчислює техніко-економічні показники:

- річну генерацію на основі інсоляції;
- LCOE (собівартість кіловат-години);
- NPV, IRR, payback period — ключові фінансові метрики інвестиційного проєкту.

Після обчислень результати передаються в PdfService [37], який на основі Blade-шаблону формує фінальний документ у форматі PDF. Це дозволяє користувачу зберегти комерційну пропозицію, поділитися нею або роздрукувати для подальшого аналізу. Застосунок підтримує як збереження документа у файловій системі (/storage/proposals/), так і відкриття у браузері у вигляді попереднього перегляду.

Загальну логіку роботи сервісного шару відображає наступна схема (рис. 2.3), яка ілюструє послідовність викликів між модулями.

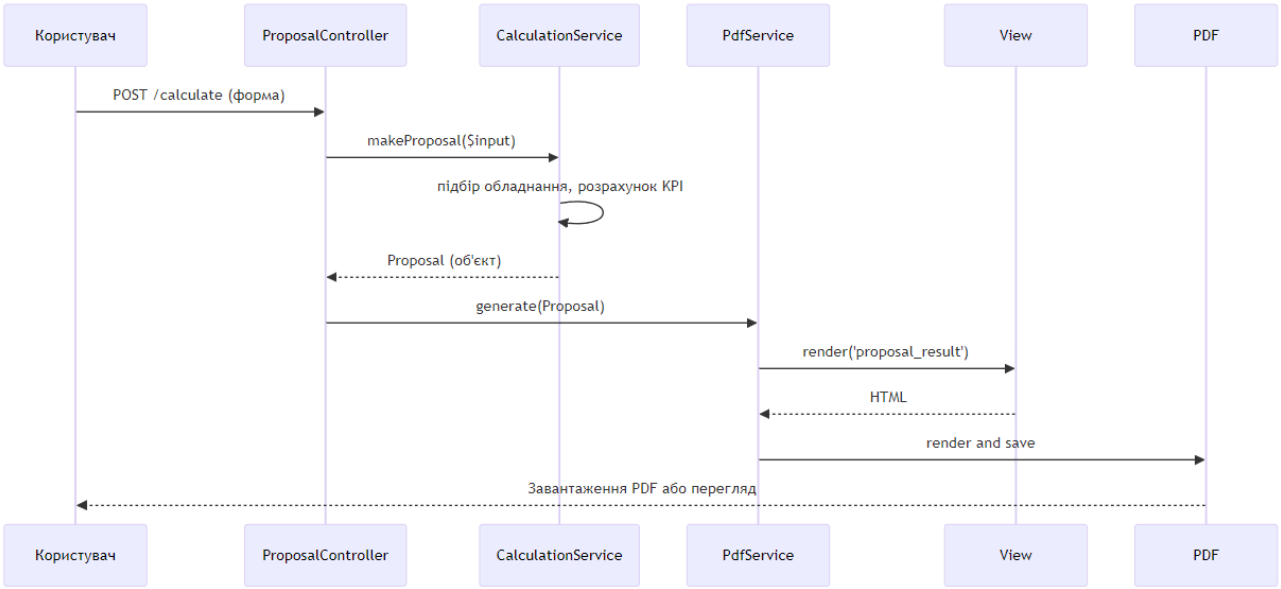


Рис. 2.3 – Схема обробки запиту та генерація PDF

Реалізація сервісного шару забезпечує інкапсуляцію обчислювальної логіки, можливість багаторазового використання сервісів (напр., в API або у майбутніх розширеннях) та зручну масштабованість: у разі потреби можна реалізувати додаткові сервіси (наприклад, аналітики, генерації порівняльних звітів), не змінюючи архітектури контролерів чи представлення.

У рамках вебзастосунку зберігання інформації реалізовано через реляційну модель бази даних, що забезпечує цілісність, послідовність та ефективність доступу до об’єктів системи. Усі сутності, з якими працює система, представлені у вигляді моделей у просторі імен App\Models та зберігаються у відповідних таблицях бази даних MySQL. Моделі виконують роль об’єктно-орієнтованої обгортки над записами в БД, включаючи логіку зчитування, запису, перевірки валідності та перетворення у формат JSON або масив [38].

Система працює з трьома ключовими сутностями:

- Proposal (пропозиція) — основна бізнес-одиниця, яка містить всю інформацію про створену клієнтом заявку, включно з обраною конфігурацією та розрахованими показниками.
- SolarPanel (фотомодуль) — довідник сонячних панелей із зазначенням

потужності, вартості, ефективності.

- Inverter (інвертор) — довідник інверторів, що містить технічні характеристики, включно з максимальною потужністю та ціною.

Зв’язки між таблицями реалізовано через зовнішні ключі. Пропозиція містить ідентифікатори обраної панелі (panel_id) та інвертора (inverter_id), що дозволяє легко здійснювати JOIN-операції при генерації звітів або PDF-пропозицій. Структуру моделей системи наведено на рис. 2.4.

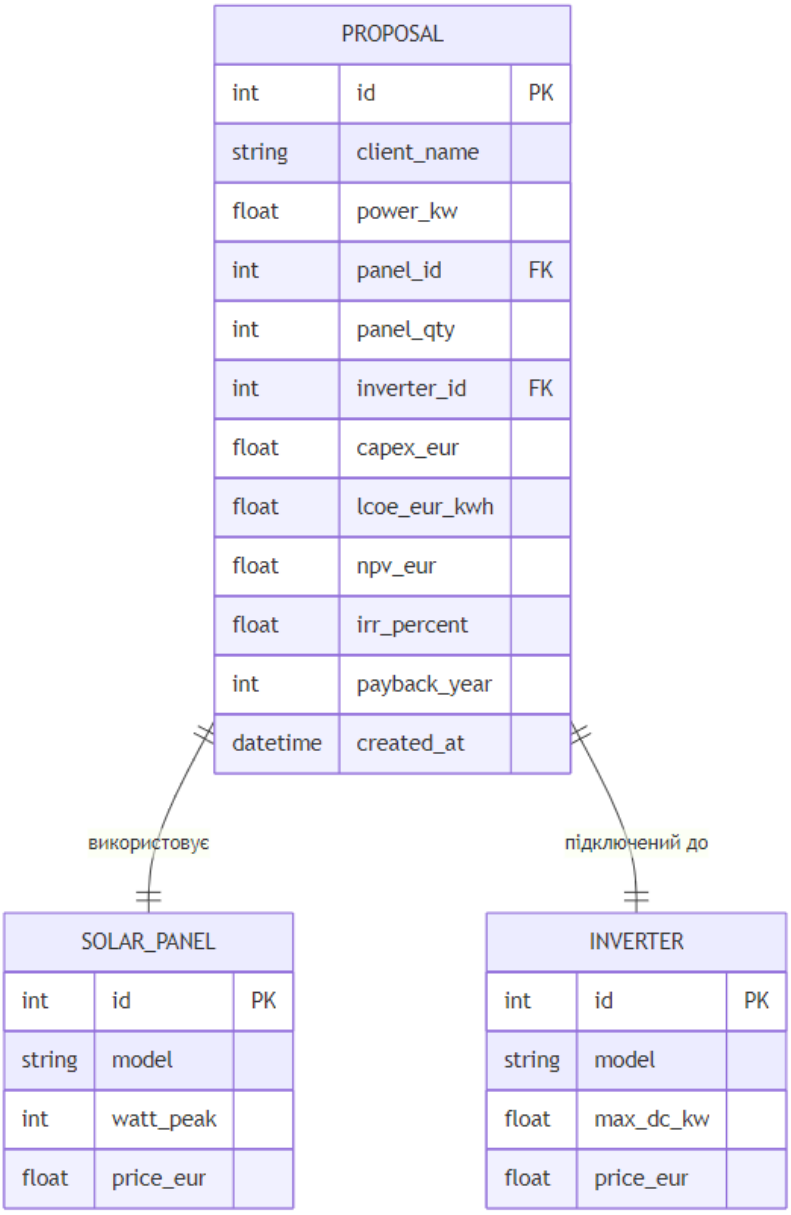


Рис. 2.4 – ER-діаграма моделей вебзастосунку

Така структура забезпечує гнучкість при формуванні нових пропозицій, дозволяє виводити історію попередніх заявок, а також реалізовувати інтерфейс адміністрування обладнання без порушення логіки даних. Моделі SolarPanel та Inverter містять також методи для отримання всіх доступних елементів та зручного представлення у формі списків у шаблонах [39].

Також передбачено, що всі об'єкти можуть бути легко серіалізовані у JSON для можливого майбутнього API-інтерфейсу, що робить проєкт відкритим до розширення.

Для кращого розуміння сценаріїв взаємодії клієнта з вебзастосунком на рисунку нижче подано діаграму прецедентів, що наочно демонструє основні функції, доступні клієнту, та логічні зв'язки між ними (Рис 2.5.).

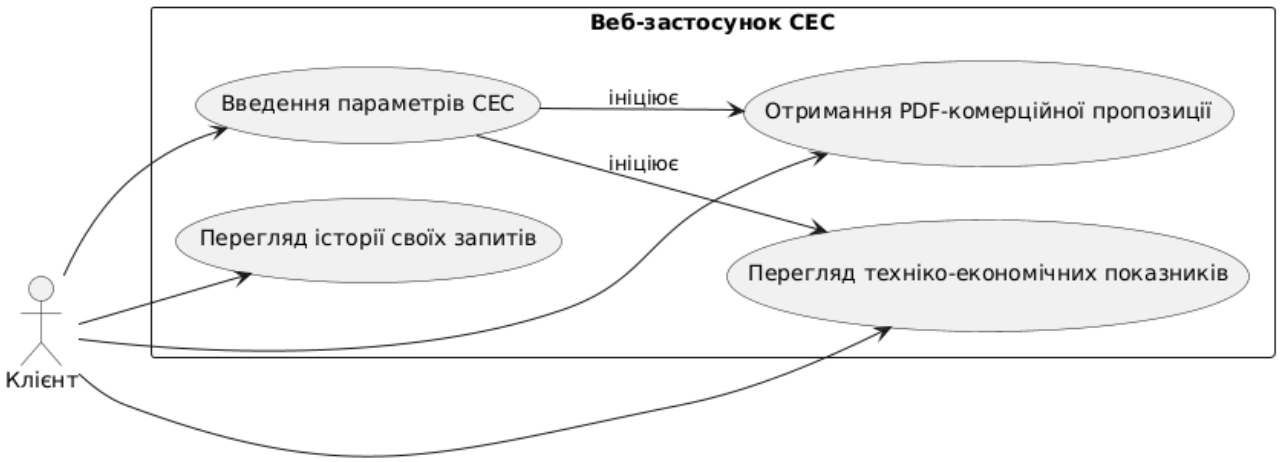


Рис. 2.5 – Діаграма прецедентів клієнта

Для адміністративної ролі, що передбачає керування базою обладнання та параметрами системи, також розроблено відповідну діаграму прецедентів (Рис. 2.6.). Вона наочно відображає основні завдання адміністратора у системі та дозволяє зрозуміти логіку його роботи з вебзастосунком.

Рис. 2.6 – Діаграма прецедентів адміністратора

Для розгортання сервера MySQL у середовищі розробки було використано технологію контейнеризації Docker. Це забезпечує ізольоване, портативне та відтворюване середовище для бази даних. Сервер MySQL 8.0 запускається у Docker-контейнері, налаштованому за допомогою файлу docker-compose.yml. Ключові параметри, такі як пароль користувача root та назва бази даних за замовчуванням (solar_proposal), визначаються через змінні середовища в цьому файлі. Порт сервера MySQL (3306 всередині контейнера) прокидається на аналогічний порт хост-машини, що робить його доступним для підключення з PHP-додатку.

Для безпечного зберігання чутливих даних, таких як облікові дані для доступу до бази даних, використовується файл .env. Цей файл містить пари ключ-значення, які завантажуються в змінні середовища PHP під час запуску додатку. Основні змінні для конфігурації бази даних включають:

- DB_CONNECTION: Тип з'єднання (наприклад, mysql).
- DB_HOST: Адреса хоста сервера MySQL (127.0.0.1 для локального Docker-контейнера).
- DB_PORT:).

- DB_DATABASE: solar_proposal
- DB_USERNAME: root
- DB_PASSWORD: Пароль користувача.

PHP-додаток використовує конфігураційний файл (Рис. 2.7.) для визначення параметрів з'єднання з базою даних. Цей файл зазвичай повертає масив налаштувань, де ключові параметри (хост, порт, назва БД, ім'я користувача, пароль) зчитуються зі змінних середовища, завантажених з файлу .env. Такий підхід дозволяє легко змінювати налаштування підключення для різних середовищ (розробка, тестування, продакшн) без модифікації коду. Файл також визначає кодування (utf8mb4) та правила сортування (utf8mb4_unicode_ci) для забезпечення коректної роботи з різними наборами символів.

```php
return [
    'default' => getenv('DB_CONNECTION') ?: 'mysql',
    'connections' => [

        'mysql' => [
            'driver'      => 'mysql',
            'host'        => getenv('DB_HOST') ?: '127.0.0.1',
            'port'        => getenv('DB_PORT') ?: '3306',
            'database'    => getenv('DB_DATABASE') ?: 'solar_proposal',
            'username'    => getenv('DB_USERNAME') ?: 'root',
            'password'    => getenv('DB_PASSWORD') ?: '12345',
            'unix_socket' => getenv('DB_SOCKET') ?: '',
            'charset'     => 'utf8mb4',
            'collation'   => 'utf8mb4_unicode_ci',
            'prefix'      => '',
            'strict'      => true,
            'engine'      => null,
            'options'     => extension_loaded('pdo_mysql') ? array_filter([
                PDO::MYSQL_ATTR_SSL_CA => getenv('MYSQL_ATTR_SSL_CA'),
            ]) : [],
        ],
    ],
];
```

Рис. 2.7 – Фрагмент файлу конфігурації database.php

Для взаємодії PHP з MySQL необхідний відповідний драйвер. Найчастіше використовується розширення PHP pdo_mysql, яке надає інтерфейс PDO (PHP Data Objects) для роботи з базами даних MySQL. Це розширення має бути встановлене та активоване у конфігурації PHP (php.ini) на сервері, де виконується додаток.

Основні таблиці:

- inverters: Зберігає інформацію про доступні моделі інверторів, включаючи їх

назву, максимальну потужність по постійному струму (DC), потужність по змінному струму (AC) та ціну. Кожен запис має унікальний ідентифікатор id.

- solar_panels: Містить дані про сонячні панелі: назва, пікова потужність (Вт), напруга в точці максимальної потужності (VMP) та ціна. Кожна панель також має унікальний id.
- proposals: Центральна таблиця, що зберігає інформацію про сформовані комерційні пропозиції. Вона включає дані клієнта, розрахункові параметри потужності, посилання на обрані панелі (panel_id) та інвертори (inverter_id) з відповідних таблиць, кількість панелей та інверторів, розраховані фінансові показники (CAPEX, LCOE, NPV, IRR, термін окупності), прапорець перевищення бюджету та дату створення.

Таблиця proposals пов'язана з таблицями solar_panels та inverters через зовнішні ключі:

- proposals.panel_id посилається на solar_panels.id.
- proposals.inverter_id посилається на inverters.id. Ці зв'язки забезпечують цілісність даних та дозволяють ефективно вибирати пов'язану інформацію. Встановлено обмеження ON DELETE RESTRICT, що запобігає видаленню панелі або інвертора, якщо на них є посилання у пропозиціях.

У PHP-додатку взаємодія з базою даних реалізована через моделі (Inverter.php, SolarPanel.php, Proposal.php). Ці моделі використовують бібліотеку illuminate/database (зокрема, її компонент Capsule Manager для роботи поза фреймворком Laravel), яка надає зручний інтерфейс для виконання SQL-запитів (конструктор запитів) або використання ORM Eloquent.

