МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК
СЕКЦІЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ПРОЕКТУВАННЯ

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: «Аналіз методів визначення впливу погодних умов на продуктивність альтернативних джерел енергії»

за спеціальністю 122 «Комп'ютерні науки»,
освітньо-професійна програма «Інформаційні технології проектування»

Виконавець роботи: студентка групи ІТ.м-91 Казлаускайте Анастасія Сергіївна

Кваліфікаційну роботу захищено на засіданні ЕК з оцінкою _____ «___» грудня 2020 р.

Науковий керівник _____ к.т.н., доц., Шендрик В.В.
(підпис)

Голова комісії _____ Шифрін Д.М.
(підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент _____
(підпис)

Суми-2020

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

ЗАТВЕРДЖУЮ
Зав. секцією ІТП

_____ В. В. Шендрик
«___» _____ 2020 р.

ЗАВДАННЯ

на кваліфікаційну роботу магістра студентові

Казлаускайте Анастасія Сергіївна
(прізвище, ім'я, по батькові)

1 Тема проекту *Аналіз методів визначення впливу погодних умов на продуктивність альтернативних джерел енергії*

затверджена наказом по університету від «26»листопада 2020 р. № 1824 - III

2 Термін здачі студентом закінченого проекту «_07__» ___грудня___ 2020 р.

3 Вхідні дані до проекту методи визначення впливу погодних умов, зібрані дані освітленості у форматі xlsx.

4 Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити) аналіз предметної області, постановка задачі та методи дослідження, аналіз методів прийняття рішень, практична реалізація кращого методу.

5 Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) актуальність проблеми, постановка задачі, моделі визначення впливу метеоданих на рівні генерації АДЕ, модель операційної логіки енергосистеми, аналіз методів, проектування практичної реалізації, практична реалізація, висновки, результати апробації**.**

6. Консультанти випускної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

Дата видачі завдання _____.

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів випускної проекту	Термін виконання етапів проекту	Примітка
1	Аналіз предметної області	01.09.20 – 10.09.20	
2	Визначення в потребі системи	11.09.20 – 17.09.20	
3	Ідентифікація ідеї проекту	18.09.20 – 21.09.20	
4	Аналіз методів прийняття рішень	22.09.20 – 05.10.20	
5	Визначення інструментарію	06.09.20 – 07.10.20	
6	Планування WBS	08.09.20 – 09.10.20	
7	Планування OBS	08.09.20 – 09.10.20	
8	Складання календарного плану	12.10.20 – 12.10.20	
9	Визначення ризиків	13.10.20 – 13.10.20	
10	Визначення застосування методів	14.10.20 – 19.10.20	
11	Порівняння методів	20.10.20 – 28.10.20	
12	Створення бази даних	29.10.20 – 30.10.20	
13	Створення програмних модулів	02.11.20 – 23.11.20	
14	Створення графічного інтерфейсу	24.11.20 – 25.11.20	
15	Компілювання програмного коду	26.11.20 – 30.11.20	
16	Тестування	01.12.20 – 03.12.20	
17	Оформлення документації	06.10.20 – 10.12.20	
18	Введення в експлуатацію	11.12.20 – 11.12.20	
19	Архівація проекту	14.12.20 – 14.12.20	

Магістрант _____ Казлаускайте А.С.

Керівник роботи _____ к.т.н., доц. Шендрик В.В.

РЕФЕРАТ

Тема кваліфікаційної роботи магістра «Аналіз методів визначення впливу погодних умов на продуктивність альтернативних джерел енергії».

Пояснювальна записка складається з переліку умовних позначень, вступу, 4 розділів, висновків, списку використаної літератури із 36 джерел і додатків. Загальний обсяг роботи – 138 сторінок, у тому числі 71 сторінка основного тексту, 3 сторінки списку використаної літератури, 63 сторінки додатків.

Кваліфікаційну роботу магістра присвячено проведенню аналізу методів визначення впливу постійно-змінних погодних умов на продуктивність альтернативних джерел енергії. У даній роботі було проведено аналіз предметної області, аналіз методів прийняття управлінських рішень, деталізовано мету проекту і визначено перелік задач. Далі було виконано моделювання практичної реалізації дипломної роботи з урахуванням математичних моделей та аналізу методів відповідно до визначених критеріїв порівняння. Результатом проведеної роботи є інформаційна технологія виявлення найбільш ефективного методу визначення впливу метеорологічних умов на продуктивність альтернативних джерел енергії з практичною реалізацією у вигляді програмного додатку, що зпрощує взаємодію користувача зі складним апаратним устаткуванням.

Ключові слова: python, аналіз методів, інформаційна технологія, АДЕ, гібридна енергосистема.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

АДЕ – альтернативні джерела енергії

ГЕСВДЕ – гібридна енергетична система з відновлювальними джерелами енергії

АБ – акумуляторна батарея

СБ – сонячна батарея

ВЕУ – вітрогенератор

БД – база даних

ВСТУП

Актуальність: Метеорологічні дані швидко змінюються територіально, що породжує певного роду невизначеність даних, тому виникає необхідність оцінювати продуктивність альтернативних джерел енергії (АДЕ) завчасно. Для вирішення цієї проблеми існує потреба в більш детальних дослідженнях методів та визначення кращого, який би став базисом для створення ефективного інструментарію управління енергією за умови результативного її виробництва. Тому існує необхідність у першочерговому аналізі впливу погодних умов на продуктивність АДЕ та аналізі методів прийняття управлінських рішень відповідно до ідентифікованих факторів впливу.

На сьогодні для управління оцінкою продуктивності такого роду існує певний ряд автоматизованих систем, де в основі операційної логіки використовуються дискретні дані і виконуються операції за методом дерева рішень. Зауважимо, що при використанні даного методу не враховується неточність та неповнота вхідних даних, що провокує неточність і обробки даних. Такий принцип управління враховує далеко не всі існуючі фактори.

Тема: Аналіз методів визначення впливу погодних умов на продуктивність альтернативних джерел енергії.

Мета: Створення інформаційної технології, яка буде виконувати аналіз методів дослідження впливу постійно-змінних погодних умов на продуктивність альтернативних джерел енергії.

Об’єкт дослідження: Процес прийняття рішень при оцінці впливу погодних умов на продуктивність альтернативних джерел енергії.

Предмет дослідження: Методи, моделі та інформаційна технологія для прийняття рішення щодо оцінки впливу погодних умов на продуктивність АДЕ.

Гіпотеза: Використання розробленої інформаційної технології дозволить виявити найефективніший з методів визначення впливу погодних умов на продуктивність АДЕ в залежності від технологічних та постійно змінних метеорологічних факторів.

Наукова новизна: Проведення аналізу створених методів визначення впливу погодних умов на продуктивність альтернативних джерел енергії.

Практична новизна: Використання інформаційної технології дозволить поєднати в єдиній екосистемі три різних метода для розрахунку потужності альтернативного джерела енергії та дозволить виявити найефективніший з цих методів з урахуванням типів вхідних даних, витрат на час та об'єм ресурсних витрат програмного забезпечення та апаратної частини ПК.

Впровадження: Результати впровадженні в навчальний процес Сумського державного університету при вивченні дисципліни «Інтелектуальний аналіз даних».

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Ідентифікація проблеми

Сучасний розвиток світу, особливо економіки, залежить від постійного розвитку в сфері енергетики. Суспільство здебільшого використовує викопне паливо, а саме нафтопродукти, природний газ та вугілля, для виробництва електроенергії. Однак через обмеженість запасів природних ресурсів та необґрунтоване використання існує небезпека енергетичної кризи та небезпека створення складних екологічних умов [30]. Тому виникає питання про формулювання заходів та вжиття певних дій для подолання цієї проблеми. Використання відновлювальних джерел енергії розглядається як ефективне рішення.

На сьогодні здебільшого електроенергія виробляється на електростанціях централізовано. Без огляду на те, що великі електростанції мають величезний енергетичний потенціал, все одно існує потреба негайно передавати електроенергію, оскільки її не можна акумулювати та зберігати у великих кількостях, не спричиняючи великих втрат, а потрібно негайно споживати [29]. Передача потужності на великі відстані також супроводжується втратою напруги. В результаті все більше і більше країн використовують концепцію розподіленого виробництва енергії, яка передбачає вироблення електроенергії кінцевими споживачами для задоволення індивідуальних потреб та передачу решти електроенергії до загальної мережі (якщо така є) [33]. Розподілені електростанції використовують місцеві відновлювані джерела енергії (сонячні, вітрові тощо). Якщо використовується кілька джерел енергії, енергетична система може бути гібридною [34].

Через регіональні відмінності в метеорологічних умовах необхідно заздалегідь оцінити продуктивність джерел відновлюваної енергії, тому що погодні умови та

технічний стан устаткування несуть в собі певного роду невизначеність. Для вирішення цієї проблеми необхідні більш детальні дослідження методів визначення кращого джерела електроенергії та створення дієвих інструментів управління енергією з точки зору її ефективного виробництва.

Власник гібридної електростанції повинен сприймати та аналізувати велику кількість інформації та приймати ефективні управлінські рішення. Технологічні системи стають дедалі складнішими, що приносить інформаційний тиск на людей. Тому саме інформаційні технології можна використовувати для вдосконалення процесу управління та вирішення проблеми взаємодії між користувачами та складними технічними об’єктами, що використовують найчастіше чіткі правила для автоматичного керування перемиканням джерел та не враховують неповноту і невизначеність вхідних даних.

Передумовою безпомилкової роботи інформаційної технології є наявність поточних даних, прогнозована ефективність сонячних панелей та вітряних турбін, а також врахування всіх факторів, що впливають на точність алгоритмів обчислення. Поточні дані збираються з метеорологічного сайту, а також з певного ряду датчиків, тоді як прогнозні дані можна отримати лише з математичних моделей.

1.2 Аналіз методів прийняття управлінських рішень

Після проведення детального аналізу предметної області, а саме виявлення актуальності проблеми визначення впливу перманентно змінних погодних умов, постає задача аналізу методів та підходів до розв’язку поставленої проблеми. Проблема полягає в тому, що за операційною логікою сучасних автоматизованих систем управління використовуються дискретні вхідні дані, а також оброблюються в більшості випадків за чіткими правилами, не враховуючи всі існуючі фактори, невизначеність і неповноту даних.

Питання визначення залежності кількості вироблюваної енергії від метеоумов зводиться до прийняття управлінських рішень, щодо кращого джерела електроенергії в даний момент часу при даних погодних показниках.

Прийняття рішень – це процес вибору найкращого рішення з допустимої безлічі рішень або впорядкування безлічі рішень. Прийняття рішень можливо на підставі знань про об'єкт управління, про процеси, які в ньому протікають, і тих, які можуть статися з плином часу. Тому необхідною є наявність адекватної моделі об'єкта, і при наявності безлічі показників (критеріїв), що характеризують ефективність прийнятого рішення [23].

Відповідно до поставленої мети дипломного проекту необхідно реалізувати інформаційну технологію, яка буде виконувати аналіз методів дослідження впливу постійно-змінних погодних умов на продуктивність альтернативних джерел енергії. В ході виконання дипломної роботі розглядається гібридна енергосистема, яка обслуговує одне або кілька домогосподарств. Електрична система складається з набору вітрогенераторів, сонячних панелей та акумулятора для зберігання електричної енергії. Крім того, існує підключення до зовнішньої електромережі для передачі надлишкової енергії. Схема цієї системи наведена на рисунку 1.1.

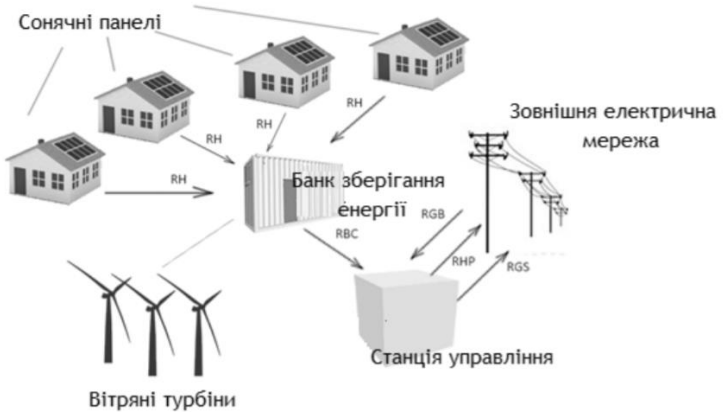

Рисунок 1.1 – Схема гібридної електросистеми

В ході виконання дипломної роботи необхідно провести аналіз методів визначення впливу постійно змінних метеоумов на продуктивність відновлювальних джерел енергії. Задача прийняття рішення може бути зведена до класифікації відновлюваного джерела енергії відповідно до погодних та технологічних характеристик. Дерева рішень є одним з найпопулярніших методів класифікації і допомагають процесу прийняття рішень. Цей метод є найбільш популярним для створення автоматизованих систем управління, та, на жаль, враховує далеко не всі фактори, а отже не є найбільш точним.

Дерево рішень – це спрямоване дерево з кореневим вузлом, який не має вхідних ребер, а всі інші ребра з рівно одним вхідним краєм, відомі як вузли прийняття рішень. На етапі навчання кожен внутрішній вузол розділяє простір екземпляра на дві або більше частин з метою оптимізації роботи класифікатора. Після цього кожен шлях від кореневого вузла до листового вузла формує правило рішення, щоб визначити, до якого класу належить новий екземпляр [3].

Дерево рішень – це класифікатор, який проводить рекурсивний розділ над простором екземпляра. Типове дерево рішень складається з внутрішніх вузлів, країв і вузлів листків. Кожен внутрішній вузол називається вузлом прийняття рішення, що представляє тест на атрибут або підмножину атрибутів, і кожне ребро позначене

певним значенням або діапазоном значення вхідних атрибутів. Таким чином, внутрішні вузли, пов'язані з їх краями, ділять простір екземпляра на два або більше розділів. Кожен листовий вузол є кінцевим вузлом дерева з міткою класу. Наприклад, на рисунку 1.2 наведено ілюстрацію основного дерева рішень, де коло означає вузол рішення, а квадрат - листовий вузол. У цьому прикладі ми маємо три атрибути розділення, а також дві мітки класів, тобто YES та NO. Кожен шлях від кореневого вузла до листового вузла утворює правило класифікації.

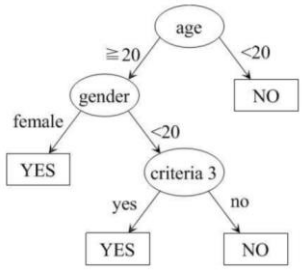

Рисунок 1.2 – Приклад дерева рішень

Загальний процес побудови дерева рішень такий. Отримавши набір навчальних даних, застосовується функція вимірювання до всіх атрибутів, щоб знайти найкращий атрибут розділення. Після визначення атрибута розділення простір екземпляра розділяється на кілька частин. У межах кожного розділу, якщо всі навчальні екземпляри належать одному класу, алгоритм припиняється. В іншому випадку процес розбиття буде виконуватися рекурсивно, поки весь розділ не буде присвоєно одному класу. Після побудови дерева рішень можна легко генерувати правила класифікації, які можна використовувати для класифікації нових екземплярів з невідомими мітками класів [8].

Порівняно з іншими загальноприйнятими методами класифікації, прогнозування або рідкісного використання, використання дерев рішень має наступні важливі переваги:

– Відсутність обробки даних. Наприклад, немає необхідності в нормалізації, додаванні фіктивних змінних та видаленні відсутніх даних. Цей метод може використовувати або категоріальні змінні, або інтервальні змінні.

– Існує можливість інтуїтивного розуміння та пояснення. Правила руху до вузлів дерева можуть мати чітку логіку ("білий ящик") і можуть бути формалізовані навіть там, де експерту важко зробити аналіз. Зауважмо, що нейронні мережі зазвичай не надають користувачам такої можливості ("чорний ящик").

– Висока надійність і точність. Висока надійність та точність моделі можуть бути статистично оціненими. Відсутність апріорних припущень про закон розподілу даних робить їх непараметричними та стабільними для різних даних із невідомими апріорними законами розподілу.

– Висока обчислювальна ефективність. Метод характеризується швидким процесом навчання (побудова дерева), і оскільки структура даних дерева рішень проста, вона має високу обчислювальну ефективність для обробки великих обсягів даних [24].

Не дивлячись на те, що метод дерево рішень має високі показники ефективності результуючих даних та є легким в експлуатації, розглянутий метод має недоліки, що перелічені далі:

– Дерево рішень чутливе до шуму у вхідних даних. Незначні зміни в навчальних зразках можуть призвести до загального коригування моделі, що вплине на зміни в правилах класифікації моделі та інтерпретації.

– Існують певні обмеження у розподілі межі, тому дерево рішень щодо якості класифікації не є таким хорошим, як інші методи.

