

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ
СІКОРСЬКОГО»**

Навчально-науковий інститут атомної та теплової енергетики

Кафедра інженерії програмного забезпечення в енергетиці

«На правах рукопису»
УДК 004.4

ДО ЗАХИСТУ ДОПУЩЕНО
В.о. завідувача кафедри
Олександр КОВАЛЬ
« » 2024 р.

Магістерська дисертація
на здобуття ступеня магістра
за освітньою програмою «Інженерія програмного забезпечення
інтелектуальних кібер-фізичних систем в енергетиці»
за спеціальності 121 Інженерія програмного забезпечення
на тему: «Рекомендаційна система підготовки оселі до блекауту»

Виконав: студент 2 курсу групи ТВ-з21мп

Кучерук Владислав Євгенович
(прізвище, ім'я, по батькові)

(підпис)

Науковий керівник: к.т.н., Кравцов Г.О.

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

(підпис)

Рецензент: с.н.с., к.т.н., Джигун О.М.

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

(підпис)

Засвідчую, що у цій магістерській
дисертації немає запозичень з праць інших
авторів без відповідних посилань.

Студент _____
(підпис)

Київ – 2024

**Національний технічний університет України
«Київський політехнічний інститут ім. Ігоря Сікорського»**

Навчально-науковий інститут атомної та теплової енергетики

Кафедра інженерії програмного забезпечення в енергетиці

Рівень вищої освіти другий, магістерський

Спеціальність 121 Інженерія програмного забезпечення

Освітньо-професійна програма «Інженерія програмного забезпечення інтелектуальних кібер - фізичних систем в енергетиці»

ЗАТВЕРДЖУЮ

В.о. завідувача кафедри

_____ Олександр КОВАЛЬ
(підпис)

«_____» _____ 2024 р.

**З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ ДИСЕРТАЦІЮ СТУДЕНТУ**

_____ Кучеруку Владиславу Євгеновичу

(прізвище, ім'я, по батькові)

1. Тема дисертації:

_____ «Рекомендаційна система підготовки оселі до блекауту»

Науковий керівник _____ Кравцов Григорій Олексійович, к.т.н.

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «08» листопада 2023 року № 5202-с

2. Строк подання студентом дисертації _____ 10 січня 2024 року

3. Об'єкт дослідження _____ процес надання рекомендації щодо вибору засобів резервного електроживлення для оселі.

4. Предмет дослідження _____ вебзастосунок рекомендаційної системи для ефективної підготовки власних осель до блекауту.

5. Вихідні дані до роботи _____ мова програмування Python, фреймворк Django, середовище розробки VS Code.

6. Перелік питань, які потрібно розробити _____ провести дослідження існуючих інструментів для підготовки осель до відключень світла, визначити основні функціональні можливості рекомендаційної системи, обрати технології для реалізації ідеї, розробити алгоритм пошуку найбільш релевантного засобу

резервного електроживлення, протестувати розроблену систему та виправити помилки за наявності.

7. Орієнтований перелік ілюстративного матеріалу структура шаблону проєктування MVT, алгоритм роботи програми, головна сторінка рекомендаційної системи.

8. Орієнтований перелік публікацій _____

9. Дата видачі завдання «01» листопада 2022 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання магістерської дисертації	Строки виконання етапів магістерської дисертації	Примітка
1	Отримання завдання	01.11.2022	виконано
2	Дослідження предметної області	01.11.2022 - 01.12.2022	виконано
3	Постановка вимог до проєктування системи	01.12.2022 - 15.12.2022	виконано
4	Дослідження існуючих рішень	15.12.2022 - 03.01.2023	виконано
5	Розробка програмного продукту	03.03.2023 - 20.09.2023	виконано
6	Тестування	20.09.2023 - 15.10.2023	виконано
7	Захист програмного продукту	16.10-2023 – 20.10.2023	виконано
8	Підготовка магістерської дисертації	21.10.2023 – 17.11.2023	виконано
9	Передзахист	18.12.2023 - 22.12.2023	виконано
10	Захист	15.01.2024 – 19.01.2024	виконано

Студент

(підпис)

Кучерук В.Є.

(прізвище та ініціали)

Науковий керівник

(підпис)

Кравцов Г.О.

(прізвище та ініціали)

РЕФЕРАТ

Структура та обсяг магістерської дисертації. Магістерська дисертація складається зі вступу, 5 розділів, висновків, 14 посилань на джерела. Обсяг дисертації 87 сторінок, котрі містять 16 рисунків.

Актуальність теми. Сьогодення характеризується глибоким проникненням комп'ютеризації у майже всі сфери людського життя. Відповідно, електроенергія відіграє не останню роль в цьому. Без неї буде порушена коректна робота багатьох сервісів, що, звісно, стає доволі актуально в наші непрості часи, коли відключення світла можуть бути тривалими і значними для робочих моментів.

Мета і задачі дослідження. Метою є підвищення рівня освіченості людей щодо вибору засобів енергоживлення для власних осель, які допоможуть підвищити готовність та стійкість оселі до блекауту, а також зменшать, різного роду, негативний вплив відключень електроенергії.

Основними завданнями є:

- аналіз поточного стану справ щодо підготовки власних осель до відключень світла;
- пропонування концепції оптимізованого рішення для легшої та швидшої підготовки оселі до блекауту;
- розробка алгоритму підбору найбільш релевантного засобу для потреб користувача;
- розробка рекомендаційної системи.

Методи дослідження. У даному дослідженні для досягнення мети роботи було використано метод порівняльного аналізу наявних засобів електроживлення між собою для рекомендації споживачу найбільш релевантного з них.

Практичне значення одержаних результатів. Результати проведеного дослідження представляються у вигляді програмного продукту, який доступно

представляє інформацію для ознайомлення та, за потреби, рекомендацію щодо вибору безперебійного засобу електроживлення оселі.

Об'єктом дослідження є процес надання рекомендації щодо вибору засобів резервного електроживлення для оселі.

Предметом дослідження є вебзастосунок рекомендаційної системи для ефективної підготовки власних осель до блекауту.

Ключові слова: рекомендаційна система, блекаут, відключення світла, засоби електроживлення, вебзастосунок, Python, Django, PostgreSQL.

ABSTRACT

The structure and scope of the thesis. The master's thesis comprises an introduction, 5 chapters, conclusions, and includes 14 source references. The dissertation spans 87 pages, containing 16 figures.

Relevance of topic. The present era is characterized by the profound integration of computerization into nearly every aspect of human life. Accordingly, electricity plays an important role in this. Without it, the proper functioning of many services would be disrupted, which, of course, becomes particularly relevant in our challenging times when power outages can be prolonged and significant, affecting work activities.

Research goal and objectives. The aim is to raise people's awareness regarding the selection of energy sources for their homes, thereby enhancing the preparedness and resilience of households to blackouts and mitigating various negative impacts resulting from power outages.

The main tasks include:

- analyzing the current state of preparedness of private residences for power outages;
- proposing a concept for an optimized solution to facilitate quicker preparation for blackouts;
- developing an algorithm for selecting the most relevant tool to meet user needs;
- creating a recommendation system.

Methods of research. In this study, to achieve the research goal, we employed the method of comparative analysis among available power supply devices to recommend the most relevant one to the consumer.

Practical value of obtained results. The results of the conducted research are presented in the form of a software product that offers information for review and, if needed, recommendations for selecting an uninterruptible power supply for homes.

The object of the research is the process of offering recommendations for selecting backup power supplies for homes.

The subject of the research is a web application of a recommendation system designed for the efficient preparation of private homes for a blackout.

Keywords: recommender system, blackout, power outage, power supply, web application, Python, Django, PostgreSQL.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ	9
ВСТУП.....	10
1 ПОСТАНОВКА ЗАДАЧІ ТА ОСНОВНІ ЗАВДАННЯ	11
1.1 Життя без електроживлення	11
1.2 Значення електроенергії для працездатності людини	13
1.3 Задача та основні завдання.....	15
1.4 Опис запропонованого рішення.....	17
Висновки до розділу 1	21
2 ОГЛЯД ІСНУЮЧИХ СИСТЕМ	22
2.1 Існуючі системи.....	22
2.1.1 Системи рекомендаційного типу	23
2.1.2 Системи оглядового типу	23
2.1.3 Системи магазинного типу	24
2.2 Головні особливості розроблюваної системи у порівнянні з конкурентами .	24
Висновки до розділу 2	26
3 ЗАСОБИ РОЗРОБКИ	28
3.1 Мова програмування Python	28
3.2 Фреймворк Django.....	34
3.3 База даних PostgreSQL.....	39
3.4 Середовище розробки VS Code	42
Висновки до розділу 3	43
4 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ.....	45
4.1 Модель MVT	45
4.2 Опис алгоритму роботи програми.....	47
4.3 Серверна частина програмного продукту	50
4.4 Користувачька частина програмного продукту	57
4.5 Передумови використання	61
4.6 Функціональні можливості	62
Висновки до розділу 4	64

5 РОЗРОБКА СТАРТАП-ПРОЄКТУ	65
5.1 Опис ідеї проєкту	65
5.2 Технологійний аудит проєкту.....	67
5.3 Аналіз ринкових можливостей стартап-проєкту	67
5.4 Розроблення ринкової стратегії програмного продукту	77
5.5 Розроблення маркетингової програми стартап-проєкту	80
Висновки до розділу 5	83
ВИСНОВКИ.....	85
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	86
ДОДАТКИ	88
ДОДАТОК А Лістинг розробленої програми (фрагменти)	88
ДОДАТОК Б Презентація	98

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

ОЕС – об'єднана енергетична система

СЕЕ – споживання електричної енергії

БД – база даних

SQL – мова структурованих запитів (*англ.* Structured Query Language)

MVT – модель представлення шаблон (*англ.* Model View Template)

ORM – об'єктно-реляційне відображення (*англ.* Object Relational Mapping)

HTML – мова гіпертекстової розмітки (*англ.* HyperText Markup Language)

CSS – каскадні таблиці стилів (*англ.* Cascading Style Sheets)

ВСТУП

Сьогодні, в умовах воєнного стану, особливого значення набуває процес належної підготовки громадян до потенційних небезпечних і критичних ситуацій. Однією з таких ситуацій – є блекаут, або ж відключення електроенергії. В поточних реаліях, такі відключення вимагають особливої уваги та підготовки, адже їхнім каталізатором вже не є рядові та незначні перебої, ремонтні роботи тощо в системі енергоживлення. Перебої в постачанні електроенергії можуть становити значні наслідки, окрім комфорту, ще й для безпеки та життя, через відсутність опалення, можливості безперервного зберігання продуктів харчування тощо. Тому, підготовка повинна бути комплексною і включати в себе прорахунок всіх можливих сценаріїв подальшого розвитку.

Також, широкого масштабу сягнув розвиток комп'ютерних технологій, на яких функціонує безліч корисних та незамінних сервісів для сучасного людського життя. Відповідним чином їхнє безперешкодне функціонування для виконання поставлених на них задач безпосередньо залежить від потокового електроживлення. Відмова в їхній роботі здатна призвести до критичних економічних, матеріальних чи правових збитків.

Одна справа, коли ці сервіси працюють під керівництвом великих компаній, які можуть забезпечити свою продукцію безперебійним живленням для цілодобової роботи. Але якщо ми будемо говорити про звичайних людей, які мають потребу в забезпеченні електроенергією, то для цих же людей, може виникнути важкість у виборі засобу для забезпечення оселі безперебійним живленням.

Для вибору найкращого варіанту потрібно витратити багацько часу та дослідити чимало різноманітних ресурсів, що може бути проблемно, адже не всі здатні приділити цьому питанню достатньо уваги через різноманітні життєві клопоти. Куди простіше було б звернутися до спеціалізованого ресурсу з короткою інформацією про поточний стан справ, наявні та актуальні засоби електроживлення та, за потреби, можливістю розрахувати індивідуальне рішення для власної оселі.

1 ПОСТАНОВКА ЗАДАЧІ ТА ОСНОВНІ ЗАВДАННЯ

1.1 Життя без електроживлення

Уявивши собі життя людини без електроживлення – відразу виникає багато питань щодо того, чи зможе вона взагалі існувати без нього і надалі. Для комфортного та функціонального життя людство роками створювало і виробляло різноманітні приладдя для полегшення власного життя. Користь таких винаходів була і залишається неоціненно великою, але, особливість цих винаходів полягає в тому, що їх переважна більшість досить сильно залежить від електропостачання до їхніх керуючих механізмів всередині них.

Зростання кількості населення, урбанізація та процеси індустріалізації, розвиток інфраструктур транспорту спричиняють зростання видобутку та споживання електричної енергії. Електроенергетика є базовою галуззю економік багатьох країн світу, вона забезпечує як функціонування всіх інших галузей господарства, так і можливості їхнього перспективного розвитку [1].

Життя сучасної людини неможливо уявити без електроенергії. Промисловість, сільське господарство, наука, медицина, радіо і телебачення, інтернет, численні види зв'язку, побутові прилади і пристрої, системи опалення та кондиціонування повітря, освітлення міських вулиць і автодоріг – це лише незначна частка застосування електрики в сучасному світі. І природно, що припинення енергозабезпечення навіть на незначний період часу призводить до повного паралічу життєдіяльності людини, що відбувається не так вже й рідко [2].

В епоху, коли електрику ще не було відкрито, людина була надзвичайно обмеженою в своїх можливостях. Величезне число ідей, задумів і винаходів не могли бути реалізовані через відсутність в ті часи електрики. Відкриття електричного струму викликало справжній переворот в існуванні і світогляді всього людства. Завдяки цьому безцінному і унікальному відкриттю людська цивілізація отримала можливість стрімко розвиватися, досягати нових висот в

науці і техніці. Людина отримала можливість зробити своє життя значно більш комфортним, зручним, ефективним і радісним [2].

Основний пік робіт і досліджень, пов'язаних з відкриттям електричного струму, припадає на період з кінця XIX століття до початку XX століття. Цей період надзвичайно багатий безцінними і дивовижними відкриттями і винаходами в області електрики. Саме в цей період було закладено міцну наукову базу, яка дала можливість стрімко розвиватися електроенергетиці і досягти сучасного найвищого рівня її розвитку практично у всьому світі [2].

Внаслідок того, що значення електрики в усіх сферах життєдіяльності сучасного суспільства просто колосальне, проблеми стабільного і якісного енергопостачання; надійної і стійкої передачі електроенергії від електричних станцій до споживачів; проблеми безперебійного функціонування різноманітного електротехнічного обладнання є надзвичайно важливими і актуальними на сьогоднішній момент часу [2].

Енергетика в сучасному світі є основою розвитку базових галузей промисловості, що визначають прогрес суспільного виробництва. В усіх промислово розвинених країнах темпи розвитку енергетики випереджали темпи розвитку інших галузей [2].

Настільки широке поширення електроенергетики також пояснюється її специфічними властивостями: можливістю перетворюватися практично в усі інші види енергії (теплову, механічну, звукову, світлову тощо); здатністю відносно просто передаватися на значні відстані у великих кількостях; величезними швидкостями протікання електромагнітних процесів; здатністю до дроблення енергії і перетворення її параметрів (зміна напруги, частоти) [2].

У промисловості електрична енергія застосовується як для приведення в дію різних механізмів, так і безпосередньо в технологічних процесах. Робота сучасних засобів зв'язку (телеграфу, телефону, радіо, телебачення) заснована на застосуванні електроенергії. Без неї неможливим був би розвиток кібернетики, обчислювальної техніки, космічної техніки [2].

Таким чином електроенергетика – важлива частина життєдіяльності людини. Рівень її розвитку відображає рівень розвитку продуктивних сил суспільства і можливості науково-технічного прогресу [2].

Електрика стала невід’ємною частиною людського життя і за такий короткий час позбутися її і відмовитися від знаряддя, яке функціонує з її допомогою так просто не вийде та й дивним було б це взагалі рішення. Найкращим варіантом стала б можливість безперебійного живлення власних осель та необхідних робочих інструментів.

1.2 Значення електроенергії для працездатності людини

Робота людини з тривалим часом її існування постійно розвивалася та вдосконалювалася у своїх методиках та функціях. Не в останню чергу, це завдяки сучасним напрацюванням: завдяки приладам широкого застосування, які також потребують електроживлення для можливості забезпечувати своїх власників своїми наявними можливостями.

Якісного, стабільного і безперебійного енергозабезпечення вкрай потребують промислові та сільськогосподарські підприємства, урядові, фінансово-економічні та оборонні структури, науково-дослідні центри, заклади охорони здоров'я та освіти. Неможливо назвати ту область сучасної життєдіяльності людини, де б не використовувалася електроенергія. Тому від якісної, ефективної та оперативної роботи структур, що забезпечують безперебійне постачання електроенергією численних об'єктів народного господарства, залежить, у вищій мірі, благополуччя і успішне існування людського суспільства [2].

Величезну роль електроенергія відіграє у транспортній промисловості. Електротранспорт не забруднює навколишнє середовище. Велику кількість електроенергії споживає електрифікований залізничний транспорт, що дозволяє

підвищувати пропускну здатність доріг за рахунок збільшення швидкості руху поїздів, знижувати собівартість перевезень, підвищувати економію палива [2].

В Україні електроенергетика функціонує як об'єднана енергетична система (ОЕС), яка розглядається як сукупність електростанцій, електричних мереж, інших об'єктів електроенергетики, що об'єднані спільним режимом виробництва, передачі та розподілу електричної енергії при централізованому управлінні цим режимом [3].

Виробництво електричної енергії здійснюється на електричних станціях, які можна поділити на дві групи – системного і місцевого значення. Електростанції системного значення забезпечують в межах країни централізоване виробництво електричної енергії і підтримання режиму роботи ОЕС. Це електростанції великої потужності – атомні, теплові, гідроелектростанції. Електростанції місцевого значення (теплові, відновлювальної електроенергетики: вітрові, сонячні, малі гідроелектростанції тощо) істотно не впливають на режим ОЕС. У деяких випадках вони працюють відокремлено від ОЕС і забезпечують електричною енергією окремих споживачів (підприємства, населені пункти) [1].

Складовими системи «Споживання електричної енергії» (СЕЕ) є споживачі й приймачі електричної енергії.

Приймач електричної енергії (електроприймач) – апарат, агрегат, механізм, призначений для перетворення електричної енергії в інший вид енергії [4]. Отже приймачем електричної енергії є електрична частина механізму або технологічної установки, що одержує електроенергію з мережі й витрачає її на виконання технологічних процесів, які супроводжуються перетворенням електроенергії в інші види енергії: механічну, теплову, світлову тощо. Деякі технологічні установки мають у своєму складі декілька електроприймачів: верстати, крани, прокатні стани і т. п. Розглядаючи технологічну установку як єдине ціле говорять про споживача електричної енергії [1].

Споживач електричної енергії (електроспоживач) – електроприймач або група електроприймачів, об'єднаних технологічним процесом, які розміщуються на певній території [4].

Таким чином, можемо зрозуміти важливість електроенергії для робочих моментів. Це є невід'ємною складовою електроприймачів для їхнього належного функціонування за безпосереднім призначенням.

На підприємствах СЕЕ поєднує групу електроприймачів технологічної установки, цеху, корпусу або підприємства в цілому, на підставі їхніх характерних функцій (ознак). Приміром, характерною функцією деякого промислового підприємства є виробництво певної продукції. Саме підприємство в цілому може розглядатися як СЕЕ, до складу якого входять інші електроспоживачі (цехи, дільниці), розташовані на відповідній території (підприємства, цеху, дільниці) [1].

У містах СЕЕ є будинки (житлові, громадського призначення тощо), мікрорайони, райони. Місто в цілому також є споживачем електричної енергії [1].

Зазначимо, що розгляд електроспоживачів, їхньої структури й технічних характеристик набуває важливого значення при дослідженні взаємодії систем споживання і розподілу електричної енергії [1].

Без належного забезпечення електроенергією користувацьких пристроїв для роботи, може бути порушена критично-важлива структура у сфері різноманітних людських послуг від найдрібніших до наймасштабніших, що може тягти за собою серйозні збитки.