Capsule Manager ініціалізується на старті додатку, завантажуючи конфігурацію з файлу database.php. Після цього моделі можуть виконувати операції CRUD (Create, Read, Update, Delete) з відповідними таблицями бази даних, використовуючи статичні методи або методи екземплярів.

Після налаштування сервера MySQL, конфігурації PHP-додатку та створення структури бази даних, працездатність системи перевіряється шляхом:

1. Спроби підключення до сервера MySQL за допомогою стороннього SQL-клієнта

(наприклад, розширення VS Code, DBeaver, HeidiSQL), використовуючи облікові дані з файлу .env. Успішне підключення та можливість перегляду структури бази даних (Рис. 2.8.) підтверджують коректність налаштувань сервера та доступність бази даних.

2. Запуску PHP-додатку та виконання операцій, що передбачають взаємодію з базою даних (наприклад, відображення списку інверторів, створення нової пропозиції). Відсутність помилок підключення та коректне відображення/збереження даних свідчать про правильну інтеграцію.

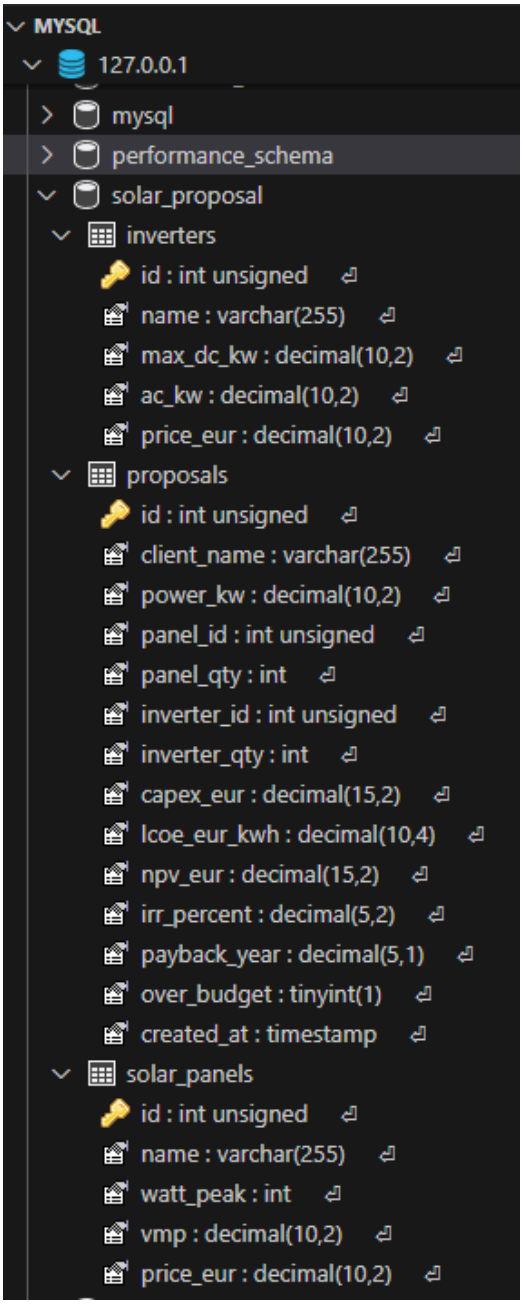


Рис. 2.8 – Відображення таблиць бази даних solar_proposal у SQL- клієнті

Налаштування підключення до MySQL у PHP-додатку включає конфігурацію сервера бази даних (часто через Docker), визначення параметрів з'єднання у файлі .env та конфігураційному файлі PHP, забезпечення наявності необхідного PHP-драйвера (pdo_mysql) та коректну реалізацію взаємодії з базою даних на рівні моделей додатку. Представлена структура бази даних (Схема 1) забезпечує логічне зберігання даних, необхідних для функціонування системи розрахунку комерційних пропозицій для сонячних електростанцій.

2.3. Математична модель розрахунку техніко-економічних показників СЕС

Побудова математичної моделі, яка лежить в основі обчислення техніко-економічних показників сонячної електростанції, базується на наборі реалістичних, але спрощених припущень, які дозволяють адаптувати складні процеси проектування до автоматизованого вебінтерфейсу. Такі припущення є необхідними для зменшення обсягу вхідної інформації, яку повинен вводити користувач, і при цьому зберігають достатню точність результатів для формування попередньої комерційної пропозиції.

У моделі передбачено, що встановлена потужність системи (вхідний параметр P_k у кВт) безпосередньо визначається користувачем у формі. Це дозволяє адаптувати конфігурацію до потреб конкретного клієнта — побутового або малого бізнесу. У якості стандартного значення, у разі відсутності введення, приймається 10 кВт [40].

Наступним ключовим припущенням є рівень інсоляції — середня річна кількість сонячної енергії, доступної на один встановлений кіловат пікової потужності (кВт$_p$). Для території України прийнято умовне значення 1 200 кВт·год/кВт$_p$. Це значення враховує середні кліматичні умови для центральної частини країни і використовується у формулі річної генерації.

Ще однією важливою складовою є оцінка системних витрат BOS (Balance

of System), до яких належать монтаж, кабельна інфраструктура, металоконструкції, щитові елементи тощо. У моделі використано спрощену формулу: «BOS = 150 €/кВт × P_k».

Це дозволяє швидко оцінити додаткову частку витрат у загальному капіталовкладенні без потреби в деталізації.

З боку користувача також вводиться бюджет проєкту ($CAPEX_{max}$) — максимально припустима сума інвестицій, що дозволяє фільтрувати конфігурації обладнання, які не перевищують доступні ресурси клієнта. У разі неможливості знайти повну конфігурацію в межах бюджету система автоматично пропонує найближчу можливу, яка лише незначно перевищує обмеження.

Додатково можуть вводитися тарифні параметри, зокрема фіксована ставка викупу електроенергії (наприклад, «зелений» тариф) або диференційовані значення для денного та нічного періодів. Це дозволяє адаптувати фінансову модель до реальних умов користувача.

Усі вхідні дані проходять валідацію та нормалізацію перед передачею в обчислювальний модуль. У табл. 2.1 подано перелік основних вхідних параметрів моделі:

Таблиця 2.1.

Основні вхідні параметри моделі

Позначення	**Назва параметра**	**Одиниці**	**Тип введення**	**Типове значення**
P_k	Встановлена потужність СЕС	кВт	Користувач вводить	10
B_{max}	Бюджет клієнта	€	Користувач вводить	15 000
E_{sol}	Середня річна інсоляція	кВт·год/$кВт_р$	Фіксоване (модель)	1 200
$C_{\beta os}$	Вартість BOS на 1 кВт потужності	€/кВт	Фіксоване (модель)	150
T_r	Тариф за продаж електроенергії	€/кВт·год	Користувач/модель	0.13

Модель ґрунтується на реалістичних техніко-економічних передумовах, які спрощують процес підготовки пропозиції, не втрачаючи аналітичної сили. Це дозволяє автоматизувати первинну оцінку інвестиційної доцільності проєкту для різних категорій споживачів.

Одним із ключових етапів техніко-економічного обґрунтування інвестицій у сонячну електростанцію є розрахунок прогнозованої річної генерації електроенергії, що визначає потенційні грошові потоки, строк окупності та екологічний ефект. У межах створеного вебзастосунку цей розрахунок автоматизовано через модуль CalculationService і реалізовано на основі простої аналітичної моделі, придатної для швидкої оцінки.

Річна генерація позначається як G_{ann} і обчислюється за формулою:

$$G_{ann} = P_{inst} \cdot E_{sol} \quad (2.1)$$

- P_{inst} – встановлена пікова потужність СЕС, кВт (вводиться користувачем),
- E_{sol} — середньорічна інсоляція, кВт·год/кВт$_р$ (приймається фіксовано як 1 200 для України),
- G_{ann} — річна генерація, кВт·год/рік.

У випадку, якщо система складається з панелей, кожна з яких має індивідуальну пікову потужність W_{panel} (Вт), і їх кількість N_{panel}, то потужність станції уточнюється як:

$$P_{inst} = \frac{N_{panel} \cdot W_{panel}}{1000} \quad (2.2)$$

Цей розрахунок реалізовано програмно, і він дозволяє виявити оптимальну кількість панелей для досягнення заданої користувачем потужності.

У вебзастосунку це формалізовано у вигляді виклику методу: «$gen = $panelQty * $panel->watt_peak / 1000 * 1200;».

Одиниці узгоджуються наступним чином:

- W_{panel} у ватах (Вт) → переводиться в кіловати;
- 1200 — інсоляція (стала для моделі), що відповідає типовим умовам Київської, Полтавської, Черкаської областей.

Цей рівень точності є достатнім для орієнтовного проєктного аналізу, який не враховує локальні метеоумови, кут нахилу панелей чи втрати в системі, але дозволяє користувачеві сформувати попереднє уявлення про економіку проєкту.

Для забезпечення адаптивності у майбутньому модель може бути легко

доопрацьована шляхом введення регіональних коефіцієнтів, сезонного профілю виробітку або втрат на тіньові зони.

У результаті отримане значення G_{ann} передається у наступні розрахунки — визначення доходу, окупності та впливу на екологію.

Для отримання цілісної техніко-економічної моделі сонячної електростанції недостатньо лише оцінити виробіток електроенергії. Не менш важливим є розрахунок витрат на реалізацію проєкту, зокрема капітальних інвестицій (CAPEX) та щорічних операційних витрат (OPEX). Саме ці показники дозволяють обчислити загальну собівартість електроенергії, строк окупності, чисту приведену вартість (NPV) тощо.

CAPEX — це загальна сума інвестицій, необхідна для побудови системи. У моделі, реалізованій у вебзастосунку, вона визначається як сума вартості основних компонентів системи:

$$C\ APEX = C_{panels} + C_{inverter} + C_{BOS} \tag{2.3}$$

де:

- $C_{panels} = N_{panel} \cdot P_{panel}$ — сукупна вартість сонячних панелей,
- $C_{inverter}$ — вартість обраного інвертора,
- $C_{BOS} = 150 \cdot P_{inst}$ — оціночна вартість компонентів BOS (кабелі, монтаж, захист, кріплення), задана як фіксована ставка 150 €/кВт.

Така модель дозволяє швидко оцінити загальні витрати, пов’язані з придбанням і встановленням СЕС, навіть без залучення детального проєктування.

OPEX — це витрати, що виникають щороку протягом усього життєвого циклу системи (наприклад, 20 років). У розрахунковій моделі використано емпіричне припущення, що щорічні операційні витрати становлять 2 % від початкового капіталовкладення:

$$OPEX = 0.02 \cdot C\ APEX \tag{2.4}$$

Це охоплює регулярне обслуговування, чистку панелей, моніторинг системи, а також незначні ремонтні роботи.

У коді реалізація подається наступним чином: «$opex = $capex * 0.02;».

Такий підхід відповідає галузевій практиці для невеликих станцій приватного або комерційного сектору та дозволяє ефективно оцінити вплив витрат на прибутковість інвестиції.

Поєднання моделі CAPEX і OPEX забезпечує основу для подальших фінансових обчислень — зокрема, розрахунку LCOE, чистої приведеної вартості (NPV), внутрішньої норми дохідності (IRR) та строку окупності (payback).

Модель зберігає точність, достатню для попередньої аналітики.

Для повноцінного економічного обґрунтування інвестицій у СЕС у рамках вебзастосунку реалізовано автоматичний розрахунок чотирьох основних фінансових показників: собівартості електроенергії (LCOE), чистої приведеної вартості (NPV), внутрішньої норми дохідності (IRR) та строку окупності. Всі вони розраховуються на основі отриманих значень капітальних/операційних витрат та генерації, описаних у попередніх пунктах.

LCOE (Levelized Cost of Electricity) Це середня приведена собівартість виробництва 1 кВт·год електроенергії протягом усього терміну експлуатації станції. Вона показує мінімальну ціну продажу електроенергії, при якій проєкт буде беззбитковим.

$$LCOE = \frac{CAPEX + OPEX \cdot T}{G_{ann} \cdot T} \qquad (2.5)$$

де:

- T — кількість років роботи (у моделі приймається 20),
- G_{ann}— річна генерація кВт·год.

NPV — це чиста приведена вартість проєкту, яка відображає різницю між приведеним доходом і витратами з урахуванням дисконтування. Якщо NPV > 0, проєкт вважається фінансово доцільним.

$$NPV = -CAPEX + \sum_{t=1}^{T} \frac{(R - OPEX)}{(1+r)^t} \qquad (2.6)$$

де:

- $R = G_{ann} \cdot$ тариф — річний дохід,
- r — ставка дисконту (у моделі: 8 % або 0.08),

- T — термін розрахунку (20 років).

NPV повертається у євро й використовується як критерій при прийнятті інвестиційного рішення.

IRR — це така ставка дисконту r∗r^*r∗, при якій NPV = 0. Вона вказує на середньорічну дохідність проєкту. У вебзастосунку обчислення IRR реалізовано методом бісекції (ітераційним), без використання зовнішніх бібліотек.

Строк окупності — це кількість років, необхідна для того, щоб накопичений чистий грошовий потік дорівнював вкладеним інвестиціям:

$$Payback = \left\lceil \frac{CAPEX}{R-OPEX} \right\rceil \quad (2.7)$$

У PHP: «$payback = ceil($capex / ($rev - $opex));», це простий показник, який наочно демонструє інвестору часовий горизонт повернення вкладених коштів.

Кожна заявка у системі автоматично супроводжується комплексною фінансовою аналітикою, яка враховує як технічні, так і економічні параметри проєкту. Завдяки цьому клієнт отримує обґрунтовану пропозицію, а адміністратор — інструмент контролю якості бізнес-моделі.

Окрім економічної доцільності, проєкти відновлюваної енергетики мають важливе екологічне значення. Одним із ключових індикаторів такого впливу є зменшення викидів парникових газів, зокрема діоксиду вуглецю (CO_2), за рахунок заміщення традиційного енергогенерування (вугілля, газ) чистою сонячною енергією. У межах вебзастосунку реалізовано спрощену модель обчислення CO_2-ефекту, яку додають до кожної сформованої пропозиції.

Розрахунок базується на стандартному галузевому коефіцієнті:

$$EF = 0.5 \text{ тонн } CO_2 \text{ / МВт\год} \quad (2.8)$$

Це усереднене значення, рекомендоване Європейською комісією та іншими міжнародними джерелами, яке враховує середній вуглецевий слід енергосистеми при генерації 1 МВт·год електроенергії. Для України діапазон коливається в межах 0.4–0.6, тож значення 0.5 вважається репрезентативним.

Екологічний ефект (CO_2-зменшення) визначається за формулою:

$$CO_2^{saved} = \frac{G_{ann}}{1000} \cdot EF \quad (2.9)$$

де:

- G_{ann} — річна генерація СЕС, кВт·год (розраховується у пункті 2),
- EF = 0.5 — коефіцієнт еквіваленту викидів.

У коді це реалізується наступним чином: @$co2 = ($production_kwh / 1000) * 0.5;». Результат подається у тоннах викидів CO_2, що не потраплять в атмосферу завдяки встановленій СЕС.

Цей показник не лише підвищує переконливість комерційної пропозиції, а й дозволяє використовувати її в маркетингових та ESG-звітах, коли клієнтом виступає бізнес, орієнтований на соціальну відповідальність.

Розрахунок CO_2-еквіваленту доповнює інші фінансово-технічні метрики, формуючи комплексний портрет ефективності інвестиції — як з погляду економіки, так і екології.

2.4. Алгоритмічне забезпечення формування комерційної пропозиції

Процес (Рис. 2.9.) формування комерційної пропозиції у вебзастосунку реалізований як послідовність взаємодій між окремими логічними компонентами: форма вводу → контролер → сервіс розрахунку → генератор документа → користувач. Весь процес є детермінованим і передбачає передачу вхідних даних, автоматичний підбір конфігурації, розрахунок фінансових метрик та підготовку PDF-файлу [42].