– Дерево рішень можна перекваліфікувати, саме тому доводиться вдаватися до методу «зрізання гілок», встановлюючи мінімальну кількість елементів у листі дерева або максимальну глибину дерева.

– Складний пошук дерева найкращих рішень: Це призводить до необхідності використання евристичних методів (наприклад, жадібний пошук

об’єктів) для максимізації інформації, що в кінцевому рахунку не забезпечує 100% гарантії пошуку найкращого дерева.

– Дерево рішень робить постійний прогноз для об’єкта в просторі ознак поза паралелепіпедом, і об’єкт не охоплює всіх об’єктів у навчальній вибірці [33].

Після детального аналізу методу дерева рішень: визначення алгоритму роботи з даними, визначення варіантів формату отриманих вихідних даних, ідентифікація переваг та недоліків використання методів, необхідно провести дослідження застосування інших методів.

Пропонується для визначення джерела електроенергії, для вдосконалення операційної логіки алгоритму використовувати метод кластеризації або нечітких правил. Такі методи враховують залежність від нечітких метеорологічних факторів, а також неповноти вхідних технологічних даних.

Метод кластеризації - це процес групування подібних наборів даних. Ця група не контролюється; це робиться без використання відомих структур у даних. Кластеризація спрямована на створення кластерів із зразками даних, які більше схожі між собою, ніж зразки даних, що належать до інших кластерів. Кожне скупчення визначається центральною точкою, центроїдом. Подібність даних в одному кластері вимірюється за допомогою різних критеріїв. Таким чином, існує безліч різних методів, які можуть вирішити загальне завдання кластеризації [8]

Кластеризація набуває цінність тоді, коли вона виступає одним з етапів аналізу даних, побудови закінченого аналітичного рішення. Аналітику часто легше виділити групи схожих об'єктів, вивчити їх особливості і побудувати для кожної групи окрему модель, ніж створювати одну загальну модель на всіх даних [22].

Кластерний аналіз є одним з основних методів аналізу даних, що представляє собою сукупність методів та алгоритмів, спрямованих на розподіл певної сукупності об’єктів дослідження на підмножини подібних один одному об'єктів [30].

Найбільша перевага кластерного аналізу полягає в тому, що він дозволяє розділити об'єкти не тільки за одним параметром, але і за набором показників. Крім

того, на відміну від більшості математичних та статистичних методів, кластерний аналіз не накладає жодних обмежень на типи об'єктів, що розглядаються, і дозволяє розглянути багато вихідних даних майже будь-якого характеру. Кластерний аналіз дозволяє розглянути велику кількість інформації та різко її зменшити, стиснути велику кількість інформації, щоб зробити її компактною та зрозумілою.

Як і будь-який інший метод, кластерний аналіз також має певні недоліки та обмеження: зокрема, склад та кількість кластерів залежать від обраних критеріїв розділення. При спрощенні вихідного набору даних до більш компактної форми можуть виникати деякі спотворення, а окремі ознаки одного об'єкта можуть бути втрачені, замінивши його ознаки на узагальнені значення параметрів кластеризації. При класифікації об'єктів зазвичай ігнорується можливість відсутності значення кластера у розглянутому наборі [22].

Останнім методом для аналізу є метод нечіткої логіки. Даний метод є найбільш податливим та гнучким до невизначеності метеорологічних даних.

Нечітка логіка має на меті формалізувати людські можливості в неточних або приблизних формах міркувань, що надає можливість більш повно описати невизначені ситуації. У варіанті нечіткої логіки, запропонованому Л.Заде, сукупність висловлювань із значеннями істини має вигляд дійсних інтервалів [0,1], що дозволяє висловлюванням приймати будь-яке значення істини з цього інтервалу. Ця величина є кількісною оцінкою правдивості твердження, і неможливо повністю визначити, чи є твердження правильним чи неправильним. Використання інтервалу [0,1] як набору значень істини дозволяє нам побудувати логічну систему, в якій можна робити невизначені міркування. У цій системі існує можливість оцінити достовірність таких тверджень: "Швидкість автомобіля досить висока", "Тиск у системі досить високий", "Молоді люди" тощо. [25]

Як і будь-яка нечітка система, що базується на правилах, нечіткі системи правил Мамдані, мають два основні компоненти:

– систему нечітких правил, яка реалізує процес нечітких міркувань, що застосовується на вході системи для отримання вихідних даних системи;

– нечітка база знань (БЗ), яка представляє знання про проблему, що вирішується.

БЗ містить нечіткі ЯКЩО - ТОДІ правила, складені з лінгвістичних змінних, які приймають значення в наборі термінів з реальних значень. Нечіткі набори, що визначають семантику мовних міток, однаково визначені для всіх правил, що включені до БЗ, тим самим полегшуючи читабельність системи для людей. Колекція нечітких лінгвістичних правила становлять описовий підхід, оскільки БЗ стає якісним виразом системи. До того ж при поділі між структурами нечітких правил та їх значенням можна розрізнити два різні компоненти –

– основу нечітких правил, що містить колекцію нечітких правил, і базу даних, що містить функції належності нечітких;

– розділи, пов'язані з мовними змінними.

Це визначає чітке розмежування між структурою нечіткої моделі та параметри, визначені в класичній ідентифікації системи [2].

Використання нечіткої логіки має велику кількість переваг:

– застосування якісних змінних в аналізі;

– використання нечітких вхідних даних для роботи;

– виконання аналізу відповідно до мовних стандартів;

– швидке моделювання складних динамічних систем та можливість їх порівняння із заданою точністю.

Проте нечітка логіка має недоліки, що перелічені нижче:

– вибір функції належності та формування нечітких правил введення є суб'єктивними;

– потрібне спеціальне програмне забезпечення та професіонали, які знають, як ним користуватися [28].

Після детального аналізу кожного з розглянутих методів, можна зробити перший висновок, що дані методи різняться між собою сценарієм виконання аналізу даних та форматом вхідних даних.

Також під час подальшого більш детального аналізу методів прийняття рішень з визначення впливу постійно змінних метеоумов на продуктивність відновлювальних джерел енергії буде проведено порівняльну характеристику методів за певними критеріями, а саме:

- складність алгоритму;
- кінцевий час виконання обробки вхідної вибірки даних;
- статична складність алгоритму;
- ємнісна складність (об’єм пам’яті);
- точність;
- корисність;
- структурованість;
- надійність.

Розглянуті підходи необхідно порівняти за визначеними кількісними та якісними критеріями для того, щоб виявити кращій метод прийняття управлінських рішень.

2 ПОСТАНОВКА ЗАДАЧІ ТА МЕТОДИ ДОСЛІДЖЕННЯ

2.1 Мета та задачі дослідження

Метою дипломної роботи є створення інформаційної технології, яка буде виконувати аналіз методів дослідження впливу постійно-змінних погодних умов на продуктивність альтернативних джерел енергії. Розроблена інформаційна технологія дозволить поєднати в єдиній екосистемі три різних метода для прийняття управлінських рішень щодо кращого джерела електроенергії та дозволить виявити ефективний з цих методів з урахуванням типів вхідних даних, витрат на час та ресурси програмної частини, відповідно до визначених раніше критеріїв. Потенційними користувачами розробленого проекту є фірми, діяльність яких полягає в установці генераторів альтернативних джерел енергії. Оскільки інформаційна технологія дозволить менеджерам фірм виконати аналіз результатів закладених методів для визначення альтернативного джерела енергії.

Розроблена інформаційна технологія повинна виконувати такі функції:

- виконувати пошук геопозиції місцерозташування АДЕ;
- доступ до метеорологічного сайту;
- збереження розрахункових даних;
- визначення АДЕ за кращим методом;
- збереження результуючих даних.

Створена інформаційна технологія повинна мати інтуїтивно зрозумілий інтерфейс користувача, мати забезпечувати повноцінне використання функціоналу з боку користувача в незалежності від його місцезнаходження.

Для досягнення поставленої мети потрібно вирішити відповідні задачі:

- виконати детальний аналіз існуючої проблеми;

- виконати експертний аналіз існуючих методів для реалізації проекту відповідно до визначених критеріїв;
- розробити математичну модель вибору ефективного режиму функціонування електросистеми;
- виконати реалізацію функціоналу програмного продукту;
- провести тестування інформаційної технології.

Під час ідентифікації мети проекту було проведено планування робіт проекту, що представлено у Додатку А.

2.2 Методи дослідження

Після визначення мети проекту, зазначивши перелік функціональних вимог до практичної реалізації методу визначення впливу погодних умов на продуктивність альтернативних джерел, а також склавши перелік задач, необхідних для реалізації проекту, потрібно було визначити методи наукового дослідження.

До методів наукового дослідження відносять такі види як:

- емпіричний метод;
- експерементальний метод;
- теоритичний метод;
- комплексний метод [21].

Емпіричний метод базується на певних спостереженнях, експериментах, що грутруються на даних досвіду. Виконуючи розробку проекту «Аналіз методів визначення впливу погодгних умов на продуктивність альтернативних джерел енергії» було застосовано емпіричний метод для визначення кращого за певними показниками методу прийняття управлінських рішень для визначення джерела

електроенергії, а також емпіричний метод було використано для тестування математичної моделі в практичній реалізації даного проекту.

Експерементальний метод є загальнонауковим методом дослідження, що представляє собою активну теоритичну та практичну діяльність експерементатора. Експерементальна перевірка забезпечує верифікацію ефективності нових способів, методів та концепцій, що були запропоновані автором дослідження. У випадку з проектом «Аналіз методів визначення впливу погодних умов на продуктивність альтернативних джерел енергії» було застосовано експерементальний метод при відлагодженні та тестуванні програмної реалізації.

Для теоритичного дослідження об’єкту необхідно вводити певні абстрактні поняття, пропонувати гіпотези та висувати теорії, що демонструють та пояснюють внутрішній механізм явища. Під час розробки даного дипломного проекту теоритичний метод було застосовано при побудові математичної моделі генерації енергії від альтернативних джерел енергії, моделі споживання електроенергії, моделі операційної логіки роботи гібридної енергосистеми, а також методах визначення впливу погодних умов на продуктивність альтернативних джерел енергії.

Комплексний метод дослідження представляє собою поєднання всіх попередньо описаних методів в єдиному методі, тому доцільно стверджувати, що при розробці дипломного проекту «Аналіз методів визначення впливу погодних умов на продуктивність альтернативних джерел енергії» було застосовано саме комплексний метод наукового дослідження.

3 Моделювання дипломного проекту

Після попереднього проведення аналізу предметної області, було визначено актуальність проблеми, ідентифіковано головну ідею даного дипломного проекту, проведено аналіз методів прийняття управлінських рішень, визначено перелік критеріїв необхідних для порівняння обраних методів, а також виконано планування робіт проекту. Далі необхідно виконати моделювання даної роботи.

3.1 Структурно-функціональне моделювання

IDEF (Integration Definition) являє собою певний набір стандартизованих методів і сімейства графічних мов, що використовуються для інформаційного моделювання в галузі програмної інженерії, бізнес-процесів, та їх вдосконалення [5].

Метод IDEF0 використовується для визначення моделей функцій, що також є моделями класу «що робити». Такі моделі є описовими та демонструють діяльність процесу на високому рівні. Моделі нотації IDEF0 демонструють основні види діяльності та вхід, вихід, контроль та механізми, що пов’язані з кожною основною діяльністю. Складні процеси можна додатково розкласти, щоб продемонструвати діяльність нижніх рівнів [10].

Після детермінації кращого методу визначення впливу погодних умов у даній дипломній роботі цей метод має практичну реалізацію в вигляді програмного продукту. Уся логіка обробки операцій, що міститься в програмному продукті, відбувається за математичними моделями. Для представлення залежностей етапів роботи самої інформаційної технології біло створено функціональні моделі

методології SADT у нотації IDEF0. На рис. 3.1 представлено контекстну діаграму нульового рівня.

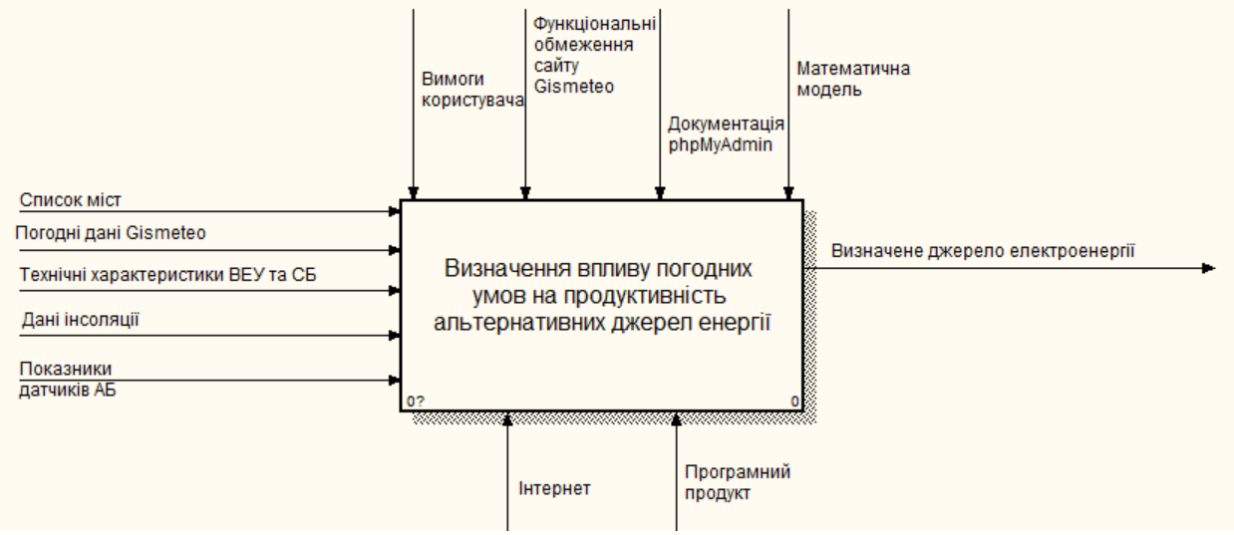

Рисунок 3.1 – Контекстна діаграма нотації IDEF0

Після формування контекстної діаграми нульового рівня, було проаналізовано складність проекту, та вирішено виконати декомпозицію на 3 головних підпроцеси:

– Заповнення бази даних;
– Розрахунок потужностей та рівня споживання;
– Визначення пріоритетності АДЕ.

Декомпозована діаграма першого рівня, що деталізує батьківську модель, представлена на рис. 3.2. Результатом всіх підпроцесів отримуємо визначене джерело електроенергії.

Далі необхідно деталізувати процес «Заповнення бази даних», тому вирішено декомпозувати контекстну діаграму до 2го рівня. Діаграму декомпозиції «Заповнення бази даних» представлено на рис. 3.3. Результатом даного підпроцесу є заповнена база даних з необхідною інформацією щодо метеорологічних показників та геолокації, необхідної для виконання подальших розрахунків.