1.3 Задача та основні завдання

Розглянута проблематика у вступі та попередніх підрозділах широко і повно описує поточний стан справ щодо механізму вибору резервного джерела живлення для оселі. Також, не менш важливими, залишаються питання щодо забезпечення нормального рутинного життя людей в ці періоди. Під визначенням «нормального

життя» варто розуміти належне забезпечення та функціонування оточення, яке виключно сприятиме розвитку і вдосконаленню людських якостей. Підсумовуючи викладені вище думки, зазначимо, що електроенергія стала невід'ємною частиною людського життя в сучасному світі. Якоюсь мірою, вона є однією зі складових частин на шляху для досягнення успіху. Тому, прибрати її з цього рівняння – означає похитнути можливість досягнення бажаних та успішних результатів в житті.

Це, в першу чергу, обумовлене глобальною комп'ютеризацією світу. Вона сягнула величезних масштабів і на комп'ютерних системах зав'язана зараз, якщо не вся, то переважна більшість роботи сучасного світу. Всі телекомунікаційні системи, новинна інформація, різноманітні транзакції, пошта, конференції та багато інших сфер людської діяльності, в сьогоденні, є залежними від комп'ютерних систем. Порушення в їхній роботі матиме вплив на всі залежні від цього робочі сфери. Проте, це може бути описом, хіба що гостросюжетного фільму. В реальному ж житті великі компанії, які займають певну нішу в робочій діяльності людства, цілком здатні забезпечити власну продукцію, яка функціонує в залежності від комп'ютерних систем, безперебійним енергоживленням в періоди відключень. Тому, проблематика щодо вибору засобу для резервного енергоживлення з урахуванням багатьох технологічних та фінансових факторів припадає на окрему когорту людей. Поточні потреби у забезпеченні пристроями для резервного електроживлення обумовлені сучасними реаліями через можливість працювати дистанційно чи то ж просте бажання не втрачати комунікативний зв'язок із сучасним світом.

Іншим важливим фактором у бажанні заволодіти пристроями резервного електроживлення полягає в тому, що є потреба у забезпеченні безперебійного виробництва власної продукції у приватних секторах. Окрім цього, під час тривалих відключеннях світла, виникає гостра потреба у допоміжному засобі, який надасть змогу отримати тепло в оселі, можливість приготування їжі та вирішення інших дрібних, але не менш важливих рутинних домашніх справ. Тому сьогодні,

важливість забезпечення оселі додатковим електроживленням постає гостро, як ніколи раніше.

1.4 Опис запропонованого рішення

Розібравшись з поточною проблематикою даного питання, слід тепер зосередити власну увагу на можливих та доцільних варіантах його рішення. Запропоноване рішення повинно вирізнятися своєю структурністю та лаконічністю у представленні інформації споживачу. Всі відображення повинні бути доступно формалізовані для зменшення рівня невизначеності у людини, щоб їй легше було досягти поставлених цілей.

Цілі споживача можуть бути різноманітними, від найменших до найбільш масштабних. Але головним пов'язуючим фактором є те, що вони всі, так чи інакше, стосуватимуться пристроїв для забезпечення оселі резервним варіантом енергоживлення. Отже, розроблена система повинна доступним чином забезпечувати цю можливість для споживача.

Для забезпечення доступності розроблюваної системи, потрібно обрати варіант, який зможе охопити більшу частку користувачів, що потребують вирішення питань щодо засобу резервного енергоживлення. Також, такий варіант повинен бути зручним у використанні та не вимагати витратити додаткові зусилля, щоб досягти бажаного результату. Адже, головною проблемою до цього, яка є у схожих існуючих системах – це їхня розмежованість від однорідної цілі. Тобто, вони не спеціалізуються на тому, щоб допомогти користувачу підібрати найоптимальніший варіант за його запитами. Вони радше дають ознайомлювальну інформацію з детальним оглядом або ж можуть допомогти з розрахунками певних параметрів для подальшого самостійного пошуку рішення. Таким чином, доцільність розробленої системи повинна виражатися через показ різноманітних суміжних компонентів пошуку та розрахунку для досягнення єдиної цілі. І найважливішим повинна стати умова розміщення всіх допоміжних компонентів в одному єдиному місці, щоб

мінімізувати даремне витрачання часу на пошуки потрібної інформації в найрізноманітніших ділянках мережевого простору.

Досягнення всебічної доступності повинно реалізовуватися через найпоширеніше поле для взаємодії з користувачами. Для досягнення такої цілі зручно будувати рекомендаційну систему у вигляді сайту, який зможе бути доступним будь-де та повсякчас, оскільки в сьогоденні, життя без використання браузера важко уявити. В такому варіанті реалізації значно спрощується процес поширення програмного продукту, оскільки, не доведеться споживачам додатково щось встановлювати чи налаштовувати, а просто перейти за необхідною адресою та приступити до використання ресурсу.

Рекомендаційна система для підготовки оселі до блекауту повинна мати зручний та інформативний інтерфейс для спрощення розуміння поточного контексту та для швидшого досягнення бажаних цілей. Сайт будуватиметься з використанням фронтенд та бекенд технологій.

Фронтенд частина, що відповідатиме за візуальне сприйняття відвідувачами ресурсу братиме на себе задачу інтуїтивно зрозумілого інтерфейсу для подальшої навігації ним. Для цього, доцільно використовувати стримані візуальні забарвлення та формально-сприйнятливі шрифти для пояснювального тексту (наочна інформація зазначена в п. 4.4).

Бекенд частина відповідатиме за основну прописну логіку сайту. Тобто, за переходи між різними сторінками сайту, переадресації сторінок та їхнє представлення. Всі введення на сайті будуть оброблятися серверною частиною і видавати необхідний результат користувачу. Окрім базових оброблювальних задач по представленню сторінок з наявною інформацією, бекенд частина повинна буде проводити розрахунки найбільш виваженого, найбільш релевантного засобу за запитом користувача. Однією з найважливіших особливостей рекомендаційної системи, яке і буде виправдовувати її назву, – стане своєрідний калькулятор. Відвідувач сайту матиме змогу ввести декілька параметрів, які є бажаними та найголовнішими для нього і на їхній основі система проведе розрахунки і видасть

найбільш релевантний варіант за запитом користувача. (наочна інформація зазначена в п. 4.3).

Зазначені параметри є наступними: кількість виробленої енергії, бажаний бюджет, час безперебійного живлення, нічний режим роботи та можливість розміщення в квартирі. Алгоритм рекомендації працює наступним чином:

- вибір всіх приладів, фільтруючи за запитом користувача;
- порівняння обраних приладів між собою, для виявлення найкращого;
- вивід на екран єдиного варіанту, який виявився найкращим серед інших.

Принцип роботи першого та останнього пунктів є зрозумілим: спочатку всі наявні засоби відбираються за критеріями користувача для подальшого аналізу, після якого, відображається найкращий варіант на екрані. Процес аналізу обраних приладів проводиться шляхом їхнього порівняння між собою за трьома параметрами. Щоб мінімізувати різноманітні похибки і неточності з величезними розбіжностями в параметрах деяких приладів доцільно виконувати порівняння з прив'язкою до одиниці виробленої енергії кожним з приладів. Це зможе показати цілісну картину в незалежності від класу виробу. Тобто, порівнюватиметься не конкретна потужність різних приладів, а кількість виробленої енергії на однакову вартість або час активної безперебійної роботи:

$$r_{pi} = \frac{e_i}{p_i}, \quad (1.1)$$

де r_{pi} – розрахований результат кількості виробленої енергії на одиницю витрачених коштів для конкретного пристрою;
 e_i – кількість виробленої енергії конкретним пристроєм;
 p_i – вартість конкретного пристрою.

Такий розрахунковий результат обчислюється для кожного приладу, який підходить під параметри фільтру вибору. Окрім вартості, ще робляться розрахунки за аналогічними до (1.1) формулою для значення часу безперебійної роботи та одноразових витрат на конкретний прилад. Після проведення таких розрахунків, в кожному з параметрів обирається прилад з найкращим значенням, який оцінюється,

як стовідсотковий варіант релевантності. Всім іншим, в свою чергу, привласнюється певний відсоток від даного стовідсоткового значення. Такий процес робиться для кожного з параметрів, а далі, для кожного пристрою, рахується середнє арифметичне відсоткових значень за формулою (1.2). Такий варіант розрахунку за кожним з параметрів допоможе визначити найкращий засіб серед представлених за критеріями користувача:

$$r_i = \frac{p_{pci} + t_{pci} + e_{pci}}{3}, \quad (1.2)$$

де r_i – загальний відсотковий результат для конкретного пристрою;

p_{pci} – відсотковий результат вартості в порівнянні з іншими пристроями;

t_{pci} – відсотковий результат часу безперебійної роботи в порівнянні з іншими пристроями;

e_{pci} – відсотковий результат одноразових витрат на один цикл роботи в порівнянні з іншими пристроями.

Також, рекомендаційна система повинна містити в собі всю актуальну інформацію про засоби безперебійного живлення за різноманітними категоріями. Для цього використовуватиметься база даних з наперед визначеними таблицями для збереження необхідної інформації. За наповнення самої бази даних повинні відповідати адміністратори системи, які мають на руках числові значення конкретних засобів та наявну необхідну інформацію про них. Інформація в ручну заноситься до потрібної таблиці і в процесі роботи сайту, може бути відображена на сторінці. За підключення до бази даних з наявною інформацією про актуальні засоби для їхнього відображення на потрібних сторінках відповідатиме бекенд частина.

Висновки до розділу 1

Глобальна комп'ютеризація людської діяльності сягнула своїх найбільших масштабів в сьогоденні. Для повноцінного функціонування необхідне постачання електроенергії в безперебійному режимі. Величезні корпорації здатні подбати про таку проблематику на корпоративному рівні, але у звичайних користувачів можуть виникати з цим труднощі.

Людям, для особистих потреб, може знадобитися наявність в їхній оселі джерела резервного живлення на випадок екстремальних відключень електроенергії. Для підбору такого засобу витрачається багато часу, а отриманий результат не завжди може задовільняти всі потреби. Процес вибору доцільно було б спростити.

Для досягнення поставленого завдання гарною ідеєю в реалізації було б використання автоматизованої системи для підбору оптимального рішення, шляхом порівняння наявних засобів за бажанням користувача.

2 ОГЛЯД ІСНУЮЧИХ СИСТЕМ

2.1 Існуючі системи

Систем для вирішення зазначеної проблематики, на поточний момент, не існує. Мова йде саме про спеціалізовані системи для підготовки власних осель до періоду затемнення. Тобто, систем, які здатні повноцінно і беззаперечно надавати інформацію щодо засобів та пристроїв енергоживлення осель, зацікавлених в цьому питанні людей, у формалізованому вигляді.

Звісно, не правильно буде сказати, що підготувати власне помешкання до незапланованих відключень неможливо через відсутність відповідних систем. Вони все ж існують, але в, певного роду, трішки інших представленнях. Вони здатні певним чином надавати інформативні дані щодо доцільних порад у підготовці власних приміщень необхідним знаряддям для незалежного, від централізованого, енергоживлення. Проте, дана інформація є надто розгалуженою, щоб користуватися нею повноцінно і отримувати максимум користі за найменш короткий проміжок часу.

Існуючі системи, мають трохи іншу спеціалізацію, основну задачу. А сам процес інформування користувачів щодо засобів та методів забезпечення власного житла додатковим енергоживленням є другорядною задачею, яка не впливає на основну, або ж є лише частиною її. Такі представлення, існуючих систем, в тематиці підготовки забезпечення житла автономним енергоживленням, будуть розглянуті в наступних підрозділах.

2.1.1 Системи рекомендаційного типу

Першою категорією подібних систем до розроблюваної є системи рекомендаційного типу. Таких систем безліч у відкритому доступі і головною їхньою ціллю є рекомендація щодо вибору того чи іншого засобу енергоживлення.

Сюди можна віднести різноманітні інформаційні ресурси, де публікуються статті з текстовим оглядом певного засобу та інструкцією його використання. Інформація, безумовно, може бути корисною, але такого роду представлення, не здатне надати повної картини огляду щодо вибору найоптимальнішого рішення.

Іншим видом рекомендаційних ресурсів є сайти або ж додатки з можливістю введення числових значень для розрахунку результату. Наприклад, вводяться значення робочого навантаження, номінальної напруги та бажаного часу роботи джерела живлення і на виході розраховується мінімальна допустима ємність акумуляторної батареї для задоволення потреб користувача при наявних введених умовах. Такі розрахунки, або ж схожого виду, мають свою користь і з цим важко сперечатися, але розраховують лише конкретний параметр. За розрахованим параметром потрібно підібрати відповідну акумуляторну батарею, або ж будь-що інше, та налаштувати власноруч. Така система не надасть інформації, щодо доцільності вибору того чи іншого засобу, пристрою чи системи для безперебійного забезпечення енергоживленням власної оселі.

2.1.2 Системи оглядового типу

Ще одним видом систем, які широко залучені до процесу вибору того чи іншого засобу енергоживлення є сервіси з оглядовою інформацією, інструкцією або ж досвідом експлуатації. Сюди доцільно відносити різноманітні відео-хостинги, основним курсуючим джерелом інформації в яких є відео. Спрямованість відео сягає своєї широкої різноманітності та всеохопності, але і засоби для забезпечення

будівлі додатковим енергоживленням не залишаються осторонь. Оглядові відео від спеціалізованих компаній або ж від звичайних користувачів ладне надати вичерпну інформацію щодо бажаного товару і більш широко ввести в хід справи щодо поточного пристрою. Отримана інформація нестиме в собі цінність, але, знову ж таки, не надаватиме повної картини щодо всіх наявних засобів із найбільш релевантним з них під потреби споживача.

2.1.3 Системи магазинного типу

Ще одним типом систем, які не менше інших залучені до процесу підбору приладів додаткового енергоживлення для оселі є інтернет-магазини. Вони дають змогу, вкотре, ознайомитися з товаром, порівняти варіативність вибору інших користувачів та за потреби придбати його. Але перш ніж дійти до цього етапу, виникає потреба у виборі найкращого з-поміж всіх інших альтернативних приладів, що можуть пропонуватися на кожному закутку. Порівняння безліччі однотипних приладів, схожих за виглядом, але різними за функціоналом може займати забагато часу та сил, але не давати належного результату на виході. В підсумку, цінність витраченого часу не завжди може конвертуватися у користь.

2.2 Головні особливості розроблюваної системи у порівнянні з конкурентами

Для захоплення цікавості людей потрібно мати певний ряд переваг над аналогічними системами. Переваги повинні бути ключовими і чітко визначеними для наочної демонстрації всіх особливостей системи, що розробляється. Чим більше унікальних особливостей представлені в програмному продукті, тим більше

ймовірність заохочення людей до усвідомлення доцільності такого рішення для їхніх різноманітних і подекуди специфічних задач.

Після проведеного аналізу існуючих рішень на ринку можна виділити одну ключову особливість: відсутня наявність розробленого рішення, яке в собі міститиме все необхідне для підбору релевантного засобу резервного електроживлення. Таке рішення, яке поєднуватиме в собі сукупність усього найкращого, що вже мається та доповнюватиме його чимось унікальним, своїми неповторними особливостями.

Враховуючи все вищезазначене, розроблювана рекомендаційна система для підготовки оселі до блекауту матиме три базові ключові особливості, які в сукупності та в поєднанні одине з одним, як єдиний суцільний інструмент, даватимуть бажаний результат у вигляді доступності, зручності та наочності результатів для підбору найбільш релевантного засобу електроживлення.

Особливості розроблюваної системи:

- першою такою особливістю є власна база знань для розробленого проєкту, яка постійно наповнюватиметься та доповнюватиметься новими компонентами для масштабування продуктів рекомендації. Збирання та наповнення даної бази покладається на адміністраторів системи, які оброблені та формалізовані дані заноситимуть до неї. Відвідувачі ресурси завжди зможуть звернутися до інформації з даної бази знань через відповідну сторінку;
- другою особливістю є підрахунок статистики використовуваних засобів резервного електроживлення іншими користувачами, який заснований на калькуляторних розрахунках для них. Статистичні дані представлені у вигляді стовпчастої діаграми використовуваних засобів та секторної діаграми типу використовуваної енергії засобами, які найчастіше рекомендуються відвідувачам ресурсу;
- третьою і найбільш важливою особливістю розроблюваної системи є сторінка для розрахунку найрелевантнішого засобу резервного

електроживлення для оселі відвідувача ресурсу. Результат такого розрахунку ґрунтується на побажаннях самого користувача, шляхом введення бажаних значень у відповідні поля (потужність, вартість, час роботи тощо). Такий варіант допоможе за мінімальних зусиль отримати максимальний результат без зайвих вагань.

Таким чином, маючи у собі три ключові особливості, які поєднуються в одній розроблюваній системі, здатні виділитися серед наявних запропонованих рішень на ринку та забезпечити всю наявну та необхідну інформацію користувачам щодо вибору резервного джерела електроживлення. Варіативність використання ресурсу не обмежується завчасно визначеним одним напрямком роботи, а масштабується в залежності від бажань відвідувачів. Неважливо, чи буде це звичайний довідник для ознайомлення з підвищення рівня інформаційної грамотності користувачів через базу знань, графічне спостереження для прийняття рішення, на основі вибору інших користувачів чи розрахунок власного індивідуально-релевантного засобу резервного живлення оселі – система розрахована на різні варіанти використання. За мінімально можливу кількість прикладених зусиль, в одному місці, можливо отримати максимум наявної інформації щодо наявних в базі знань засобів резервного електроживлення та значно полегшити процес вибору і подальшого забезпечення власної оселі.

Висновки до розділу 2

Підбиваючи підсумки, з огляду існуючих систем щодо належної підготовки оселі до періоду блекауту, доцільно сказати, що в повному і одноманітному вигляді немає. Це пояснюється тим, що тривалий час до цього, не було гострої потреби в цьому. Існували рідко-періодичні відключення електроенергії, які ґрунтувалися на реконструкційних чи аналізуючих роботах комунальних служб. Такі відключення не створювали гострої проблеми в нагальності вирішення питання із забезпечення

власної оселі електроживленням через свою нетривалість. Відповідно страху перед зимовим періодом не було, так само, як і перед відсутністю тепла, можливості належного збереження продуктів харчування тощо.

Проте, ситуація, на даний момент, змінилась кардинально. Тому потреба в нагальному забезпеченні власної оселі механізмом резервного електроживлення все ще існує і, на превеликий жаль, не зникає. Існує, звісно, безліч різнопланових сайтів, додатків тощо, для відслідковування необхідних параметрів, розрахунку бажаних величин чи ознайомлення з товаром. Але вони ладні надати лише крупинку необхідної інформації для складання повної картини щодо поточної ситуації.

Звісно, цього може бути цілком достатньо в деяких конкретних випадках. Проте, якщо взяти цілісну картину і розглянути потреби користувачів, то можна зробити впевнений висновок, що цього буде замало. Наявність повної та одноманітної системи із зібраною усією необхідною інформацією щодо поточної проблематики в одному місці є зручним. Це зможе значно економити час користувачам та не розподіляти свою увагу на різних ресурсах з різними задачами для пошуку бажаного рішення.

Побудова системи з наявною ознайомлювальною інформацією щодо приладів для забезпечення власної оселі безперебійним енергоживленням, статистичними даними найпопулярніших видів забезпечення енергії, а також розрахунковою системою для підбору найрелевантнішого рішення за вказаними параметрами є доцільним, як мінімум, за властивістю зручності та користі.

Таке рішення, дозволило б вирішити суттєву проблему щодо вибору найкращого варіанту серед наявних за миттєвий проміжок часу і значно спростити вирішення настільки клопіткої задачі.

3 ЗАСОБИ РОЗРОБКИ

3.1 Мова програмування Python

Автором мови Python є нідерландський програміст Гвідо ван Россум. На той час він був співробітником голландського інституту CWI і працював над мовою ABC, яка мала стати максимально простою і зрозумілою декларативною мовою. Мова ABC була достатньо цікавою, але вона мала значний недолік: нею ніхто не користувався. Тому в грудні 1989 року Гвідо розпочав свій «хобі проект» Python, в основу якого лягли окремі елементи мови ABC (фактично, Python створювався як спроба виправити помилки, допущені при проектуванні ABC). Датою появи мови Python вважається 20 лютого 1991 року [5].

Python — це потужна мова програмування, яка проста у вивченні. Він має ефективні структури даних високого рівня та простий, але ефективний підхід до об'єктно-орієнтованого програмування. Елегантний синтаксис і динамічна типізація Python разом з його інтерпретованим характером роблять його ідеальною мовою для створення сценаріїв і швидкої розробки додатків у багатьох сферах на більшості платформ [6].