Початкову точку ініціює користувач, який заповнює форму на головній сторінці, вказуючи:

- ім’я клієнта;
- бажану потужність СЕС (у кВт);
- орієнтовний бюджет (€);
- (опційно) тариф або тарифну пару день/ніч.

Надіслані дані надходять до ProposalController@calculate, який:

- виконує первинну перевірку та нормалізацію;
- викликає CalculationService@makeProposal($input) для розрахунку;
- отримує готовий об’єкт Proposal;
- передає його у PdfService, який генерує фінальний PDF-документ.

Якщо розрахунок успішний, користувач бачить результат і може завантажити документ або зберегти посилання на нього.

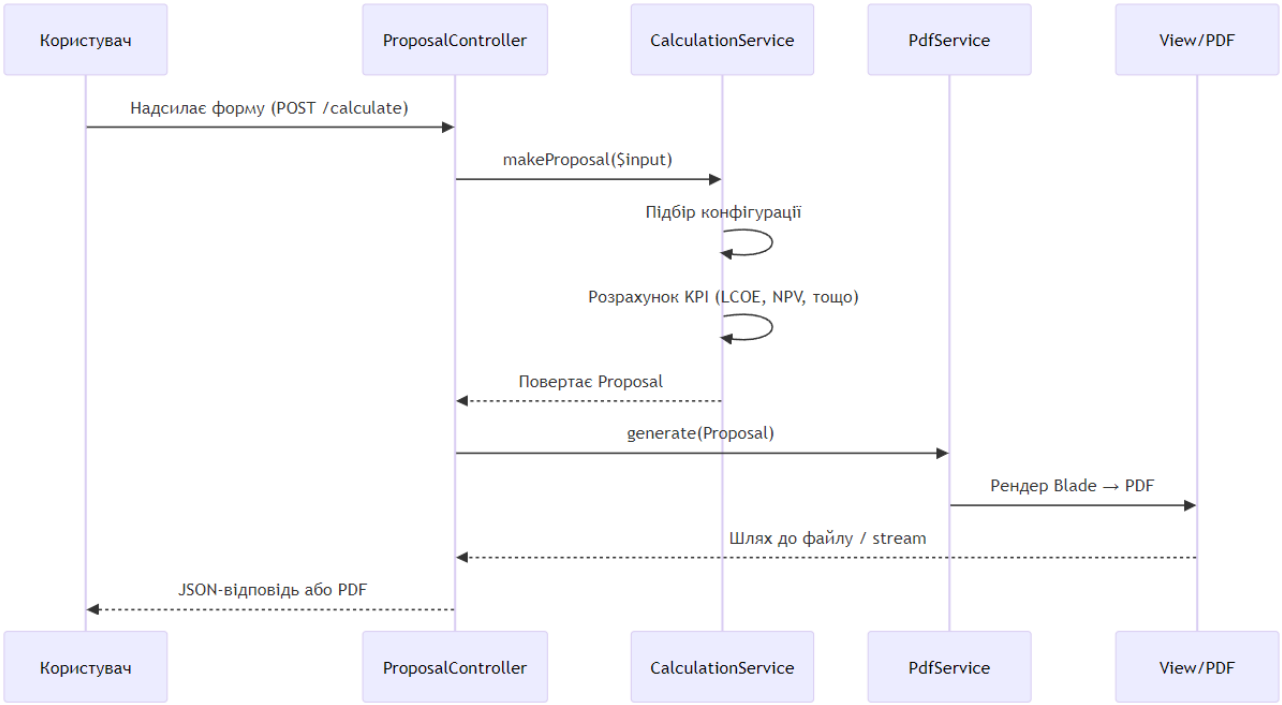


Рис. 2.9 – Послідовність формування комерційної пропозиції

Кожен з етапів працює ізольовано:

- CalculationService не взаємодіє з HTML або запитами;
- PdfService працює виключно з шаблоном і модельними даними;
- ProposalController лише координує взаємодію.

Це забезпечує чисту архітектуру і дозволяє в подальшому використовувати ті самі сервіси в інших контекстах — наприклад, у REST-API або мобільному застосунку.

У разі помилки (наприклад, неможливості знайти конфігурацію), обробка винятків здійснюється у контролері, а користувач отримує відповідне

повідомлення або підказку.

Ключовим етапом формування комерційної пропозиції є підбір оптимальної конфігурації СЕС, яка задовольняє технічні вимоги (задана потужність) і не перевищує бюджет (або перевищує мінімально). Для цього у сервісі CalculationService реалізовано алгоритм, який перебирає всі комбінації обладнання з довідників — панелей (SolarPanel) та інверторів (Inverter), — і обирає найкращу з погляду вартості.

Логіка підбору:

1. Для кожної панелі обчислюється необхідна кількість модулів, щоб сумарна потужність дорівнювала або перевищувала задану користувачем:

$$N_{panel} = \left\lceil \frac{P_{target} \cdot 1000}{W_{panel}} \right\rceil \quad (2.10)$$

2. Для кожного інвертора перевіряється, чи його максимальна потужність достатня для цієї конфігурації:

$$P_{inv}^{max} \geq P_{target} \quad (2.11)$$

3. Загальна вартість конфігурації обчислюється як:

$$CAPEX = C_{panels} + C_{inverter} + BOS \quad (2.12)$$

4. Якщо CAPEX ≤ бюджет — конфігурація зберігається як підходяща. Якщо CAPEX > бюджет — зберігається як найдешевша понад бюджет.
5. Наприкінці порівнюються дві групи:

– найкраща у межах бюджету (якщо є),
– або найдешевша понад бюджет (альтернативний варіант з позначкою over_budget = true).

Цей підхід дозволяє завжди пропонувати користувачу максимально близьку до бюджету конфігурацію, навіть якщо точне потрапляння неможливе.

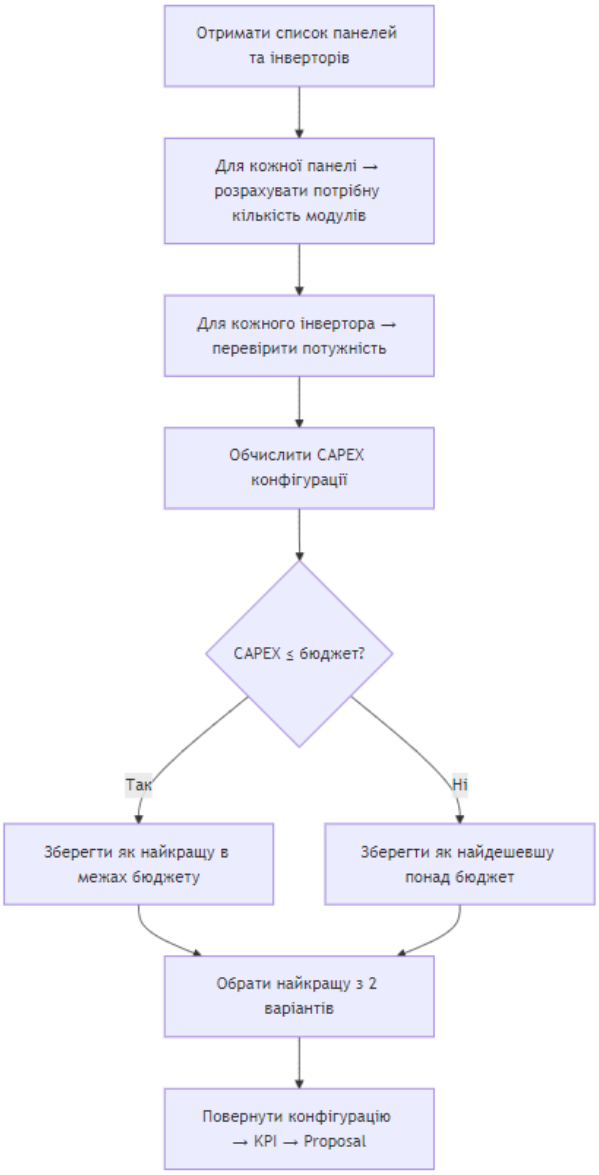


Рис. 2.10 – Логіка підбору конфігурації

У реалізації використано масив $bestWithin для варіантів у межах бюджету та $bestOver — для понад бюджет. Завдяки цьому користувач не стикається з повідомленням «немає варіантів» і завжди отримує реалістичну оцінку.

У процесі розробки алгоритмічного забезпечення вебзастосунку важливо передбачити поведінку системи у випадках, коли не вдається знайти точну відповідність між вимогами користувача та наявною конфігурацією обладнання.

У контексті підбору сонячної електростанції це означає ситуації, коли задана потужність не може бути реалізована в межах вказаного бюджету або коли в базі даних узагалі відсутні компоненти, здатні задовольнити технічні

параметри. Саме тому система побудована так, щоб не лише виконувати прямий підбір, а й забезпечувати обробку винятків та пошук найближчого (наближеного) рішення у разі відсутності ідеального варіанту [43].

Під час перебору всіх можливих комбінацій панелей та інверторів, алгоритм формує дві множини рішень: одна — це конфігурації, які повністю відповідають бюджету (тобто капітальні витрати не перевищують його), інша — ті, що трохи виходять за межі бюджету, але при цьому технічно задовольняють вимогу по потужності. У разі, якщо вдалося знайти хоча б одну конфігурацію в межах бюджету, саме вона обирається як оптимальна. Якщо ж усі варіанти перевищують бюджет, система не припиняє обчислення, а автоматично повертає найдешевший із понадбюджетних варіантів. Такий підхід дозволяє забезпечити відповідь навіть у разі жорстких обмежень з боку користувача.

Програмно це реалізується через умовний блок, у якому спочатку перевіряється наявність конфігурації у межах бюджету, і лише у разі її відсутності обирається альтернатива з позначкою over_budget = true. Цей прапорець пізніше може бути використаний для виводу попередження в інтерфейсі або в PDF-пропозиції (наприклад, текстового повідомлення "обрана конфігурація перевищує вказаний бюджет").

У випадку, якщо жоден варіант не задовольняє задану потужність — наприклад, у базі немає інверторів з достатньою вихідною потужністю — система формально припиняє обчислення і викидає виняток (RuntimeException).

Цей виняток перехоплюється на рівні контролера, який повертає користувачеві повідомлення у формі JSON з відповідним текстом помилки. Така поведінка унеможливлює генерацію хибної або порожньої пропозиції, зберігаючи логічну цілісність і достовірність даних.

Механізм обробки винятків та пошуку наближених рішень робить систему гнучкою і стійкою до обмежень користувача, дозволяючи їй залишатися корисною навіть у випадках нестачі ресурсів або обладнання. Це також створює передумови для розширення логіки в майбутньому — наприклад, для автоматичного повідомлення адміністратора про нестачу інвентаря або генерації

рекомендацій щодо оновлення бази.

Після завершення підбору конфігурації обладнання та розрахунку ключових техніко-економічних показників система переходить до побудови об’єкта, який представляє підсумкову комерційну пропозицію. У структурі вебзастосунку цей об’єкт реалізовано як екземпляр класу Proposal, що належить до простору імен App\Models. Він виступає єдиним джерелом істини для подальшого рендерингу шаблону, генерації PDF-документа або збереження інформації в базу даних [44].

Формування об’єкта Proposal виконується в межах методу makeProposal($input) класу CalculationService. Під час створення заповнюються всі ключові поля: ім’я клієнта (введене вручну), бажана потужність станції, кількість панелей, обраний інвертор, а також результати обчислень — капітальні витрати (CAPEX), собівартість кВт·год (LCOE), чиста приведена вартість (NPV), внутрішня норма дохідності (IRR) та строк окупності (Payback). Додатково додається позначка over_budget, яка вказує, чи вдалося вкластися в бюджет.

На цьому етапі об’єкт Proposal є транзитним контейнером — тобто він не обов’язково зберігається в базі, але вже готовий до передачі у шаблон Blade або до модуля PDF-генерації. У разі потреби (наприклад, при повторному перегляді) об’єкт може бути збережений у таблиці proposals, що дозволяє сформувати історію звернень або звітність.

З технічного погляду, Proposal є об’єктом із підтримкою автоматичної серіалізації у JSON або масив, що використовується для API-відповідей або передачі у шаблон. Кожне з полів має типізацію, і вся модель готова до перевірки або логічного розширення. Наприклад, додавання кількох інверторів або модулів панелей буде можливе без порушення структури об’єкта.

Факт створення об’єкта також означає завершення обчислень — після чого пропозиція може бути передана до PdfService, виведена у браузері або повернена через JSON-інтерфейс. Завдяки єдиній моделі, кожен крок роботи з результатом є уніфікованим та ізольованим від деталей реалізації обчислень.

Об’єкт Proposal виконує функцію стандартизованого підсумкового

контейнера, який поєднує в собі усю логіку конфігурації, розрахунків та презентації, слугуючи проміжною ланкою між внутрішніми сервісами та зовнішнім інтерфейсом користувача.

Після того як система побудувала об'єкт Proposal, що містить технічну конфігурацію та всі розраховані показники ефективності, наступним кроком є формування кінцевого результату, придатного для ознайомлення, збереження або друку. Цей етап відповідає за генерацію повноцінного документа у форматі PDF, який є головним артефактом взаємодії користувача з вебзастосунком.

У вебзастосунку для генерації документів використовується сервіс PdfService, який є інтерфейсом до бібліотеки Dompdf. Перевагою такого підходу є можливість створення PDF-документів безпосередньо з HTML-шаблонів, що використовують Blade — той самий шаблонізатор, який застосовується для виводу у браузері. Це забезпечує однаковість візуального стилю та дозволяє адміністратору редагувати дизайн документа незалежно від бізнес-логіки.

Процес формування PDF включає кілька послідовних кроків:

1. Рендеринг шаблону proposal_result.blade.php з передачею об’єкта Proposal.
2. Підготовка HTML — результат рендерингу.
3. Ініціалізація Dompdf з відповідними налаштуваннями (кодування UTF-8, підтримка шрифтів, дозволи на зовнішні ресурси).
4. Генерація PDF з HTML.
5. Збереження документа у файловій системі (директорія /storage/proposals/) або передача у браузер через stream().

У користувача є два сценарії отримання результату:

- Миттєвий перегляд у браузері (inline),
- Збереження файлу у файловій системі для повторного завантаження за посиланням /proposal/{id}.

Після успішної генерації пропозиція також може бути збережена в базу даних, що дозволяє реалізувати історію звернень. Завдяки такому підходу користувач отримує повноцінний документ, що включає технічні, фінансові та екологічні аспекти, та є придатним для подання у банк, інвестору або

внутрішнього погодження.

Водночас, у разі помилки (наприклад, недоступності шаблону, нестачі шрифтів або неправильної структури вхідних даних), система перехоплює виключення і повертає користувачеві інформативне повідомлення про неможливість згенерувати пропозицію.

Останній етап логіки — це перетворення розрахованої конфігурації в формальний, структурований і візуально оформлений документ, який завершує логіку користувацького запиту і є готовим до практичного використання.

2.5. Реалізація веб-інтерфейсу та бізнес-логіки

Інтерфейс користувача є ключовим елементом вебзастосунку, особливо для взаємодії з нетехнічними користувачами. У роботі використано сучасні підходи, орієнтовані на простоту й інтуїтивність без додаткового навчання.

Реалізовано адаптивний дизайн і структурний мінімалізм, завдяки використанню CSS-фреймворку Bootstrap 5, що забезпечує компонування, адаптацію до різних пристроїв та респондентне верстання [45]. Це дозволяє коректно працювати на десктопних та мобільних пристроях.

Інтерфейс умовно поділяється на дві зони: публічну (форма створення комерційної пропозиції) та адміністративну (доступ після авторизації). Обидві частини використовують спільний шаблон з елементами: шапкою <header>, зоною контенту <main> та футером <footer>.

Шапка орієнтує користувачів: ліворуч — назва застосунку з тематичною піктограмою, праворуч — кнопка входу/виходу відповідно до статусу користувача.