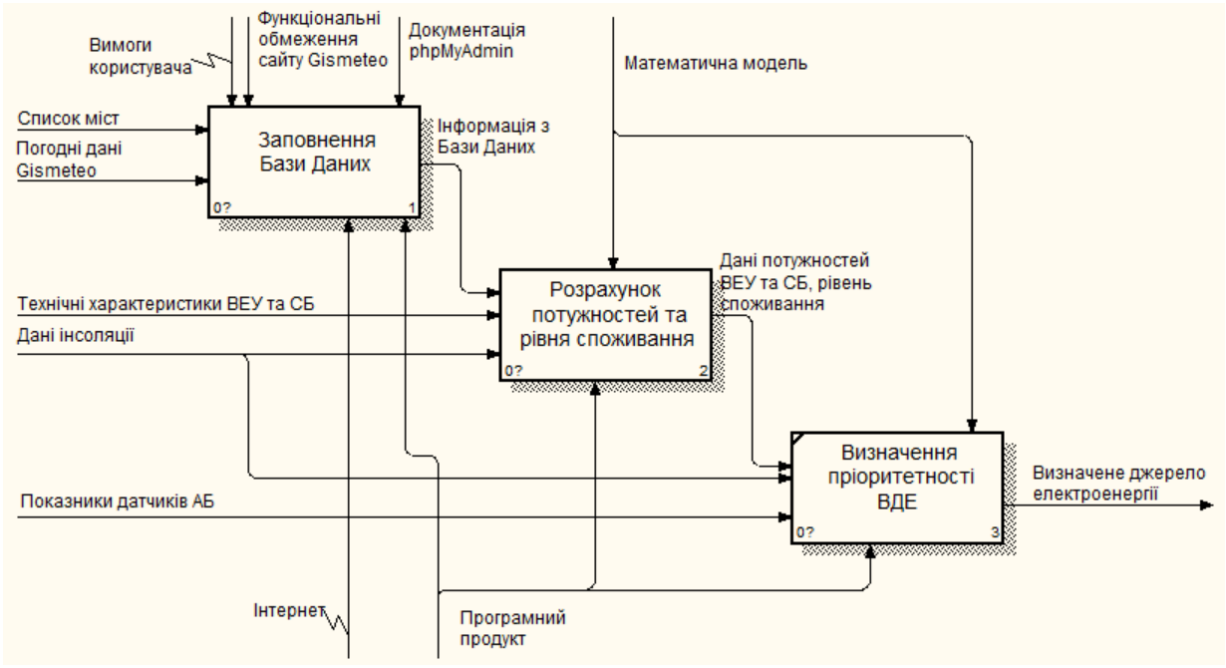

Рисунок 3.2 – Діаграма декомпозиції першого рівня

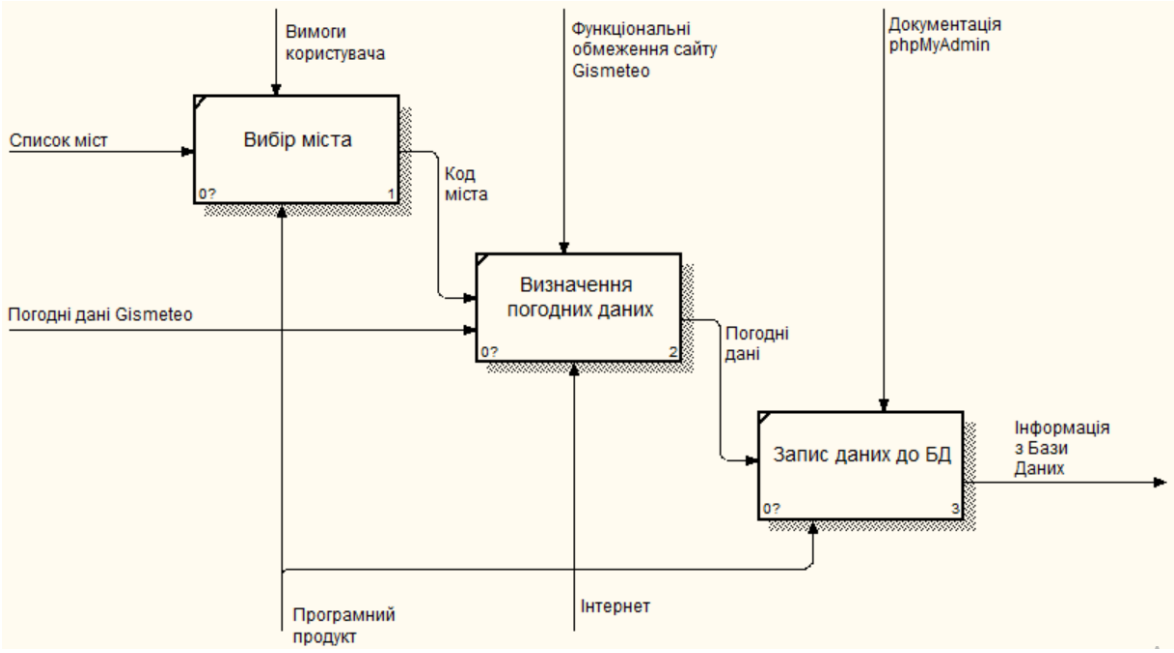

Рисунок 3.3 – Декомпозиція процесу «Заповнення бази даних»

Наступним було декомпозовано процес «Розрахунок потужностей та рівня споживання» на 3 підпроцеси. Як результат отримуємо обраховані потужності вітрогенератора та сонячних панелей, а також рівень споживання електроенергії. Діаграму декомпозиції процесу «Розрахунок потужностей та рівня споживання» представлено на рис. 3.4.

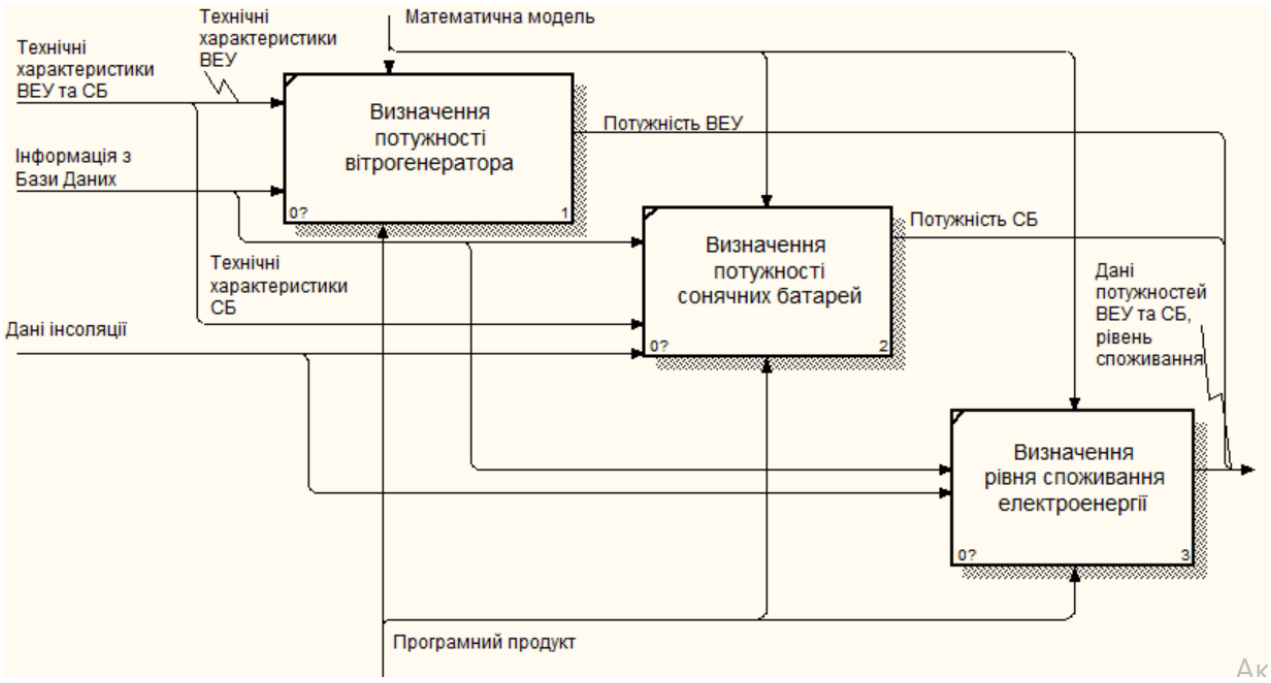

Рисунок 3.4 – Декомпозиція процесу «Розрахунок потужностей та рівня споживання»

3.2 Моделювання бази даних

На сьогоднішній день моделі «сутність-зв’язок» найчастіше використовуються для проектування баз даних. Модель використовує загальну схему блоку для опису концептуальної діаграми. Діаграма «сутність-зв’язок» в основному визначає дані та

взаємозв'язок між ними, а також використовує їх для проектування сховища даних системи, яка повинна бути детально розроблена.

За допомогою засобів СУБД MySQL отримані при розрахунках обраним кращім методом дані зберігаються в базу даних. Взаємодія з базою даних також відбувається під керівництвом СУБД MySQL. Сам програмний продукт координує всі ці дії та отримує дані з бази даних, а також публікує їх у певному форматі: на екрані користувача та у вигляді звіту відповідно до запиту користувача. ER-діаграма бази даних представлено на рис.3.5.

До бази даних заноситься інформація щодо місцезнаходження гібридної електросистеми (назва міста, країни та номер міста на сайті gismeteo.com), метеорологічних умов з погодного сайту (температура повітря, хмарність, швидкість та напрямок вітру, а також дата вимірювання) та отримані результати обрахунків (потужності вітрогенератора та сонячних панелей, залишок електроенергії в акумуляторній батареї, рівень споживання та тип підключеного джерела АДЕ).

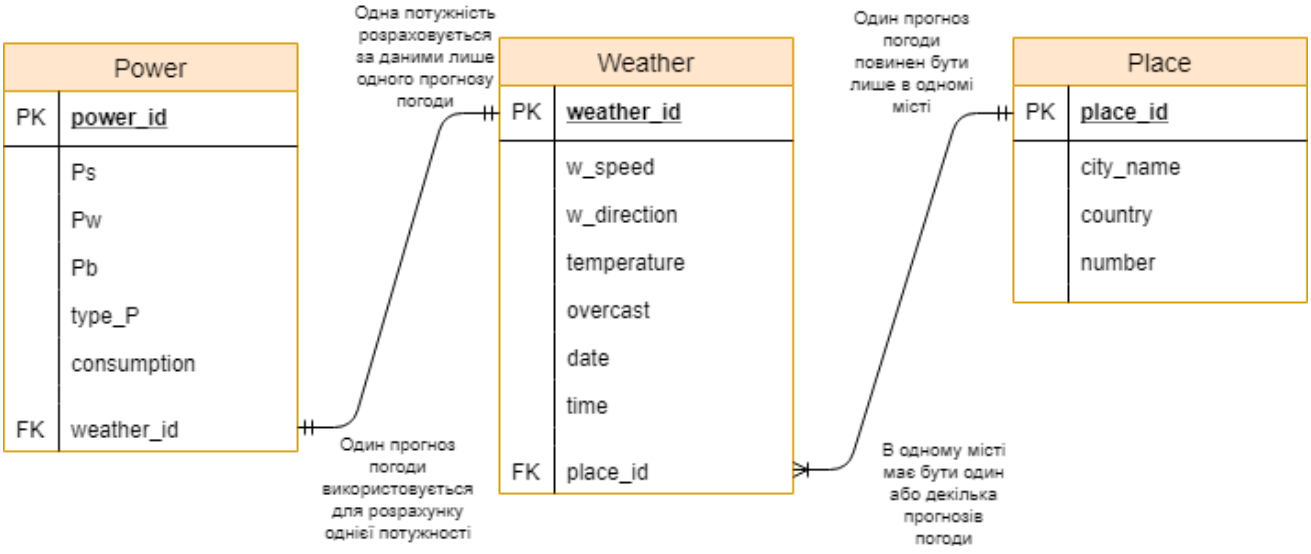

Рисунок 3.5 – Концептуальна модель бази даних

3.3 Моделювання варіантів використання

Діаграми варіантів використання UML є основною формою системних / програмних вимог для нових слаборозвинених програм. У сценаріях використання вказується очікувана поведінка (що), а не точний спосіб її реалізації (як). Ключова концепція моделювання варіантів використання полягає в тому, що воно може допомогти розробити систему з точки зору кінцевого користувача. Діаграма варіантів використання зазвичай має простий вигляд. Вона не відображає деталей варіантів використання:

- узагальнює деякі взаємозв'язки між варіантами використання, акторами та артефактами;
- не відображає порядок, у якому виконуються кроки для досягнення цілей кожного варіанту використання[19].

Для даного дипломного проекту було визначено перелік акторів: користувач (user), сайт прогнозу погоди (gismeteo.com), excel файл (ms excel), база даних (data base) та шаблон звіту (ms word). Далі представлено перелік сценаріїв варіантів використання:

- Parsing – збирання метео данних з сайту прогнозу погоди;
- Search – пошук геолокації;
- Calculation – виконання обчислень потужностей та рівня споживання електроенергії;
- Priority – визначення найбільш відповідного моменту джерела електроенергії;
- Save – збереження отриманих результатів;

На основі сформованих даних було розроблено діаграму Use Case, що представлено на рис. 3.6.

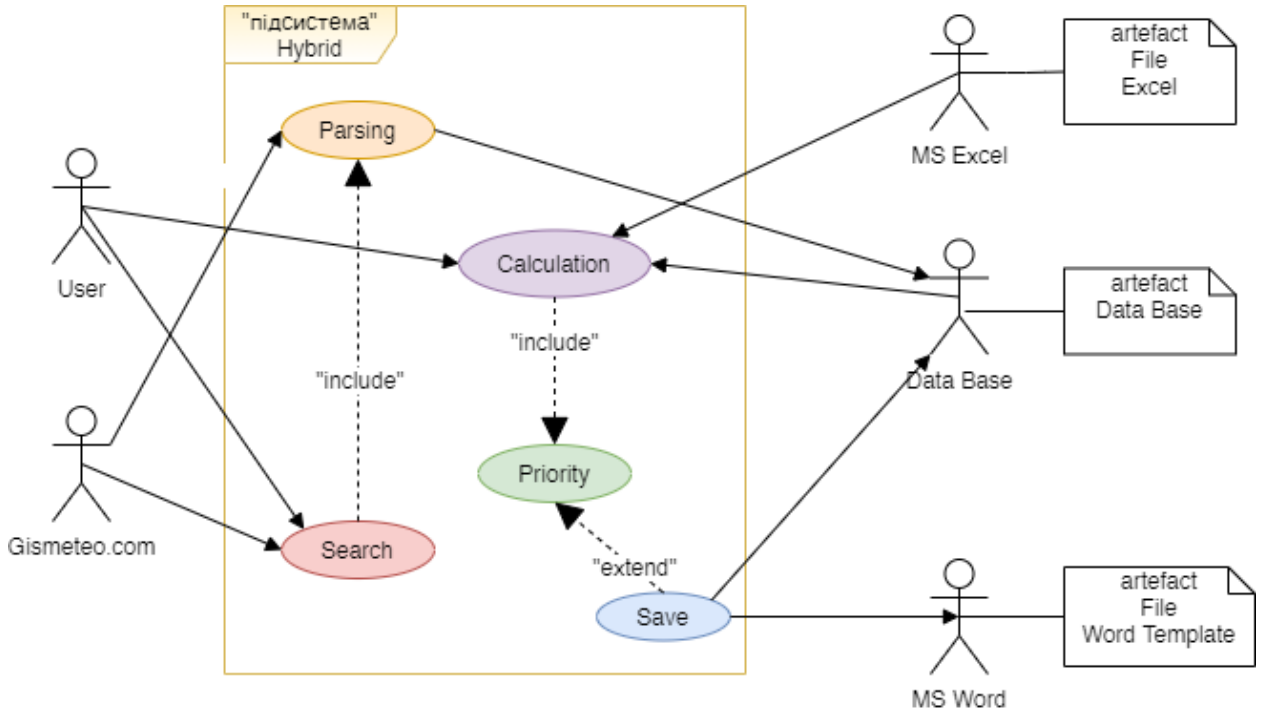

Рисунок 3.6 – Діаграма Use Case

3.4 Математична модель основних процесів

3.4.1 Модель генерації електроенергії в залежності від метеорологічних умов

У даному проекті розглядається електросистема, що складається з вітрогенераторів, сонячних батарей та акумуляторів. Для визначення джерела електричної енергії перш за все необхідно визначити рівень генерації електричної енергії від кожного джерела альтернативної енергії.

Пропонується визначити залежність електрогенерації ВЕУ від аеродинамічних характеристик вітроротора, фактору впливу температури повітря та швидкості вітру.

Останні дослідження аеродинамічної характеристики вітроротора $C_p(\lambda)$ демонструють, що даний показник не є постійним значенням, а має певну закономірність зміни відповідно до швидкості вітру [22]:

$$C_p(\lambda, V_w) = 1{,}14(\frac{9{,}47}{\lambda} - 1)e^{\frac{-f(V_w)}{\lambda}}, \text{ де } f(V_w) = 0{,}003869V_w^2 - 0{,}128V_w + 6{,}627 \quad (3.1)$$

Для визначення аеродинамічної характеристики вітроротора також врахуємо залежність густини повітря від його температури $t_{\text{п}}$:

$$\rho(t_{\text{п}}) = 0{,}00001661t_{\text{п}}^2 - 0{,}004764t_{\text{п}} + 1{,}2924 \quad (3.2)$$

Кількість енергії, що згенерована ВЕУ за проміжок часу t визначається за наступною формулою:

$$Ew(t) = \rho(t_{\text{п}}) \times S \times C_p(\lambda, V_w) \times V_w^3(t) \quad (3.3)$$

де S – площа омивання ротора ВЕУ, м2; $C_p(\lambda, V_w)$ – залежність коефіцієнта використання потужності вітру від швидкохідності вітроротора $\lambda = \omega r / V_w$; ω – кутова швидкість вітроротора, рад/с; r – радіус вітроротора, м; V_w – швидкість вітру, м/с; t – час, год.

Відповідно до наявної невизначеності, що несуть в собі метеорологічні показники, пропонується визначати потужність генерації електроенергії від сонячних батарей як трикутне нечітке число [21]:

$$Ep = < Ep_{min}, Ep_{mod}, Ep_{max} > \quad (3.4)$$

де Ep_{mod} – модальне значення; Ep_{min}, Ep_{max} – ліва та права межа інтервалу невизначеності.