З'явившись порівняно пізно, Python створювався під впливом багатьох мов програмування:

- ABC – на відміну від значної кількості мов програмування, в мові ABC відразу використовувалися відступи (поля) замість операторних дужок для групування операторів. Такий же принцип був обраний і для Python;
- Modula-3 – популярна в академічних кругах, але майже без практичного програмування мова була однією з перших, де були винятки (try ... except), які і були взяті для Python;

- C, C++ – для Python були взяті деякі синтаксичні конструкції (як пише сам Гвідо ван Россум - він використовував найбільш несуперечливі конструкції з C, щоб не викликати неприязнь у C-програмістів до Python);
- Smalltalk – елементи об'єктно-орієнтованого програмування;
- Lisp – окремі риси функціонального програмування;
- Fortran – зрізи масивів, комплексна арифметика;
- Miranda – спискові вирази;
- Java – модулі logging, unittest, threading (частина можливостей оригінального модуля не реалізована), xml.sax стандартної бібліотеки, спільне використання finally та except при обробці винятків, використання @ для декораторів;
- Icon – генератори.

На даний момент існують дві актуальні версії мови: Python 2.x і Python 3.x. Це пов'язане з тим, що за свій час «життя» мова Python 2.x акумулювала певну кількість недоліків, які не можна було виправити, не зламавши чийсь код. Основний недолік пов'язаний з рядками (спочатку рядки були байтовими, потім було додано підтримку Юнікоду (Unicode) з неявним приведенням).

Тому в більшості випадків Python 2.x використовується для забезпечення сумісності з раніше написаними програмами. Проте підтримка гілки 2.x була завершена в січні 2020 року.

Python 3.0 обернено не сумісний з попередньою серією 2.x. Код Python 2.x швидше за все буде видавати помилки при виконанні в Python 3.x. Динамічна типізація Python, разом з планами зміни декількох методів словників, робить механічне портування з Python 2.x в Python 3.x дуже складним [5].

Мову Python можна охарактеризувати як: інтерпретовану високорівневу мову програмування загального призначення. В різній літературі можна зустріти набагато більше складових мови, які вказують при її описі, тому просто сформулюємо її особливості [5].

Інтерпретатор Python легко розширюється за допомогою нових функцій і типів даних, реалізованих у C або C++ (або інших мовах, які можна викликати з C). Python також підходить як мова розширення для налаштування програм [6].

Читання хорошої програми на Python дуже нагадує читання англійського тексту, хоча і досить строгого. Така псевдокодова природа Python є однією з його найсильніших сторін. Python надає можливість програмісту зосередитися на розв'язуванні задачі, а не на самій мові. Ядро мови без додаткових модулів досить просте і мінімальне - це надає мові простоти та логічності, що відповідно спрощує її вивчення [5].

Мовою Python надзвичайно легко почати програмувати. Python має виключно простий синтаксис. Досить часто в закладах освіти обирають Python як першу мову програмування для своїх студентів.

Python - це приклад вільного і відкритого програмного забезпечення – FLOSS (Free / Libre and Open Source Software). Простіше кажучи, ви маєте право вільно використовувати копії цього програмного забезпечення, переглядати його вихідні тексти, вносити зміни, а також використовувати його частини в своїх програмах. Python був створений і постійно поліпшується співтовариством, яке просто хоче зробити його краще [5].

Машина (комп'ютер, процесор) розуміє лише певний обмежений та примітивний набір спеціальних команд. Коли пишеться код деякою мовою, по суті, вказуються інструкції, які повинна виконати машина. В залежності від того, які ці інструкції, мови і поділяються на високорівневі та низькорівневі. Якщо код пишеться в машинних інструкціях (командах), то це низькорівневі мови. Якщо ж додатково необхідне певне програмне забезпечення, яке буде переводити код, написаний мовою програмування, в машинні інструкції (команди) то це високорівнева мова програмування. Програмне забезпечення, за яким відбувається вказане переведення, називається компіляторами або інтерпретаторами [5].

Завдяки своїй відкритій природі, Python був портований на багато платформ (змінений таким чином, щоб працювати на різних пристроях та різних операційних системах). Один і той самий код буде запускатися на різних платформах без будь-яких змін за умови, що в коді відсутні системно-залежні функції.

Як зазначалося раніше, для виконання програми на комп'ютері необхідно команди мови програмування перетворити у команди, які здатен виконувати процесор комп'ютера. Існує два типи таких перетворень – інтерпретація і компіляція, які здійснюються спеціальними програмами-трансляторами [5].

Програма, написана компільованою мовою програмування, як наприклад, Pascal або C++, аналізується і при відсутності помилок перетворюється, за допомогою компілятора, в мову, зрозумілу комп'ютеру (бінарний код), після чого записується в оперативну пам'ять комп'ютера або на диск. Утворений на диску файл є виконуваним файлом і може бути виконаний засобами операційної системи без повторного використання транслятора.

При інтерпретації транслятор аналізує кожну команду програми окремо і, якщо команда не містить синтаксичних помилок, перетворює її в машинний код і відразу виконує [8]. Проте в Python цей процес дещо складніший. Після запуску програми Python аналізує кожну команду програми і, якщо команда не містить синтаксичних помилок, перетворює її в байт-код (машино-незалежний код низького рівня). Далі байт-код передається віртуальній машині Python, яка і виконує інструкції байт-коду, перетворюючи їх в машинний код [5].

Коли байт-код сформовано, його можна виконати на різних інтерпретаторах, що підтримують цей байт-код. Так, для Python можна виділити такі інтерпретатори:

- CPython – інтерпретатор, реалізований на C. Вважається еталонною реалізацією Python і підтримує більшість активно використовуваних платформ;
- IronPython – Python для .NET Framework і Mono. Компілює Python програми в MSIL (Microsoft Intermediate Language), таким чином надаючи повну інтеграцію з .NET-системою;

- Jython – реалізація Python, що використовує JVM (Java Virtual Machine);
- Stackless – патчі до CPython, який надає розширені можливості багатопотокового програмування і значно більшу глибину рекурсії;
- PyPy – реалізація Python, написана на RPython. В PyPy вбудований трасуючий JIT-компілятор (Just-in-time compilation), який може перетворювати код Python в машинний код під час виконання програми [5].

В підсумку, можна виділити 5 основних переваг Python:

- легка читаність коду. При реалізації високонавантажених проєктів, над якими працюють відразу кілька програмістів, читабельність коду грає вирішальне значення, тому що кожен розробник повинен розуміти, що роблять його колеги. Python диктує суворі вимоги до оформлення коду і влаштована таким чином, щоб мінімізувати кількість рядків. Тому читати її завжди зручно і легко;
- широка область застосування. Python використовується для розробки онлайн і мобільних додатків, машинного навчання, при створенні ігор і зручна для автоматизації математичних розрахунків. Тому її можна використовувати для вирішення практично будь-яких завдань;
- гарне ком'юніті і стійкість. Мова Python може похвалитися великим фан-клубом розробників і підтримується гігантами IT-індустрії, такими, як Google, Facebook і Spotify;
- портативність. У багатьох мовах програмування, для запуску програми на різних платформах потрібно вносити зміни в код. Python працює по іншому. Написавши код один раз, ви зможете запускати його на будь-яких пристроях без додаткових коригувань;
- велика база бібліотек. Працюючи з Python, розробник не залежить від зовнішніх бібліотек. Мова оснащена потужною стандартною базою

функцій, які можна використовувати для вирішення комерційних завдань, що істотно спрощує роботу програміста [7].

Багатогранність мови Python робить її затребуваною у багатьох сферах бізнесу. Найчастіше її застосовують:

- у системному адмініструванні. Тут Python дозволяє автоматизувати більшість процесів. Вона проста, надійна і підтримує додаткові пакети, що підвищують ефективність роботи. До того ж, ця мова є встановленою на всіх серверах, що працюють на ОС Linux;
- у наукових дослідженнях. Мова Python проста в освоєнні, і володіє бібліотеками, призначеними спеціально для дослідницьких робіт, наприклад SciPy, Matplotlib і NumPy. Це робить мову надзвичайно популярною серед вчених, особливо в областях точних наук – фізики і математики;
- в Data Science. Мова Python застосовується для написання алгоритмів, використовуваних програмами, пов'язаними з машинним навчанням і ШІ. Крім того, мова застосовується різними сервісами для хмарного зберігання, парсингу і обробки даних. Зокрема, корпорація Google використовує Python для індексації веб-ресурсів;
- у веб розробці. У веб-програмуванні для роботи з Python часто підключають готові фреймворки, включаючи Pyramid, Pylons, TurboGears, Flask, CherryPy. Найпопулярнішим з них є Django [7].

3.2 Фреймворк Django

Django – веб-фреймворк повного стека, який підходить для великих і складних проєктів. Допомогає швидко та чисто зробити веб-проєкт. Створено і підтримується сотнею досвідчених веб-розробників. Згідно з документацією Django деякі з перерахованих у ній функцій максимально швидкі, надійно захищені, надзвичайно масштабовані та універсальні. Django підтримує роботу з основними базами даних – MySQL, MariaDB, SQLite, PostgreSQL та Oracle. Особливості Django – це його автентифікація, маршрутизація URL-адрес, механізм шаблонів, об'єктно-реляційна проєкція (ORM) та міграція схем БД [9].

За допомогою Django можливо перевести веб-додатки від концепції до запуску за лічені години. Django бере на себе більшу частину клопоту веб-розробки, тому розробнику можна зосередитися на написанні свого додатка без потреби винаходити колесо. Фреймворк є безкоштовним та з відкритим кодом [10].

Словом, Django – це фреймворк, який полегшує створення веб-сайтів за допомогою Python. Він бере на себе найскладніші завдання, щоб розробники могли зосередитися на створенні власних веб-додатків [11].

Django наголошує на багаторазовому використанні компонентів, що також називається DRY (Don't Repeat Yourself), і постачається з готовими до використання функціями, такими як механізм входу в систему, підключення до бази даних та CRUD-операції (Create Read Update Delete – створення, читання, оновлення, видалення). Django особливо корисний для веб-сайтів, керованих базами даних [11]. Фреймворк допоможе розробити CRUD додаток під ключ. З Django не доведеться винаходити велосипед. Він працює з коробки і дозволить зосередитися на бізнес-логіці і продуктах для звичайних людей [12].

Django надає багату основу для полегшеного створення форм та подальшого маніпулювання даними з форм, наявність безлічі різноманітних компонентів та інструментів, які допомагають у розробці та тестуванні додатків, має «з коробки» автоматизований інтерфейс панелі адміністратора, яка автоматично генерується під

час створення додатку. Окрім цього, пропонує власну систему локалізації, яка значно спрощує процес розробки програм для різних мов. А також безпека є темою першочергової важливості при розробці веб-застосунків і Django здатен надати величезну розсип інструментів та механізмів для організації захисту вже за замовчуванням.

Django серйозно ставиться до безпеки та допомагає розробникам уникати багатьох типових помилок безпеки, таких як SQL-ін'єкція, міжсайтовий скриптинг, підробка міжсайтового запиту та клікджекінг. Його система автентифікації користувачів забезпечує безпечний спосіб керування обліковими записами та паролями користувачів [10].

Також Django містить величезну кількість функціональності для вирішення більшості завдань веб-розробки. Фреймворк задає структуру проекту, яка допомагає розробникам розуміти, де і як додавати нову функціональність. Завдяки однакової для всіх проектів структурі набагато простіше знайти вже готові рішення або отримати допомогу від спільноти [12].

Функціональність Django можливо розширити за допомогою додаткових плагінів, які представляються у вигляді програмних модулів, які надають змогу швидко додати на сайт необхідні функції.

Фреймворк пропонує інструменти по типу:

- аутентифікації;
- логування;
- кешування;
- пагінації;
- валідації даних;
- сесій;
- управління статичними файлами;
- карт сайту (sitemaps).

Дані інструменти є вкрай важливими та необхідними для розробників при створенні веб-застосунків.

Також, фреймворк Django, за замовчуванням, здатен захистити від найбільш розповсюджених мережових небезпек на сайтах:

1. Захист від міжсайтових скриптів (*англ.* cross site scripting, XSS) – XSS-атаки дозволяють користувачеві впроваджувати клієнтські скрипти в браузері інших користувачів. Зазвичай це досягається шляхом зберігання шкідливих скриптів у базі даних, звідки вони будуть завантажені та показані іншим користувачам, або шляхом спонукання користувачів натиснути на посилання, яке призведе до виконання JavaScript зловмисника у браузері користувача. Однак XSS-атаки можуть походити з будь-якого ненадійного джерела даних, такого як файли cookie або веб-сервіси, коли дані не проходять достатню санітарну обробку перед включенням на сторінку. Використання шаблонів Django захищає користувачів від більшості XSS-атак. Однак, важливо розуміти, який захист він забезпечує і які його обмеження. Шаблони Django уникають специфічних символів, які є особливо небезпечними для HTML. Хоча це і захищає користувачів від більшості шкідливого введення, але не є повністю безвідмовним.
2. Захист від підробки міжсайтових запитів (*англ.* cross site request forgery, CSRF) – CSRF-атаки дозволяють зловмиснику виконувати дії з використанням облікових даних іншого користувача без його відома або згоди. Django має вбудований захист від більшості типів CSRF-атак, за умови, що ви ввімкнули і використовуєте його там, де це доречно. Однак, як і у випадку з будь-якою технікою захисту, існують обмеження. Наприклад, можна вимкнути модуль CSRF глобально або для певних представлень (views). Захист CSRF працює шляхом перевірки наявності секрету в кожному POST-запиті. Це гарантує, що зловмисник не зможе "відтворити" форму POST на сайті і змусити іншого користувача, який

- увійшов до системи, мимоволі відправити цю форму. Зловмисник повинен знати секрет, який залежить від користувача (за допомогою файлу cookie).
3. Захист від SQL-ін'єкцій – SQL-ін'єкція – це тип атаки, коли зловмисник може виконати довільний SQL-код у базі даних. Це може призвести до видалення записів або витіку даних. Набори запитів Django захищені від SQL-ін'єкцій, оскільки вони побудовані з використанням параметризації запитів. SQL-код запиту визначається окремо від параметрів запиту. Оскільки параметри можуть бути надані користувачем і тому небезпечні, вони екрануються драйвером баз даних. Django також надає розробникам можливість писати необроблені запити або виконувати користувацький sql.
 4. Захист від клікджекінгу (*англ.* clickjacking) – це тип атаки, коли шкідливий сайт обгортає інший сайт у фрейм. Ця атака може призвести до того, що користувач, який нічого не підозрює, буде обманом змушений виконати небажані дії на цільовому сайті. Django містить захист від перехоплення кліків у вигляді проміжного програмного забезпечення X-Frame-Options, яке у відповідному браузері може запобігти відображення сайту у фреймі. Захист можна вимкнути для кожного представлення (view) або налаштувати точне значення заголовка, що надсилається. Проміжне програмне забезпечення настійно рекомендується для будь-якого сайту, якому не потрібно, щоб його сторінки були загорнуті у фрейм сторонніми сайтами, або потрібно дозволити це лише для невеликої частини сайту.
 5. Перевірка заголовка хосту – Django використовує заголовок Host, наданий клієнтом, для побудови URL-адрес у певних випадках. Хоча ці значення очищуються для запобігання атакам міжсайтового скриптингу, фальшиве значення хосту може бути використане для підробки міжсайтових запитів, атак отруєння кешу та отруєних посилань в електронних листах. Оскільки навіть на перший погляд безпечні конфігурації веб-серверів вразливі до фальшивих заголовків хостів, Django перевіряє заголовки хостів на

відповідність параметру `ALLOWED_HOSTS` у методі `django.http.HttpRequest.get_host()`.

6. Політика щодо посилань – браузери використовують заголовок `Referer` як спосіб надіслати на сайт інформацію про те, як користувачі потрапили на нього. Встановивши політику рефералів, розробники можуть допомогти захистити конфіденційність користувачів, обмеживши, за яких обставин встановлюється заголовок `Referer`.
7. Політика перехресного відкриття (*англ.* cross-origin opener policy, COOP) – заголовок політики відкриття перехресних джерел (COOP) дозволяє браузерам ізолювати вікно верхнього рівня від інших документів, поміщаючи їх в іншу контекстну групу, щоб вони не могли безпосередньо взаємодіяти з вікном верхнього рівня. Якщо документ, захищений COOP, відкриває спливаюче вікно перехресного походження, властивість `window.opener` цього спливаючого вікна буде дорівнювати нулю. COOP захищає від перехресних атак.
8. Безпека сесій – подібно до обмежень CSRF, які вимагають розгортання сайту таким чином, щоб ненадійні користувачі не мали доступу до субдоменів, `django.contrib.sessions` також має обмеження [10].

Головні переваги фреймворку Django, які значно допомагають у процесі розробки.

1. Швидкість розробки. Django допомагає швидко просувати проєкт від ідеї до робочого релізу продукту.
2. Універсальність. В Django входять ключові інструменти веб-розробки: CRM, міграція баз даних, аутентифікація користувачів, форми, адмін панель. У багатьох інших фреймворків для роботи з цими функціями потрібно підключати окремі бібліотеки.
3. Гнучкість. Джанго – дуже легко налаштувати. Замість стандартної бази даних, CRM та системи шаблонів можна підключити свої інструменти.
4. Адмін панель і аутентифікація «з коробки».

5. Модулі та додатки. До проектів на Django можна підключати готові модулі.
6. Найбільше співтовариство серед усіх фреймворків Python-a.
7. Архітектура. Джанго слідує паттерну проектування моделі-вид-контролер (MVC). Він має на увазі розмежування бізнес-логіки і її візуалізації. Це дозволяє розробляти компоненти паралельно, змінювати їх окремо. А також по-різному представляти дані, в залежності від контексту.
8. Масштабованість. Django підходить для створення високонавантажених сайтів з великим трафіком.
9. Безпека. Django підтримує захист від таких видів атак, як впровадження SQL-коду і міжсайтових підробок запитів.
10. Створення API. Підтримка інструментарію для створення API: REST API і GraphQL.
11. DRF, або Django REST Framework, спрощує створення просунутих API. За допомогою RESTful API до одного сервера можна підключити декілька оболонок. Наприклад, браузерні фронтенд на Vue, і мобільні додатки для Android і iOS [13].

3.3 База даних PostgreSQL

PostgreSQL — це потужна об'єктно-реляційна база даних з відкритим вихідним кодом, яка використовує та розширює мову SQL у поєднанні з багатьма функціями, які безпечно зберігають і масштабують найскладніші дані [14].

PostgreSQL заслужив міцну репутацію завдяки своїй перевірній архітектурі, надійності, цілісності даних, надійному набору функцій, розширюваності та відданості спільноти з відкритим кодом, яка стоїть за програмним забезпеченням, щоб постійно надавати ефективні та інноваційні рішення [14].

Дана система керування БД має багато функцій, які допомагають розробникам створювати додатки, адміністраторам — захищати цілісність даних і створювати відмовостійке середовище та допомагати керувати своїми даними, незалежно від того, наскільки великий чи малий їх набір. Окрім того, що PostgreSQL є безкоштовним і з відкритим вихідним кодом, він дуже розширюваний. Наприклад, надає можливість визначати власні типи даних, створювати власні функції, навіть писати код з різних мов програмування без перекомпіляції бази даних [14].

PostgreSQL намагається відповідати стандарту SQL, якщо така відповідність не суперечить традиційним функціям або може призвести до невдалих архітектурних рішень. Багато функцій, необхідних стандарту SQL, підтримуються, хоча іноді з дещо відмінним синтаксисом або функціями [14].

Нижче наведено вичерпний перелік різноманітних можливостей, які можна знайти в PostgreSQL, і які додаються з кожним новим випуском:

1. Типи даних:

- примітиви: цілі, числові, рядкові, логічні;
- структуровані: дата/час, масив, діапазон/мультидіапазон, UUID;
- документні: JSON/JSONB, XML, ключ-значення (Hstore);
- геометрія: точка, лінія, коло, багатокутник;
- налаштування: складені, кастомні типи.

2. Цілісність даних:

- UNIQUE, NOT NULL;
- первинні ключі;
- зовнішні ключі;
- обмеження виключення;
- явні блокування, рекомендаційні блокування.