Основна зона містить специфічний контент: форма введення для публічної частини та табличні списки, кнопки керування для адміністративної. Така уніфікація спрощує підтримку та забезпечує єдність стилю.

Футер виконує естетичну й інформаційну функцію, позначає дипломний

характер роботи із зазначенням року й автора. Його фіксація знизу забезпечує цілісність верстання.

Усі стилі згруповані у файлі style.css, що підключається через CDN Bootstrap, що спрощує подальшу адаптацію до інших проєктів чи брендування.

Інтерфейс побудовано з урахуванням принципів доступності (accessibility) — елементи мають логічну послідовність, супроводжуються підписами, кнопки та форми використовують семантичні HTML-елементи. Шрифти, кольори та відступи підібрані так, щоб користувач не відчував перевантаження або візуального дискомфорту.

Організація інтерфейсу забезпечує зрозумілу взаємодію для клієнта, ефективне адміністрування для менеджера, а також гнучкість масштабування в межах проєкту. Прийнята структура є універсальною та відкритою для майбутнього розширення (наприклад, додавання API або мобільної версії).

Користувацький інтерфейс публічної частини вебзастосунку має бути простим, інтуїтивним і орієнтованим на одного з основних сценаріїв використання — формування персоналізованої комерційної пропозиції на основі базових вхідних параметрів. Особливістю реалізованого рішення є мінімізація необхідних дій з боку користувача: для ініціації розрахунку достатньо заповнити кілька ключових полів у формі.

Головна форма розташована по центру стартової сторінки та включає наступні обов’язкові поля:

- ім’я або назва клієнта (ідентифікаційна позначка);
- бажана встановлена потужність сонячної електростанції у кВт;
- орієнтовний бюджет на реалізацію проєкту в євро.

Опціонально можуть бути вказані тарифні умови: фіксований тариф або розділення на денний і нічний тарифи з коефіцієнтом споживання. Ці дані використовуються для більш точного фінансового прогнозування у рамках обчислювальної моделі.

Після натискання кнопки «Розрахувати» дані з форми відправляються асинхронно на маршрут /calculate, де відбувається автоматизований розрахунок

конфігурації, вибір оптимального обладнання, формування об’єкта пропозиції та генерація підсумкового PDF-документа. У разі успіху користувач отримує миттєве відображення ключових результатів (генерація, строк окупності, економія, скорочення викидів CO_2) та кнопку для перегляду або збереження PDF.

Результат подається у формі однієї сторінки, що дозволяє уникнути перевантаження інтерфейсу переходами. Якщо розрахунок неможливий (наприклад, через обмежений бюджет або відсутність відповідного обладнання), виводиться відповідне попередження з підказками.

Навігаційні елементи зведені до мінімуму: у публічній частині відображається лише кнопка входу адміністратора (у шапці) та футер з авторством.

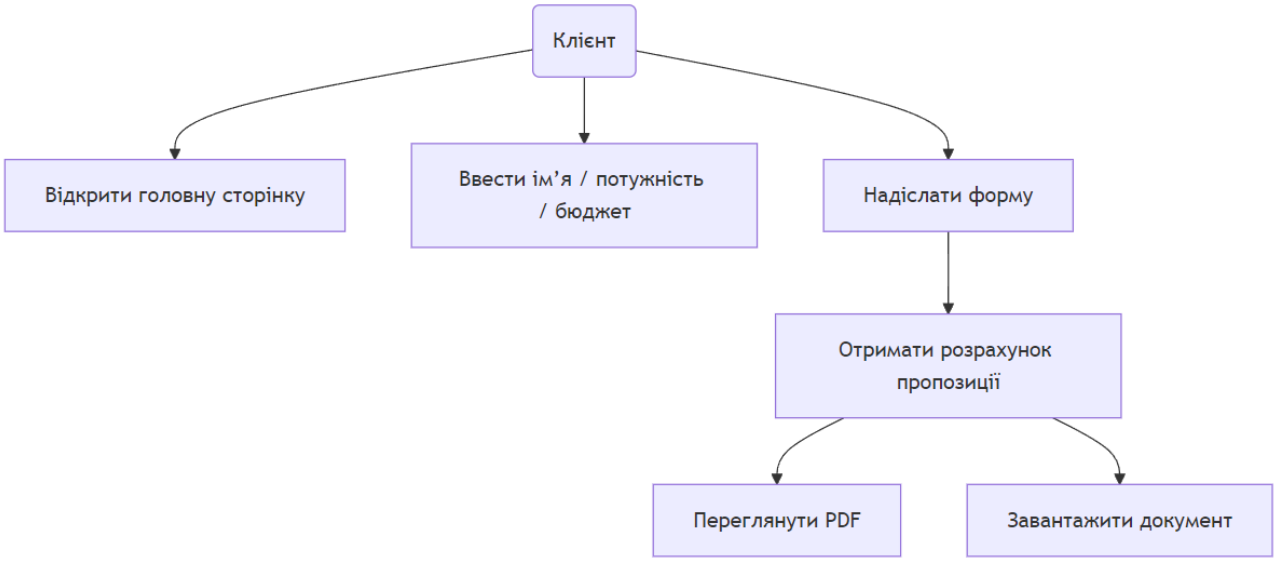


Рис. 2.11 – Use case-діаграма взаємодії клієнта із системою

Уся логіка взаємодії реалізована так, щоб процес отримання пропозиції був лінійним, без розгалужень, які могли б заплутати користувача.

Кожна дія супроводжується відповідним візуальним зворотним зв’язком — завантаженням, помилкою або підтвердженням успішного розрахунку.

Інтерфейс користувача спрямований на забезпечення доступності, швидкості і зручності, що особливо важливо при використанні системи клієнтами без технічної підготовки або безпосередньо на місці продажу.

Адміністративна частина вебзастосунку створена для забезпечення

повноцінного управління каталогом обладнання, що використовується під час автоматичного формування комерційних пропозицій. Завдяки наявності окремої панелі адміністрування, система набуває гнучкості, масштабованості й адаптивності до ринкових змін, таких як поява нових моделей інверторів або зміна цін на фотомодулі.

Доступ до адміністративного інтерфейсу є обмеженим та реалізується через механізм авторизації. Після входу за допомогою логіну та пароля адміністратор потрапляє до центральної панелі, з якої має доступ до всіх керованих ресурсів: списків інверторів, панелей, сформованих пропозицій, а також можливості вийти із системи.

Кожен довідник реалізований у вигляді таблиці з полями, які відображають ключові характеристики відповідного типу обладнання (наприклад, потужність інвертора, ефективність панелі, вартість, виробник тощо). Для кожного запису доступні дії редагування та видалення. Передбачена також кнопка «Додати новий», що відкриває форму для створення нового екземпляра обладнання з валідацією введених даних.

Усі CRUD-операції (Create, Read, Update, Delete) виконуються через вебінтерфейс, що не вимагає знання програмування або прямого доступу до бази даних. Це дозволяє адміністратору оперативно реагувати на зміни у специфікаціях та актуалізувати базу перед кожним комерційним циклом [44].

Важливим аспектом є розмежування прав доступу. Користувачі, які не пройшли автентифікацію, не мають жодного доступу до сторінок, URI яких починається з /admin. Спроба прямого переходу перенаправляє таких користувачів на сторінку входу.

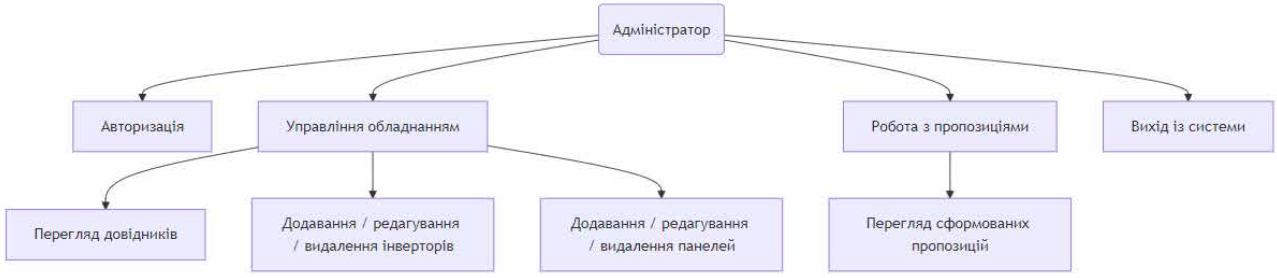


Рис. 2.12 – Use case-діаграма для адміністратора

Інтерфейс адміністратора побудовано з дотриманням принципів функціонального мінімалізму: жодних зайвих елементів, лише найнеобхідніші функції. Це забезпечує високу продуктивність взаємодії навіть у разі використання системи в умовах підвищеного навантаження (наприклад, під час виставок або консультацій клієнтів у режимі реального часу).

Адміністративна панель є інструментом налаштування та керування параметрами системи, що дозволяє підтримувати актуальність застосунку без змін у коді.

При розробці вебзастосунку особлива увага приділена користувацькому досвіду (UX): навігації, повідомленням про стан і взаємодії з елементами інтерфейсу. Навігація реалізована через єдину шапку сторінки (navbar), яка містить логотип системи, кнопку входу чи виходу адміністратора, а також за необхідності — повернення до адмін-панелі. Головна сторінка не має зайвих елементів, що спрощує навігацію для клієнта. Всі URL мають логічні REST-назви (/, /login, /admin тощо).

Реакція системи на дії користувача включає:

– спливаючі повідомлення при успішних або помилкових діях;
– статусні повідомлення після надсилання форми;
– підсвічування обов'язкових полів при помилках;
– індикатори фонового завантаження (асинхронні AJAX-запити).

Інтерфейс відповідає рекомендаціям UX: кнопки чітко підписані, елементи таблиць подані текстовими посиланнями, відстані між елементами зменшують випадкові натискання, розміри полів форм відповідають очікуваним введенням. Всі повідомлення сформульовані зрозуміло, без технічної термінології. Дизайн однорідний, з адаптивною версією для мобільних пристроїв.

Проєкт зосереджений на функціональності та візуальній чіткості, без брендових елементів, що полегшує оцінку якості UX при демонстрації або рецензуванні.

Взаємодія інтерфейсу із внутрішньою бізнес-логікою побудована за принципом розподілу відповідальності: інтерфейс збирає дані, а сервер виконує

обчислення через сервіси. Контролери обробляють запити користувачів, взаємодіють із сервісами та моделями, забезпечуючи валідацію і оновлення даних у базі. Результати повертаються структуровано (масив, JSON), відображаються через HTML або формуються у PDF-документ [45].

Інтерфейс ізольований від бізнес-логіки, що дозволяє незалежно вносити зміни в кожну частину без ризику порушення працездатності. Це забезпечує високу модульність і стабільність системи, знижуючи навантаження на кінцевого користувача та підвищуючи загальну якість продукту.

РОЗДІЛ 3. ТЕСТУВАННЯ ТА ОЦІНКА ЯКОСТІ СИСТЕМИ

3.1. Підхід до тестування

Після завершення розробки вебзастосунку для формування комерційних пропозицій щодо встановлення сонячних електростанцій (СЕС) виникає необхідність у всебічному тестуванні програмного продукту з метою виявлення та усунення помилок, забезпечення стабільності його функціонування та перевірки відповідності функціоналу вимогам, визначеним у розділі про постановку задачі. Тестування є важливим етапом життєвого циклу програмного забезпечення, оскільки дозволяє оцінити якість реалізованої системи та підтвердити її готовність до використання кінцевими користувачами — менеджерами, консультантами, адміністраторами, а також потенційними клієнтами [46].

Основною метою тестування є перевірка того, наскільки система реалізує задані функціональні та нефункціональні вимоги, зокрема: правильність розрахунку ключових показників ефективності СЕС (CAPEX, LCOE, NPV, IRR, строк окупності), стабільність збереження даних, коректне відображення користувацького інтерфейсу, безпомилкове формування PDF-документів, а також надійність механізмів аутентифікації адміністратора.

Завданням тестування також є перевірка працездатності різних сценаріїв використання, зокрема: створення нової пропозиції з різними параметрами, зміна довідників обладнання, генерація звітів, виведення історії пропозицій, а також поведінка системи у разі некоректного вводу або відсутності даних у каталозі.

Мета тестування охоплює не лише перевірку відповідності розробленого вебзастосунку специфікаціям, а й забезпечення його зручності, надійності та адаптованості до реальних умов експлуатації [47].

У межах даного дипломного проєкту було прийнято рішення застосувати комбінацію кількох типів тестування, які відповідають особливостям

архітектури вебзастосунку та характеру його функціональності. Враховуючи структуру системи, реалізовану на основі шаблону MVC (Model–View–Controller), тестування охоплює як окремі програмні компоненти, так і інтегровану роботу всієї системи в реальних умовах.

Першим типом тестування, що був застосований, стало функціональне тестування. Воно передбачає перевірку реалізації основних бізнес-вимог, таких як: створення та збереження пропозицій, розрахунок техніко-економічних показників, вибір конфігурації СЕС, авторизація адміністратора, робота з довідниками обладнання. Метою функціонального тестування є встановлення того, що кожна функція системи працює відповідно до очікуваних результатів на основі заданих вхідних даних [48].

Другим важливим напрямом є інтеграційне тестування, яке дозволяє перевірити взаємодію між модулями, зокрема: обробку форми користувача контролером, зв’язок із моделями бази даних, правильність виведення результатів у представленнях. Оскільки обробка запиту та генерація PDF-документів включає координацію кількох підсистем, інтеграційне тестування допомагає виявити помилки в логіці передачі даних між ними.

Крім того, використовувалося регресійне тестування після кожного етапу впровадження нової функції, з метою впевнитись, що нові зміни не порушили вже реалізовані можливості. Це забезпечило підтримку цілісності всієї системи упродовж розробки.

Елементи юзабіліті-тестування були використані при перевірці взаємодії з інтерфейсом користувача, доступності інформації, зрозумілості кнопок і повідомлень. Хоча це не було формалізованим тестуванням із залученням фокус-груп, зворотний зв’язок, отриманий під час перевірок, допоміг покращити зручність використання застосунку [49].

Обрані типи тестування забезпечили комплексну оцінку роботи вебзастосунку на різних рівнях — від окремих функцій до повної інтеграції системи.

Для ефективного проведення тестування вебзастосунку було створене

контрольоване середовище, яке максимально імітує умови реального розгортання. Основою для цього слугувала локальна машина розробника з операційною системою Windows 10, на якій встановлено вебсервер PHP вбудованими засобами (через команду php -S). У якості середовища виконання використовувалася версія PHP 8.4.6 [50], яка забезпечує сумісність із синтаксисом сучасного PHP та гарантує підтримку актуальних можливостей мови.

Фреймворк, що був використаний у проєкті — власноруч побудований мікрофреймворк на базі шаблону MVC із застосуванням компонента Blade для шаблонізації інтерфейсу. База даних у вигляді файлів зберігалась у форматі SQLite, що дало змогу швидко розгортати систему без зовнішніх залежностей. Усі моделі взаємодіють із базою даних через реалізовану ORM-структуру [51].

Для перегляду результатів роботи інтерфейсу та тестування користувацьких сценаріїв застосовувався браузер Google Chrome останньої версії, що дозволяло проводити перевірку відображення інтерфейсу, коректності CSS-стилів, перевірки форм та навігації. Логування активності вебсервера здійснювалося безпосередньо у консолі через PHP Development Server, де фіксувалися усі запити GET, POST, відповіді сервера з кодами 200, 302, 404 тощо.

Для перевірки правильності генерації PDF-документів використовувалася функціональність браузера, а також ручне зчитування збережених файлів. Завдяки модульності системи, тестування логіки обчислень (розрахунок LCOE, NPV, IRR тощо) здійснювалося через подачу різних вхідних даних у відповідну форму на головній сторінці застосунку [52].