Для детермінації залежності $Ep_{min}, Ep_{mod}, Ep_{max}$ від конструктивних та зовнішніх факторів, був оброблений набір експерементальних даних для визначення впливу цих факторів на електричні характеристики сонячних батарей. Досліджено 16 наборів сонячних батарей, площа яких становить 0,0403 м2 при сонячному світлі (550-1260) Вт/м2 та температурі (12-70)°С. Набір містить фотоелементи різного розміру для усунення невизначеності і такого роду [20].

Як результат обробки даних, були отримані залежності нечіткої моделі потужності сонячних батарей (3.4) [4]:

$$\begin{aligned} Ep_{mod} &= (0{,}0901E + 0{,}0873t - 0{,}00032Et)S, \\ Ep_{min} &= (0{,}0876E + 0{,}0499t - 0{,}00027Et)S, \\ Ep_{max} &= (0{,}0918E + 0{,}1055t - 0{,}00035Et)S, \end{aligned} \tag{3.5}$$

де S – загальна площина сонячних батарей, що є складовими гібридної електромережі, м2; Е – освітленість, Вт/м2; t – температура повітря, °С.

Згідно з представленням потужності СБ у вигляді нечіткого кортежу Ep, де $Ep_{min} < Ep_{mod} < Ep_{max}$, то функція належності $\mu_P(x)$ має наступний вигляд:

$$\begin{aligned} \mu_P(x) &= \frac{x - Ep_{min}}{Ep_{mod} - Ep_{min}}, x \in [Ep_{min}, Ep_{mod}]; \\ \mu_P(x) &= \frac{Ep_{max} - x}{Ep_{max} - Ep_{mod}}, x \in [Ep_{mod}, Ep_{max}]; \\ \mu_P(x) &= 0, x \notin [Ep_{min}, Ep_{max}], \end{aligned} \tag{3.6}$$

де для довільного числа $x \in [Ep_{min}, Ep_{mod}]$ справедливим є представлення $x = Ep_{min} + \alpha(Ep_{mod} - Ep_{min})$, а для довільного $x \in [Ep_{mod}, Ep_{max}]$ –

$x = Ep_{max} - \alpha(Ep_{max} - Ep_{mod})$, де $\alpha \in [0,1]$ – заданий рівень належності числа x нечіткій множині Ep [4].

Для того, щоб визначити значення коефіцієнта α, було оброблено дані про освітленість при різних умовах хмарності протягом року (таблиця 3.1). Оскільки дані про освітленість визначаються якісно і характеризуються ступенем хмарності (ясно, слабка хмарність тощо) у відсотках на різних сайтах порізному, було вирішено використовувати в якості основних дані про рівено хмарності з веб-сайту Gismeteo.com:

- «Ясно» - 0%;
- «Слабка хмарність» - 25%;
- «Помірна хмарність» - 50%;
- «Хмарно» - 100%.

Крім того, необхідно враховувати залежність рівнів інсоляції від сезону року, дані про рівень освітленості по місяцях при різних умовах хмарності представлено в таблиці 3.1.

Таблиця 3.1 – Інсоляція в умовах хмарності

Місяць	При безхмарному небі, Вт*год/м2	В умовах повної хмарності, Вт*год/м2
Січень	63,25	38,75
Лютий	94,37	59,08
Березень	160,99	95,48
Квітень	197,34	131,4
Травень	240,51	175,15
Червень	245,57	175,5
Липень	244,57	187,24
Серпень	207,51	165,23

Продовження таблиці 3.1 – Інсоляція в умовах хмарності

Вересень	166,81	117,9
Жовтень	119,06	78,12
Листопад	76,31	42,16
Грудень	56,44	32,34

Було визначено, що при сильній та помірній хмарності рівень потужності СБ належить проміжку $[Ep_{min}, Ep_{mod}]$, а за умови безхмарної погоди або малої хмарності – $[Ep_{mod}, Ep_{max}]$.

За попередніми даними коефіцієнт α для умов «Хмарно» та «Помірна хмарність» визначається за формулою:

$$\alpha = \frac{E2(n)+(E1(n)-E2(n)\times\varphi)}{E1(n)} \quad (3.7)$$

де $E2(n)$ – значення інсоляції за умови повної хмарності, кВт*год/м2; $E1(n)$ – значення інсоляції в умовах безхмарності, кВт*год/м2; n – номер місяця в році; φ – рівень безхмарності,% (для «Хмарно» - 0, для «Помірна хмарність» - 0,5).

Для малохмарної та ясної погоди коефіцієнт α визначається за такою формулою:

$$\alpha = 1 - \frac{E2(n)+(E1(n)-E2(n))\times\varphi}{E1(n)} \quad (3.8)$$

де $E2(n)$ – значення інсоляції при умові повної хмарності, кВт*год/м2; $E1(n)$ – значення інсоляції при умові ясного неба, кВт*год/м2; n – номер місяця у році; φ – рівень безхмарності,% (для «Ясно» - 0, для «Помірна хмарність» - 0,25).

У таблиці 3.2 наведені результати розрахунку коефіцієнта α щомісяця.

З наведених результатів розрахунку коефіцієнта α видно, що в безхмарних умовах потужність СБ приймає максимальне значення за заданих погодних та технологічних умовах.

Таблиця 3.2 – Значення коефіцієнта α

Місяць	Сильна хмарність	Хмарність	Слабка хмарність	Ясно
Січень	0,6126	0,8063	0,0970	0
Лютий	0,6261	0,8130	0,0935	0
Березень	0,5931	0,7966	0,1017	0
Квітень	0,6658	0,8329	0,0835	0
Травень	0,7282	0,8641	0,0679	0
Червень	0,7146	0,8573	0,0713	0
Липень	0,7656	0,8828	0,0586	0
Серпень	0,7963	0,8982	0,0509	0
Вересень	0,7068	0,8534	0,0733	0
Жовтень	0,6561	0,8281	0,0860	0
Листопад	0,5525	0,7763	0,1119	0
Грудень	0,5713	0,7857	0,1072	0

Відповідно до операційної логіки гібридної енергосистеми, у разі недостатнього рівня генерування енергії СБ та вітрових турбін буде використовуватися накопичена енергія акумулятора, що працює в режимі заряду та розряду.

Якщо акумулятор заряджається, тоді енергія E_B в батареї в момент часу t дорівнює:

$$E_B(t) = E_B(t-1) + (E_{Gen}(t) - E_L(t)) \quad (3.9)$$

де $E_{Gen}(t)$ –значення згенерованої електроенергії СБ та ВЕУ щогодини у час t, кВт×год; $E_L(t)$ – необхідна споживачу електроенергія за час t, кВт×год; $E_B(t-1)$ – кількість електроенергії в акумуляторі за годину до часу t, кВт×год.

Кількість енергії акумуляторної батареї E_B за час t, якщо батарея розряджається, визначається за наступною формулою:

$$E_B(t) = E_B(t-1) + (E_L(t) - E_{Gen}(t)) \quad (3.10)$$

Якщо згенерована потужність перевищує попит споживача, надлишок енергії передаватиметься на акумулятор, поки кількість енергії в батареї не досягне максимального значення $E_{B\,max}$.

Якщо генерується надмірна кількість енергії, і акумулятор повністю заряджений, надлишок $EPG(t)$ буде передано на зовнішнє джерело живлення. Годинне значення $EPG(t)$ визначається за наступною формулою:

$$EPG(t) = E_{Gen}(t) - (E_L(t) + \frac{E_{B\,max} - E_{Gen}(t-1)}{\eta_B} \quad (3.11)$$

де $E_{B\,max}$ – максимальне значення кількості енергії в акумуляторі, кВт×год.

3.4.2 Модель споживання електроенергії

Наступним кроком є визначення кількості споживаної електричної енергії. Згідно з дослідженням, процес споживання електричної енергії має циклічний характер і залежить від багатьох внутрішніх факторів (наприклад, календар вихідних та святкових днів) та зовнішніх факторів (кліматичних характеристик тощо). На

рисунку 3.7 показано незбалансований щоденний план споживання електроенергії, який можна чітко розділити на три періоди часу: ранок, вечір та фон [12].

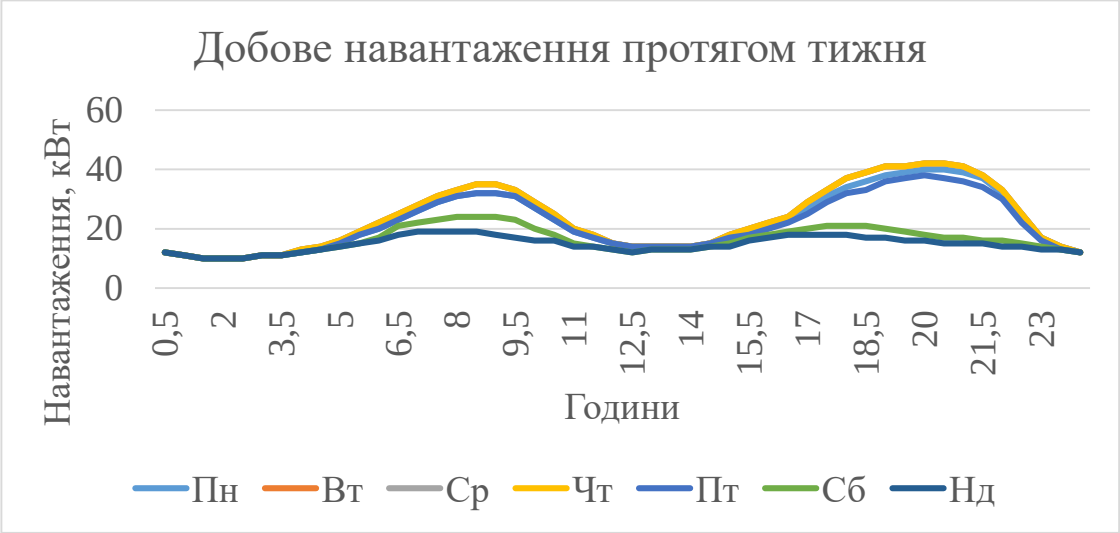

Рисунок 3.7 – Приклад графіка споживання електроенергії протягом доби

Оскільки отриманий графік демонструє чіткі піки, рекомендується виразити прогнозну функцію пікового споживання електроенергії як суму функцій Гаусових кривих, а прогнозну функцію фонового споживання виразити як пряму (рис. 3.8).

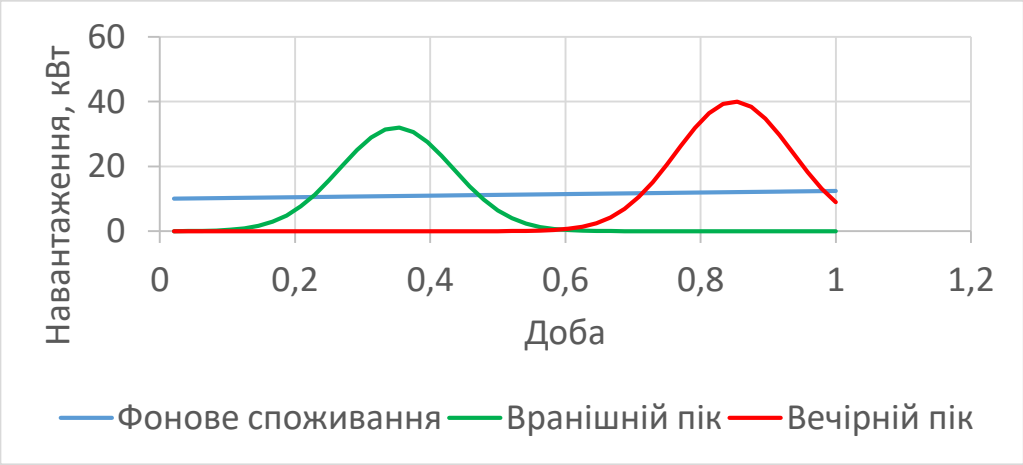

Рисунок 3.8 – Прогнозна функція споживання для доби

Процес споживання електричної енергії має певний ступінь невизначеності, тому миттєве споживання електроенергії будується у нечіткій формі і апроксимується нечітким трикутним числом $W(t) =< W^{mod}(t), W^{min}(t), W^{max}(t) >$. Отже, добовий цикл споживання електричної енергії описується функцією наступного виду:

$$\begin{aligned} &W(t) =< W^{\mathrm{mod}}(t), W^{\mathrm{min}}(t), W^{\mathrm{max}}(t) >, \\ &W^{\mathrm{mod}}(t) = W^{\mathrm{mod}}_{фон}(t) + W^{\mathrm{mod}}_{1пік}(t) + W^{\mathrm{mod}}_{2пік}(t), \\ &W^{\mathrm{min}}(t) = W^{\mathrm{min}}_{фон}(t) + W^{\mathrm{min}}_{1пік}(t) + W^{\mathrm{min}}_{2пік}(t), \\ &W^{\mathrm{max}}(t) = W^{\mathrm{max}}_{фон}(t) + W^{\mathrm{max}}_{1пік}(t) + W^{\mathrm{max}}_{2пік}(t). \end{aligned} \tag{3.12}$$

$$\begin{aligned} &W^{\mathrm{mod}}_{фон}(t) = a_1 t + a_2; W^{\mathrm{min}}_{фон}(t) = b_1 t + b_2; W^{\mathrm{max}}_{фон}(t) = c_1 t + c_2; \\ &W^{\mathrm{mod}}_{1пік}(t) = \mathrm{a}_3 \cdot \mathrm{e}xp(-(\mathrm{t} - \mathrm{a}_4)^2 / \mathrm{a}_5); W^{\mathrm{min}}_{1пік}(t) = \mathrm{b}_3 \cdot \mathrm{e}xp(-(\mathrm{t} - \mathrm{b}_4)^2 / \mathrm{b}_5); \\ &W^{\mathrm{max}}_{1пік}(t) = \mathrm{c}_3 \cdot \mathrm{e}xp(-(\mathrm{t} - \mathrm{c}_4)^2 / \mathrm{c}_5); \\ &W^{\mathrm{mod}}_{2пік}(t) = \mathrm{a}_6 \cdot \mathrm{e}xp(-(\mathrm{t} - \mathrm{a}_7)^2 / \mathrm{a}_8); W^{\mathrm{min}}_{2пік}(t) = \mathrm{b}_6 \cdot \mathrm{e}xp(-(\mathrm{t} - \mathrm{b}_7)^2 / \mathrm{b}_8); \\ &W^{\mathrm{max}}_{2пік}(t) = \mathrm{c}_6 \cdot \mathrm{e}xp(-(\mathrm{t} - \mathrm{c}_7)^2 / \mathrm{c}_8). \end{aligned} \tag{3.13}$$

де t – поточний час; a1-a8, b1-b8, c1-c8 – коефіцієнти. Отже, функція добового споживання електричної енергії набуває вигляду, що представлено на рис. 3.9 [12].

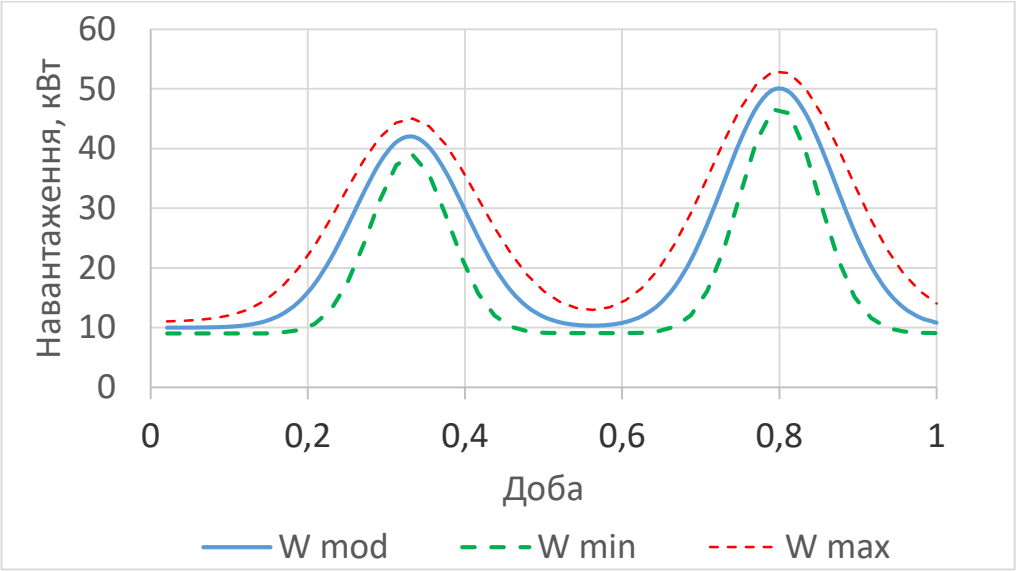

Рисунок 3.9 – Функціональна залежність прогнозної моделі електроспоживання

Коефіцієнти a1-a8, b1-b8 та c1-c8 містять інформацію про річний та тижневий цикли. Оскільки сезонні зовнішні коливання є нелінійними, оптимально представити модель як поліном:

$$\begin{aligned} a_i &= a_{i1} d_n^2 + a_{i2} d_n + a_{i3}, \\ b_i &= b_{i1} d_n^2 + b_{i2} d_n + b_{i3}, \\ c_i &= c_{i1} d_n^2 + c_{i2} d_n + c_{i3}, \\ i &= \overline{1,8}, \end{aligned} \tag{3.14}$$

де d_n – номер дня в тижні. У свою чергу отриманий коефіцієнт залежать від номера тижня в році в такому виді, де n – номер тижня в році:

$$
\begin{aligned}
a_{ij} &= a_{ij1}n^2 + a_{ij2}n + a_{ij3}, \\
b_{ij} &= b_{ij1}n^2 + b_{ij2}n + b_{ij3}, \\
c_{ij} &= c_{ij1}n^2 + c_{ij2}n + c_{ij3}, \\
i &= \overline{1,8},\ j = \overline{1,3},
\end{aligned}
\tag{3.15}
$$

Отже, за допомогою запропонованих моделей ми можемо отримати прогнозовану залежність споживання дня від будь-якого дня року, що враховує добове, тижневе та річне споживання енергії [12].