3. Паралелізм, продуктивність:

- індексування: Б-дерево, багатостовпчикове, виразове, часткове;

- розширене індексування: GiST, SP-Gist, KNN Gist, GIN, BRIN, індекси покриття, фільтри Блума;
- складний планувальник/оптимізатор запитів, сканування лише за індексами, багатоклонкова статистика;
- транзакції, вкладені транзакції (через точки збереження) ;
- багатоверсійний контроль паралелізму (MVCC) ;
- розпаралелювання запитів на читання та побудова індексів Б-дерева;
- розбиття таблиць на розділи;
- всі рівні ізоляції транзакцій, визначені в стандарті SQL, включаючи Serializable;
- компіляція виразів Just-in-time (JIT).

4. Надійність, аварійне відновлення:

- журналювання з випередженням запису (Write-ahead Logging, WAL) ;
- реплікація: Асинхронна, Синхронна, Логічна;
- відновлення в момент часу (PITR), активні резерви;
- табличні простори.

5. Безпека:

- аутентифікація: GSSAPI, SSPI, LDAP, SCRAM-SHA-256, сертифікат та інші;
- надійна система контролю доступу;
- безпека на рівні стовпців і рядків;
- багатфакторна автентифікація за допомогою сертифікатів та додаткового методу.

6. Розширюваність:

- збережені функції та процедури;

- процедурні мови: PL/pgSQL, Perl, Python та Tcl. Існують інші мови, доступні через розширення, наприклад, Java, JavaScript (V8), R, Lua та Rust;
- конструктори SQL/JSON та вирази шляхів;
- зовнішні обгортки даних: підключення до інших баз даних або потоків за допомогою стандартного інтерфейсу SQL;
- налаштовуваний інтерфейс зберігання таблиць;
- багато розширень, які надають додаткову функціональність, включаючи PostGIS.

7. Інтернаціоналізація, текстовий пошук:

- підтримка міжнародних наборів символів, наприклад, через зіставлення ICU;
- нечутливі до регістру та наголосу зіставлення;
- повнотекстовий пошук.

PostgreSQL має високу масштабованість як щодо величезної кількості даних, якими він може керувати, так і щодо кількості одночасних користувачів, яких він може обслуговувати. У виробничих середовищах є активні кластери PostgreSQL, які керують багатьма терабайтами даних, і спеціалізовані системи, які керують петабайтами [14].

3.4 Середовище розробки VS Code

VS Code є досить легким, але при тому, доволі потужним редактором програмного коду та доступний для всіх наявних десктопних платформ. Має в своєму арсеналі підтримку для незліченної кількості популярних мов програмування. Ця підтримка полягає базові аспекти, такі як підсвічування синтаксису мови, позиціонування та зіставлення дужок, згортання готових блоків коду тощо.

Також, не менш важливою особливістю даного середовища розробки коду є можливість його внутрішнього наповнення під специфічні потреби розробника шляхом спеціальних розширень. Не зайвою виявиться можливість керування версіями розробленого програмного забезпечення, що надасть змогу переглянути внесені зміни до коду та вплине на прийняття найоптимальнішого рішення для продовження розробки. Для функціональності керування версіями є необхідність у зв'язаному акаунті з будь-якою підтримуваною системою керування версіями (Git, Apache Subversion, Perforce тощо).

Visual Studio Code містить декілька розширень для FTP (File Transfer Protocol), що дозволяє використовувати його як безкоштовну альтернативу для веб-розробки. Код можна синхронізувати між редактором і сервером без завантаження додаткового програмного забезпечення.

Окрім цього наявна можливість встановлювати кодування сторінки, на якій буде збережено активний документ, визначити бажаний символ нового рядка та мову програмування поточного документа.

Всі ці наявні особливості дозволяють з мінімальними затратами сил та часу для налаштування виконувати всі необхідні базові роботи щодо розробки бажаного програмного забезпечення. Зручність у використанні у момент розробки дозволить легко реалізувати всі задуми та в майбутньому підтримувати розроблений застосунок безперешкодно.

Висновки до розділу 3

Для розробки потужного та, в своєму роді, інноваційного рішення потрібно використовувати найкращі інструменти для досягнення поставлених цілей. Простота в розробці та надійність в подальшому використанні на бойовому сервері можуть наблизити до омріяних цілей.

В даній роботі використовувалися високорівнева мова програмування Python, яка цілковито та повністю здатна забезпечити базові математичні обчислення для виконання необхідної роботи. Під її керівництвом, систематизувати структура файлів та папок розроблюваного проєкту вдалося зв допомогою потужного фреймворку Django. Даний фреймворк дозволяє виконати базове налаштування веб додатку для спрощення побудови нових та обширних функцій в майбутньому. Не в останню чергу, корисною і не менш важливою необхідністю стала база даних PostgreSQL, яка здатна забезпечити високу функціональність при роботі з величезними об'ємами даних, які в майбутньому, стануть нормою для рекомендаційної системи.

4 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

4.1 Модель MVT

Django слідує шаблону проєктування MVT (Model View Template) (рис. 4.1) [11]:

1. Модель (Model) – дані, які потрібно представити, зазвичай це дані з БД.
2. Представлення (View) – обробник запиту, який повертає відповідний шаблон і вміст на основі запиту користувача.
3. Шаблон (Template) – текстовий файл (наприклад, HTML-файл), що містить макет веб-сторінки з логікою відображення даних.

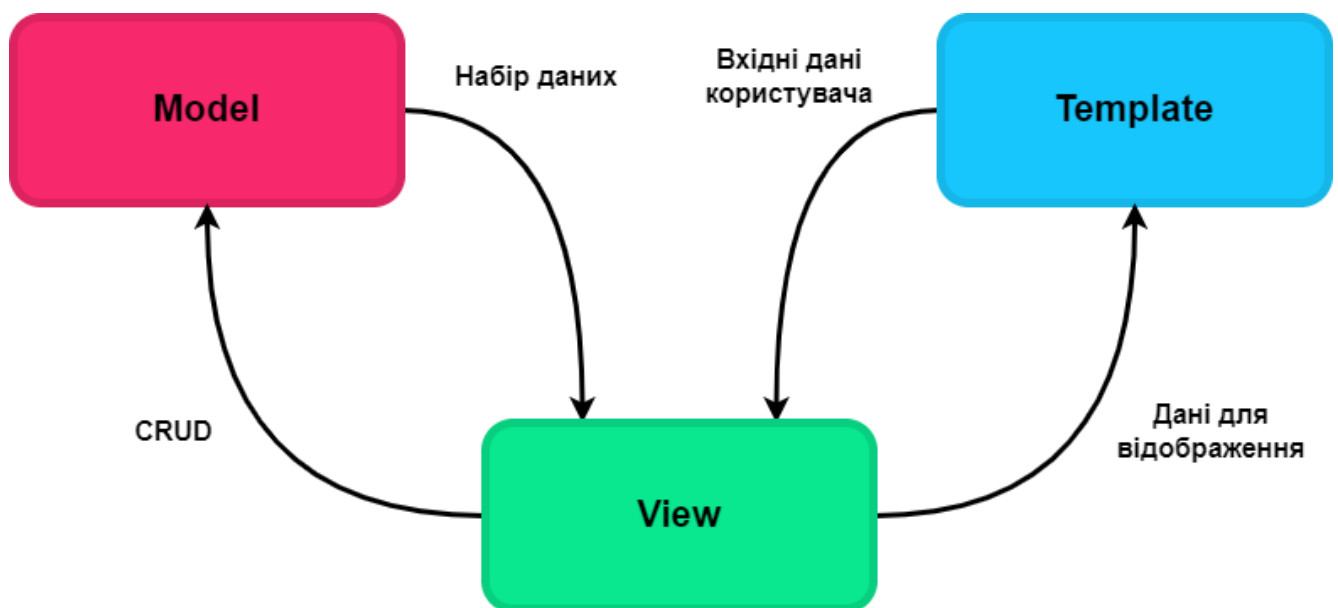


Рисунок 4.1 – Структура шаблону проєктування MVT

Модель надає дані з бази даних. У Django дані надаються у вигляді об'єктно-реляційного відображення (*англ.* Object Relational Mapping, ORM), що є технікою, призначеною для полегшення роботи з базами даних.

Найпоширенішим способом вилучення даних з бази даних є SQL. Однією з проблем SQL є те, що потрібно мати досить гарне розуміння структури бази даних, щоб мати можливість працювати з нею. Django, завдяки ORM, полегшує спілкування з базою даних, не вимагаючи написання складних SQL-запитів. Моделі зазвичай знаходяться у файлі з назвою `models.py`.

Представлення – це функція або метод, який приймає http-запити в якості аргументів, імпортує відповідну модель, з'ясовує, які дані потрібно відправити шаблону, і повертає кінцевий результат. Зазвичай вони знаходяться у файлі з назвою `views.py`.

Шаблон – це файл, в якому описується, як має бути представлений результат. Шаблони часто є html-файлами з кодом, що описує макет веб-сторінки, але вони можуть бути і в інших форматах для представлення інших результатів. Django використовує стандартний HTML для опису макета та застосовує теги Django для додавання логіки (рис. 4.2). Шаблони програми знаходяться в папці з назвою `templates`.

```
<h1>Домашня сторінка</h1>

<p>Мене звати {{ firstname }}.</p>
```

Рисунок 4.2 – Шаблон сторінки Django-проєкту з власною прописаною логікою

Django також надає спосіб навігації по різних сторінках веб-сайту. Коли користувач запитує URL-адресу, Django вирішує, до якого представлення його відправити. Це робиться у файлі з назвою `urls.py`.

На практиці, коли браузер запитує URL-адресу, відбувається наступне:

1. Django отримує URL, перевіряє файл `urls.py` і викликає представлення, яке відповідає URL.

2. У представленні, розташованому у `views.py`, перевіряється наявність відповідних моделей.
3. Моделі імпортуються з файлу `models.py`.
4. Потім представлення надсилає дані до вказаного шаблону в папці шаблонів.
5. Шаблон містить теги HTML і Django, і разом з даними повертає браузеру готовий HTML-контент.

4.2 Опис алгоритму роботи програми

Алгоритм роботи програми описується блок-схемою зображеною на рисунку 4.3. Описується наочно саме процес розрахунку найкращого засобу резервного електроживлення серед наявних в БД.

Процес роботи даного алгоритму починається в момент введення даних характеристик бажаного користувачем засобу на сторінці розрахунку. Після цього відбувається вибір всіх наявних засобів з бази знань ресурсу за визначеними критеріями користувачем. Робота алгоритму на даному етапі може завершитися, при умові, що таких засобів просто не існують за визначеними параметрами. У зворотному випадку, якщо є один та більше наявних засобів в базі даних, які відповідають критеріям вибору, то вони, так би мовити, «витягаються» з бази у вигляді масиву даних для проведення подальшого порівняння.

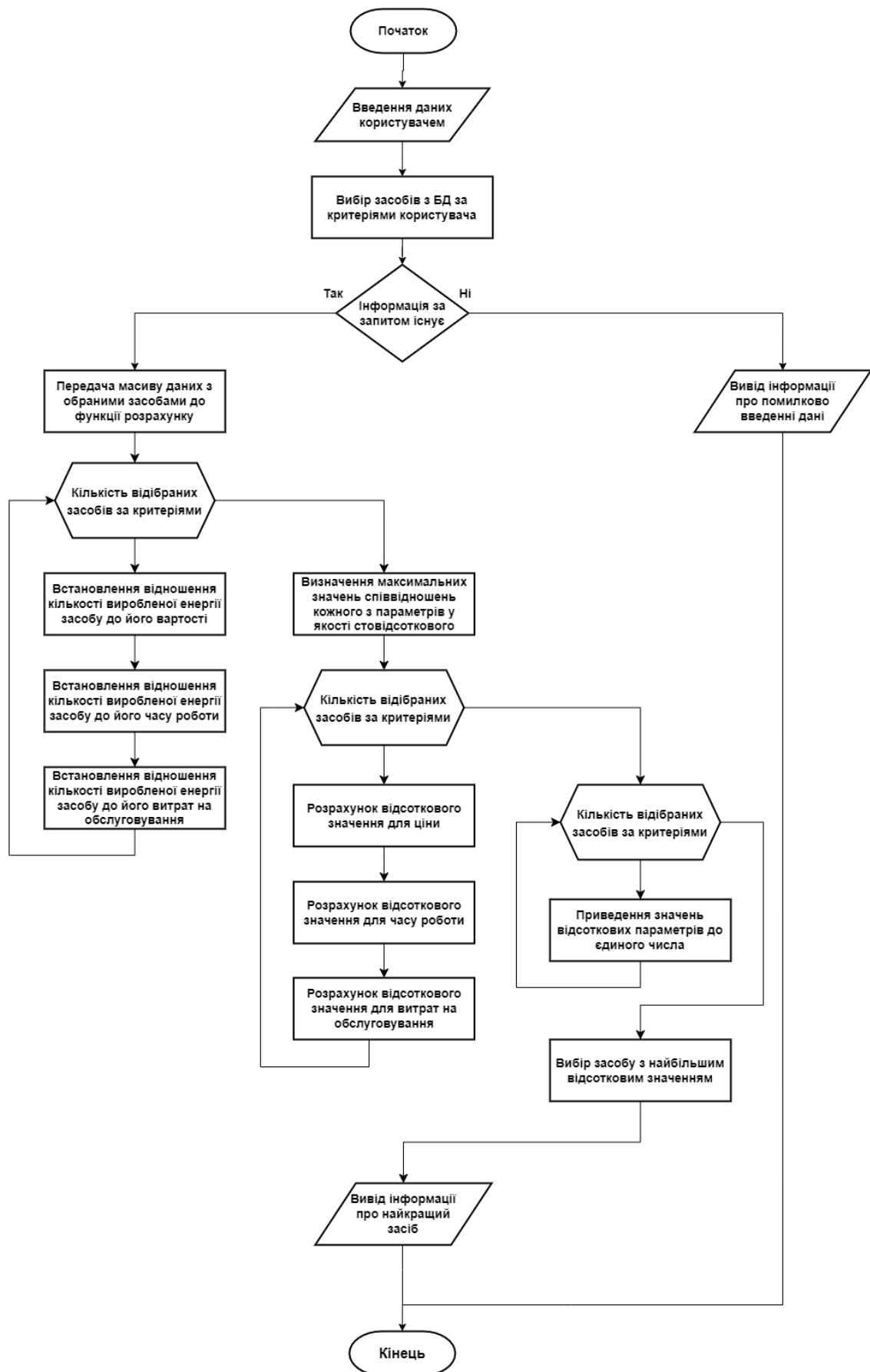


Рисунок 4.3 – Алгоритм роботи програми

Дані передаються до спеціальної функції розрахунку, де циклічно оброблюються дані всіх наявних засобів з вибірки. На кожній ітерації циклу за трьома числовими параметрами засобів їх привласнюється значення відношення кількості виробленої енергії до самого параметру. Оскільки енергія є важливим фактором у виборі засобу резервного електроживлення, тому доцільно всі розрахунки проводити з прив'язкою до цієї кількісної характеристики пристроїв. А для зменшення рівня неточності доцільно проводити порівняння шляхом приведення значень у вигляді відношень, до параметрів вартості, часу роботи та одноразових витрат. Таким чином, можна буде наочно порівняти засоби резервного електроживлення різних класів без урахування їхньої потужності та призначення.

Далі обираються найбільші значення за кожним із критеріїв та ставиться у відповідність до стовідсоткового значення, оскільки це значення є найкращим серед наявних, тому його доцільно вважати ідеально релевантним у порівнянні з наявними засобами резервного електроживлення. Потім в черговому циклі, на кожній ітерації відбувається розрахунок відсоткового значення для інших параметрів пристрою, які не є стовідсотковими.

Після даної частини виконання алгоритму миттю відбувається перехід до наступної частини блок-схеми: приведення значень відсоткових параметрів до єдиного числа. Тобто розрахунок для кожного засобу середнього арифметичного за трьома характеристиками для зручного порівняння обраних засобів між собою.

На вирішальному етапі алгоритму відбувається порівняння всіх наявних обраних засобів за критеріями за відсотковим значенням розрахованим на попередніх етапах. Обирається, в підсумку, засіб з найбільшим значенням відсотку, який і ототожненням релевантності засобу за запитом користувача.

4.3 Серверна частина програмного продукту

Серверна частина програмного продукту складається з двох ключових частин: коду на мові програмування Python та бази даних PostgreSQL. Для зазначеної мови використовувався спеціалізований фреймворк Django для побудови серверної частини, де зазначається різноманітна та функціональна логіка роботи сайту.

Частина, яка функціонує на сервері і виконує непомітні для користувача дії, але достатньо важливі для належної роботи застосунку має своєрідну структуру (рис. 4.4).

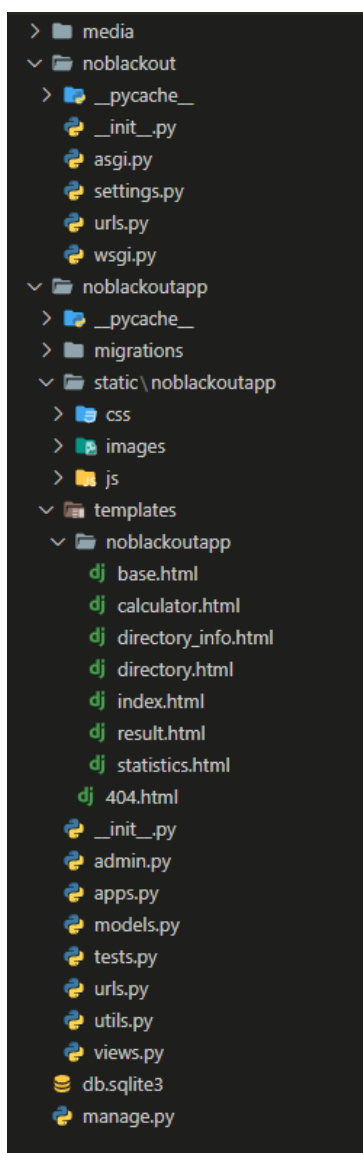


Рисунок 4.4 – Структура папок та файлів програмного продукту

Розглянемо основні ключові папки та файли із зазначеної структури на рисунку 4.4:

- папка `media` – місце, де зберігаються медіа даних для різноманітних засобів енергоживлення. В базі даних вказується шлях до цієї папки з відповідним ім'ям файлу для потрібного засобу;
- файл `settings.py` – потрібен для налаштування параметрів, що контролюють загальну поведінку розробленого проєкту. Тут представляються різноманітні налаштування: бази даних, статичних і медіа-файлів, безпеки, роботи з кешем тощо;
- папка `migrations` – необхідна для керування та внесення змін у структуру бази даних. Зберігання історії змін схеми бази даних;
- папка `static` – потрібна для зберігання статичних файлів: `.css`, `.js`, зображення, шрифти та інші ресурси, які можуть знадобитися для побудови динамічних сторінок додатку;
- папка `templates` – зберігає в собі шаблони файлів `.html`, що використовуються для відображення веб-сторінок;
- файл `admin.py` – відповідає за налаштування адміністративного інтерфейсу вед-додатку. Дозволяє реєструвати моделі розробленого додатку, щоб зробити їх доступними для моніторингу або ж редагування в адмін-панелі;
- файл `models.py` – дозволяє визначити структуру бази даних для розроблюваного проєкту. В даному файлі представляється опис моделей (таблиць), які використовуватимуться для зберігання та взаємодії з інформацією з бази даних;
- файл `urls.py` – використовується для визначення маршрутів (URL-шляхів) та їх приналежності до відповідного представлення (`views.py`). Цей файл визначає, як HTTP-запити мають бути оброблені та які сторінки повинні

бути відображенні користувачам при відвідуванні тієї чи іншої сторінки сайту;

- файл `utils.py` – використовується для зберігання корисних функцій та класів, які можуть бути використанні в різних частинах проєкту. Таким чином, зменшує в рази дублювання однотипного коду;
- файл `views.py` – використовується для визначення представлень розробленого веб-додатку. Розуміння представлень у Django полягає в тому, що вони відповідають за обробку HTTP-запитів від користувачів та у відображенні, в подальшому, відповідей на ці ж запити. Тобто, представлення можна назвати центром серверної частини, адже відповідають за обробку всієї прописаної логіки, взаємодії з моделями, шаблонами та користувачем;
- файл `manage.py` – використовується виключно для керування та управління Django-проєктом та не вимагає різноманітних сторонніх дописів від розробників. Деякі з найбільш корисних функцій: створення нових додатків проєкту, запуск локального сервера для розробки, створення адміністративного користувача, виконання міграцій тощо.