Тестування відбувалося у середовищі, що повністю відображає потенційні умови експлуатації, з мінімальними сторонніми залежностями, що забезпечує стабільність і відтворюваність результатів.

Приймальне тестування (англ. acceptance testing) виступає ключовим етапом перевірки готовності системи до експлуатації користувачами. У випадку вебзастосунку для автоматизованого формування комерційних пропозицій СЕС,

критерії приймального тестування були сформульовані з урахуванням вимог до функціональності, точності розрахунків, зручності інтерфейсу та відповідності очікуванням замовника.

Одним із основних критеріїв була коректність виконання техніко-економічних розрахунків. Для цього проводилася перевірка правильності обчислення ключових показників: капіталовкладення (CAPEX), рівень LCOE, чиста приведена вартість (NPV), внутрішня норма дохідності (IRR) та строк окупності. Для підтвердження правильності застосовувалися тестові сценарії з контрольними вхідними параметрами, що мали відомі очікувані результати.

Другим критерієм була працездатність функціональності створення PDF-файлу з готовою пропозицією. Систему вважалося прийнятною, якщо PDF-файл коректно відображав усі необхідні дані, не містив порушень верстки, та був придатним для друку або надсилання клієнту.

Третім важливим аспектом було забезпечення стабільної роботи при некоректному введенні даних. Система мала попереджати користувача про некоректне значення бюджету або потужності та не допускати аварійного завершення виконання. Було перевірено реакцію на граничні значення, порожні поля, надто великі або малі числа [53].

Також тестувалася логіка взаємодії адміністратора з системою: доступ до довідників обладнання, можливість додавання, редагування, видалення панелей та інверторів. Вхід до адмін-панелі перевірявся через форму авторизації, і лише за коректним логіном і паролем надавався доступ до внутрішніх сторінок.

Останнім критерієм була перевірка сумісності інтерфейсу із сучасними браузерами та коректного масштабування елементів на різних розмірах екрана.

Це гарантувало відповідність системи основним принципам UX/UI-дизайну та її придатність до практичного використання.

Приймальне тестування було спрямоване на перевірку не лише функціональності, але й якості реалізації, стабільності, зручності, та готовності до експлуатації.

У рамках цього проєкту основним методом перевірки працездатності

системи було ручне тестування, що дозволило покроково перевірити функціональність кожного модуля, виявити логічні помилки та оцінити зручність користувацького інтерфейсу. Цей підхід був обраний через специфіку проєкту, який передбачає взаємодію користувача з візуальним середовищем, введенням параметрів та спостереженням за результатами у реальному часі.

Ручне тестування охопило кілька основних сценаріїв використання. Перш за все, перевірявся процес створення нової комерційної пропозиції: користувач переходив на головну сторінку, вводив ім’я клієнта, потужність установки та бюджет, після чого натискав кнопку розрахунку. Було перевірено, що система обробляє дані без помилок, виконує розрахунок техніко-економічних показників та виводить результат із можливістю попереднього перегляду або завантаження PDF-файлу [54].

Другим тестовим сценарієм була перевірка поведінки системи при введенні некоректних даних: порожніх полів, негативних значень, надмірно великих або нульових величин. У таких випадках система повинна відмовляти у розрахунку та повертати відповідне попередження або повідомлення про помилку. Це забезпечує захист від критичних збоїв та покращує користувацький досвід.

Окремо проводилося тестування адміністративної частини системи. Авторизація перевірялася введенням як правильних, так і неправильних облікових даних. Після успішного входу адміністратор мав змогу переглядати довідники, додавати нові позиції, редагувати існуючі та видаляти їх. Кожна дія супроводжувалася оновленням інтерфейсу та повідомленням про результат. Особливу увагу приділяли збереженню внесених змін і відсутності дублікатів.

Тестування також охопило перегляд сформованих раніше пропозицій, що зберігались у базі, з можливістю повторного відкриття PDF-документа [55]. Це дозволило переконатися в коректному збереженні та відтворенні даних з історії.

Ручне тестування проводилось поетапно, з послідовною перевіркою кожної функції системи відповідно до реальних сценаріїв використання. Такий підхід виявився ефективним у межах дипломного проєкту, дозволив виявити й

усунути помилки до передачі системи на розгляд, та забезпечив належний рівень довіри до працездатності реалізованого програмного забезпечення.

3.2. Результати тестування критичних функцій і продуктивності

Основним функціональним сценарієм використання вебзастосунку є формування нової комерційної пропозиції для сонячної електростанції на основі введених клієнтом параметрів. Цей процес охоплює декілька етапів: введення вхідних даних, розрахунок конфігурації, генерацію показників ефективності, формування структури пропозиції та вивід її у форматі PDF. У ході тестування була проведена перевірка повного циклу цього процесу з використанням реалістичних даних.

На рис. 3.1 представлено вигляд головної форми користувача після заповнення необхідних полів — імені клієнта, бажаної потужності установки та бюджету. Окремо в інтерфейсі передбачено поле для тарифного варіанту, але для спрощення перевірки використовувався базовий сценарій без диференційованих тарифів.

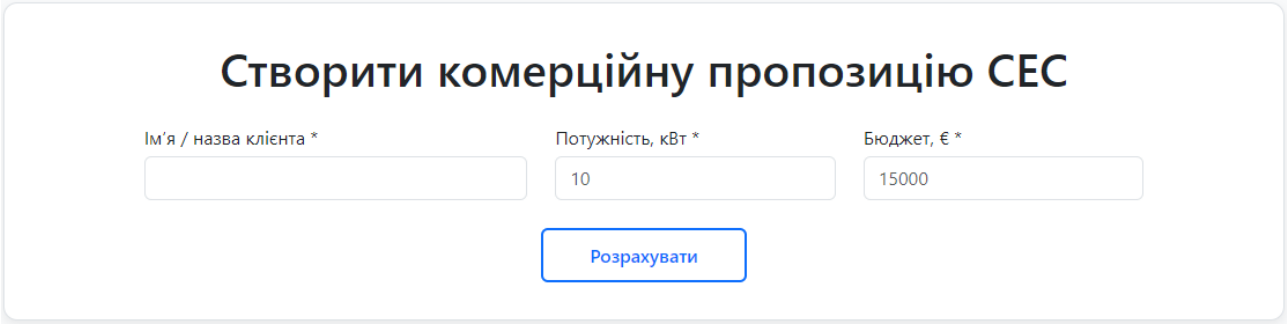
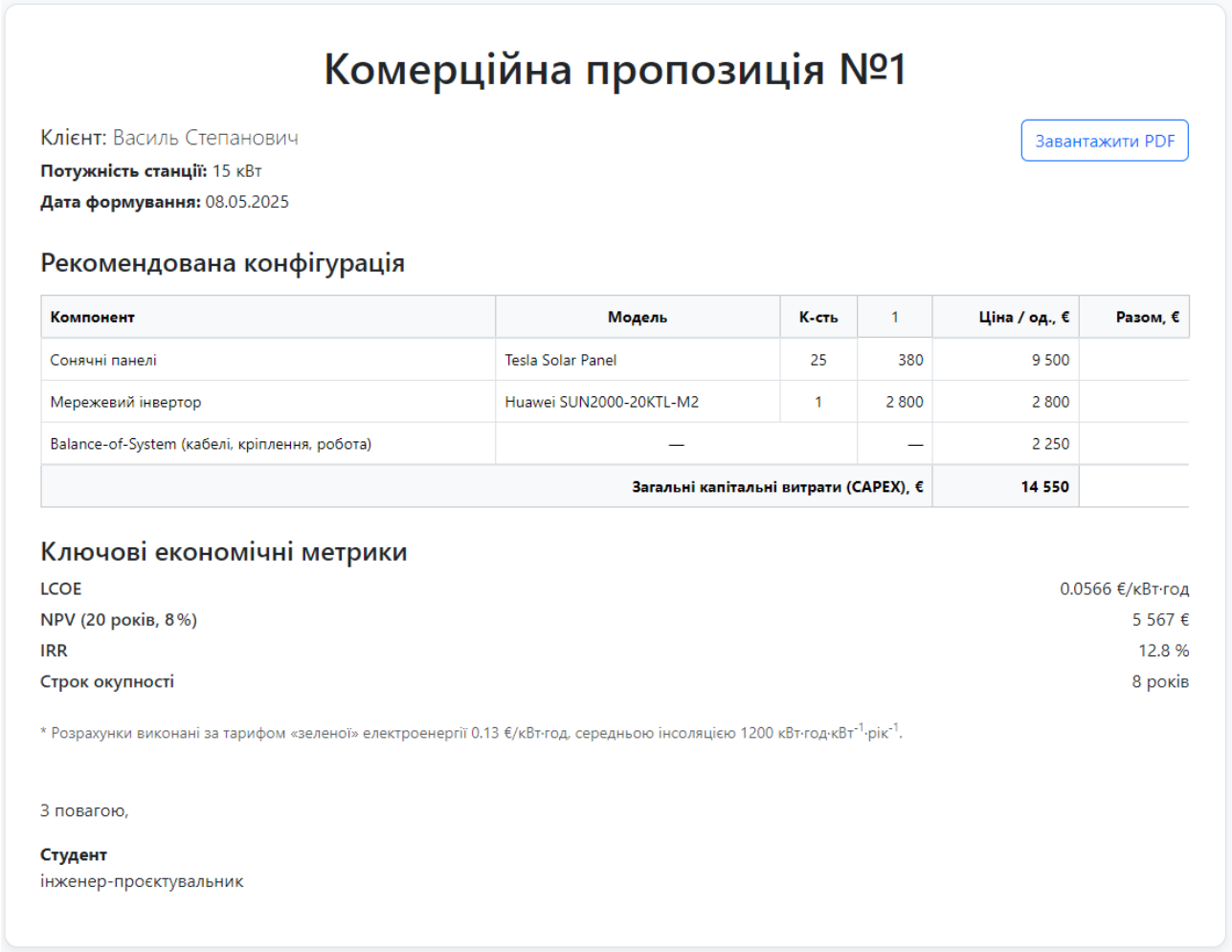


Рис. 3.1 – Форма введення параметрів комерційної пропозиції

Після натискання кнопки «Розрахувати та сформувати пропозицію» здійснюється передача даних на сервер. Система аналізує довідники обладнання, здійснює підбір відповідної конфігурації, обчислює техніко-економічні показники та повертає результат. На рис. *3.2* зображено екран з результатами —

показники річної генерації електроенергії (у кВт·год), строку окупності (у роках), економії (у євро) та скорочення викидів CO_2.



Комерційна пропозиція №1

Клієнт: Василь Степанович

Потужність станції: 15 кВт

Дата формування: 08.05.2025

Завантажити PDF

Рекомендована конфігурація

Компонент	Модель	К-сть	1	Ціна / од., €	Разом, €
Сонячні панелі	Tesla Solar Panel	25	380	9 500	
Мережевий інвертор	Huawei SUN2000-20KTL-M2	1	2 800	2 800	
Balance-of-System (кабелі, кріплення, робота)	—		—	2 250	
			Загальні капітальні витрати (CAPEX), €	**14 550**	

Ключові економічні метрики

LCOE	0.0566 €/кВт·год
NPV (20 років, 8 %)	5 567 €
IRR	12.8 %
Строк окупності	8 років

* Розрахунки виконані за тарифом «зеленої» електроенергії 0.13 €/кВт·год, середньою інсоляцією 1200 кВт·год·кВт$^{-1}$·рік$^{-1}$.

З повагою,

Студент
інженер-проєктувальник

Рис. 3.2 – Результати розрахунку комерційної пропозиції

Водночас система дозволяє переглянути та завантажити пропозицію у форматі PDF, що може бути збережена, надіслана клієнту або використана для подальшого оформлення. На *рис. 3.3* продемонстровано зовнішній вигляд згенерованої PDF-пропозиції. Як видно, вона містить ім’я клієнта, обрану конфігурацію, кількість обладнання, економічні показники та коротку аналітику — усе на одній сторінці.

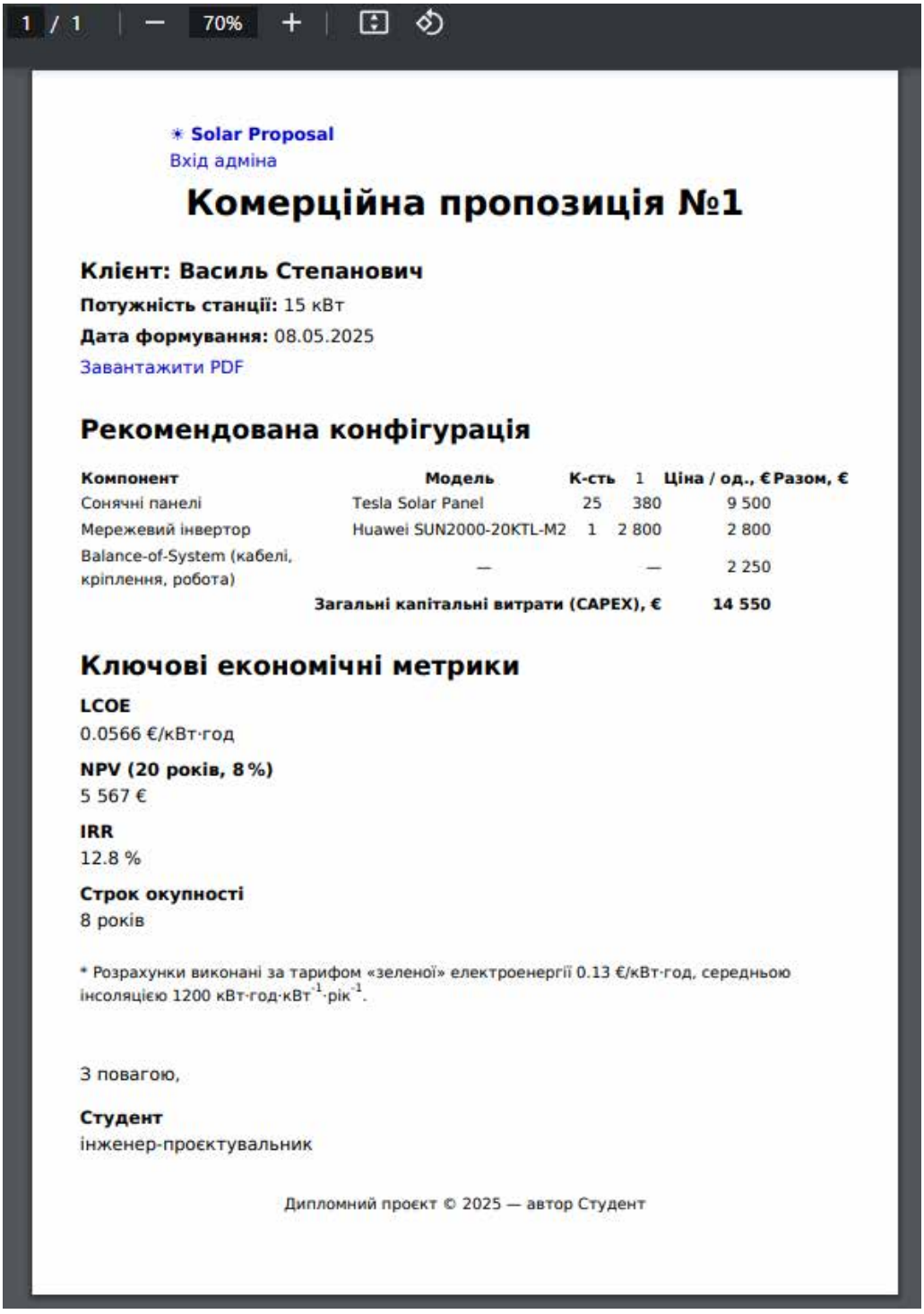


Рис. 3.3 – Попередній перегляд PDF-документа комерційної пропозиції

Результати тестування підтвердили працездатність механізму генерації пропозицій: усі поля обробляються коректно, а результат повертається без помилок за умов, коли в довідниках наявне відповідне обладнання. PDF-документ формується стабільно, підтримує українські шрифти та масштабований макет.