3.4.3 Модель операційної логіки

Відповідно до логіки роботи гібридної енергосистеми, на першому етапі необхідно визначити потужність вітряної установки, сонячних батарей та рівень заряду акумуляторної батареї, що описано в математичній моделі, наведеній у параграфі 3.4. Далі виконується прогнозування споживання електроенергії протягом певного періоду. За даними сайту прогнозу погоди, було вирішено взяти інтервал часу, рівний кожним 3 годинам у день.

Далі визначаємо загальну кількість споживаної та генерованої енергії на основі погодних даних та технічних факторів. Поточна потужність E_{LG}та потужність зовнішньої мережі E_{GRID} можуть бути виражені як [22]:

$$E_{GEN} = E_P + E_w \tag{3.16}$$

$$E_{LG} = E_{LOAD} - E_{GEN} \tag{3.17}$$

$$E_{GRID} = E_{LG} - E_B, \tag{3.18}$$

де E_{GEN} – загальна потужність всіх АДЕ, кВт×год; E_p – потужність СБ, кВт×год; E_w – потужність ВЕУ, кВт×год; E_B – рівень заряду АБ, кВт×год; E_{LOAD} – потужність навантаження на електросистему, кВт×год.

Потужність акумулятора безпосередньо залежить від його заряду SOC, який повинен бути в будь-який час між максимальним та мінімальним значеннями, відповідно SOC_{MIN} та SOC_{MAX}, щоб продовжити термін служби акумулятора, а саме:

$$SOC_{MIN} \leq SOC(n) \leq SOC_{MAX}, \quad (3.19)$$

$$SOC_{MIN} = (1 - DOD) * SOC_{MAX}, \quad (3.20)$$

де DOD – глибина розряду батареї. У цьому дослідженні вважається, що максимальна глибина розряду становить 50% від загальної потужності батареї, щоб продовжити термін служби таких акумуляторних батарей.

Потім на основі отриманих даних за обраним методом визначається ефективний режим роботи відповідно до метеорологічних факторів, тобто визначається тип або комбінацію підключених АДЕ. Якщо виробленої енергії недостатньо, виконується підключення акумуляторної батареї. В іншому випадку надмірна потужність буде спрямована на акумулятор або зовнішнє джерело живлення.

3.4.4 Модель вибору ефективного режиму функціонування електросистеми

Дерево рішень. На першому етапі було визначено потужності сонячних батарей Es, вітряних установок Ew, а також визначено рівень заряду акумуляторної батареї на період часу t. Далі визначаєється рівень споживання електроенергії та загальний запит від домогосподарства на кількість електроенергії RH. Наступним етапом необхідно визначити джерело живлення, при умові генерації недостатньої кількості енергії, виконується підключення акумулятора або зовнішньої мережі. В

іншому випадку надлишок згенерованої електроенергії буде спрямовано на акумулятор або зовнішню мережу.

Для вибору джерела електроенергії або їх композиції визначимо перелік чітких правил, що є основою алгоритму роботи гібридної енергосистеми (рис.3.10).

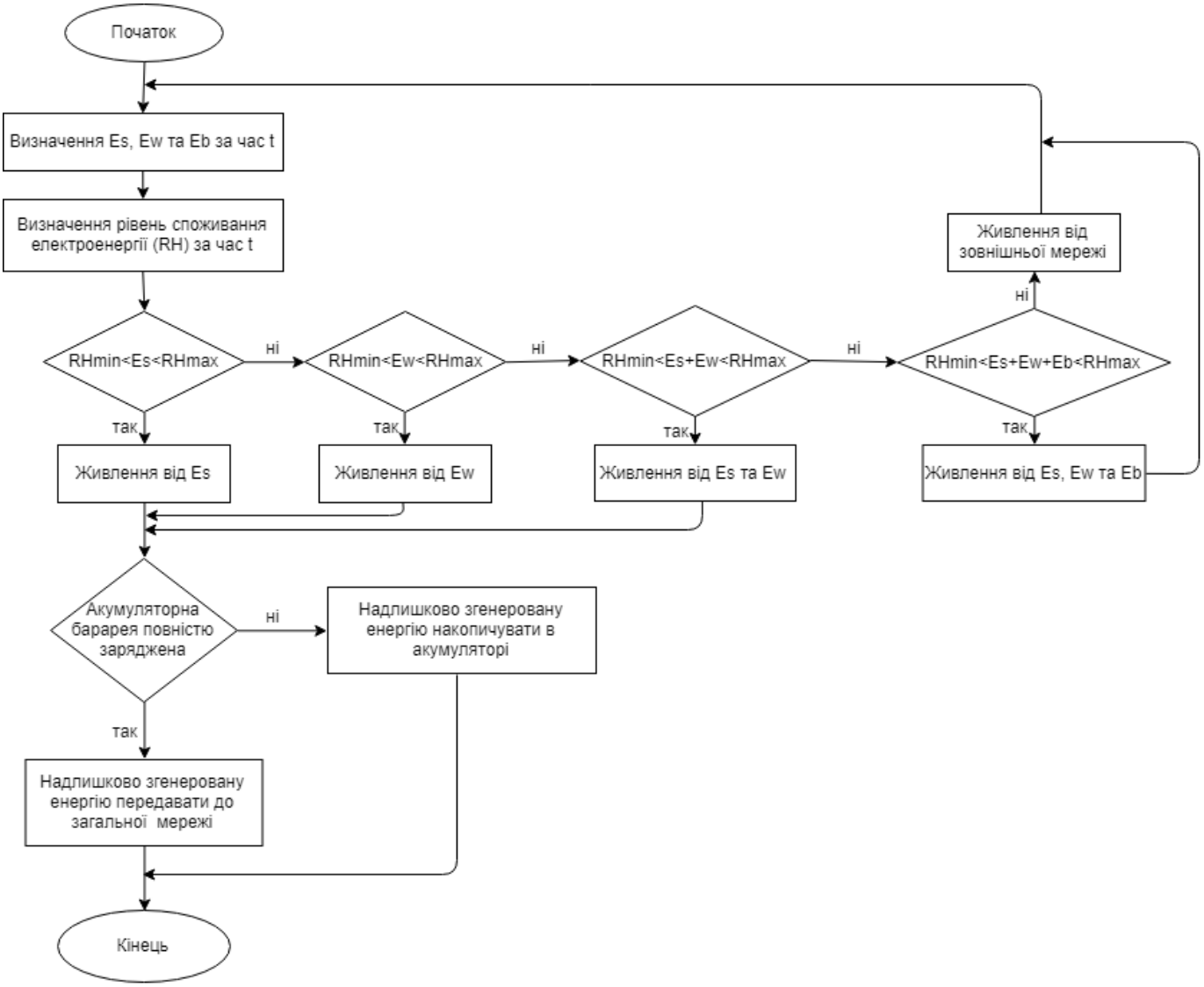

Рисунок 3.10 – Алгоритм вибору джерела електроенергії

Кластеризація. Наступним розглянемо метод кластеризації для вибору режиму функціонування електросистеми. Для обробки отриманих прогнозних даних

необхідно представити нечітке трикутне число споживання електричної енергії у вигляді кластеру, для якого необхідним є знаходження його центру, що представляє собою середнє геометричне місце точок у просторі змінних:

$$Ws_i(t) = \sqrt[n]{\left|\prod_{i=0}^{n} W_i\,(t)\right|} \tag{3.21}$$

де n – кількість елементів кластеру; W_i – кількість споживаної електричної енергії за період часу t, кВт×год; t – період часу, год.

Для перевірки даного методу було проведено його тестування на найбільш характерних днів протягом року, а саме, дні зимового та літнього сонцестояння, весняного та зимового рівнодення. Обчислення виконувались кожні три години протягом доби при відповідних погодних умовах для різних рівнів заряду акумуляторної батареї (Додаток Б). Розглянемо можливі зміни режиму роботи енергосистеми в найбільш типові дні кожного сезону та визначимо ефективність цього методу вибору кращого джерела АДЕ за певних метеорологічних та технічних умов. Вибір ефективного режиму здійснюється методом кластеризації, який включає знаходження найменшої евклідової відстані, від центру кластеру рівня споживання електроенергії до певного джерела АДЕ або їх комбінації за показниками рівня генерації електроенергії.

Розглянемо найбільш типові можливі варіанти роботи, щоб продемонструвати характерність генерації енергії протягом певного періоду року. Рисунок 3.11 демонструє рівні енергії різних комбінацій АДЕ о 8 ранку 21го березня 2020.

Рівень заряду акумулятора визначається показником відповідного датчика, а споживання та генерація електроенергії розраховуються відповідно до моделей, запропонованих в розділі 3, а дані про погоду на певний час збираються з метеорологічного сайту. Наприклад, якщо заряд акумулятора становить 90%, споживання енергії – 11,29 кВт×год, то при заданих умовах найближчим джерелом

живлення є В. Енергія, вироблена вітрогенератором, повністю покриває споживання енергії, а надлишок електроенергії знаходиться в межах допустимого діапазону значень, не викликаючи втрат енергії, забезпечуючи тим самим ефективне функціонування енергетичної системи.

Рисунок 3.11 – Приклад вибору джерела електроенергії

Рівень заряду акумулятора менше 50% буде неефективним через провокування сильного зносу і скорочення терміну служби батареї, а також призведе до недостатнього живлення в системі, тому рекомендується не допускати такий рівень заряду акумулятора. Такі випадки розраховуються, але як можливий режим роботи не враховується. Рекомендується працювати на рівні заряду більше 50%, і заряджати його у разі надмірної генерації електроенергії, що забезпечує метод кластеризції.

Модель нечітких правил. Останнім з методів прийняття управлінських рішень для порівняння є метод нечітких правил. Даний метод має перевагу серед інших в

тому, що враховує фактор прогнозу погоди нечіткого характеру. Визначимо функцію належності для критерію « Джерело електроенергії». Приклад правила нечіткого виведення представлено в таблиці 3.3, що використовується для визначення джерела живлення.

Передумови правил виражаються у формі предикатів, де змінні «Хмарний покрив» (OY), «Швидкість вітру» (WY), «Заряд АБ» (AY), «Потужність СБ» (SY) та «Потужність ВЕУ» (BY) є частиною передумови правила.

$$Z = \{Z1, Z2, Z3, Z4\} \quad (3.22)$$

$$Z = F(OY, WY, AY, BY, SY) \quad (3.23)$$

Таблиця 3.3 – Приклад нечіткого правила

Передумова	OY is OY1, WY is WY1, AY is AY1
Формулювання правила	If OY is OY1 and WY is WY1 then If AY is AY1 then Z=Z1, else Z=Z4.
Інтерпретація	Якщо «Хмарний покрив»=OY1 і «Швидкість вітру»=WY1, то Якщо «Заряд АБ»=AY1, То Z=Z1, Інакше Z=Z4.

$$OY = OY1, OY2, OY3, OY4 \quad (3.24)$$

$$WY = WY1, WY2, WY3 \quad (3.25)$$

$$AY = AY1, AY2 \quad (3.26)$$

$$BY = BY1, BY2 \quad (3.27)$$

$$SY = SY1, SY2 \quad (3.28)$$

Лінгвістична змінна OY "Хмарний покрив" визначена в універсальному наборі U (OY) = [0,1]. Множина термів лінгвістичної змінної T (OY) = <ясно, слабка хмарність, хмарність, сильна хмарність> включає в себе терми лінгвістичних змінних OY1, OY2, OY3, OY4.

Лінгвістична змінна WY «Швидкість вітру» визначена в універсальному наборі U(WY) = [0,30] (м/с). Множина термів лінгвістичної змінної T (WY) = <низька, середня, висока> включає в себе терми лінгвістичних змінних WY1, WY2, WY3.

Лінгвістична змінна AY «Заряд АБ» визначена в універсальному наборі U(AY) = [0,10] (кВт). Множина термів лінгвістичної змінної T (AY) = <достатня, недостатня> включає в себе терми лінгвістичних змінних AY1, AY2.

Лінгвістична змінна BY «Потужність ВЕУ» визначена в універсальному наборі U(BY) = [0,10] (кВт). Множина термів лінгвістичної змінної T (BY) = <достатня, недостатня> включає в себе терми лінгвістичних змінних BY1, BY2.

Лінгвістична змінна SY «Потужність СБ» визначена в універсальному наборі U(SY) = [0,10] (кВт). Множина термів лінгвістичної змінної T (SY) = <достатня, недостатня> включає в себе терми лінгвістичних змінних SY1, SY2.

Лінгвістична змінна Z «Джерело електроенергії» визначена в універсальному наборі U(Z) = [0,1]. Множина термів лінгвістичної змінної T(Z) = <АБ, ВЕУ, СБ, Зовнішня мережа> включає в себе терми лінгвістичних змінних Z1, Z2, Z3, Z4.

Отже, далі наводиться повний перелік нечітких правил, що використовуються в моделі прогнозування роботи гібридної енергосистеми:

1. Якщо Хмарний покрив є «сильна хмарність» і Швидкість вітру є «низька», то Якщо Заряд АБ є «достатній», то АБ підключено, Інакше Зовнішня мережа підключена;

2. Якщо Хмарний покрив є «сильна хмарність» і Швидкість вітру є «середня», то Якщо Потужність ВЕУ є «достатня», то ВЕУ підключено, Інакше Якщо Заряд АБ є «достатній», то АБ підключено, Інакше Зовнішня мережа підключена;

3. Якщо Хмарний покрив є «сильна хмарність» і Швидкість вітру є «висока», то Якщо Потужність ВЕУ є «достатня», то ВЕУ підключено, Інакше Якщо Заряд АБ є «достатній», то АБ підключено, Інакше Зовнішня мережа підключена;

4. Якщо Хмарний покрив є «хмарно» і Швидкість вітру є «низька», то

5. Якщо Заряд АБ є «достатній», то АБ підключено, Інакше Зовнішня мережа підключена;

6. Якщо Хмарний покрив є «хмарно» і Швидкість вітру є «середня», то

7. Якщо Потужність ВЕУ є «достатня» і Потужність СБ є «достатня», то ВЕУ підключено і СБ підключено, Інакше Якщо Заряд АБ є «достатній», то АБ підключено, Інакше Зовнішня мережа підключена;

8. Якщо Хмарний покрив є «хмарно» і Швидкість вітру є «сильна», то

9. Якщо Потужність ВЕУ є «достатня» і Потужність СБ є «достатня», то ВЕУ підключено і СБ підключено, Інакше Якщо Заряд АБ є «достатній», то АБ підключено, Інакше Зовнішня мережа підключена;

10. Якщо Хмарний покрив є «слабка хмарність» і Швидкість вітру є «низька», то Якщо Потужність ВЕУ є «достатня» і Потужність СБ є «достатня», то ВЕУ підключено і СБ підключено, Інакше Якщо Заряд АБ є «достатня», то АБ підключено, Інакше Зовнішня мережа підключена;

11. Якщо Хмарний покрив є «слабка хмарність» і Швидкість вітру є «середня», то Якщо Потужність ВЕУ є «достатня» і Потужність СБ є «достатня», то ВЕУ підключено і СБ підключено, Інакше Якщо Заряд АБ є «достатній», то АБ підключено, Інакше Зовнішня мережа підключена;

12. Якщо Хмарний покрив є «слабка хмарність» і Швидкість вітру є «висока», то Якщо Потужність ВЕУ є «достатня» і Потужність СБ є «достатня», то ВЕУ підключено і СБ підключено, Інакше Якщо Заряд АБ є «достатня», то АБ підключено, Інакше Зовнішня мережа підключена;

13. Якщо Хмарний покрив є «ясно» і Швидкість вітру є «низька», то Якщо Потужність ВЕУ є «достатня» і Потужність СБ є «достатня», то ВЕУ підключено і

СБ підключено, Інакше Якщо Заряд АБ є «достатня», то АБ підключено, Інакше Зовнішня мережа підключена;

14. Якщо Хмарний покрив є «ясно» і Швидкість вітру є «середня», то Якщо Потужність ВЕУ є «достатня» і Потужність СБ є «достатня», то ВЕУ підключено і СБ підключено, Інакше Якщо Заряд АБ є «достатня», то АБ підключено, Інакше Зовнішня мережа підключена;

15. Якщо Хмарний покрив є «ясно» і Швидкість вітру є «висока», то Якщо Потужність ВЕУ є «достатня» і Потужність СБ є «достатня», то ВЕУ підключено і СБ підключено, Інакше Якщо Заряд АБ є «достатній», то АБ підключено.