У файлі `models.py` визначення структури таблиці бази даних для відображення приладів резервного електроживлення виглядає наступним чином (рис. 4.5).

```

27 class Directory(models.Model):
28     title = models.CharField(max_length=255, verbose_name="Заголовок")
29     slug = models.SlugField(max_length=255, unique=True, db_index=True, verbose_name="URL")
30     content = models.TextField(blank=False, verbose_name="Опис")
31     price = models.FloatField(max_length=10, blank=False, verbose_name="Приблизна ціна, грн")
32     one_time_expense = models.FloatField(max_length=10, blank=False, verbose_name="Приблизні витрати на час безперебійної роботи, грн")
33     time_to_work = models.FloatField(max_length=10, blank=False, verbose_name="Приблизний час безперебійної роботи, год")
34     energy = models.FloatField(max_length=10, blank=False, verbose_name="Приблизна кількість виробленої енергії, кВт/год")
35     night_mode_work = models.BooleanField(default=False, verbose_name="Нічний режим роботи")
36     inside_apartment = models.BooleanField(default=False, verbose_name="Можливість розміщення в квартирі")
37     image = models.ImageField(upload_to="photos/%Y/%m/%d/", blank=False, verbose_name="Зображення")
38     usage_quantity = models.IntegerField(blank=False, default=0, verbose_name="Кількість використання")
39     is_published = models.BooleanField(default=True, verbose_name="Публікація")
40     per_cent = models.DecimalField(max_digits=5, decimal_places=2, null=True, blank=True)
41     type_of_energy = models.ForeignKey('categoryOfEnergy', on_delete=models.PROTECT, verbose_name="Тип енергії")
42
43
44     def __str__(self):
45         return self.title
46
47     def get_absolute_url(self):
48         return reverse('directory_info', kwargs={'directory_slug': self.slug})
49
50     class Meta:
51         verbose_name = "Довідник"
52         verbose_name_plural = "Довідники"
53         ordering = ['title']

```

Рисунок 4.5 – Структура таблиці для зберігання засобів резервного електроживлення в базі даних

За замовчуванням, проєкт на Django працює з простою вбудованою системою управління базами даних SQLite3. Вона здатна забезпечити базові потреби розробникам при побудові власного проєкту. Проте, для більш розгорнутого рішення з масштабним охопленням, доцільно використовувати потужніші системи керування БД: PostgreSQL.

За допомогою пакетного менеджера Python з відповідною командою: `pip install`, встановлюються необхідні компоненти спеціалізованої бібліотеки для коректної роботи системи керування БД PostgreSQL, які мають загальну назву – `psycopg2`.

Після чого, переведення проєкту на функціональність PostgreSQL відбувається у файлі `settings.py` (рис. 4.6).

```

88 DATABASES = {
89     'default': {
90         'ENGINE': 'django.db.backends.postgresql',
91         'NAME': 'brdb',
92         'USER': 'postgres',
93         'PASSWORD': 'Lbgkjvyf1Hj,jnf',
94         'HOST': 'localhost',
95         'PORT': '5432',
96     }
97 }

```

Рисунок 4.6 – Підключення системи керування БД PostgreSQL до Django-проєкту

На рисунку 4.6 відповідним чином зазначаються дані для підключення до БД. В прикладі вказані дані підключення до локально розміщеної бази даних. Після проведених дій, проєктна БД буде працювати під керуванням PostgreSQL і доступ до неї, після проведених міграцій (спеціального механізму, який дозволяє автоматично зберігати та виконувати зміни в БД на основі змін у моделях файлу models.py), можливо отримати з адміністративної панелі. Інтерфейс адміністративної панелі зображений на рисунках 4.7 та 4.8.

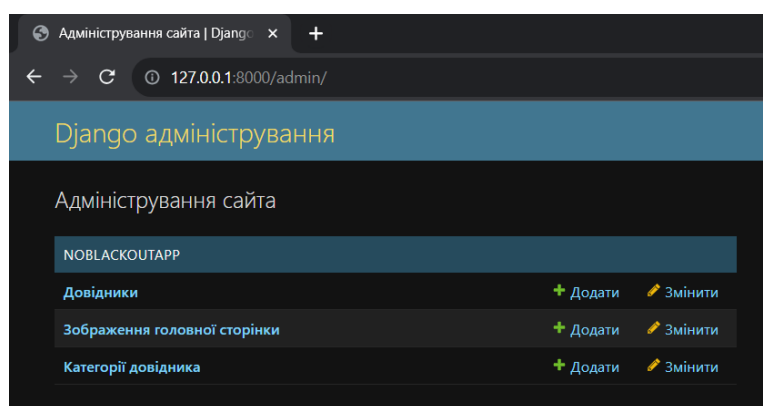


Рисунок 4.7 – Адміністративна панель проєкту зі списком наявних таблиць визначеної бази даних

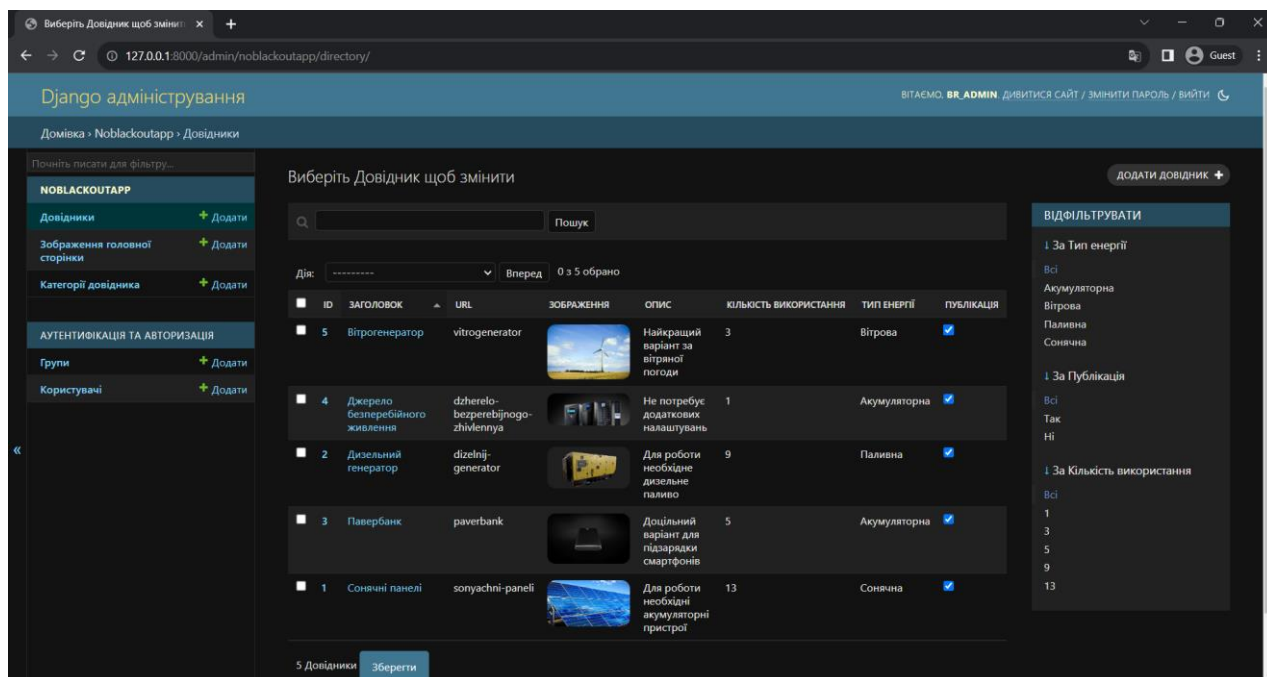


Рисунок 4.8 – Таблиця «Довідники» із наповненням вмістом відповідно до своєї тематики

Як вже було зазначено, обробка всієї логіки сайту у взаємодії користувача із серверною частиною відбувається у файлі `views.py`. Тут визначаються спеціальні класи або функції для, які в собі поєднують обробку запитів від користувачів, вибір даних з БД та видачу результату у відповідь на запит. Головним таким класом у розробленому проєкті є клас `Result` (рис. 4.9).

```
157 class Result(DataMixin, ListView):
158     model = Directory
159     template_name = 'noblackoutapp/result.html'
160     context_object_name = 'result'
161     infoToCalculate = None
162     finalResult = None
163
164
165 > def calculate(self, calculator_info):...
211
212 > def get_context_data(self, *, object_list = None, **kwargs):...
236
237
```

Рисунок 4.9 – Клас для представлення сторінки результатів розрахунку

До класу представлення, зображеного на рисунку 4.9, входять дві допоміжні функції, які і виконують базовий функціонал класу. Функція `get_context_data()` відповідає обробку першочергового контексту (спеціальний словник з необхідними даними, які слід передати до шаблону веб-сторінки) перш ніж відобразити результат на сторінці браузера (рис. 4.10).

```
215 def get_context_data(self, *, object_list = None, **kwargs):
216     context = super().get_context_data(**kwargs)
217     data = self.request.session.get('data')
218
219
220
221     if data['night_mode'] == 'true': self.infoToCalculate = Directory.objects.filter(is_published = True, energy_gte = float(data["energy"]), price_lte =
222     elif data['inside_apartment'] == 'true': self.infoToCalculate = Directory.objects.filter(is_published = True, energy_gte = float(data["energy"]), price_lte =
223     elif data['night_mode'] == 'true' and data['inside_apartment'] == 'true': self.infoToCalculate = Directory.objects.filter(is_published = True, energy_gte = float(data["energy"]), price_lte = float(data["money"]), time
224     else: self.infoToCalculate = Directory.objects.filter(is_published = True, energy_gte = float(data["energy"]), price_lte = float(data["money"]), time
225
226
227
228     if not self.infoToCalculate.exists():
229         c_def = self.get_user_context(title = 'Результат', result = "error")
230         return dict(list(context.items()) + list(c_def.items()))
231
232     else:
233         self.finalResult = self.calculate(self.infoToCalculate)
234
235
236
237     c_def = self.get_user_context(title = 'Результат', result = Directory.objects.get(title = self.finalResult[0]), per_cent = self.finalResult[1])
238     return dict(list(context.items()) + list(c_def.items()))
239
```

Рисунок 4.10 – Вміст функції класу представлення для першочергової обробки контексту для відображення

На рисунку 4.10 перші рядки коду даної функції відповідають за отримання базового контексту та значення з сесії в Django. Сесія в Django є механізмом, який дозволяє зберігати дані між HTTP-запитами та HTTP-відповідями на стороні сервера.

Умовними виразами перевіряються умови, задані користувачем та відбираються фільтровані записи з БД для порівняння між собою у другій функції класу представлення: `calculate()`, якщо ці дані існують (рис. 4.11).

```
165 def calculate(self, calculator_info):
166     price = {}
167     time = {}
168     one_time_expense = {}
169     generalResult = {}
170
171
172     for i in range(len(calculator_info)):
173         price[calculator_info[i].title] = calculator_info[i].energy / calculator_info[i].price
174         time[calculator_info[i].title] = calculator_info[i].energy / calculator_info[i].time_to_work
175         one_time_expense[calculator_info[i].title] = calculator_info[i].energy / calculator_info[i].one_time_expense
176
177
178
179
180     max_price = max(price.values())
181     max_time = max(time.values())
182     one_time_expense_max = max(one_time_expense.values())
183
184
185
186     for j in price:
187         price[j] = (100*price[j]) / max_price
188         time[j] = (100*time[j]) / max_time
189         one_time_expense[j] = (100*one_time_expense[j]) / one_time_expense_max
190
191
192
193     for k in calculator_info:
194         generalResult[k.title] = round((price[k.title] + time[k.title] + one_time_expense[k.title]) / 3, 2)
195
196
197
198     generalResultKey = max(generalResult, key=generalResult.get)
199     generalResultValue = max(generalResult.values())
200
201
202     directory_instance_to_update = Directory.objects.get(title = generalResultKey)
203     directory_instance_to_update.usage_quantity += 1
204     directory_instance_to_update.save()
205
206     directory_instance_to_update2 = CategoryOfEnergy.objects.get(name = directory_instance_to_update.type_of_energy)
207     directory_instance_to_update2.usage_quantity += 1
208     directory_instance_to_update2.save()
209
210     return generalResultKey, generalResultValue
211
```

Рисунок 4.11 – Вміст функції класу представлення для порівняння обраних даних для виявлення найкращого з них

На рисунку 4.11 зображена програмна реалізація порівняння різних приладів за кількістю виробленої енергії на однакову вартість, час активної безперебійної

роботи та одноразових витрат. Як і було описано в п.1.4, це є необхідною дією, щоб мінімізувати різного роду похибки та неточності у величезних розбіжностях деяких параметрів приладів порівняння. Це зможе показати цілісну картину в незалежності від класу виробу.

Отриманні результати повертаються у оглянути функцію до цього та передаються у якості контексту до зазначеного шаблону в класі представлення Result, щоб відобразити його на сторінці користувачу.

4.4 Користувацька частина програмного продукту

Як зазначалося раніше, користувацька частина програмного продукту представляється у вигляді сайту з власною базою даних. Сайт формалізований до інтуїтивно доступного рівня в плані навігації та розуміння тематичного контексту. Сайт має 3 основні ключові напрямки, які визначають його основну структуру сторінок та деякі допоміжні.

Однією з допоміжних сторінок є головна, де представляється коротка ознайомлювальна інформація на тематику сайту з миттєвою рекомендацією популярних рішень на поточний момент. Можливо на даному етапі основне завдання рекомендаційної системи для підготовки оселі до блекауту вдасться зацікавити користувача бажаним пристроєм (рис. 4.12).

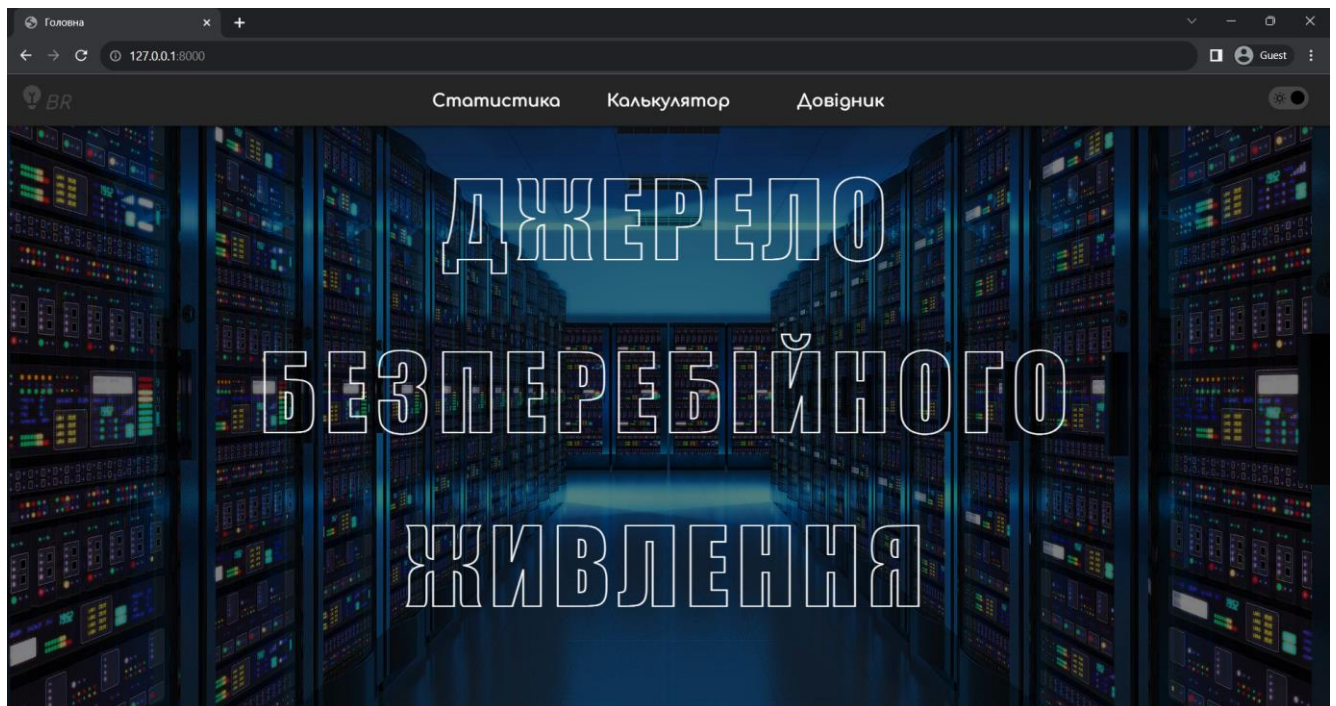


Рисунок 4.12 – Головна сторінка рекомендаційної системи

Перейдемо до огляду трьох основних напрямків діяльності сайту. Перший з них – це сторінка статистики, яка відображена на рисунку 4.13.

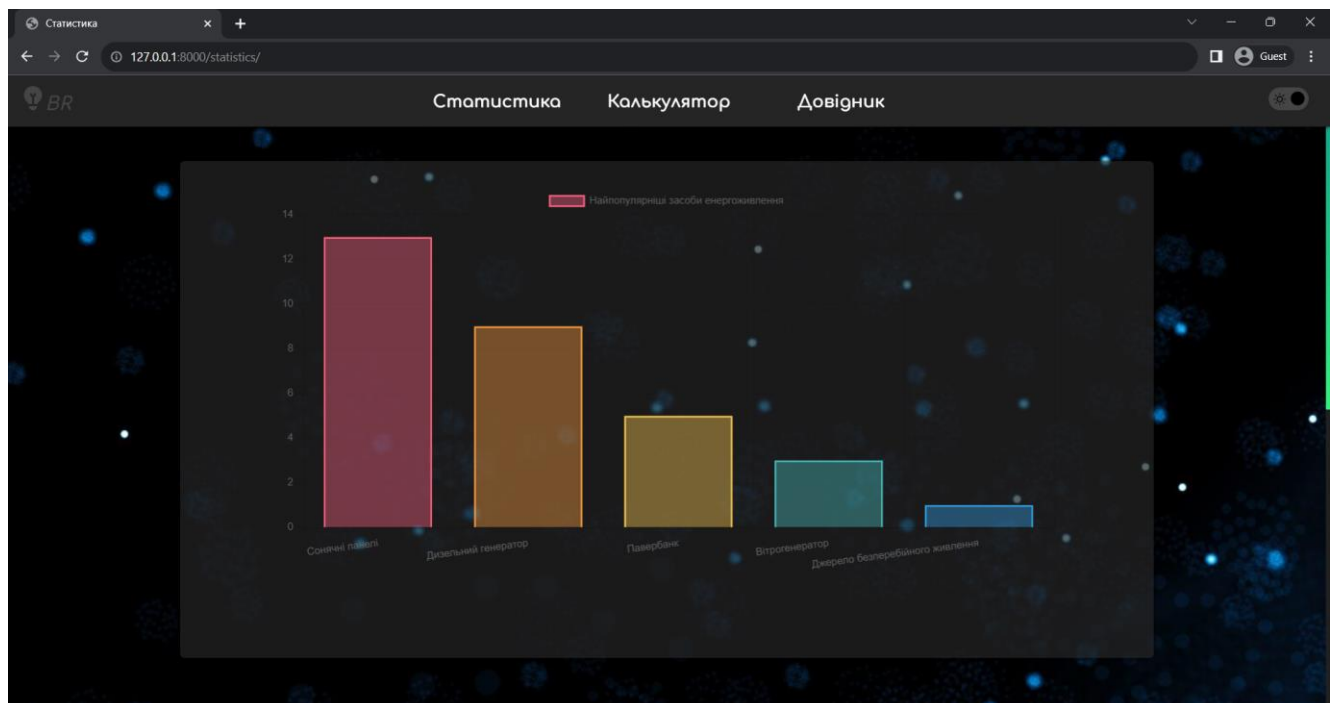


Рисунок 4.13 – Сторінка зі статистичними даними

Сторінка, відображена на рисунку 4.13, містить в собі наочно-інформативні дані щодо найкращого засобу резервного енергоживлення, заснованого на його популярності серед інших користувачів. Не менш важливим є демонстрація найпопулярнішого типу отримання електроенергії, що теж здатне вплинути на свідомість відвідувача сайту для прийняття власного рішення щодо вибору засобу електроживлення.