Базовий сценарій використання — від введення даних до формування готової пропозиції — виконується повністю автоматизовано та без збоїв, що свідчить про правильність реалізації основної бізнес-логіки вебзастосунку.

Функціональна можливість збереження сформованих комерційних пропозицій є критично важливою складовою роботи вебзастосунку. Вона дозволяє зберігати кожну сформовану конфігурацію в базі даних, забезпечуючи можливість подальшого перегляду, повторного відкриття PDF-документа або ведення історії взаємодій з клієнтами. У процесі тестування було перевірено правильність запису об’єкта Proposal, його збереження у базі та подальше відображення в інтерфейсі.

На рис. 3.4 представлено сторінку «Історія сформованих пропозицій». У цьому розділі автоматично відображаються всі пропозиції, що були створені через головну форму. Для кожного запису зберігаються ключові параметри — ім’я клієнта, дата створення, потужність СЕС, а також посилання на перегляд PDF-документа.

Рис. 3.4 – Відображення історії сформованих пропозицій

Після натискання на посилання «Переглянути PDF» система відкриває відповідний документ, який був згенерований під час створення пропозиції. Як показано на рис. 3.3, PDF-документ повністю відповідає первинній версії — збережено верстку, значення всіх розрахованих показників та індивідуальні параметри конфігурації.

Цей механізм дозволяє не лише забезпечити зручність повторного доступу до результатів, але й підвищити професійність роботи з клієнтами. У реальному

сценарії менеджер або інсталятор може повернутися до раніше сформованої пропозиції без необхідності повторного введення даних, що економить час і знижує ймовірність помилок.

Крім того, при повторному відкритті PDF-файлів система використовує збережену версію без повторного обчислення, що забезпечує стабільність даних незалежно від подальших змін у довідниках обладнання або тарифах.

Механізм збереження і повторного доступу до комерційних пропозицій є важливою ознакою надійності системи та її готовності до використання в умовах реального бізнес-процесу.

Адміністративна панель вебзастосунку є окремим функціональним модулем, що забезпечує повний контроль над каталогами обладнання: інверторами, сонячними панелями та іншими компонентами системи. Її правильна робота критично важлива для забезпечення актуальності розрахунків та відповідності комерційних пропозицій наявному асортименту.

У межах тестування було перевірено доступ до адмін-панелі через форму авторизації, що відображається у правій частині шапки на головній сторінці (рис. 3.5). Авторизація потребує введення логіна й пароля адміністратора, після чого відкривається доступ до сторінки керування.

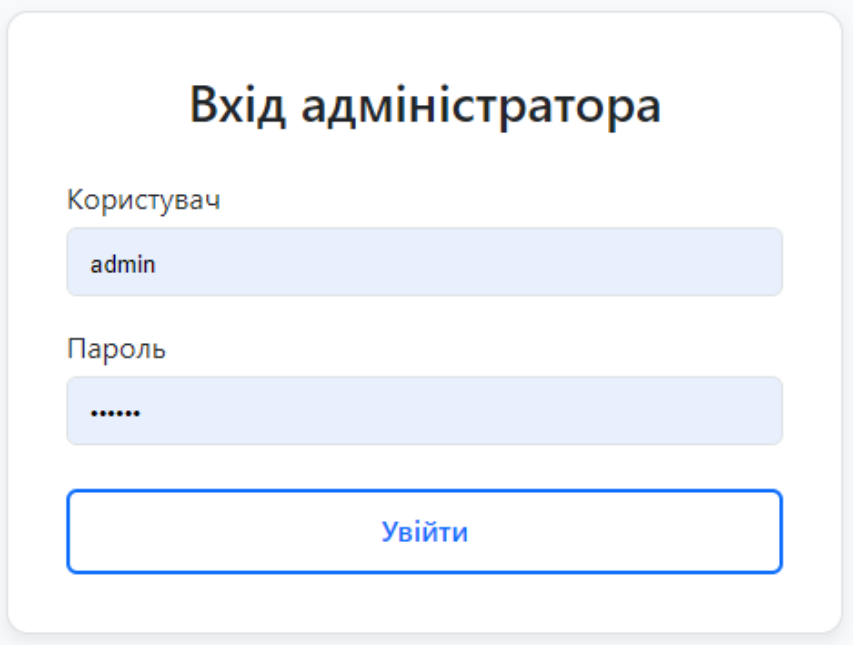


Рис. 3.5 – Вхід адміністратора через окрему кнопку в навігації

Після авторизації адміністратор потрапляє до адмін-панелі, яка дозволяє виконувати CRUD-операції над довідниками (створення, перегляд, редагування, видалення). На рис. 3.6 представлено таблицю наявних інверторів. Для кожного з них відображаються технічні характеристики, зокрема потужність (DC/AC), максимальне навантаження, виробник, ціна, а також кнопки «Редагувати» і «Видалити», які реалізовані у вигляді текстових посилань.

Рис. 3.6 – Таблиця інверторів у адміністративній частині

Особливу увагу було приділено зручності інтерфейсу: кнопки не перекривають одна одну, поля вирівняні, а вся таблиця адаптована до ширини екрана. Валідація форм редагування та створення також була протестована — система повідомляє про помилки у разі некоректного або порожнього введення.

Тестування адміністративної частини підтвердило, що система дозволяє зручно та безпомилково керувати технічними параметрами, від яких залежить точність комерційних розрахунків. Роль адміністратора чітко відокремлена від ролі клієнта, а інтерфейс реалізовано з урахуванням принципів безпеки й ефективності.

Крім перевірки стандартних сценаріїв, тестування вебзастосунку передбачало також імітацію нестандартних ситуацій, які можуть виникнути під час взаємодії з системою в реальних умовах. Йдеться про випадки, коли користувач вводить неповні або некоректні дані, подає запит при відсутності обладнання, або коли бюджет значно менший за мінімальну конфігурацію.

Метою такого тестування було переконатись, що система не лише не допускає критичних помилок, а й чітко інформує користувача про суть проблеми.

На рис. 3.7 показано приклад ситуації, коли користувач подав заявку з надто низьким бюджетом. У відповідь система виводить повідомлення про неможливість підібрати конфігурацію. Текст сформульовано зрозумілою мовою без технічних термінів, що дозволяє користувачеві самостійно скоригувати параметри.

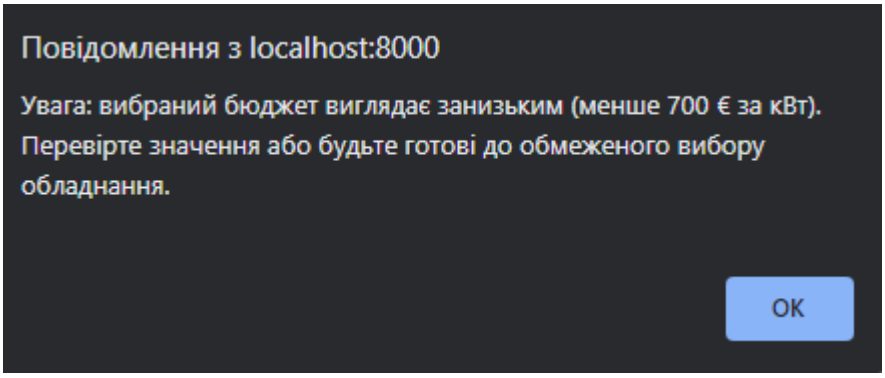


Рис. 3.7 – Повідомлення про відсутність конфігурацій у межах бюджету

Ще одним важливим сценарієм є випадок, коли довідник обладнання порожній або не містить жодної моделі, яка покриває задану потужність. У цьому разі система не допускає формування пропозиції, а натомість повертає відповідне повідомлення.

Система демонструє високий рівень толерантності до помилок користувача: вона не лише утримується від аварійного завершення виконання, а й забезпечує інформування з поясненням причин та рекомендаціями. Це сприяє покращенню користувацького досвіду та відповідає вимогам до надійності сучасних вебсистем.

У межах оцінювання продуктивності вебзастосунку здійснено спостереження за обробкою HTTP-запитів під час реального використання системи. Для цього було використано вбудований сервер PHP, запущений у режимі відладки, що дозволяло відслідковувати всі вхідні запити, їх тип, маршрут та відповідний HTTP-статус.

На рис. 3.8 представлено консольний лог сервера, який фіксує послідовність подій при відкритті головної сторінки, подачі розрахунку, перегляді результатів та PDF-версії комерційної пропозиції. Особливу увагу слід звернути на такі аспекти:

- Всі GET- та POST-запити обробляються коректно — сервер повертає статус [200] OK, що вказує на успішну відповідь;
- Редіректи (наприклад, після розрахунку) позначені статусом [302] Found, що свідчить про реалізовану логіку перенаправлення;
- Динамічні маршрути, як-от /proposal/1 і /proposal/1?pdf=1, обробляються швидко й без помилок, навіть коли підключається модуль генерації PDF;
- Лише запит до /favicon.ico повертає [404] Not Found, що є несуттєвою втратою, оскільки не впливає на бізнес-логіку або працездатність системи.

```
PS C:\Users\Apex\Desktop\solar-proposal-app> php -S localhost:8000 -t public
[Thu May  8 17:58:49 2025] PHP 8.4.6 Development Server (http://localhost:8000) started
[Thu May  8 17:58:52 2025] [::1]:62420 Accepted
[Thu May  8 17:58:52 2025] [::1]:62421 Accepted
[Thu May  8 17:58:52 2025] [::1]:62420 [200]: GET /
[Thu May  8 17:58:52 2025] [::1]:62420 Closing
[Thu May  8 17:58:52 2025] [::1]:62422 Accepted
[Thu May  8 17:58:52 2025] [::1]:62421 [200]: GET /assets/css/style.css
[Thu May  8 17:58:52 2025] [::1]:62422 [200]: GET /assets/js/app.js
[Thu May  8 17:58:52 2025] [::1]:62421 Closing
[Thu May  8 17:58:52 2025] [::1]:62422 Closing
[Thu May  8 17:58:52 2025] [::1]:62423 Accepted
[Thu May  8 17:58:52 2025] [::1]:62423 [404]: GET /favicon.ico
[Thu May  8 17:58:52 2025] [::1]:62423 Closing
[Thu May  8 17:59:18 2025] [::1]:62435 Accepted
[Thu May  8 17:59:18 2025] [::1]:62436 Accepted
[Thu May  8 17:59:19 2025] [::1]:62435 [302]: POST /calculate
[Thu May  8 17:59:19 2025] [::1]:62435 Closing
[Thu May  8 17:59:19 2025] [::1]:62436 [200]: GET /proposal/1
[Thu May  8 17:59:19 2025] [::1]:62436 Closing
[Thu May  8 17:59:19 2025] [::1]:62439 Accepted
[Thu May  8 17:59:19 2025] [::1]:62439 [200]: GET /assets/css/style.css
[Thu May  8 17:59:19 2025] [::1]:62440 Accepted
[Thu May  8 17:59:19 2025] [::1]:62440 [200]: GET /assets/js/app.js
[Thu May  8 17:59:19 2025] [::1]:62439 Closing
[Thu May  8 17:59:19 2025] [::1]:62440 Closing
[Thu May  8 17:59:34 2025] [::1]:62447 Accepted
[Thu May  8 17:59:34 2025] [::1]:62448 Accepted
[Thu May  8 17:59:35 2025] [::1]:62447 [200]: GET /proposal/1?pdf=1
[Thu May  8 17:59:35 2025] [::1]:62447 Closing
[Thu May  8 17:59:51 2025] [::1]:62457 Accepted
[Thu May  8 17:59:51 2025] [::1]:62448 [200]: GET /proposal/1
[Thu May  8 17:59:51 2025] [::1]:62448 Closing
[Thu May  8 17:59:52 2025] [::1]:62458 Accepted
[Thu May  8 17:59:52 2025] [::1]:62457 [200]: GET /login
[Thu May  8 17:59:52 2025] [::1]:62457 Closing
[Thu May  8 17:59:52 2025] [::1]:62458 [200]: GET /assets/css/style.css
[Thu May  8 17:59:52 2025] [::1]:62459 Accepted
[Thu May  8 17:59:52 2025] [::1]:62459 [200]: GET /assets/js/app.js
```

Рис. 3.8 – Консоль сервера під час тестування швидкодії

Система демонструє стабільну роботу під навантаженням у межах типового сценарію використання. Всі ключові маршрути працюють без затримок або внутрішніх помилок, що дозволяє зробити висновок про належну реалізацію маршрутизатора та логіки обробки запитів.

3.3. Ергономічні вимоги до робочого місця розробника та безпека роботи з ПК

Сучасна професійна діяльність розробника програмного забезпечення передбачає тривале перебування за комп’ютером, що вимагає дотримання чітко визначених ергономічних вимог до робочого місця. Невідповідність цих вимог може призводити до фізичного дискомфорту, хронічної втоми, зниження продуктивності та розвитку професійних захворювань. Тому важливим етапом при організації робочого середовища є створення умов, що відповідають як фізіологічним особливостям користувача, так і нормативним санітарним стандартам.

Робоче місце розробника повинно включати ергономічний стіл, крісло, комп’ютерну техніку та аксесуари, розміщені з урахуванням природного положення тіла під час роботи. Висота столу має становити 72–75 см, що відповідає зручному положенню передпліч при наборі тексту. Крісла повинні мати регульовану спинку, підлокітники, змінну висоту та наявність підтримки поперекового відділу хребта. Це забезпечує правильне положення хребта, знижує статичне навантаження на м’язи спини та шиї [56].

Монітор має бути розміщений на відстані 50–70 см від очей, верхній край екрана — на рівні горизонтальної лінії погляду або трохи нижче. Такий підхід запобігає перенапруженню зорового апарату та зменшує ймовірність розвитку синдрому «сухого ока». Кут нахилу екрана має бути змінним, щоб уникнути відблисків та забезпечити комфортне зчитування інформації.

Клавіатура розташовується на висоті, що дозволяє тримати передпліччя горизонтально, а зап’ястки — у прямому положенні. Рекомендується використання підставок для рук, які знижують ризик виникнення тунельного синдрому. Миша повинна знаходитися на одній висоті з клавіатурою, бажано з підтримкою зап’ястка.

Робоча зона має бути організована таким чином, щоб всі необхідні предмети знаходилися в межах досяжності руки, з мінімальною потребою в

обертанні тулуба або тривалому нахилі. Простір під столом повинен бути вільним для розміщення ніг без перешкод. Рекомендується використання підставки для ніг при роботі у сидячому положенні понад 4 години на день.

Дотримання ергономічних принципів при організації робочого місця не лише підвищує рівень комфорту працівника, а й сприяє збереженню здоров'я та покращенню продуктивності, що є особливо важливим для представників інтелектуальної праці, зокрема в галузі ІТ [57].

Згідно з вимогами ДСанПіН 3.3.2-007-98, тривалість безперервної роботи з ПК не повинна перевищувати 2 години поспіль. Після цього необхідно робити технологічні перерви тривалістю 10–15 хвилин. При загальному навантаженні понад 6 годин на день рекомендується встановлення двох регламентованих перерв по 15 хвилин. Під час перерв бажано виконувати вправи для зняття статичного напруження, зміну положення тіла або гімнастику для очей [57].

Освітлення має відповідати норми освітленості не нижче 300 лк, без мерехтіння, з переважанням нейтрального білого спектру (4000–5000 К). Робоче місце повинно мати достатній рівень природного освітлення в денний час, а джерела штучного світла — не створювати тіней або відблисків на екрані. Перевага надається світильникам із дифузним розсіюванням світла та можливістю регулювання яскравості.