3.4.5 Аналіз методів вибору ефективного режиму функціонування електросистеми

Аналіз методів визначення впливу постійно змінних погодних умов на продуктивність альтернативних джерел енергії зводиться до задачі вибору кращого методу прийняття рішень відповідно до визначених кількісних та якісних критеріїв.

Метод дерева рішень є найбільш популярним серед інших при використанні в автоматизованих системах управління, тому що є високо надійним та точним. Даний метод використовує чіткі правила та дискретні дані для класифікації джерела електроенергії та не враховує фактори нечіткості, що несуть в собі прогнозні метеорологічні дані.

Як альтернативний підхід до визначення джерела альтернативної енергії пропонується розглянути метод кластеризації. В просторі даних, що складається з даних визначеного рівня генерації електроенергії кожного з АДЕ, даних зпрогнозованого рівня споживання електроенергії на визначений час та рівня заряду акумуляторної батареї, знаходимо ефективне джерело енергії як коротшу відстань до центру кластеру, що представляє собою рівень споживання. Даний підхід є більш

гнучким, ніж дерево рішень, адже не накладає певних обмежень на типи даних, що розглядаються.

Останнім з розглядаємих є метод нечітких правил. Даний метод краще за інших демонструє гнучкість до невизначеності метеорологічних даних, оскільки більш точно враховує не тількі кількісні, але і якісні дані.

По-перше, необхідно порівняти дані методи за складністю алгориму. Складність алгоритму зазвичай оцінюється за часом виконання або використаною пам'яттю. В обох випадках складність залежить від розміру вхідних даних: масив, що містить 100 елементів, буде швидшим, ніж обробка подібних 1000 елементів. Менш важливим є точний час, бо час залежить від процесора, типу даних, мови програмування та багатьох інших параметрів. Важливою є лише асимптотична складність, тобто складність, коли розмір вхідних даних близиться до нескінченості [27].

При однаковоій кількості виконання розрахунків складність методу дерева рішень – O(log(n)), такий порядок зростання означає, що час виконання алгоритму збільшується логарифмічно із збільшенням розміру вхідного масиву, методу кластеризації – O(n2), а методу нечітких правил – O(n3) [9].

Наступним етапом необхідно порівняти дані методи за швидкістю виконання розрахунків, для чого було використано бібліотеку datetime мови програмування Python. Програмний код, який було застосовано для обислення часу виконання кожної функції представлено на рис. 3.12. Результати дослідження представлено в таблиці 3.4. У зв'язку з тим, що обислення виконувалось лише для однієї ітерації, то отримані дані часу є досить малими, близькими до 0.

```
start_time = datetime.datetime.now()
end_time = datetime.datetime.now()
print('%.15f' %(end_time - start_time).seconds)
```

Рисунок 3.12 – Програмний код обчислення часу виконання функції

Ємнісна складність алгоритму характеризує кількість одиниць пам’яті, необхідних для виконання даного алгоритму, а статична складність алгоритму представляє собою довжину опису алгоритмів [31]. Для визначення об'єму пам'яті, що використовується для виконання кожного методу програмно було застосовано наступний фрагмент коду (рис.3.13). Результати дослідження ємнісноїскладності у байтах та статичної складності алгоритмів представлено в таблиці 3.4.

```
process = psutil.Process(os.getpid())
print(process.memory_info().rss)  # in bytes
```

Рисунок 3.13 – Програмний код обчислення об’єму пам’яті

Таблиця 3.4 – Порівняння методів

Порівняльні характеристики	Дерево рішень	Кластеризація	Нечіткі правила
Алгоритмічна складність	O(log(n))	O(n2)	O(n3)
Ємнісна складність (Б)	122449920	122474496	122494976
Статична складність	23	88	133
Час виконання алгоритму (с)	0.000000000000186	0.000000000000289	0.000000000000489

До якісних критеріїв можна віднести точність, структурованість, корисність та надійність. Оцінка методів відповідно до даних критеріїв виконується експертом-розробником. Результат порівняння методів представлено на рис. 3.14. Очевидно, що найбільш структурованим є метод дерева рішень, що працює з дискретними даними.

Та попри це, метод нечітких правил можна вважати кращим, оскільки критерій корисності та надійності є більш важливими, аніж структурованість. Більш того саме метод нечітких правил враховує фактори невизначеності метеорологічних даних та справедливо може вважатися кращим, не дивлячись на алгоритмісну складність даного методу.

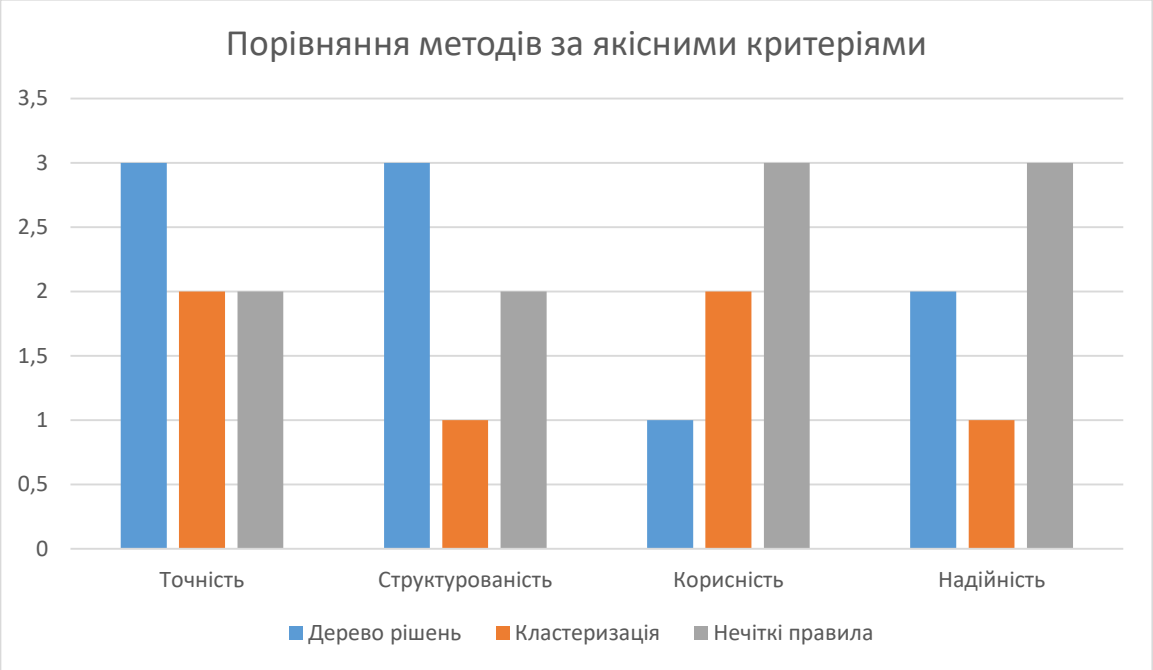

Рисунок 3.14 – Порівняння методів за якісними критеріями

4 РОЗРОБКА ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

Визначений в попередньому розділі кращий метод має практичну реалізацію у вигляді програмного продукту, де вся логіка обробки операцій здійснюється згідно з отриманими моделями.

Архітектуру програмної реалізації зображено на рис.4.1. Оскільки дані оброблюються з зовнішніх джерел, програмне забезпечення збирає дані прогнозу погоди та дані про інсоляцію за поточний рік у вигляді електронної таблиці Excel.

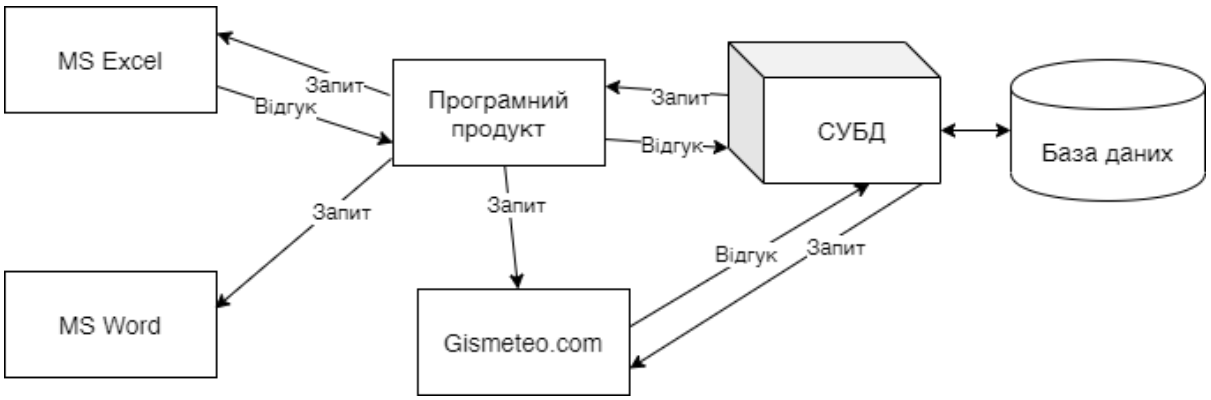

Рисунок 4.1 – Архітектура програмної реалізації

Програмний продукт розроблений за допомогою мови програмування Python. Інструмент QT Designer використовується для реалізації діалогових вікон із відповідними функціями.

Перш ніж розробляти додаток з використанням мови програмування Python, потрібно розробити прототип майбутнього діалогового вікна. Прототип повинен містити всі необхідні блоки, що відповідають функціональним вимогам додатку.

Прототип представлено на рисунку 4.2, що дає наближене уявлення про розташування основних блоків.

Рисунок 4.2 – Прототип програмної реалізації

Функціональна панель повинна дозволяти користувачеві зберігати результати у вигляді звіту на запит користувача,за що відповідає кнопка «Звіт». Крім того, одним із завдань є надання користувачам доступу до сайту прогнозу погоди, у нашому випадку Gismeteo.com. Необхідно натиснути кнопку "прогноз погоди", щоб виконати цю функцію.

Далі, за умовами програми, потрібно визначити місцезнаходження користувача для подальшого визначення метеоумов. Ці функції виконує блок "Пошук міста", що складається з рядку введення та функціонльної кнопки.

Наступним кроком виконуємо деталізацію блоку «Технічні характеристики». Дані для заповнення даного блоку визначаються за технічними паспортами

складових енергосистеми та датчиків заряду акумуляторної батареї, а також використовуються для розрахунків рівня генерації електроенергії та прогнозування рівня споживання.

Блоки «Результат» демонструють визначене джерело живлення, рівень заряду акумулятора, індикатор режиму заряду-розряду акумулятора, надлишок генерованої електроенергії, а також нагадування про наступне виконання обчислень. На рисунку 4.3 представлено загальний вигляд інтерфейсу програмної реалізації.

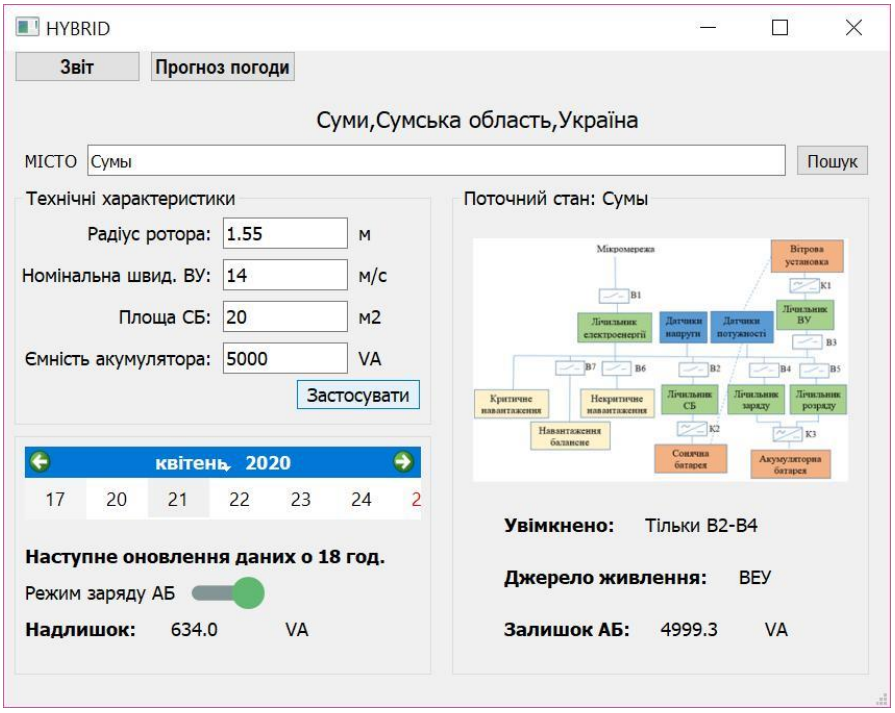

Рисунок 4.3 – Інтерфейс програмної реалізації

4.1 Розробка бази даних

Відповідно до операційної логіки програмної реалізації, отримані результати розрахунків та прогнозу споживання електроенергії необхідно зберігати в базі даних.

База даних «hybrid» зберігає в собі метеорологічні дані, дані місцезнаходження гібридної енергосистеми та дані розрахунків потужності ВЕУ, СБ, визначене джерело електроенергії, а також зпрогнозований рівень споживання. На рисунку 4.4 зображено структуру бази даних.