Наступний напрям є довідниковим і представляється відповідною сторінкою (рис. 4.14).

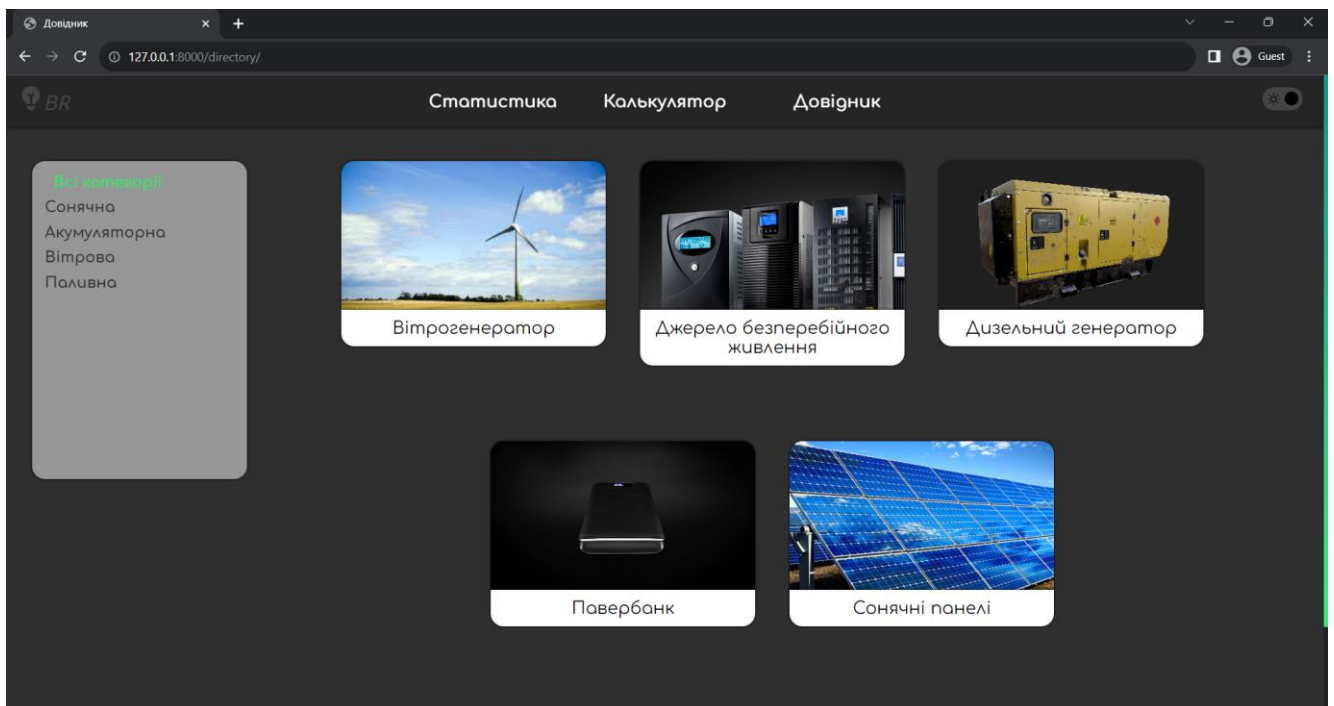


Рисунок 4.14 – Довідкова сторінка

Інформація з рисунку 4.14 надає змогу ознайомитися з наявними засобами електроживлення та переглянути їх за типом отримання енергії. Окрім цього, можна ще ознайомитися з детальною інформацією про засіб, який зміг зацікавити (рис. 4.15).

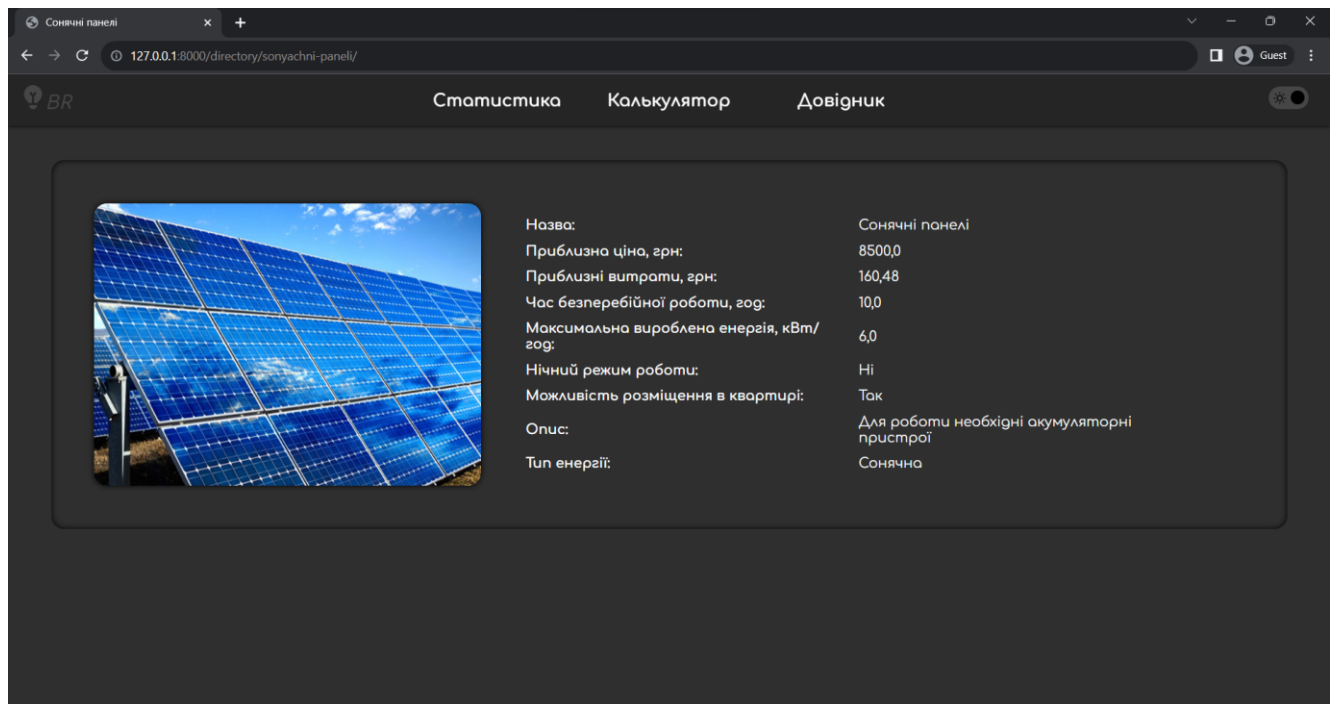


Рисунок 4.15 – Детальна довідкова інформація про конкретний засіб

Наступною сторінкою є сторінка розрахунку найбільш релевантного засобу за введеними параметрами користувача (рис. 4.16).

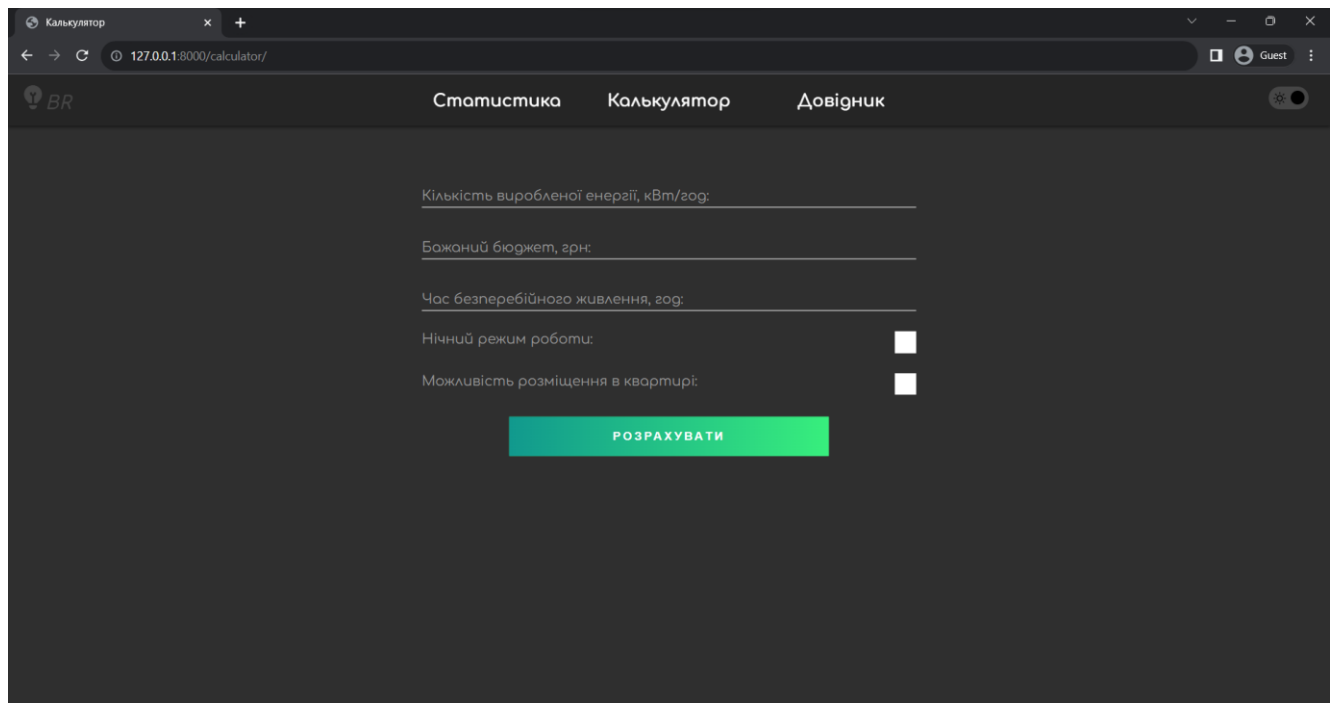


Рисунок 4.16 – Сторінка розрахунку найбільш доцільного засобу за власноруч зазначеними параметрами

На рисунку 4.16 зображено можливо один з найкорисніших напрямків функціонування рекомендаційної системи, оскільки поточний розрахунок дозволяє за короткий проміжок часу отримати найбільш доцільний варіант для власної оселі.

Користувацька частина розробленої системи передбачає роботу за декількома напрямками для досягнення результату у виборі найкращого варіанту резервного електроживлення для оселі користувача. Всі ці напрямки роботи зібрані в одному місці, що значно спробує процес вибору бажаного засобу та підвищує рівень доцільності такої системи.

4.5 Передумови використання

Своєрідні рекомендаційні системи стали невід’ємною частиною людського життя. Вони здатні зберігати в собі величезні об’єми інформації, обробляти її та видавати найбільш потрібні результати під конкретні запити користувачів. Такі системи застосовуються в багатьох галузях сучасного світу. Інколи вони можуть бути не такими вже й помітними через свою не новизну, але користі від цього менше не стає. Аналогічно до багатьох сфер людської діяльності, потребою в рекомендаційній системі є сфера безперебійних блоків, акумуляторів та генераторів. Оскільки складені умови вимагають прийняття відповідних рішень для забезпечення належної роботи та комфортному і помірному проживанню періоду блекауту.

Тому, передумовою використання даної рекомендаційної системи, перш за все звичайно, є необхідність у забезпеченні оселі приладом резервного енергоживлення для найкритичніших випадків, що можуть скластися. Для цього, необхідно провести, певного роду, аналіз всіх наявних рішень на ринку для обрання найоптимальнішого варіанту. Використовувати для такої задачі безліч ресурсів з різним профільним спрямуванням все ще можна, але ж і плата за таке використання

буде величезною у еквіваленті витраченого часу. Значно легшим варіантом було б відвідування спеціалізованого ресурсу з усією наявною інформацією на долоні з можливістю швидкого розрахунку найрелевантнішого рішення для власної оселі. Таке рішення обумовлюється людськими потребами до наявного комфорту та заощаджені власних ресурсів для профільних задач.

В цьому і полягають передумови використання розробленої системи: її простота та функціональність у поєднанні з нагальними потребами на поточний момент здатні змусити людей звертати увагу на неї та оцінювати її користь. Причини економити власний час, цінувати кожну секунду лежать на поверхні і надають найголовнішу відповідь у питанні доцільності використання такої системи та її необхідності. Окрім економії часу, до передумов можна віднести загальну потребу в обізнаності найактуальнішими параметричними даними та видами наявних засобів для забезпечення оселі безперебійним енергоживленням.

4.6 Функціональні можливості

Будь-яка система повинна мати у своєму складі різноманітні інструменти для надання максимальної функціональності власних наявних ресурсів своїм відвідувачам. Це є головним принципом для успішного існування розроблених систем. Тому, система рекомендаційного типу, яка спеціалізується на засобах енергоживлення для оселі повинна, відповідно, даний функціонал надавати і забезпечувати повною мірою.

Тож першочерговими функціональними можливостями розробленої системи – надання інформації щодо наявних та актуальних приладів для забезпечення оселі безперебійним енергоживленням в будь-який період.

Від загального опису можливостей доцільно перейти до детальних пояснень. Слід виділити три основні функціональні можливості, які виділяють дану

рекомендаційну систему на фоні інших існуючих систем: представлення статистики, розрахунок особистого рішення та інформативна довідка.

Детальний огляд функціональних можливостей рекомендаційної системи:

- представлення статистики – відображення спеціальної сторінки, де зазначається статистична інформація у вигляді графіків. Даного роду представлення спрощує наочне сприйняття відображуваної інформації, що слугує гарним фундаментом для побудови подальшого рішення щодо необхідності того чи іншого засобу. В статистиці відображається інформація про кількість згенерованих засобів у якості результату у відповідь на запит користувача у другій функціональній можливості. Також, зазначається інформація про найбільш популярний тип отримання енергії, що, в свою чергу, теж повинно позитивно впливати на ознайомлювальний процес та на легкість прийняття рішення в подальшому;
- розрахунок особистого рішення – сторінка для розрахунку найоптимальнішого рішення енергоживлення за введеними параметрами користувача. Простіше це можна назвати калькулятором, де на основі введених параметрів: кількість виробленої енергії, бажаний бюджет, час безперебійного живлення, нічний режим роботи та можливість розміщення в квартирі проводиться розрахунок. Дані беруться з бази даних, попередньо фільтруючись за критеріями користувача та порівнюються між собою для виявлення найоптимальнішого варіанту енергоживлення під зазначені потреби користувача;
- інформативна довідка – сторінка-довідник, де відображається інформація про всі наявні засоби резервного електроживлення на сайті. Представляється можливість фільтрування за типом отримання енергії та можливістю детально ознайомитися з інформацією про конкретний бажаний засіб. Інформація щодо конкретного засобу відображається у вигляді таблиці з умовними характеристиками вартості, кількості

виробленої енергії, приблизно середнього часу безперебійної роботи та іншою інформацією. Така функціональна можливість добре сприяє процесу ознайомлення відвідувачів з наявними засобами та їхніми характеристиками для наочного порівняння.

Отже, основними функціональними можливостями розробленої системи є статистика, розрахункова сторінка та довідник, що в поєднанні надають різнопланову, але конкретну та спеціалізовану інформацію і надійну допомогу у виборі оптимального рішення для власної оселі.

Висновки до розділу 4

Серверна частина розробленого застосунку працює під керуванням високорівневих технологій. Завдяки їм, має широку масштабованість, що дозволить легко розширювати поточний проєкт за необхідністю в майбутньому.

Користувацька частина вирізняється лаконічним інтерфейсом з інтуїтивно зрозумілою логікою навігацій та наочності отриманих даних. Поєднання функціональної простоти та наочної дійсності дозволить користувачам системи досягти значно швидше поставлених цілей щодо ознайомлення з наявними засобами та пошуку найкращого для них.

Для використання даної системи є необхідна потреба в сьогоденні, оскільки, наявність резервного засобу живлення дозволить перебувати у зв'язку з цивілізованим світом. Це є важливою задачею в непрості теперішні часи постійно загрози та небезпеки. Наявний функціонал в системі дозволить користувачам досліджувати питання актуальних засобів резервного електроживлення через статистичні та довідникові дані, а також за рахунок проведених розрахунків за власними критеріями. Зручність і простота є основними функціональними можливостями для користувачів, які якнайкраще сприятимуть вирішенню нагальної потреби щодо інформування та пошуку.

5 РОЗРОБКА СТАРТАП-ПРОЄКТУ

5.1 Опис ідеї проєкту

Основна ідея, яка покладена в основу даної системи – це забезпечення людей централізованим місцем перегляду, дослідження та вибору засобів резервного електроживлення. Актуальність такого рішення в сьогоденні набуває широкого значення та всебічного охоплення. Тому, спочатку доцільно провести аналіз ідеї стартап-проєкту (табл. 5.1).

Таблиця 5.1 – Опис ідеї стартап-проєкту

Зміст ідеї	Напрямок застосування	Вигоди для користувачів
1	2	3
Створення рекомендаційної системи для підготовки оселі до блекауту	Моніторинг актуальної інформації з бази знань	Підвищення рівня обізнаності населення у сфері засобів резервного електроживлення
	Перегляд кількісної характеристики забезпечення іншими користувачами	Зручність перегляду актуальності тих чи інших засобів на основі вибору інших користувачів
	Розрахунок оптимального рішення для власних потреб	Економія часу для вибору засобу резервного електроживлення для власної оселі

Підсумовуючи зміст таблиці 5.1 можна виділити три основні напрямки функціонування ро розроблюваної системи – база знань, кількісно-графічна

статистика та механізм розрахунку. Таке поєднання дозволить основній цільовій аудиторії, якими є люди з ціллю забезпечити власну оселю безперебійним електроживленням, отримати краще розуміння щодо наявних засобів, їх актуальності в суспільстві та можливість швидкого підбору.

Наступним етапом буде аналіз потенційних техніко-економічних переваг розроблюваної рекомендаційної системи в порівнянні з існуючими аналогами, що описано у таблиці 5.2.

Таблиця 5.2 – Визначення головних особливостей стартап-проєкту

Техніко-економічні характеристики ідеї проєкту	Продукція конкурентів				Слабка сторона	Нейтральна сторона	Сильна сторона
	Розроблювана система	Системи рекомендаційного типу	Системи оглядового типу	Системи магазинного типу			
1	2	3	4	5	6	7	8
Опис характеристик	Так	Ні	Так	Так		+	
Наявність статистичної інформації	Так	Так	Ні	Ні			+
Можливість придбання	Ні	Ні	Ні	Так	+		
Наявні рекомендації щодо вибору	Так	Так	Ні	Ні			+
Різносторонність системи	Так	Ні	Ні	Ні			+

З таблиці 5.2 можна зробити висновок, що наявні потенційні конкуренти не зможуть запропонувати весь повноцінний функціонал розроблюваної системи в повному обсязі, оскільки кожен з конкурентів спеціалізується на окремих його складових, а не цілеспрямовано масштабуючись на всіх.

5.2 Технологійний аудит проєкту

Наступним кроком доцільно провести технологічний аудит проєкту, щоб конкретно визначити технології, які є вкрай необхідними для розробки. Для цього потрібно спершу визначити самі технології для майбутньої реалізації, їх широку наявність та доступність (табл. 5.3).

Таблиця 5.3 – Технологічний аудит проєкту

№ п/п	Ідея проєкту	Технології її реалізації	Наявність технології	Доступність технології
1	2	3	4	5
1.	Серверна логіка	Python, Django	+	+
2.	Користувацьке відображення	HTML, CSS JavaScript	+	+
3.	База даних	PostgreSQL	+	+
4.	Візуалізація графіків	Chart.js, JavaScript	+	+

За результатами проведеного аналізу доцільно сказати, що всі необхідні технології для розробки бажаної системи наявні та доступні в повному обсязі. Тобто не вимагають будь-якого доопрацювання з боку розробників запропонованої системи та додаткових фінансових вливань.

5.3 Аналіз ринкових можливостей стартап-проєкту

Наступним кроком стане визначення ринкових можливостей даного стартап-проєкту, які можуть бути використаними для впровадження та загроз, які можуть

цьому завадити. Таким чином можна буде спланувати подальший розвиток проекту, враховуючи всі особливості конкурентів та користувачів системи.