Робоче середовище повинно бути вільним від надмірного шуму, вібрацій і візуальних подразників. Рівень шуму не повинен перевищувати 50 дБА — це забезпечує комфортну концентрацію під час інтелектуальної діяльності. Рекомендується розміщення робочих місць на відстані не менше 1,5 м один від одного у разі використання відкритих офісів. Дотримання санітарно-гігієнічних норм є ключовим чинником у профілактиці професійного перевантаження працівників ІТ-сфери. Воно забезпечує фізіологічну підтримку працездатності, знижує ризик розвитку хронічних захворювань і формує безпечне середовище для виконання високоточних і творчих завдань.

Першим кроком до покращення умов праці є використання ергономічних аксесуарів. До них належать підставки під ноутбуки, кронштейни для моніторів,

ортопедичні килимки під зап’ястки, підставки для ніг, анатомічні крісла та регульовані столи. Застосування таких елементів дозволяє адаптувати робоче місце під індивідуальні анатомічні особливості працівника, знижуючи ризик розвитку опорно-рухових розладів і перенапруження.

Важливо також впроваджувати програмне забезпечення для моніторингу режиму роботи. Це можуть бути додатки, які нагадують про перерви, пропонують короткі фізичні вправи або попереджають про перенавантаження. Наприклад, утиліти типу Stretchly, EyeLeo чи TimeOut допомагають реалізовувати техніку «помідора» або зберігати гігієнічні перерви щогодини. Це позитивно впливає на концентрацію та довготривалу увагу [58].

Створення умов праці, орієнтованих на людину, забезпечує високий рівень безпеки, зниження ризиків та зростання продуктивності, що особливо актуально у сфері ІТ, де інтелектуальне навантаження і когнітивна втома є постійним викликом.

ВИСНОВКИ

У ході виконання дипломного проєкту було розроблено та реалізовано вебзастосунок для формування комерційної пропозиції сонячної електростанції (СЕС), що дозволяє автоматизувати процес підготовки індивідуалізованих техніко-економічних рішень на основі заданих параметрів клієнта. Проєкт охоплює повний цикл створення інформаційної системи — від постановки задачі та аналізу предметної області до технічного проєктування, реалізації, тестування та оцінки якості.

У теоретичній частині роботи було досліджено сучасні тенденції розвитку ринку сонячної енергетики, структуру типової СЕС, ключові показники її ефективності, нормативно-правові вимоги та потреби потенційних клієнтів. Проведено аналіз існуючих програмних рішень, що дозволило сформулювати обґрунтовані функціональні та нефункціональні вимоги до системи.

В рамках проєктної частини здійснено обґрунтований вибір технологічного стеку, побудовано архітектуру веб застосунку та реалізовано окремі компоненти: інтерфейс користувача, бізнес-логіку, математичну модель розрахунків, алгоритми вибору оптимальної конфігурації обладнання та генерацію пропозицій у форматі PDF. Система включає також адміністративний модуль для управління довідниками обладнання та налаштувань.

Особливу увагу приділено забезпеченню якості, що підтверджено результатами функціонального та візуального тестування. Проведена оцінка продуктивності свідчить про ефективність і стабільність роботи застосунку у звичайних користувацьких сценаріях. В розділі безпеки життєдіяльності описано вимоги до охорони праці, ергономіки, протипожежного захисту та екологічних аспектів використання СЕС і програмного забезпечення.

Поставлена мета — створити практично застосовний веб застосунок для формування комерційної пропозиції СЕС — повністю досягнута. Розроблена система відповідає вимогам сучасного ринку, має зручний інтерфейс, точну аналітичну модель та здатна забезпечити високий рівень автоматизації в процесі

підготовки техніко-економічного обґрунтування для клієнтів.

Отримані результати мають прикладне значення та можуть бути використані в діяльності підприємств, що працюють у сфері проєктування та продажу сонячних електростанцій. Перспективи подальшого розвитку проєкту полягають у розширенні бази обладнання, інтеграції з картографічними сервісами, реалізації багатомовної підтримки та впровадженні мобільної версії застосунку.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Про альтернативні джерела енергії : Закон України від 20.02.2003 р. № 555-IV. URL: https://zakon.rada.gov.ua/laws/show/555-15 (дата звернення: 08.05.2025).
2. Про ринок електричної енергії : Закон України від 13.04.2017 р. № 2019-VIII. URL: https://zakon.rada.gov.ua/laws/show/2019-19 (дата звернення: 08.05.2025).
3. Про внесення змін до деяких законів України щодо відновлення та зеленої трансформації енергетичної системи України : Закон України від 21.07.2023 р. (Законопроект 9011-д). URL: https://eba.com.ua/zakonoproekt-9011-d-shhodo-vidnovlennya-ta-zelenoyi-transformatsiyi-energetychnoyi-systemy-ukrayiny/ (дата звернення: 08.05.2025).
4. Національна комісія, що здійснює державне регулювання у сферах енергетики та комунальних послуг. Про встановлення «зелених» тарифів на електричну енергію, вироблену генеруючими установками приватних домогосподарств : Постанова від 25.03.2025 р. № 465. URL: https://www.nerc.gov.ua/acts/pro-vstanovlennya-zelenih-tarifiv-na-elektrichnu-energiyu-viroblenu-generuyuchimi-ustanovkami-privatnih-domogospodarstv-12 (дата звернення: 08.05.2025).
5. Державне агентство з енергоефективності та енергозбереження України. Публічний звіт Голови Держенергоефективності про підсумки діяльності у 2022 році. URL: https://saee.gov.ua/sites/default/files/Valerii_Bezus_SAEE_zvit_2022.pdf (дата звернення: 08.05.2025).
6. Державне агентство з енергоефективності та енергозбереження України. Плани та звіти. URL: https://saee.gov.ua/spivrobitnytstvo-z-derzhenerhoefektyvnosti/plany-ta-zvity (дата звернення: 08.05.2025).
7. Сонячна енергетика України. Вікіпедія. URL: https://uk.wikipedia.org/wiki/%D0%A1%D0%BE%D0%BD%D1%8F%D1%87%D0%BD%D0%B0_%D0%B5%D0%BD%D0%B5%D1%80%D0%B3%D0%B5%D1%82%D0%B8%D0%BA%D0%B0_%D0%A3%D0%BA%D1%80%D0%B0%D1%97%D0%BD%D0%B8 (дата звернення: 08.05.2025).
8. Інститут відновлюваної енергетики НАН України. Журнал "Відновлювана енергетика". URL: https://www.ive.org.ua/?page_id=185&lang=uk (дата звернення: 08.05.2025).

9. Державна служба статистики України. Енергетичний баланс України. URL: https://stat.gov.ua/uk/datasets/enerhetychnyy-balans-ukrayiny-0 (дата звернення: 08.05.2025).
10. Які тенденції розвитку сонячної енергетики в Україні у 2024 році. SVOYA energy. URL: https://svoya-energy.com.ua/tendentsiyi-rozvytku-sonyachnoyi-energetyky-v-ukrayini-u-2024-rotsi/ (дата звернення: 08.05.2025).
11. Енергетичний сектор України. UkraineInvest. URL: https://ukraineinvest.gov.ua/wp-content/uploads/2024/08/energy-14.08-ua.pdf (дата звернення: 08.05.2025). (Примітка: дата у URL може бути прикладом, перевірте актуальність звіту на сайті).
12. Сонячна енергетика – середньострокові перспективи 2022-2026. AW-Therm. URL: https://aw-therm.com.ua/sonyachna-energetika-serednostrokovi-perspektivi-2022-2026/ (дата звернення: 08.05.2025).
13. Галузь сонячної енергетики в України. ТЕПЛА енергетична компанія. URL: https://tepla.com.ua/galuz-sonyachnoyi-energetiki-v-ukrayini/ (дата звернення: 08.05.2025).
14. Сонячні електростанції в Україні: потужність, розвиток, перспективи. EcoTech Ukraine. URL: https://www.ecotech.ua/sonyachni-elektrostancziyi-v-ukrayini-suchasnyj-stan-perspektyvy-ta-tehnologiyi/ (дата звернення: 08.05.2025).
15. Коваль В. Р. та ін. Інноваційні проєкти для економічного відродження та конкурентного розвитку України. Київський національний економічний університет імені Вадима Гетьмана. URL: https://ir.kneu.edu.ua/bitstreams/00ab5583-515c-4182-ad33-c0116ae583d6/download (дата звернення: 08.05.2025). (Згадується сонячна енергетика).
16. Збірник тез доповідей III Міжнародної науково-практичної конференції (26 квітня, 2024 р.). Західноукраїнський національний університет. Тернопіль, 2024. 310 с. URL: https://www.wunu.edu.ua/pdf/conference/2024/ME_2024.pdf (дата звернення: 08.05.2025). (Пошук статей за тематикою ВДЕ).
17. Сектор відновлюваної енергетики України до, під час та після війни. Центр Разумкова. URL: https://razumkov.org.ua/statti/sektor-vidnovlyuvanoyi-energetyky-ukrayiny-do-pid-chas-ta-pislya-viyny (дата звернення: 08.05.2025).
18. Сонячна енергетика: показники ринку. ProfBuild. URL: https://profbuild.in.ua/uk/vsi-statti-zhurnala-prof-build/6501-sonyachna-energetika-pokazniki-rinku (дата звернення: 08.05.2025).

19. Міністерство енергетики України. Звіт про стратегічну екологічну оцінку проєкту Водневої стратегії України на період до 2050 року. Київ, 2024. URL: https://www.mev.gov.ua/sites/default/files/field/file/zvit-pro-seo.pdf (дата звернення: 08.05.2025). (Містить дані про ВДЕ).
20. Державна науково-технічна бібліотека України. Сонячна енергетика. Сонячні батареї : бібліографічний покажчик (2018–2021 рр.). Київ, 2021. URL: https://dntb.gov.ua/wp-content/uploads/2022/01/7.pdf (дата звернення: 08.05.2025).
21. Інститут відновлюваної енергетики НАН України. Науковці Академії передали українським захисникам автономно-переносну сонячну електростанцію (30.03.2022). Національна академія наук України. URL: https://www.google.com/search?q=https://www.old.nas.gov.ua/UA/Messages/Pages/View.aspx%3FMessageID%3D8828 (дата звернення: 08.05.2025). (Як приклад діяльності інституту).
22. Енергетичний потенціал сонячної радіації на території України. Львівська політехніка. URL: https://science.lpnu.ua/sites/default/files/journal-paper/2017/jun/4199/21204.pdf (дата звернення: 08.05.2025). (Хоча стаття 2017 року, дані можуть бути корисними для аналізу потенціалу).
23. Доцільність використання сонячних панелей: переваги та недоліки. Вінницький національний технічний університет. URL: https://ir.lib.vntu.edu.ua/bitstream/handle/123456789/28772/8993.pdf?sequence=3&isAllowed=y (дата звернення: 08.05.2025).
24. Характеристика сонячної енергетики в Україні та світі. Вінницький національний технічний університет. URL: https://iq.vntu.edu.ua/repository/getfile.php/4151.pdf (дата звернення: 08.05.2025).
25. ДСТУ 8302:2015. Інформація та документація. Бібліографічне посилання. Загальні положення та правила складання. (Вам потрібно буде знайти повний текст стандарту через офіційні джерела або бібліотеки вашого ЗВО, оскільки він є основою для оформлення списку літератури). Приклади оформлення можна знайти на сайтах бібліотек, наприклад: Сайт: як оформлювати посилання за ДСТУ 8302:2015? Grafiati. URL: https://www.grafiati.com/uk/info/dstu-8302-2015/website/ (дата звернення: 08.05.2025).
26. Мельник Р. А. Програмування WEB-серверів та WEB-клієнтів : навчальний посібник. Вінниця : ВНТУ, 2018. 248 с. URL: http://pdf.lib.vntu.edu.ua/books/2024/Melnyk_2018_248.pdf (дата звернення: 08.05.2025). (Стосується PHP та серверної розробки).

27. Балик Н. Р., Мандзюк В. І. Бази даних MySQL: Навчальний посібник. Тернопіль : Навчальна книга – Богдан, 2010. 160 с. URL: https://bohdan-books.com/upload/iblock/ebe/ebe73ff67804bc6dc1f6f464a3ca4469.pdf (дата звернення: 08.05.2025).
28. Трофименко О. Г., Задерейко О. В., Плачінда О. Є. Веб-технології та веб-дизайн : навчальний посібник. Херсон : вид-во ФОП Вишемирський В.С., 2019. 284 с. URL: https://document.kdu.edu.ua/info_zab/061_75.pdf (дата звернення: 08.05.2025). (Охоплює HTML, CSS, основи веб-розробки).
29. Лавріщева К. М. Програмна інженерія : підручник. Київ : ВПЦ "Київський університет", 2008. 319 с. URL: https://csc.knu.ua/library/books/lavrishcheva-6.pdf (дата звернення: 08.05.2025). (Загальні принципи програмної інженерії).
30. Пірог О. В. Безпека вебдодатків : навчальний посібник. Житомир : Державний університет «Житомирська політехніка», 2025. 290 с. URL: https://eztuir.ztu.edu.ua/bitstream/handle/123456789/8791/%D0%9F%D1%96%D1%80%D0%BE%D0%B3.pdf (дата звернення: 08.05.2025).
31. ДСТУ ISO/IEC/IEEE 23026:2016. Інженерія систем і програмних засобів. Розроблення та керування WEB-сайтами для систем, програмних засобів та інформаційних послуг (ISO/IEC/IEEE 23026:2015, IDT). Київ : ДП «УкрНДНЦ», 2016. URL: https://online.budstandart.com/ua/catalog/doc-page?id_doc=71811 (дата звернення: 08.05.2025).
32. ДСТУ ISO/IEC TR 9126-3:2012. Програмна інженерія. Якість продукту. Частина 3. Внутрішні метрики (ISO/IEC TR 9126-3:2003, IDT). Київ : Мінекономрозвитку України, 2012. URL: https://online.budstandart.com/ua/catalog/doc-page.html?id_doc=53230 (дата звернення: 08.05.2025).
33. PHP Data Objects (PDO). PHP Manual (українська версія). URL: https://www.php.net/manual/uk/book.pdo.php (дата звернення: 08.05.2025). (Офіційна документація щодо PDO).
34. W3Schools українською. PHP MySQL База даних. URL: https://w3schoolsua.github.io/php/php_mysql_intro.html (дата звернення: 08.05.2025). (Навчальні матеріали з PHP та MySQL).
35. Побудова динамічних вебзастосунків за допомогою MVC. Блог IT-школи Hillel. URL: https://blog.ithillel.ua/articles/building-dynamic-web-applications-using-mvc (дата звернення: 08.05.2025). (Огляд архітектури MVC).
36. Composer частина 1. Навіщо його використовувати у PHP проектах та як з ним працювати? Блог IT-школи Hillel. URL:

https://blog.ithillel.ua/articles/composer-navishho-iogo-vikoristovuvati-u-php-proektax-ta-yak-z-nim-pracyuvati (дата звернення: 08.05.2025).
37. Ознайомлення з Bootstrap. ArmedSoft. URL: https://armedsoft.com/ua/blog/oznayomlennya-z-bootstrap (дата звернення: 08.05.2025). (Огляд Bootstrap).
38. Математичні основи моделювання систем : навчальний посібник / уклад. : І. В. Федосова, О. В. Щербина. Запоріжжя : ЗНУ, 2013. 95 с. URL: https://moodle.znu.edu.ua/pluginfile.php/752583/mod_folder/content/0/%D0%9F%D0%BE%D1%81%D1%96%D0%B1%D0%BD%D0%B8%D0%BA_7.pdf?forcedownload=1 (дата звернення: 08.05.2025). (Загальні підходи до моделювання).
39. Рекомендації CERT-UA з безпеки вебресурсів. Державна служба спеціального зв'язку та захисту інформації України. URL: https://cert.gov.ua/recommendation/19 (дата звернення: 08.05.2025).
40. Пасічник О. Г., Пасічник О. В., Стеценко І. В. Основи веб-дизайну : навчальний посібник. Київ : BHV, 2009. 336 с. (Приклад підручника, може бути знайдений в бібліотеках). URL: https://ktpu.kpi.ua/wp-content/uploads/2014/02/Pasichnik-O.-G.-Pasichnik-O.-V.-Stetsenko-I.-V.-Osnovi-veb-dizajnu.pdf (дата звернення: 08.05.2025)
41. Найкращі PHP фреймворки: огляд і порівняння популярних рішень. FoxmindEd. URL: https://foxminded.ua/php-freimvorky/ (дата звернення: 08.05.2025). (Для загального розуміння екосистеми PHP, хоча в роботі кастомний роутер).
42. Шаблонизатор Blade Laravel. Офіційна документація Laravel (російськомовний ресурс, але Blade є компонентом, описаним у "чп2.docx"). URL: https://laravel.su/docs/11.x/blade (дата звернення: 08.05.2025). (Примітка: Наведено для ознайомлення з концепцією Blade, оскільки прямих україномовних підручників саме з Blade може бути обмаль. У роботі використовується як окремий компонент).
43. Dompdf. Офіційний сайт/репозиторій (англійською). URL: https://github.com/dompdf/dompdf (дата звернення: 08.05.2025). (Примітка: Наведено для ознайомлення з бібліотекою, згаданою в "чп2.docx". Шукайте українські статті про інтеграцію PHP та PDF).
44. Вступ. Офіційна документація Bootstrap (російськомовний ресурс, але Bootstrap 5 згадано в "чп2.docx"). URL: https://getbootstrap.su/docs/5.0/getting-started/introduction/ (дата звернення: 08.05.2025). (Примітка: Наведено для ознайомлення. Шукайте українські ресурси або офіційну документацію англійською bootstrap.com).