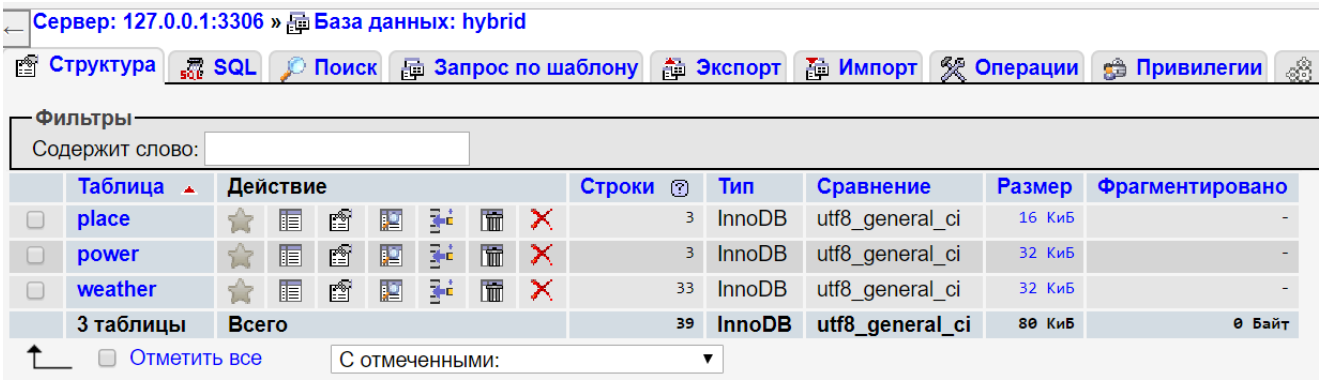

Рисунок 4.4 – База даних «hybrid»

У таблиці «place» (рис. 4.5) зберегаються назви міст, регіонів та країн, а також ідентифікаційний номер обраного міста на веб-сайті Gismeteo.com. Ці дані є необхідними для парсінгу метеорологічних даних.

place_id	city_name	country	number	region
1	Сумы	Украина	4936	Сумская область
2	Киев	Украина	4944	Киев
3	Полтава	Украина	4957	Полтавская область

Рисунок 4.5 – Склад таблиці «place»

Дані про погоду з сайту зберігаються в таблиці «weather» (рис. 4.6). Хмарний покрив, показники вітру та температура повітря необхідні для розрахунку

потужностей складових гібридної енергосистеми. Потім отримані результати розрахунків записуються у таблицю «power» (рисунок 4.7).

weather_id	w_speed	w_direction	temperature	overcast	dat_e	tim_e	place_id
1	2	ЮВ	3	Пасмурно	2020-11-28	18:00:00	1
2	2	ЮВ	3	Пасмурно	2020-11-28	18:00:00	1
3	1	Ю	3	Пасмурно	2020-11-28	21:00:00	1
4	1	Ю	3	Пасмурно	2020-11-28	21:00:00	1

Рисунок 4.6 – Склад таблиці «weather»

Параметры

power_id	Ps	Pw	Pb	type_P	consumption	overflow	weather_id
1	55.195914779999995	9.599873406383796	4638.683639982538	Es+Ew+Eb	426.11214820384595	0	4
2	32.283	9.666094305235895	4838.7791676070165	Es+Ew+Eb	203.1699266982193	0	32
3	32.283	9.666094305235895	4838.7791676070165	Es+Ew+Eb	203.1699266982193	0	32

Рисунок 4.7 – Склад таблиці «power»

Для виконання підключення бази даних до програмного додатку, а також подальшої роботи з нею, було створено програмний модуль myconnutils.py. Фрагмент коду, який виконує підключення до бази даних, зображено на рис. 4.8.

```
#the function returns connection
def get_connection():
    connection = pymysql.connect(host='127.0.0.1',
                                 user='mysql',
                                 password='mysql',
                                 db='hybrid',
                                 charset='utf8mb4',
                                 cursorclass= pymysql.cursors.DictCursor)
    return connection
```

Рисунок 4.8 – Функція підключення до бази даних

Даний модуль роботи з базою даних також має функції, що виконують запити додавання інформації до кожної таблиці бази даних, а також окремі функції виконання запитів зі змінними (рис 4.9 – 4.10).

```
def insert_city(connection, place):
    # Prepare SQL query to INSERT a record into the database.
    id = str(place['id'])
    sql_check =("SELECT number FROM place")
    cursor = execute_select(connection, sql_check)
    result = cursor.fetchall()
    n=0
    print(place)
    print(result)
    for x in range(len(result)):
        i=result[x]
        ID = i['number']
        if id == str(ID):
            n+=1
    if n==0 :
        sql = """INSERT INTO place (city_name, country, number, region)
                        VALUES ('%s', '%s', '%s', '%s');"""
        data = (place['city'], place['country'], place['id'], place['district'])
        execute_query(connection, sql, data)
```

Рисунок 4.9 – Функція додавання до бази даних

```
def execute_query(connection, sql, values):
    cursor = connection.cursor()
    try:
       # Execute the SQL command
       cursor.execute(sql % values)
       # Commit changes in the database
       connection.commit()
    except:
        # Rollback in case there is any error
        connection.rollback()
```

Рисунок 4.10 – Функція виконання запиту

4.2 Реалізація парсера

Одним з важливих етапів роботи програмного додатку є збір даних з метеорологічного сайту Gismeteo.com. Відповідно до логіки обробки даних, погодні дані збираються на основі введеної назви міста. Для визначення потужності ВЕУ, потрібно знати напрямок та швидкість вітру, для СБ – рівень хмарного покриву і температуру повітря. Тому було вирішено створити програмний модуль gparser.py, який має всі необхідні функції для парсінгу даних.

Функція extract_datetimes() використовується для збору даних про дату та час прогнозу погоди. Дані записуються до бази даних для подальшого використання. Програмний код цієї функції показано на рисунку 4.11.

```
def extract_datetimes(row):
    dates = []
    for el in row.xpath('./div/div/@title'):
        dates.append(split_todatetime(el))
    return dates
```

Рисунок 4.11 – Функція парсінгу дати та часу

Функція extract_temperature() визначає температуру повітря кожні 3 години на день (рис. 4.12). Оскільки дані про температуру повітря зберегаються на сайті в градусах Цельсія і Фаренгейта, необхідно виконати чистку даних для подальшої роботи.

```
def extract_temperature(row):
    temperatures = []
    for el in row.xpath(".//div[@class='value']"):
        temperatures.append(el.xpath('./span/text()'))
    return temperatures
```

Рисунок 4.12 – Функція парсінгу температури повітря

Для визначення рівня генерації енергії від ВЕУ необхідно визначити швидкість і напрямок вітру, що виконується за допомогою функції extract_wind(). На рисунку 4.13 зображено програмний код даної функції.

```
def extract_wind(row):
    winds = []
    for wind in row:
        w = [ el.strip() for el in wind.xpath(".//span/text()")]
        w.append(wind.xpath(".//*[contains(@class, 'gray')]/text()")[0].strip())
        winds.append(w)
    return winds
```

Рисунок 4.13 – Функція парсінгу даних вітру

Основною функцією підключення до сайту та збору даних є req_city_info(). Дана функція збирає дані щодо введеної назви міста, а саме ідентифікаційний номер, місто, регіон, країну (рис. 4.14). Для спрощення роботи з отриманими даними, представляємо їх у вигляді списку словників.

```python
def req_city_info(city, lang):
    url = 'https://www.gismeteo.ua/api/v2/search/searchresultforsuggest/'
    params = '/?lang' + lang + "&domain=ua"

    response = requests.get(url + city + params, headers=BASE_HEADERS)

    data = json.loads(response.text)
    if(data['total'] == 0): raise Exception('Error : no such city ' + suggest)

    res = {}
    res['id'] = data['items'][0]['id']
    res['city'] = data['items'][0]['name']
    res['district'] = data['items'][0]['district']['name']
    res['country'] = data['items'][0]['country']['name']
    res['url'] = data['items'][0]['url']

    return res
```

Рисунок 4.14 – Функція парсінгу даних місцезнаходження

4.3 Реалізація головного модулю програми

Пакети програм Tkinter або PyQt зазвичай використовуються для розробки інтерфейсів програмування на Python. Останній є більш зручним для розробки віконних додатків, тому його було обрано одним із інструментів. Використовуючи програму QT Designer було створено діалогове вікно з відповідними функціями, як показано на рисунку 4.15, яке включає поля введення, функціональні кнопки та зображення (завантажені за допомогою бібліотеки QPixmap) і календар як додаткові елементи. вікна.

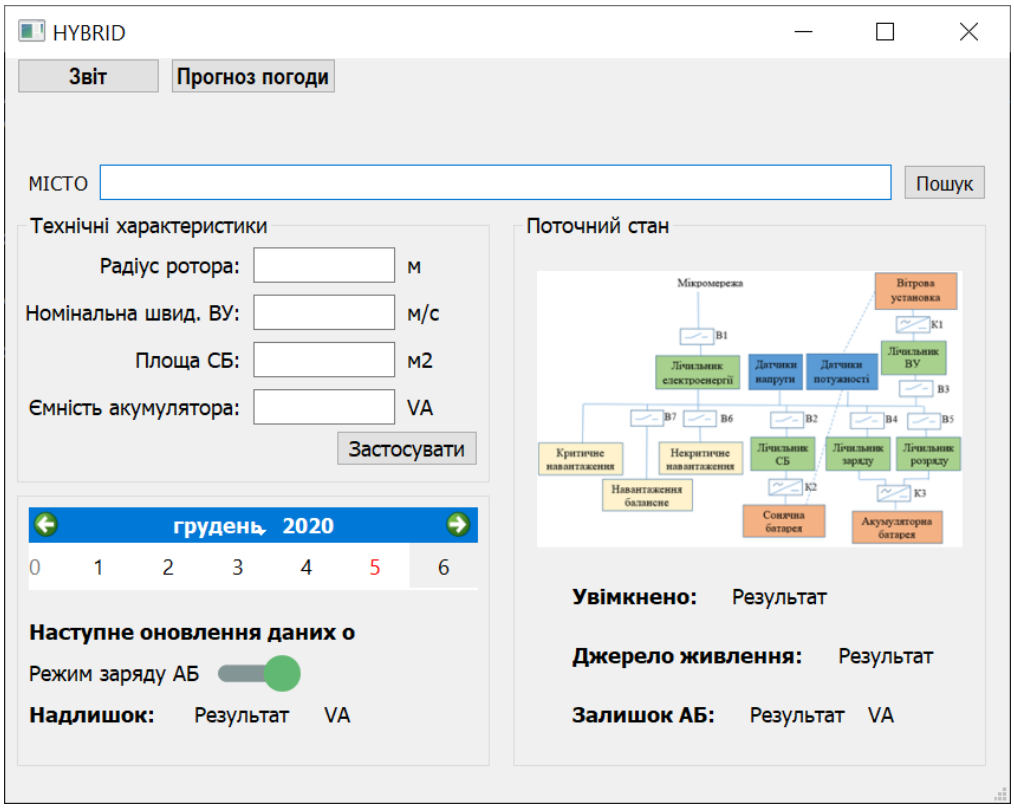

Рисунок 4.15 – Головне вікно програмного додатку

Після розташування елементів у необхідному порядку було зконвертовано головне вікно додатку в програмний файл mydesign.py, що відображає всі елементи вікна та їх характеристики. До головного модулю програми даний файл імпортується разом з іншими необхідними бібліотеками за допомогою команди import.

Спочатку, щоб в подальшому отримати необхідні результати, користувач повинен ввести назву потрібного міста в поле введення, потім натиснути кнопку пошуку. Результат запиту відображається у верхній частині вікна та в блоці «Поточний стан» (рис. 4.16).

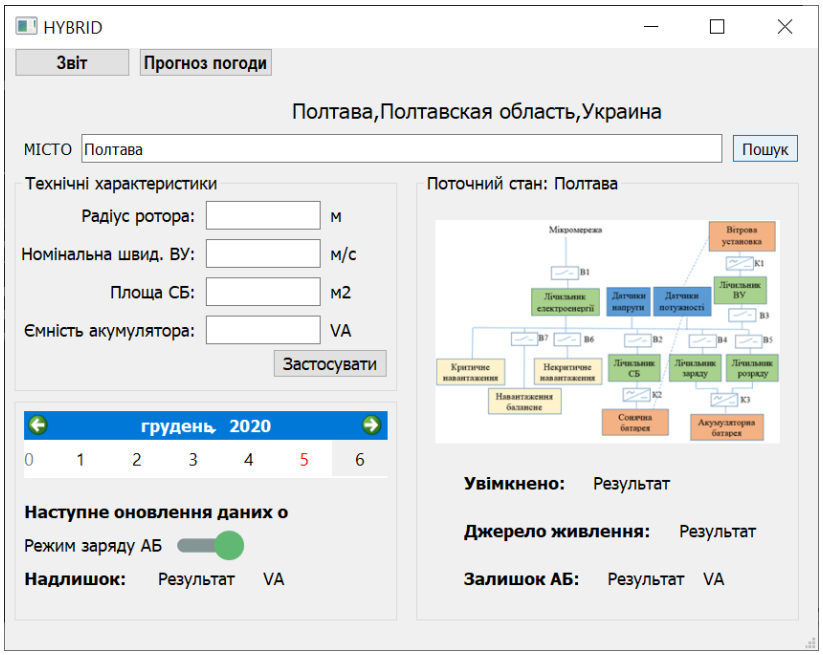

Рисунок 4.16 – Результат пошуку міста

Натискання кнопки пошуку активує функцію searchCity(), код якої показаний на рисунку 4.17. Спочатку використовується функція gparser.req_city_info() для пошуку введеного міста на веб-сайті Gismeteo.com, а потім використовується функція myconnutils.insert_city(), щоб записати дані міста в базу даних (якщо такі ще не існують). Далі, відповідно до даних отриманого міста, використовується функцію myconnutils.insert_weather() для запису отриманих даних про погоду за останню годину в базу даних (рис. 4.18).

```
def searchCity(self):
    res=gparser.req_city_info(self.ui.input_city.text(), lang='ua')

    self.ui.groupBox.setTitle('Поточний стан: ' + res['city'])
    con = myconnutils.get_connection()
    myconnutils.insert_city(con, res)

    sql_check =("SELECT * FROM place WHERE number = %s")
    d = (str(res['id']), )
    cursor = con.cursor()
    cursor.execute(sql_check, d)
    l = cursor.fetchone()
    s =  ','.join([l['city_name'], l['region'], l['country']])
    self.ui.lbl_main_city.setText(s)

    weather_list = gparser.weather(l['city_name'])
    t = int(time_define()//3.0)
    weather = weather_list[t]

    myconnutils.insert_weather(con, weather, l['place_id'])
```

Рисунок 4.17 – Код функції searchCity

weather_id	w_speed	w_direction	temperature	overcast	dat_e	tim_e	place_id
1	2	ЮВ	3	Пасмурно	2020-11-28	18:00:00	1

Рисунок 4.18 – Результат запису погодних даних

Наступним кроком необхідно ввести технічні характеристики ВЕУ та СБ. Наприклад, візьмемо ВЕУ з довжиною лопаті 1,55 і номінальною швидкістю обертання 14 м/с, дані показники визначаються з технічними паспортами виробів. Тоді, наприклад, для невеликого будинку вирішено обрати площину сонячних батарей рівною 20 м^2 та акумулятор ємністю 5000 Вт. Після заповнення необхідних полей користувач натискає кнопку «Застосувати», що викликає функцію applyData (рис. 4.19). Результати розрахунку відображаються справа (визначене джерело живлення, залишок в АБ та перемикачі, що вмикаються) та знизу (згенерований

надлишок електроенергії, індикатор заряду/розряду акумулятора), результат розрахунків представлено на рисунку (4.20).

```
def applyData(self):
    connection = myconnutils.get_connection()
    cursor = connection.cursor()
    r = float(self.ui.input_veu_r.text())
    v = float(self.ui.input_veu_n.text())
    s = float(self.ui.input_sb_s.text())
    e = float(self.ui.input_bt_e.text())

    cit = self.ui.lbl_main_city.text()
    place = cit.split(',')
    city = place[0]
    t= time_define()
    if t<10:
        time = '0'+str(t) + ':00:00'
    else: time = str(t) + ':00:00'
    #data = gparser.weather(city)
    sql ="SELECT w.w_speed, w.temperature, w.overcast, w.dat_e, w.ti
    a = sql+city+"' AND w.tim_e = '"+time+"' "
    print(a)
    cursor.execute(a)
    weth = cursor.fetchone()
    print(weth)

    ins=pd.read_excel('ins.xlsx', index_col = 0)
    lst = list(ins.index)
    lst = [str(s) for s in lst]
    ins.index = lst

    n = int(t/3.0)
    dat = weth['dat_e']
    mounth = dat.split('-')
    m = int(mounth[1])
    hour = weth['tim_e']
    tp = int(weth['temperature'])
    over = weth['overcast']
    speed = float(weth['w_speed'])

    P_w = E_veu(tp, t, r, speed, v )
    P_sb = E_sb(tp, s, float(ins.loc[hour,m]), over, m-1)# power of
    W = W_concumption(A, B, C, dat)#electricity consumprion at curre
    P_b = e
    #print(P_sb, P_w, P_b, W)
    Prior = priority(P_sb, P_w, P_b, W[1], P_b)

    #insert_power(connection, power)
    self.ui.label_veu.setText(str(round(P_w, 2)))
    self.ui.label_sb.setText(str(round(P_sb, 2)))
    self.ui.label_bt.setText(str(round(P_b, 2)))
    self.ui.lbl_source_value.setText(Prior[0])
    self.ui.label_cons.setText(str(round(W[1], 2)))
    self.ui.lbl_overflow_value.setText(str(round(Prior[2])))

    w_id = weth['weather_id']
    power = (P_sb, P_w, Prior[1], Prior[0], W[1],Prior[2], w_id)
    myconnutils.insert_power(connection, power)
```