Спочатку буде проведений аналіз попиту у таблиці 5.4.

Таблиця 5.4 – Попередня характеристика потенційного ринку стартап-проекту

№ п/п	Показники стану ринку	Характеристика
1	2	3
1.	Кількість головних гравців, од	4
2.	Загальний обсяг продаж, грн/міс	249
3.	Динаміка ринку (якісна оцінка)	Очікується зростання
4.	Наявність обмежень для входу (якщо є — вказати характер обмежень)	Немає
5.	Специфічні вимоги до стандартизації та сертифікації	Немає
6.	Середня норма рентабельності в галузі (або по ринку), %	25

Підбиваючи підсумки з вищенаведеної таблиці, доцільно сказати, що розроблювана система має високі перспективи та зможе в майбутньому мати широкий попит у користувачів.

Далі буде визначено потенційні групи користувачів розроблюваної системи та сформований приблизний перелік вимог до програмного продукту (табл. 5.5).

Таблиця 5.5 – Характеристика потенційних користувачів стартап-проєкту

№ п/п	Потреба, що формує ринок	Цільові аудиторія	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до програмного продукту
1	2	3	4	5
1.	Автоматизований процес вибору засобів резервного електроживлення	Фізичні особи	Немає	Швидкість отримання результату, можливість введення власних значень для більш релевантної відповіді
2.	Довідникова інформація щодо засобу	Фізичні особи	Немає	Поглиблена інформація про засіб для ознайомлення на поглиблення власних знань
3.	Графічна статистика обраних засобів	Фізичні особи	Немає	Зменшення рівня невизначеності при виборі засобів резервного електроживлення шляхом графічного представлення статистики

Після визначення потенційні користувачі стартап-проекту, необхідно провести аналіз ринкового середовища: зазначити фактори, які перешкоджатимуть ринковому впровадженню проекту (табл. 5.6) та, навпаки, допомагатимуть йому в цьому (табл. 5.7).

Таблиця 5.6 – Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1	2	3	4
1.	Збільшення конкуренції на ринку	Поява нових конкурентів на ринку може призвести до значної втрати потенційних користувачів розроблюваної системи	Наймання нових кваліфікованих працівників для вдосконалення наявного функціоналу та доповнення новим, щоб залучити увагу користувачів
2.	Економічна складова	Через економічну складову можлива відсутність попиту на використання сервісу	Дослідження поточної ситуації на ринку та коригування власних послуг для доступності більшій верстві населення
3.	Нові тенденції на ринку	Поява на ринку продукції з кардинально іншим підходом, яка зможе вирішувати ці ж завдання розроблюваної системи краще та пропонуватиме набагато більше на додачу	Адаптація до нової реальності та доопрацювання власного продукту, щоб відповідати сучасним новітнім тенденціям ринку

Таблиця 5.7 – Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1	2	3	4
1.	Якість розробленого продукту	Масштабування власного продукту на міжнародному ринку	Збільшення терміну та якості підтримки програмного продукту за кордоном
2.	Подальший технологічний розвиток	Колосальний розвиток технологій, що використовуються для розробки та потенційна поява нових	Впровадження сучасних технологій розробки програмного забезпечення та оптимізація власного рішення для досконалості у майбутньому
3.	Актуальність розробленої системи	Широка потреба в системі в періоди найбільшої актуальності для населення	Підтримка розробленого застосунку, доповнення новими функціями та доопрацювання вже розроблених

Після зазначення факторів загроз та можливостей необхідно визначити тип конкуренції (табл. 5.8).

Таблиця 5.8 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1	2	3
1. Тип конкуренції – чиста	На ринку присутня достатня кількість компаній зі своїми різноманітними підходами, що створює конкуренцію	Підтримування належної якості розробленої продукції та впровадження нових інновацій
2. За рівнем конкурентної боротьби – національний	Продукт розробляється, в першу чергу, для українців під їхні особисті споживчі потреби	Розвиток та просування власної продукції більшій масі людей
3. За галузевою ознакою – міжгалузева	Використання розробленого застосунку не має чіткої прив'язки до конкретної галузі та використовується виключно за потреби громадян	Поліпшення власного застосунку для кращого досвіду використання користувачами
4. Конкуренція за видами товарів: товарно-видова	Конкуренція здійснюється між програмним забезпеченням одного виду	Вдосконалення розробленого застосунку, шляхом випуску нових якісних версій оновлення

Продовження таблиці 5.8

1	2	3
5. За характером конкурентних переваг – нецінова	Ціна не являється основним показником для конкуренції, яка, в першу чергу, заснована на функціональних можливостях системи	Оптимізація цінової політика для більшої доступності людям та вдосконалення надійності та швидкості роботи розробленої системи
6. За інтенсивністю – не марочна	На перший план ставиться не компанія, а якість розробленої продукції	Просування продукту всіма можливими способами для більшого залучення користувачів та підтримка актуальності та працездатності розробленого продукту

Після проведеного ступеневого аналізу конкуренції на ринку, проаналізуємо сильні та слабкі сторони стартап-проекту у порівнянні з конкурентами (табл. 5.9).

Таблиця 5.9 – Порівняльний аналіз сильних та слабких сторін стартап-проєкту

№	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні з розробленим продуктом						
			-3	-2	-1	0	+1	+2	+3
1	2	3	4	5	6	7	8	9	10
1.	Унікальність функціоналу	20			+				
2.	Вигідна цінова політика для користувачів	19				+			
3.	Підтримка програмного забезпечення	12					+		
4.	Швидкість вибору засобів резервного електроживлення	20	+						
5.	Простота у використанні розробленого додатку	18		+					

Для завершення аналізу ринкових можливостей стартап-проєкту, його запуску та впровадження на ринку використовується SWOT-аналіз – матриці аналізу сильних (Strength), слабких (Weak), можливостей (Opportunities) та загроз (Threat).

Матриця SWOT-аналізу представлена у таблиці 5.10.

Таблиця 5.10 – SWOT-аналіз стартап-проєкту

Сильні сторони (Strength): • ціна; • поєднання трьох головних складових системи в одному місці; • швидкість при виборі засобу резервного електроживлення; • використання сучасних методів розробки, що дозволить з меншими зусиллями підтримувати розроблений застосунок; • унікальність системи, якої немає у конкурентів.	Слабкі сторони (Weak): • вимагається значна кількість розробників та адміністраторів системи для належного функціонування; • наповнення бази знань для довідникового та рекомендаційного аспектів потребує багато часу.
Можливості (Opportunities): • вихід на міжнародний ринок; • інтеграція з сервісами для продажу засобів резервного електроживлення.	Загрози (Threats): • поява конкурентів з набагато кращою продукцією; • зміни в законодавстві, які передбачатимуть обладнання осель, за замовчуванням, засобами резервного електроживлення; • можлива не постійна затребуваність у наявній продукції звичайними користувачами.

За результатами складеного SWOT-аналізу ще необхідно розробити альтернативу ринкової поведінки для виведення стартап-проекту на ринок, визначення орієнтованого оптимального часу для реалізації ідеї, оглядаючись на застосунки конкурентів (табл. 5.11).

Таблиця 5.11 – Альтернативи ринкового впровадження стартап-проекту

№	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	2	3	4
1.	Розробка базової версії рекомендаційної системи у вигляді вебзастосунку	85%	5 місяців
2.	Проведення демонстраційного заходу для міжнародного ринку	54%	11 місяців
3.	Безкоштовний режим використання рекомендаційної системи	40%	2 місяці
4.	Створення мобільного застосунку	62%	6 місяців
5.	Впровадження купівельної можливості в межах розробленої системи	70%	15 місяців

Після проведеного аналізу альтернативних варіантів, доцільно прийняти рішення щодо першочергового виходу на основний ринок з розробленим вебзастосунком, оскільки такий варіант здатен забезпечити більшу ймовірність отримання ресурсів за менший термін реалізації ідеї розробки.

5.4 Розроблення ринкової стратегії програмного продукту

Для розроблення ринкової стратегії спочатку необхідно визначити стратегію охоплення ринку, яке передбачає собою опис цільових груп потенційних споживачів рекомендаційної системи (табл. 5.12).

Таблиця 5.12 – Вибір цільових груп потенційних споживачів

№	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	2	3	4	5	6
1.	Підприємства, ІТ-компанії	Середня	Низький	Середня	Помірно
2.	Фізичні особи	Висока	Високий	Невелика	Легко

Виходячи з вмісту вищенаведеної таблиці у якості основної цільової групи було обрано фізичних осіб, яким за більшою затребуваністю може стати в нагоді розроблюваний застосунок. Оскільки була обрана лише одна цільова група потенційних споживачів як основна, тому доцільно обрати підхід концентрованого маркетингу, спрямувавши всі сили при розробці для задоволення потреб осіб з даної цільової групи потенційних споживачів.

Далі доцільно сформулювати базову стратегію розвитку (табл. 5.13).

Таблиця 5.13 – Визначення базової стратегії розвитку

№	Обрана альтернатива розвитку проєкту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
1	2	3	4	5
1.	Розробка базової версії рекомендаційної системи у вигляді вебзастосунку	Концентрований маркетинг	Три різнопланові ключові складові розроблюваної системи, які значно спрощують та пришвидшують процес вибору засобів резервного електроживлення	Надійна та тривала підтримка розробленого продукту

З інформації наведеної у таблиці вище зрозуміло, що поширення розроблюваного продукту здійснюватиметься, шляхом повноцінної концентрації на одній цільовій групі, оскільки вона і є потенційно потребуючою.

Далі буде визначено стратегії конкурентної поведінки (табл. 5.14)

Таблиця 5.14 – Визначення базової стратегії конкурентної поведінки

Чи є проєкт «першопрохідцем» на ринку?	Так
1	2
Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	В основному будуть нові споживачі, адже, як такого, аналогу не існує на поточний момент. Проте, користувачі, які використовують застосунки

Продовження таблиці 5.14

1	2
	найбільш наближених конкурентів також будуть заохочуватися до переходу
Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Так, але з власним поглядом на ситуацію і вкрапленням власних особливостей у базовий товар наслідування. Адже конкуренція сприяє прогресу
Стратегія конкурентної поведінки	Подальший розвиток задля конкурентоспроможності на ринку

Окрім цього всього, ще розробимо стратегію позиціонування, за якою користувачі на ринку повинні будуть ідентифікувати розроблену продукцію (табл. 5.15).

Таблиця 5.15 – Визначення стратегії позиціонування

Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проєкту	Вибір асоціацій, які мають сформулювати комплексну позицію власного проєкту
1	2	3	4
Поєднання зручності та легкості в одному місці для	Концентрований маркетинг	Швидкість у виборі засобу резервного електроживлення, зручність у	Швидкість, простота, доступність, функціональність

Продовження таблиці 5.15

1	2	3	4
швидкого вибору засобів резервного електроживлення		дослідженні інформації щодо вибору	

5.5 Розроблення маркетингової програми стартап-проекту

Для розроблення маркетингової програми стартап-проекту спочатку необхідно сформулювати маркетингові концепції товару. Дана інформація зазначена у таблиці 5.16.

Таблиця 5.16 – Визначення ключових переваг концепції потенційного товару

Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами
1	2	3
Швидкість у виборі засобу	Розрахункова сторінка, де за зазначеними користувачем параметрами підбирається засіб резервного електроживлення	Наявність безлічі засобів резервного електроживлення в базі, яка постійно поповнюватиметься
Поширення знань щодо наявних засобів	Можливість дослідження різноманітних засобів за їх ключовими характеристиками	Зручне відображення наявної інформації про засіб
Проведення аналізу	Графічна статистика найбільш популярних засобів серед інших користувачів для	Наочні графіки з інформацією як про самі засоби так і про тип

Продовження таблиці 5.16

1	2	3
	розуміння поточної ситуації на ринку та формування власного рішення	використовуваної ними енергії

Далі буде розроблена трирівнева маркетингова модель товару з урахуванням його головної ідеї, загальних послуг та особливостей процесу функціонування (табл. 5.17).

Таблиця 5.17 – Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові
1	2
Товар за задумом	Рекомендаційна система для підготовки оселі до блекауту
Товар у реальному виконанні	Реалізовано вебзастосунок з можливістю швидкого розрахунку засобу резервного електроживлення за власними введеними параметрами, дослідження ключових характеристик засобів та моніторингу графічної статистики щодо найпопулярніших засобів на поточний момент часу
Товар із підкріпленням	Підтримка розробленого продукту після запуску на ринку оновленнями та виправленнями для належного та якісного функціонування в майбутньому. Також, поповнення бази знань новими засобами для розширення можливостей рекомендаційної системи в цілому

Далі необхідно визначити ціну на продукцію з урахуванням цінової політики найближчих конкурентів та доходів потенційної цільової групи (табл. 5.18).

Таблиця 5.18 – Визначення меж встановлення ціни

Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар
1	2	3	4
500 грн	500 грн	від 15 000 грн	0 – 249 грн

Далі, в таблиці 5.19, будуть визначенні системи збуту.

Таблиця 5.19 – Формування системи збуту

Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
1	2	3	4
Місячна підписка на сервіс	Продаж та підтримка	Взаємодія безпосередньо між покупцем та продавцем	Пряма (взаємодія між продавцем та покупцем)

В підсумку, в таблиці 5.20, наведено концепції маркетингових комунікацій, де визначається поведінка цільових клієнтів та канали комунікації з ними.

Таблиця 5.20 – Концепція маркетингових комунікацій

Специфіка поведінки цільових клієнтів	Канали комунікацій цільових клієнтів	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
1	2	3	4	5
Клієнти дізнаються про рекомендаційну систему через інтернет мережу	Інтернет	Зручність використання, швидкість визначення бажаного засобу, широкий	Донести до кінцевого користувача зручність використання сервісу та його доцільність при вирішенні завдання пошуку засобів резервного електроживлення	Поєднання ключових особливостей щодо вибору засобів резервного електроживлення в одному місці

Висновки до розділу 5

В даному розділі було оглянуто основні пункти щодо розробки стартап-проекту розроблюваної рекомендаційної системи. Дані дії робились з урахуванням поточної ситуації на ринку, можливостей конкурентів та недоліків поточного стану справ в напрямку вибору засобів резервного електроживлення.

Було обрано цільову групу потенційних користувачів, яким більше за все дана розробка може стати в нагоді. Зазначено можливі альтернативи розвитку проекту та ймовірність залучення ресурсів для кожного запропонованого варіанту.

Враховуючи все вищезазначене, доцільність виходу на ринок із запропонованою стартап ідеєю є високою через відсутність цілеспрямованих конкурентів, які спеціалізуються в даній тематиці та поточну широку актуальність можливості рекомендаційного вибору для звичайних людей у сфері засобів резервного електроживлення.

ВИСНОВКИ

Провівши аналіз в сфері допоміжних систем для вибору джерел резервного енергоживлення доцільно сказати, що функціонального рішення не було. Наявні існуючі системи пропонують лише мінімум можливостей з кожного напрямку, який би хотілось бачити у повноцінній спеціалізованій системі. Розподіл уваги на безліч ресурсів для моніторингу наявних засобів, порівняння їхніх характеристик та розрахунку числових значень не є зручним в роботі та потребує достатньої кількості часу для досягнення власних поставлених цілей.

Зібрання в одному місці всіх напрямків щодо підбору оптимального пристрою для забезпечення власної оселі резервним живленням є логічним рішенням. Зручність такої системи прослідковується відразу, а наявний функціонал здатен значно зменшити кількість витраченого часу на пошуки бажаного рішення у підборі оптимально приладу.

Розробка запропонованого рішення полягала у наданні користувачам ресурсу можливості в одному місці відшукати всю детальні інформацію про поточний стан справ на ринку приладів резервного енергоживлення. Поєднання статистичних даних, довідника з детальною інформацією про прилади та розрахункового калькулятора здатні покрити повну потребу користувачів щодо відслідковування актуальних приладів та пошуку оптимальних для них.

Система має перспективу широкого масштабування, тому для її побудови використовувалися сучасні засоби розробки. Використаний потужний фреймворк Django надає весь наявний функціонал для легкої розробки оптимального рішення для поточної проблематики із забезпеченням надійного взаємозв'язку з своєрідною базою знань щодо приладів резервного енергоживлення для оселі користувачів.

В перспективі розвитку даної системи в майбутньому, доцільно було б впровадити можливість покупки обраних приладів на місці або ж переадресації на відповідні ресурси для ще більшої економії часу відвідувачів та підвищення зручності використання ресурсу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Охріменко, В. М. Споживачі електричної енергії: підручник. Харків: ХНУМГ ім. О. М. Бекетова, 2019. 286 с.
2. Електрика у житті людини: НЕК. URL: <https://n-e-c.com.ua/uk/node/179> (дата звернення: 16.12.2023).
3. Про ринок електричної енергії : Закон України від 01.01.2024 р. № 2019-VIII. *Відомості Верховної ради України*. 2017. № 27-28. Ст. 312.
4. Правила улаштування електроустановок: 5 вид., переробл. й доповн. Харків: Видавництво «Форт», 2014. 800 с.
5. Костюченко, А.О. Основи програмування мовою Python: навчальний посібник. Ч.: ФОП Баликіна С.М., 2020. 180 с.
6. Підручник з Python: документація. URL: <https://docs.python.org/uk/3/tutorial/index.html> (дата звернення: 21.10.2023).
7. Python – високорівнева мова програмування: AVADA-MEDIA. URL: <https://avada-media.ua/ua/services/python-plyusy-i-minusy-yazyka-kakiye-zadachi-reshayet-i-stoit-li-izuchat/> (дата звернення: 21.10.2023).
8. Рамський, Ю.С., Цибко, Г.Ю. Основи програмування (мова Паскаль): навчальний посібник. Київ, 2003. 140 с.
9. William S. Django for Beginners: Build websites with Python and Django. London: WelcomeToCode, 2022. 335 p.
10. Why Django?: офіційний електронний ресурс. URL: <https://www.djangoproject.com/start/overview/> (дата звернення: 22.10.2023).
11. Django Introduction: W3Schools. URL: https://www.w3schools.com/django/django_intro.php (дата звернення: 17.12.2023).
12. Плюси і мінуси Django: Укрнеймс.БЛОГ. URL: <https://blog.ukrnames.com/veb-master/plyusi-i-minusi-django> (дата звернення: 22.10.2023).

13. Розробка веб додатків з використанням Python і Django: WEBCASE. URL: <https://webcase.com.ua/uk/blog/razrabotka-veb-prilozhenij-s-ispolzovaniem-python-i-django/> (дата звернення: 22.10.2023).
14. What is PostgreSQL?: офіційний електронний ресурс. URL: <https://www.postgresql.org/about/> (дата звернення: 23.10.2023).