45. Технології розробки Web-ресурсів : навчальний посібник / уклад. В.М. Солдаткін. Запоріжжя: ЗНУ, 2019. 100 с. URL: https://moodle.znu.edu.ua/pluginfile.php/889006/mod_resource/content/1/%D0%9F%D0%BE%D1%81%D1%96%D0%B1%D0%BD%D0%B8%D0%BA%204.pdf (дата звернення: 08.05.2025). (Загальні технології веб-розробки, згадується PHP).
46. ДСТУ ISO/IEC/IEEE 29119-1:2017. Інженерія систем і програмних засобів. Тестування програмних засобів. Частина 1. Поняття та визначення (ISO/IEC/IEEE 29119-1:2013, IDT). Київ : ДП «УкрНДНЦ», 2017. URL: https://online.budstandart.com/ua/catalog/doc-page?id_doc=75488 (дата звернення: 08.05.2025).
47. ДСТУ ISO/IEC/IEEE 12207:2018. Інженерія систем і програмних засобів. Процеси життєвого циклу програмних засобів (ISO/IEC/IEEE 12207:2018). Київ : ДП «УкрНДНЦ», 2018. (Посилання на документ у базах стандартів або через бібліотеку ЗВО).
48. Тестування програмних систем : навчально-методичний посібник / уклад. Т. М. Райчев. Ізмаїл : ІДГУ, 2019. URL: http://idgu.edu.ua/wp-content/uploads/2019/02/testuvannja-prohramnyh-system.pdf (дата звернення: 08.05.2025). (Охоплює функціональні та нефункціональні види тестування).
49. Методологія тестування програмного забезпечення : конспект лекцій / уклад. М. М. Клименко. Львів : ЛНУ ім. Івана Франка. URL: https://financial.lnu.edu.ua/wp-content/uploads/2019/09/konspekt-testuvannia.pdf (дата звернення: 08.05.2025). (Містить розділ про ручне тестування).
50. Лабортас М. О. Алгоритми мультикритеріального тестування веб-додатків. Тернопіль : ЗУНУ. URL: http://dspace.wunu.edu.ua/bitstream/316497/17543/1/%D0%9A%D0%A1%D0%9C%D0%B7%D0%BC-21_%D0%9B%D0%B0%D0%B1%D0%BE.pdf (дата звернення: 08.05.2025).
51. Бойко Л. М. Аналіз якості програмного забезпечення. Збірник наукових праць конференції «Інформатика, інформаційні системи та технології». 2012. URL: http://oldconf.neasmo.org.ua/node/430 (дата звернення: 08.05.2025). (Згадуються стандарти якості ПЗ).
52. Види функціонального та нефункціонального тестування. Блог DAN.IT education. URL: https://dan-it.com.ua/uk/blog/vidy-funkcionalnogo-i-nefunkcionalnogo-testirovanija/ (дата звернення: 08.05.2025). (Описуються різні види тестування, включно з регресійним).

53. ДСТУ 2850-94. Програмні засоби ЕОМ. Забезпечення якості. Показники та методи оцінювання якості програмного забезпечення. (Стандарт згадується в джерелі, актуальність та текст слід перевіряти в базах стандартів).
54. Куликов С. Тестирование программного обеспечения. Базовый курс. 3-е изд. Минск : Четыре четверти, 2018. 312 с. (Примітка: Книга російськомовного автора, але часто рекомендується в українських QA-колах як базовий курс; перевірте наявність перекладу або використовуйте як приклад з відповідним коментарем).
55. Савин Р. Тестирование Дот Ком, или Пособие по жестокому обращению с багами в интернет-стартапах. Москва : Litres, 2011. 291 с. (Примітка: Аналогічно до попереднього пункту, це популярна книга для початківців, автор російськомовний. Перевірте можливість використання або знайдіть україномовні огляди/адаптації).
56. Журнал «Охорона праці». (Рекомендується пошук тематичних статей щодо безпеки на об'єктах енергетики, приклад загальної інструкції: https://pro-op.com.ua/article/1556-nstruktsya-z-ohoroni-prats-dlya-dorojnogo-robtnika) (дата звернення: 08.05.2025).
57. Журнал «Екологія і промисловість». (Рекомендується пошук статей, пов'язаних з впливом енергетики на довкілля). Приклад публікації з дотичною тематикою: https://mepr.gov.ua/wp-content/uploads/2025/02/Zdijsnennya-otsinky-ryzykiv-ta-vrazlyvosti-bioriznomanittya-do-zminy-klimatu.pdf (дата звернення: 08.05.2025).
58. Кодекс цивільного захисту України від 02.10.2012 р. № 5403-VI (зі змінами, актуальна редакція). URL: https://zakon.rada.gov.ua/laws/show/5403-17 (дата звернення: 08.05.2025). (Стосується загальних питань безпеки та реагування на надзвичайні ситуації).

ДОДАТОК А

Лістинг коду «Inverter.php»

```
<?php
namespace App\Models;

/**
 * Мережеві інвертори зберігаються у storage/data/inverters.json
 */
class Inverter
{
    public int    $id;
    public string $name;
    public float  $max_dc_kw;
    public float  $ac_kw;
    public float  $price_eur;

    /* ---------- доступ до файлу ---------- */
    private static function file(): string
    {
        $dir = dirname(__DIR__, 2) . '/storage/data';
        if (!is_dir($dir)) mkdir($dir, 0775, true);
        return $dir . '/inverters.json';
    }

    /** @return static[] */
    public static function all(): array
    {
        $rows = json_decode(@file_get_contents(self::file()),
true) ?: [];
        return array_map(fn($r) => new self($r), $rows);
    }

    public static function find(int $id): ?self
    {
        foreach (self::all() as $inv) {
            if ($inv->id === $id) return $inv;
        }
        return null;
    }

    public static function create(array $d): void
    {
        $rows = json_decode(@file_get_contents(self::file()),
true) ?: [];
        $next = $rows ? max(array_column($rows, 'id')) + 1 : 1;
        $rows[] = [
            'id'         => $next,
            'name'       => trim($d['name']),
            'max_dc_kw'  => (float)$d['max_dc_kw'],
```

```
            'ac_kw'      => (float)$d['ac_kw'],
            'price_eur'  => (float)$d['price_eur'],
        ];
        file_put_contents(self::file(), json_encode($rows,
JSON_PRETTY_PRINT | JSON_UNESCAPED_UNICODE));
    }

    public static function update(int $id, array $d): void
    {
        $rows = json_decode(@file_get_contents(self::file()),
true) ?: [];
        foreach ($rows as &$r) {
            if ($r['id'] === $id) {
                $r['name']      = trim($d['name']);
                $r['max_dc_kw'] = (float)$d['max_dc_kw'];
                $r['ac_kw']     = (float)$d['ac_kw'];
                $r['price_eur'] = (float)$d['price_eur'];
            }
        }
        file_put_contents(self::file(), json_encode($rows,
JSON_PRETTY_PRINT | JSON_UNESCAPED_UNICODE));
    }

    public static function delete(int $id): void
    {
        $rows = array_filter(
            json_decode(@file_get_contents(self::file()), true) ?:
[],
            fn($r) => $r['id'] !== $id
        );
        file_put_contents(self::file(),
json_encode(array_values($rows), JSON_PRETTY_PRINT |
JSON_UNESCAPED_UNICODE));
    }

    private function __construct(array $r)
    {
        $this->id         = $r['id'];
        $this->name       = $r['name'];
        $this->max_dc_kw  = $r['max_dc_kw'];
        $this->ac_kw      = $r['ac_kw'];
        $this->price_eur  = $r['price_eur'];
    }
}
```

ДОДАТОК Б

Лістинг коду «Proposal.php»

```
<?php
namespace App\Models;

use DateTime;
use RuntimeException;

/**
 * DTO-/Active-record-подібна модель комерційної пропозиції.
 * Зберігає та витягає файли *.json* зі storage/proposals.
 */
class Proposal
{
    public ?int    $id           = null;
    public string  $client_name  = '';
    public float   $power_kw     = 0.0;
    public int     $panel_id     = 0;
    public int     $panel_qty    = 0;
    public int     $inverter_id  = 0;
    public float   $capex_eur    = 0.0;
    public float   $lcoe_eur_kwh = 0.0;
    public float   $npv_eur      = 0.0;
    public float   $irr_percent  = 0.0;
    public float     $payback_year = 0;
    public DateTime $created_at;

    /** Кількість інверторів у конфігурації */
    public int  $inverter_qty = 1;

    /** Чи перевищує CAPEX заявлений бюджет */
    public bool $over_budget  = false;

    /* ---------- ЗБЕРЕЖЕННЯ / ЗАВАНТАЖЕННЯ ---------- */

    /** Повертає шлях до каталогу з пропозиціями. */
    private static function dir(): string
    {
        $dir = dirname(__DIR__, 2) . '/storage/proposals';
        if (!is_dir($dir)) {
            mkdir($dir, 0775, true);
        }
        return $dir;
    }

    /** Серіалізує об’єкт і пише файл; генерує id, якщо порожній.
*/
    public function save(): void
    {
        if ($this->id === null) {
            $this->id = $this->nextId();
```

```
        }
        $path = self::dir() . "/{$this->id}.json";
        file_put_contents($path, json_encode($this->toArray(),
JSON_PRETTY_PRINT | JSON_UNESCAPED_UNICODE));
    }

    /** Завантажує пропозицію за ID або кидає виняток. */
    public static function find(int $id): self
    {
        $path = self::dir() . "/{$id}.json";
        if (!file_exists($path)) {
            throw new RuntimeException("Proposal {$id} not
found.");
        }
        $data = json_decode(file_get_contents($path), true);
        return self::fromArray($data);
    }

    /** Повертає всі збережені пропозиції. @return static[] */
    public static function all(): array
    {
        $list = [];
        foreach (glob(self::dir() . '/*.json') as $file) {
            $data   = json_decode(file_get_contents($file), true);
            $list[] = self::fromArray($data);
        }
        // Новіші — першими
        usort($list, fn($a, $b) => $b->id <=> $a->id);
        return $list;
    }

    /* ---------- ВСПОМОГІЖНІ МЕТОДИ ---------- */

    /** Конвертує масив → об’єкт. */
    public static function fromArray(array $a): self
    {
        $p = new self();
        foreach ($a as $k => $v) {
            if (property_exists($p, $k)) {
                $p->$k = $k === 'created_at' ? new DateTime($v) :
$v;
            }
        }
        return $p;
    }

    /** Конвертує об’єкт → масив. */
    public function toArray(): array
    {
        return [
            'id'            => $this->id,
            'client_name'   => $this->client_name,
```

```
            'power_kw'      => $this->power_kw,
            'panel_id'      => $this->panel_id,
            'panel_qty'     => $this->panel_qty,
            'inverter_id'   => $this->inverter_id,
            'capex_eur'     => $this->capex_eur,
            'lcoe_eur_kwh'  => $this->lcoe_eur_kwh,
            'npv_eur'       => $this->npv_eur,
            'irr_percent'   => $this->irr_percent,
            'payback_year'  => $this->payback_year,
            'created_at'    => $this->created_at-
>format(DateTime::ATOM),
        ];
    }

    /** Генерує наступне ID на основі існуючих файлів. */
    private function nextId(): int
    {
        $files = glob(self::dir() . '/*.json');
        $ids   = array_map(fn($f) => (int) basename($f, '.json'),
$files);
        return $ids ? max($ids) + 1 : 1;
    }
}
```

ДОДАТОК В

Лістинг коду «SolarPanel.php»

```
<?php
namespace App\Models;

/**
 * Модель Сонячної панелі з JSON-сховищем
storage/data/solar_panels.json
 */
class SolarPanel
{
    public int    $id;
    public string $name;
    public int    $watt_peak;
    public float  $vmp;
    public float  $price_eur;

    /* ---------- Статичні сервіси ---------- */

    private static function file(): string
    {
        $dir = dirname(__DIR__, 2) . '/storage/data';
        if (!is_dir($dir)) { mkdir($dir, 0775, true); }
        return $dir . '/solar_panels.json';
    }

    /** @return static[] */
    public static function all(): array
    {
        $rows = json_decode(@file_get_contents(self::file()),
true) ?: [];
        return array_map(fn($r) => new self($r), $rows);
    }

    public static function find(int $id): ?self
    {
        foreach (self::all() as $p) {
            if ($p->id === $id) return $p;
        }
        return null;
    }

    public static function create(array $data): void
    {
        $rows = json_decode(@file_get_contents(self::file()),
true) ?: [];
        $next = count($rows) ? max(array_column($rows, 'id')) + 1
: 1;
        $rows[] = [
            'id'         => $next,
            'name'       => trim($data['name']),
```

```
            'watt_peak'  => (int) $data['watt_peak'],
            'vmp'        => (float)$data['vmp'],
            'price_eur'  => (float)$data['price_eur'],
        ];
        file_put_contents(self::file(), json_encode($rows,
JSON_PRETTY_PRINT | JSON_UNESCAPED_UNICODE));
    }

    public static function update(int $id, array $data): void
    {
        $rows = json_decode(@file_get_contents(self::file()),
true) ?: [];
        foreach ($rows as &$r) {
            if ($r['id'] === $id) {
                $r['name']      = trim($data['name']);
                $r['watt_peak'] = (int) $data['watt_peak'];
                $r['vmp']       = (float)$data['vmp'];
                $r['price_eur'] = (float)$data['price_eur'];
            }
        }
        file_put_contents(self::file(), json_encode($rows,
JSON_PRETTY_PRINT | JSON_UNESCAPED_UNICODE));
    }

    public static function delete(int $id): void
    {
        $rows = array_filter(
            json_decode(@file_get_contents(self::file()), true) ?:
[],
            fn($r) => $r['id'] !== $id
        );
        file_put_contents(self::file(),
json_encode(array_values($rows), JSON_PRETTY_PRINT |
JSON_UNESCAPED_UNICODE));
    }

    /* ---------- Внутрішнє ---------- */
    private function __construct(array $r)
    {
        $this->id        = $r['id'];
        $this->name      = $r['name'];
        $this->watt_peak = $r['watt_peak'];
        $this->vmp       = $r['vmp'];
        $this->price_eur = $r['price_eur'];
    }
}
```