ДОДАТКИ

ДОДАТОК А Лістинг розробленої програми (фрагменти)

Файл `models.py`

```
from django.db import models
from django.urls import reverse

class HomePage(models.Model):
    title = models.CharField(max_length = 255, blank = False,
        verbose_name="Заголовок")

    slug = models.SlugField(max_length = 255, unique = True, db_index = True,
        verbose_name = "URL")

    image = models.ImageField(upload_to="photos/home/%Y/%m/%d/", blank = False,
        default = None, verbose_name="Завантажити фонове зображення")

    def __str__(self):
        return self.title

    def get_absolute_url(self):
        return reverse('homepage', kwargs = {'homepage_slug': self.slug})

class Meta:
    verbose_name = "Зображення головної сторінки"
    verbose_name_plural = "Зображення головної сторінки"
    ordering = ['title']

class Directory(models.Model):
    title = models.CharField(max_length = 255, verbose_name="Заголовок")

    slug = models.SlugField(max_length = 255, unique = True, db_index = True,
        verbose_name = "URL")
```

```

        content = models.TextField(blank = False, verbose_name="Опис")

        price = models.FloatField(max_length=10, blank=False, verbose_name="Приблизна
ціна, грн")

        one_time_expense = models.FloatField(max_length=10, blank=False,
verbose_name="Приблизні витрати на час безперебійної роботи, грн")

        time_to_work = models.FloatField(max_length=10, blank=False,
verbose_name="Приблизний час безперебійної роботи, год")

        energy = models.FloatField(max_length=10, blank=False, verbose_name="Приблизна
кількість виробленої енергії, кВт/год")

        night_mode_work = models.BooleanField(default = False, verbose_name="Нічний
режим роботи")

        inside_apartment = models.BooleanField(default = False,
verbose_name="Можливість розміщення в квартирі")

        image = models.ImageField(upload_to="photos/%Y/%m/%d/", blank = False,
verbose_name="Зображення")

        usage_quantity = models.IntegerField(blank=False, default=0,
verbose_name="Кількість використання")

        is_published = models.BooleanField(default = True, verbose_name="Публікація")

        per_cent = models.DecimalField(max_digits=5, decimal_places=2, null=True,
blank=True)

        type_of_energy = models.ForeignKey('CategoryOfEnergy',
on_delete=models.PROTECT, verbose_name="Тип енергії")


    def __str__(self):
        return self.title


    def get_absolute_url(self):
        return reverse('directory_info', kwargs = {'directory_slug': self.slug})


class Meta:
    verbose_name = "Довідник"
    verbose_name_plural = "Довідники"
    ordering = ['title']


class CategoryOfEnergy(models.Model):

```

```

    name = models.CharField(max_length = 100, db_index = True, verbose_name =
'Категорія')

    slug = models.SlugField(max_length = 255, unique = True, db_index = True,
verbose_name = "URL")

    usage_quantity = models.IntegerField(blank=False, default=0,
verbose_name="Кількість використання")

    def __str__(self):
        return self.name

    def get_absolute_url(self):
        return reverse('category', kwargs = {'category_slug': self.slug})

    class Meta:
        verbose_name = "Категорія довідника"
        verbose_name_plural = "Категорії довідника"
        ordering = ['name']

```

Файл **views.py**

```

from django.shortcuts import render
from django.views.generic import ListView, DetailView

from .models import *
from .utils import *

from django.http import JsonResponse
import json
import logging

def user_data(request):
    if request.method == 'POST' and request.META.get('HTTP_X_REQUESTED_WITH') ==
'XMLHttpRequest':
        try:

```

```

        data = request.POST
        request.session['data'] = data

        if data["energy"] == '' or data["money"] == '' or data["time"] == '':
            return JsonResponse({'message': 'calculator', 'redirect':
reverse('calculator')})
        else:
            return JsonResponse({'message': 'result', 'redirect':
reverse('result')})

    except json.JSONDecodeError as e:
        print(e)
        JsonResponse({"success": False, "message": "Error decoding JSON
data"})
    except Exception as e:
        print(e)
        JsonResponse({"success": False, "message": "Error processing data"})

    else:
        return JsonResponse({'error': 'Invalid request method'}, status=400)

class Home(DataMixin, ListView):
    model = HomePage
    template_name = 'noblackoutapp/index.html'
    context_object_name = 'home'

    def get_context_data(self, *, object_list = None, **kwargs):
        context = super().get_context_data(**kwargs)
        slide_info = HomePage.objects.all().order_by('pk')
        c_def = self.get_user_context(title = 'Головна', slides = slide_info)
        return dict(list(context.items()) + list(c_def.items()))

```

```

def diagramsData(request):
    try:
        dataFromDB = CategoryOfEnergy.objects.filter()
        dataFromDB2 = Directory.objects.filter(is_published = True).order_by('-usage_quantity')

        dataToSend1 = {}
        dataToSend2 = {}

        for i in range(len(dataFromDB)):
            dataToSend1[dataFromDB[i].name] = dataFromDB[i].usage_quantity

        for j in range(len(dataFromDB2)):
            dataToSend2[dataFromDB2[j].title] = dataFromDB2[j].usage_quantity

        dataToSend = {}
        dataToSend[0] = dataToSend1
        dataToSend[1] = dataToSend2

        return JsonResponse(dataToSend)

    except Exception as e:
        logging.error(str(e))
        return JsonResponse({'error': 'Internal Server Error'}, status=500)

def Statistics(request):
    context = {'title': 'Статистика', 'navigation': navigation}
    return render(request, 'noblackoutapp/statistics.html', context = context)

class DirectoryView(DataMixin, ListView):
    model = Directory
    template_name = 'noblackoutapp/directory.html'
    context_object_name = 'directories'

```

```

def get_context_data(self, *, object_list = None, **kwargs):
    context = super().get_context_data(**kwargs)
    c_def = self.get_user_context(title = 'Довідник')
    return dict(list(context.items()) + list(c_def.items()))

def get_queryset(self):
    return Directory.objects.filter(is_published =
True).select_related('type_of_energy')

class DirectoryByCategory(DataMixin, ListView):
    model = Directory
    template_name = 'noblackoutapp/directory.html'
    context_object_name = 'directories'
    allow_empty = False
    paginate_by = 20

    def get_context_data(self, *, object_list = None, **kwargs):
        context = super().get_context_data(**kwargs)
        category = CategoryOfEnergy.objects.get(slug =
self.kwargs['category_slug'])
        c_def = self.get_user_context(title = 'Катеропія - ' + str(category.name),
category_selected = category.pk)
        return dict(list(context.items()) + list(c_def.items()))

    def get_queryset(self):
        return Directory.objects.filter(type_of_energy__slug =
self.kwargs['category_slug'], is_published =
True).select_related('type_of_energy')

class DirectoryInfo(DataMixin, DetailView):
    model = Directory
    template_name = 'noblackoutapp/directory_info.html'
    slug_url_kwarg = 'directory_slug'
    context_object_name = 'directory_info'

```

```

def get_context_data(self, *, object_list = None, **kwargs):
    context = super().get_context_data(**kwargs)
    directory_info = Directory.objects.get(title = context['directory_info'])
    c_def = self.get_user_context(title = context['directory_info'],
category_selected = directory_info.type_of_energy_id)
    return dict(list(context.items()) + list(c_def.items()))

```

```

class Calculator(DataMixin, ListView):
    model = Directory
    template_name = 'noblackoutapp/calculator.html'
    context_object_name = 'calculator'

```

```

def get_context_data(self, *, object_list = None, **kwargs):
    context = super().get_context_data(**kwargs)
    c_def = self.get_user_context(title = 'Калькулятор')
    return dict(list(context.items()) + list(c_def.items()))

```

```

class Result(DataMixin, ListView):
    model = Directory
    template_name = 'noblackoutapp/result.html'
    context_object_name = 'result'
    infoToCalculate = None
    finalResult = None

```

```

def calculate(self, calculator_info):
    price = {}
    time = {}
    one_time_expense = {}

```

```

generalResult = {}

for i in range(len(calculator_info)):
    price[calculator_info[i].title] = calculator_info[i].energy /
calculator_info[i].price
    time[calculator_info[i].title] = calculator_info[i].energy /
calculator_info[i].time_to_work
    one_time_expense[calculator_info[i].title] = calculator_info[i].energy
/ calculator_info[i].one_time_expense

max_price = max(price.values())
max_time = max(time.values())
one_time_expense_max = max(one_time_expense.values())

for j in price:
    price[j] = (100*price[j]) / max_price
    time[j] = (100*time[j]) / max_time
    one_time_expense[j] = (100*one_time_expense[j]) / one_time_expense_max

for k in calculator_info:
    generalResult[k.title] = round((price[k.title] + time[k.title] +
one_time_expense[k.title]) / 3, 2)

generalResultKey = max(generalResult, key=generalResult.get)
generalResultValue = max(generalResult.values())

```

```

        directory_instance_to_update = Directory.objects.get(title =
generalResultKey)

        directory_instance_to_update.usage_quantity += 1

        directory_instance_to_update.save()

        directory_instance_to_update2 = CategoryOfEnergy.objects.get(name =
directory_instance_to_update.type_of_energy)

        directory_instance_to_update2.usage_quantity += 1

        directory_instance_to_update2.save()

    return generalResultKey, generalResultValue

```

```

def get_context_data(self, *, object_list = None, **kwargs):

    context = super().get_context_data(**kwargs)

    data = self.request.session.get('data')

    if data['night_mode'] == 'true': self.infoToCalculate =
Directory.objects.filter(is_published = True, energy__gte = float(data["energy"]),
price__lte = float(data["money"]), time_to_work__gte = float(data["time"]),
night_mode_work = True)

    elif data['inside_apartment'] == 'true': self.infoToCalculate =
Directory.objects.filter(is_published = True, energy__gte = float(data["energy"]),
price__lte = float(data["money"]), time_to_work__gte = float(data["time"]),
inside_apartment = True)

    elif data['night_mode'] == 'true' and data['inside_apartment'] == 'true':
self.infoToCalculate = Directory.objects.filter(is_published = True, energy__gte =
float(data["energy"]), price__lte = float(data["money"]), time_to_work__gte =
float(data["time"]), night_mode_work = True, inside_apartment = True)

    else: self.infoToCalculate = Directory.objects.filter(is_published = True,
energy__gte = float(data["energy"]), price__lte = float(data["money"]),
time_to_work__gte = float(data["time"]))

```

```

if not self.infoToCalculate.exists():
    c_def = self.get_user_context(title = 'Результат', result = "error")
    return dict(list(context.items()) + list(c_def.items()))

else:
    self.finalResult = self.calculate(self.infoToCalculate)

    c_def = self.get_user_context(title = 'Результат', result =
Directory.objects.get(title = self.finalResult[0]), per_cent =
self.finalResult[1])
    return dict(list(context.items()) + list(c_def.items()))

```

ДОДАТОК Б Презентація



НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
Кафедра Інженерії програмного забезпечення в енергетиці



«Рекомендаційна система підготовки оселі до блекауту»

Виконав: студент магістерського рівня 2-го року навчання, групи ТВ-з21мп

Кучерук Владислав Євгенович

Керівник: к.т.н. Кравцов Григорій Олексійович



Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

АКТУАЛЬНІСТЬ ТЕМИ ДОСЛІДЖЕННЯ

Сьогодення характеризується глибоким проникненням комп'ютеризації у майже всі сфери людського життя. Вона відіграє не останню роль у звичній людській рутині поточного століття.



Рисунок 1 – Дослідження компанії Gradus Research за жовтень 2022



Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

2/12

ПОСТАНОВКА ЗАДАЧІ

Мета: підвищення рівня освіченості людей щодо вибору засобів електроживлення для власних осель, які допоможуть підвищити готовність та стійкість оселі до блекауту, а також зменшать, різного роду, негативний вплив відключень електроенергії.

Об'єктом дослідження є процес надання рекомендації щодо вибору засобів резервного електроживлення для оселі.

Предметом дослідження є вебзастосунок рекомендаційної системи для ефективної підготовки власних осель до блекауту.



ПОСТАНОВКА ЗАДАЧІ

Задачі, які необхідно розв'язати для досягнення мети:

- аналіз поточної ситуації щодо підготовки власних осель до відключень світла у порівнянні з конкурентами;
- пропонування концепції зручного рішення для легшої та швидшої підготовки оселі до блекауту;
- розробка алгоритму підбору найбільш релевантного засобу для потреб користувача;
- розробка рекомендаційної системи.



ІСНУЮЧІ АНАЛОГИ НА РИНКУ

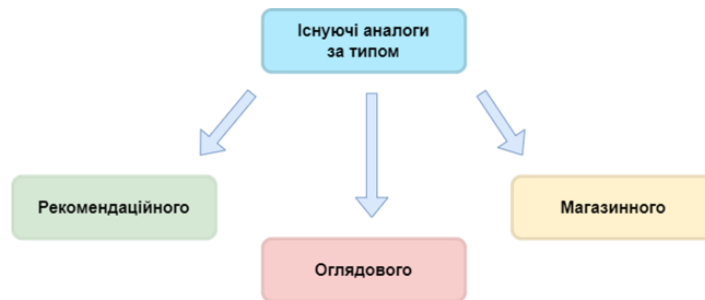


Рисунок 2 – Аналоги для рекомендаційної системи підготовки оселі до блекауту



КЛЮЧОВІ ОСОБЛИВОСТІ ПРОПОНОВАНОГО РІШЕННЯ



1. Статистика



2. Розрахунок



3. Знання



ОСОБЛИВОСТІ РОЗРОБЛЕНОЇ СИСТЕМИ

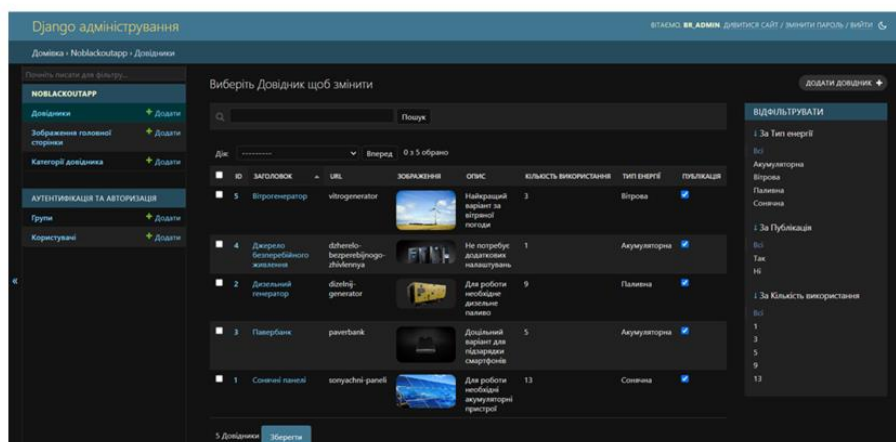


Рисунок 3 – Адміністративна панель



ОСОБЛИВОСТІ РОЗРОБЛЕНОЇ СИСТЕМИ

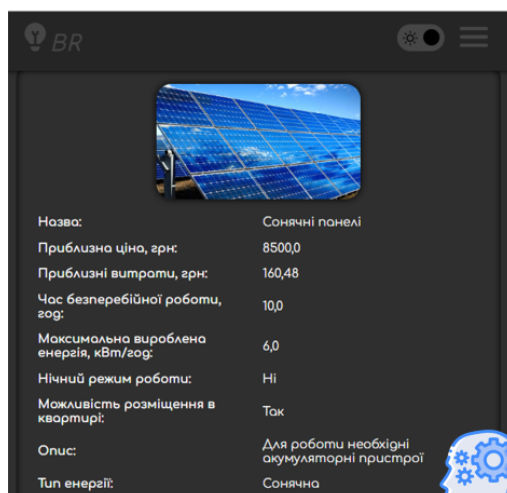


Рисунок 4 – Сторінка бази знань

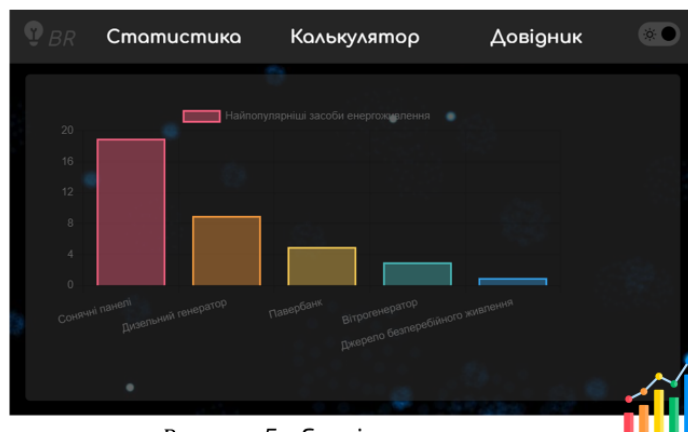


Рисунок 5 – Сторінка статистики



ПРИКЛАД ПІДБОРУ РЕКОМЕНДАЦІЙ

Таблиця 1 – Найкращі засоби за витрачені кошти

Найменування	К-ть енергії, кВт, e_i	Вартість, грн, p_i	Результат, r_{pi}	Відсотковий результат, p_{pci}
Сонячні панелі	6	8 500	0,0007	100%
Вітрогенератор	18	32 000	0,00056	80%



$$r_{pi} = \frac{e_i}{p_i}, \quad (1)$$

де r_{pi} – розрахований результат кількості виробленої енергії на одиницю витрачених коштів для конкретного пристрою;
 e_i – кількість виробленої енергії конкретним пристроєм;
 p_i – вартість конкретного пристрою.



ПРИКЛАД ПІДБОРУ РЕКОМЕНДАЦІЙ

Таблиця 2 – Релевантність засобів за запитом користувача

Найменування	p_{pci}	t_{pci}	e_{pci}	Результат, r_i
Сонячні панелі	100%	86%	100%	95%
Вітрогенератор	80%	100%	78%	86%

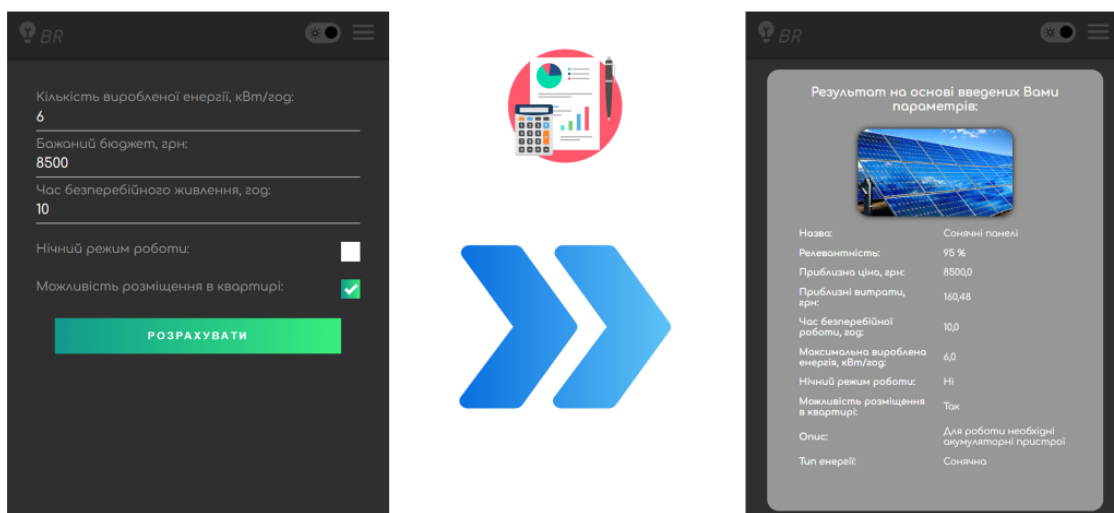


$$r_i = \frac{p_{pci} + t_{pci} + e_{pci}}{3}, \quad (2)$$

де r_i – загальний відсотковий результат для конкретного пристрою;
 p_{pci} – відсотковий результат вартості в порівнянні з іншими пристроями;
 t_{pci} – відсотковий результат часу безперебійної роботи в порівнянні з іншими пристроями;
 e_{pci} – відсотковий результат одноразових витрат на один цикл роботи в порівнянні з іншими пристроями.



ПРИКЛАД ПІДБОРУ РЕКОМЕНДАЦІЙ



The diagram illustrates the process of selecting recommendations. It starts with a form on the left for inputting parameters, followed by a central icon representing calculation, and ends with a result screen on the right.

Input Parameters (Left Screen):

- Кількість виробленої енергії, кВт/год: 6
- Бажаний бюджет, грн: 8500
- Час безперебійного живлення, год: 10
- Нічний режим роботи: ☐
- Можливість розміщення в квартирі: ☒
- Button: РОЗРАХУВАТИ

Calculation (Center): Represented by a calculator icon and a large blue double arrow pointing right.

Result (Right Screen):

Результат на основі введених Вами параметрів:



Назва:	Сонячні панелі
Релевантність:	95 %
Приблизна ціна, грн:	8500,0
Приблизні витрати, грн:	160,48
Час безперебійної роботи, год:	10,0
Максимально вироблена енергія, кВт/год:	6,0
Нічний режим роботи:	Ні
Можливість розміщення в квартирі:	Так
Опис:	Для роботи необхідні акумуляторні пристрої
Тип енергії:	Сонячна

Рисунок 6 – Сторінка розрахунку та кінцевого результату



ВИСНОВКИ

Задачі, які було розв'язано для досягнення мети:

- проаналізовано поточну ситуацію щодо підготовки власної оселі до відключень світла та визначено доцільність розроблюваного рішення;
- запропоновано концепцію рішення підготовки оселі до блекауту у вигляді вебзастосунку, яка полегшує процес ознайомлення та вибору засобів резервного електроживлення;
- розроблено алгоритм підбору найбільш релевантного засобу шляхом зіставлення відсоткових показників порівнюваних засобів;
- розроблено рекомендаційну систему з використанням технології Python Django.