Рисунок 4.19 – Код функції applyData

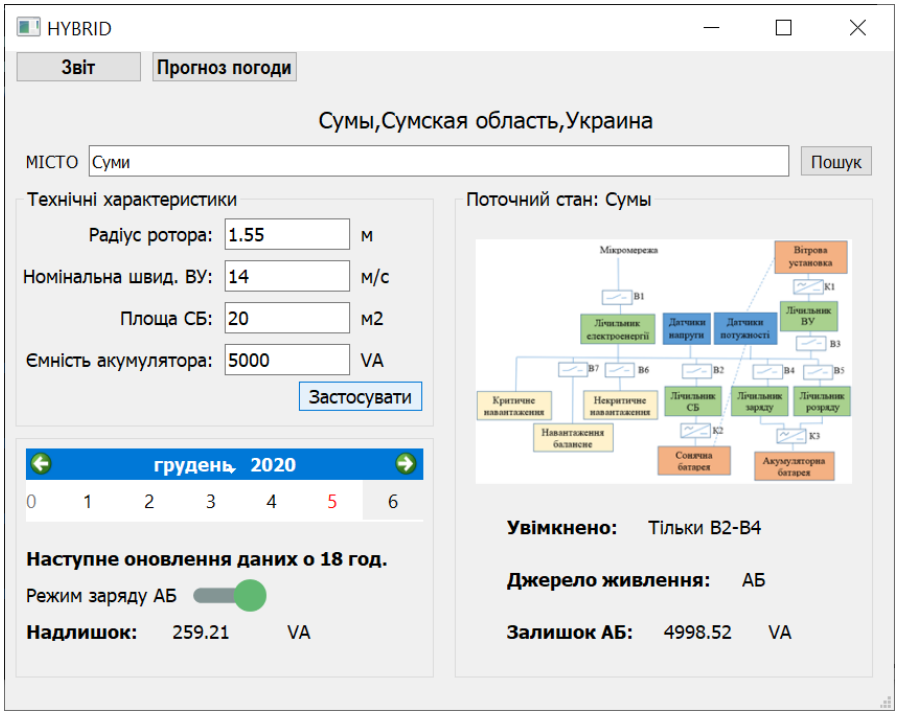

Рисунок 4.20 – Результат виконання розрахунків

Функція E_veu() використовується для визначення потужності вітрогенератора (рис. 4.21), функція E_sb() - для обчислення потужності сонячних батарей (рис. 4.22), а функція W_consumption() - для визначення рівня споживання енергії (рис. 4.23) та Bon(), яка визначає джерело живлення методом нечітких правил (рис. 4.24).

```
def E_veu(tp, t, R, Vw, V):
    Q=0.00001661*tp**2-0.004764*tp+1.2924
    f=0.003869*Vw**2-0.128*Vw+6.650827154
    w=V/R
    Cp=0.351
    E=Q*Cp*Vw*math.pi*R**2
    return E
```

Рисунок 4.21 – Код функції E_veu()

```python
def E_sb(tp, Square, E, overcast, month):
    N1=0.0901 * E + 0.0873 * tp - 0.00032 * E * tp
    P_mod = Square*N1
    N2 = 0.0876*E + 0.0499*tp - 0.00027*E*tp
    P_min = N2*Square
    N3 = 0.0918*E + 0.1055*tp - 0.00035*E*tp
    P_max = N3*Square
    alfa = np.array([[0.6126,   0.8063, 0.0970, 0],
                     [0.6261,   0.8130, 0.0935, 0],
                     [0.5931,   0.7966, 0.1017, 0],
                     [0.6658,   0.8329, 0.0835, 0],
                     [0.7282,   0.8641, 0.0679, 0],
                     [0.7146,   0.8573, 0.0713, 0],
                     [0.7656,   0.8828, 0.0586, 0],
                     [0.7963,   0.8982, 0.0509, 0],
                     [0.7068,   0.8534, 0.0733, 0],
                     [0.6561,   0.8281, 0.0860, 0],
                     [0.5525,   0.7763, 0.1119, 0],
                     [0.5713,   0.7857, 0.1072, 0]])
    if overcast == 'Пасмурно':
        x = P_min + alfa[month, 0]*(P_mod - P_min)
    elif overcast == 'Облачно':
        x = P_min + alfa[month, 1]*(P_mod - P_min)
    elif overcast == 'Малооблачно':
        x = P_max - alfa[month, 2]*(P_max - P_mod)
    elif overcast == 'Ясно':
        x = P_max - alfa[month, 3]*(P_max - P_mod)
    return x
```

Рисунок 4.22 – Код функції E_sb()

```python
W_concumption(A, B, C, data):

#D=data[0]
curr_date = datetime.datetime.strptime(data, '%Y-%m-%d')
n = float(curr_date.isocalendar()[1])
d = float(curr_date.weekday()+1)

#---------------
k = time_define()
t = k/24.0
#---------------
a1=A[1]
a2=A[2]
a3=A[3]
a4=A[4]
a5=A[5]
a6=A[6]
a7=A[7]
a8=A[8]
Wmod_f = (a1[1,0]*d+a1[2,0])*n**2+(a1[1,1]*d+a1[2,1])*n+(a1[1,2]*d+a1[2,2])*t+a2[0,2]*d**2+a2[1,2]*d+a2[2,2]
Wmod_1 = ((a3[0,0]*d**2+a3[1,0]*d+a3[2,0])*n**2+(a3[0,1]*d**2+a3[1,1]*d+a3[2,1])*n+(a3[0,2]*d**2+a3[1,2]*d+a3[2,2]))*math.exp(-((t-a5[2,2])**2)/a4[2,2])
Wmod_2 = ((a6[0,0]*d**2+a6[1,0]*d+a6[2,0])*n**2+(a6[0,1]*d**2+a6[1,1]*d+a6[2,1])*n+(a6[0,2]*d**2+a6[1,2]*d+a6[2,2]))*math.exp(-((t-a8[2,2])**2)/a7[2,2])

b1=B[1]
b2=B[2]
b3=B[3]
b4=B[4]
b5=B[5]
b6=B[6]
b7=B[7]
b8=B[8]
Wmin_f = (b1[1,0]*d+b1[2,0])*n**2+(b1[1,1]*d+b1[2,1])*n+(b1[1,2]*d+b1[2,2])*t+b2[0,2]*d**2+b2[1,2]*d+b2[2,2]
Wmin_1 = ((b3[0,0]*d**2+b3[1,0]*d+b3[2,0])*n**2+(b3[0,1]*d**2+b3[1,1]*d+b3[2,1])*n+(b3[0,2]*d**2+b3[1,2]*d+b3[2,2]))*math.exp(-((t-b5[2,2])**2)/b4[2,2])
Wmin_2 = ((b6[0,0]*d**2+b6[1,0]*d+b6[2,0])*n**2+(b6[0,1]*d**2+b6[1,1]*d+b6[2,1])*n+(b6[0,2]*d**2+b6[1,2]*d+b6[2,2]))*math.exp(-((t-b8[2,2])**2)/b7[2,2])

c1=C[1]
c2=C[2]
c3=C[3]
c4=C[4]
c5=C[5]
c6=C[6]
c7=C[7]
c8=C[8]
Wmax_f = (c1[1,0]*d+c1[2,0])*n**2+(c1[1,1]*d+c1[2,1])*n+(c1[1,2]*d+c1[2,2])*t+c2[0,2]*d**2+c2[1,2]*d+c2[2,2]
Wmax_1 = ((c3[0,0]*d**2+c3[1,0]*d+c3[2,0])*n**2+(c3[0,1]*d**2+c3[1,1]*d+c3[2,1])*n+(c3[0,2]*d**2+c3[1,2]*d+c3[2,2]))*math.exp(-((t-c5[2,2])**2)/c4[2,2])
Wmax_2 = ((c6[0,0]*d**2+c6[1,0]*d+c6[2,0])*n**2+(c6[0,1]*d**2+c6[1,1]*d+c6[2,1])*n+(c6[0,2]*d**2+c6[1,2]*d+c6[2,2]))*math.exp(-((t-c8[2,2])**2)/c7[2,2])

#---------------
W_mod=Wmod_f+Wmod_1+Wmod_2
W_min=Wmin_f+Wmin_1+Wmin_2
W_max=Wmax_f+Wmax_1+Wmax_2

W=[W_min, W_mod, W_max]
return W
```

Рисунок 4.23 – Код функції W_consumption()

```
def Bon(s, P_sb, r, P_w, e, W):
    # розрахунок K2U, K0U визначаються за держстандартом з якості електроенергії

    UA = np.random.uniform(100.0, 320.0)
    UB = np.random.uniform(100.0, 320.0)
    UC = np.random.uniform(100.0, 320.0)

    UAB = np.random.uniform(223.0, 537.0)
    UAC = np.random.uniform(223.0, 537.0)
    UBC = np.random.uniform(223.0, 537.0)

    UMAX = max(UAB, UAC, UBC)
    UMIN = min(UAB, UAC, UBC)
    Umax2 = max(UA, UC, UB)
    Umin2 = min(UA, UC, UB)

    K2U = 0.62*(UMAX-UMIN)*100/380
    K0U = 0.62*(Umax2-Umin2)*100/220

    #потужність сонячної батареї Psb
    Psbmod = 34.237*s
    Psbmin = 11.495*s
    Psbmax = 64.793*s

    MsbL = max(0, min(1, ((Psbmod - P_sb)/(Psbmod - Psbmin))))
    MsbN = max(0, min(1, ((P_sb - Psbmin)/(Psbmod - Psbmin)), (Psbmod - P_sb)/(Ps
    MsbH = max(0, min(1, (P_sb - Psbmod)/(Psbmax - Psbmod)))

    if max(MsbL, MsbN, MsbH) == MsbL:
        Msb = 1
    elif max(MsbL, MsbN, MsbH) == MsbN:
        Msb = 2
    else: Msb = 3

    #потужність вітроустановки PW
    Pwmin = 1.405*pow(r,2)
    Pwmod = 7.729*pow(r,2)
    Pwmax = 14.052*pow(r,2)
```

```
    MpwL = max(0, min(1, ((Pwmod - P_w)/(Pwmod - Pwmin))))
    MpwN = max(0, min(1, ((P_w - Pwmin)/(Pwmod - Pwmin)), (Pwmod - P_w)/(Pwmax - Pwmod)))
    MpwH = max(0, min(1, (P_w - Pwmod)/(Pwmax - Pwmod)))

    if max(MpwL, MpwN, MpwH) == MpwL:
        Mpw = 1
    elif max(MpwL, MpwN, MpwH) == MpwN:
        Mpw = 2
    else: Mpw = 3

    #ємність акумуляторної батареї PB
    Pbmax = 5000

    MpbL = max(0, min(1, (0.75*Pbmax - e)/(0.5*Pbmax)))
    MpbH = max(0, min(1, (e - 0.25*Pbmax)/(0.5*Pbmax)))

    if max(MpbL, MpbH) == MpwL:
        Mpb = 1
    else: Mpb = 3

    #поточна потужність електроспоживання W
    Wmin = 0.54
    Wmod = 6.49
    Wmax = 31

    MwL = max(0, min(1, ((Pwmod - W/100)/(Wmod - Wmin))))
    MwN = max(0, min(1, ((W/100 - Wmin)/(Wmod - Wmin)), (Wmod - W/100)/(Wmax - Wmod)))
    MwH = max(0, min(1, (W/100 - Wmod)/(Wmax - Wmod)))

    if max(MwL, MwN, MwH) == MwL:
        Mw = 1
    elif max(MwL, MwN, MwH) == MwN:
        Mw = 2
    else: Mw = 3

    B1 = 2
    B5 = 2
    B6 = 2
```

Рисунок 4.24 – Фрагмент функції Bon()

Якщо користувачеві потрібно створити звіт на основі отриманих даних, рекомендується натиснути кнопку «Звіт». Файл буде збережено в папці проекту з назвою "Report.docx". Приклад виконання програмою даної функції показано на рисунку 4.25, який містить відповідні повідомлення про результати запису та сам файл звіту - рис. 4.26. Код програми для виконання цієї команди показано на рисунку 4.27.

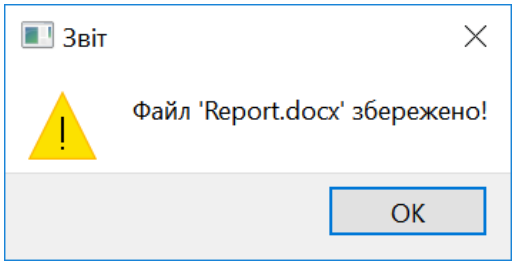

Рисунок 4.25- Результат збереження звіту

РЕЗУЛЬТАТИ·ЕЛЕКТРОСПОЖИВАННЯ¶

¶

Потужність·ВЕУ¤	10¤
Потужність·СБ¤	268¤
Ємність·акумулятора¤	4999¤
Рівень·споживання·електроенергії¤	149¤
Джерело·енергії¤	АБ¤
Передано·до·загальної·мережі¤	0¤

¶

Дата:2020-12-07¶

Рисунок 4.26 – Сформований звіт

```
def create_report(self):
    e1=self.ui.label_veu.text()
    e2=self.ui.label_sb.text()
    e3=self.ui.label_bt.text()
    e4=self.ui.label_cons.text()
    e5=self.ui.lbl_source_value.text()
    e6=self.ui.lbl_overflow_value.text()
    report_create(e1, e2, e3, e4, e5, e6)
    msgBox = QMessageBox()
    msgBox.warning(self, 'Звіт ',"Файл 'Report.docx' збережено!")
```

Рисунок 4.27 – Код функції створення звіту

Як додаткова можливість для користовуча – наявність доступу до метеорологічног сайту, для чого необхідно натиснути кнопку «Прогноз погоди». Як результат користувачу у браузері відкриється сайт Gismeteo.com, що відображає погоду відповідно до місцезнаходження користувача (рис. 4.28).

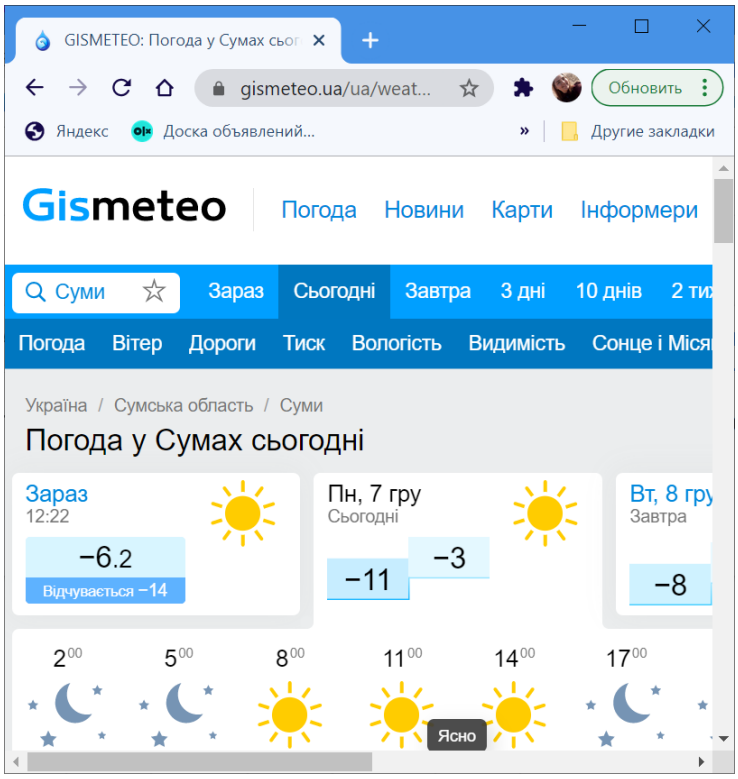

Рисунок 4.28 – Сайт Gismeteo.com

Оскільки дані погоди на сайті можуть змінюватись протягом дня, було вирішено збирати дані згідно з поточним часом, тому програма виконує обрахунки кожні 3 години, оскільки сайт Gismeteo.com забезпечує оновлення прогнозу погоди для кожного третього часу. Для такого запуску програми створено клас UpdateResults(threading.Thread), який містить функцію самозапуску run() (Додаток С).

Після проведення необхідних операцій розроблені файли коду hybrid.py, gparcer.py, mydesign.py, та myconnutils.py представлено у Додатку С. Інструкцію користувача для коректної роботи представлено у Додатку В.

ВИСНОВКИ

Під час виконання дипломного проекту було проведено аналіз методів визначення впливу погодних умов на продуктивність альтернативних джерел енергії та обрано кращий відповідно до визначеного переліку критеріїв, визначено, що кращим є метод нечітких правил, створено практичну реалізацію у вигляді програмного додатку, що відповідає всім поставленим функціональним вимогам:

- виконувати пошук геопозиції місцерозташування АДЕ;
- доступ до метеорологічного сайту;
- збереження розрахункових даних;
- визначення АДЕ за кращим методом;
- збереження результуючих даних кожного методу.

Використання розробленої інформаційної технології дозволить виявити ефективний з методів визначення впливу погодних умов на продуктивність АДЕ в залежності від технологічних та постійно змінних метеорологічних факторів. Практична реалізація даного дипломного проекту матиме попит серед менеджерів фірм, що мають на меті впровадження складних обчислювальних систем для визначення джерела електроенергії, а також серед домогосподарств, що мають вже встановлену гібридну енергосистему, для полегшення взаємодії зі складним апаратним устаткуванням.

Під час реалізації дипломного проекту роботи було виконано детальний аналіз предметної області: визначено актуальності поставленої проблеми в сучасному світі, проведено аналіз існуючих підходів у прийнятті управлінських рішень, щодо вибору кращого джерела електроенергії в заданий момент часу при заданих погодних показниках, сформовано кількісні та якісні критерії для порівняльної характеристики методів визначення впливу погодних умов на продуктивність альтернативних джерел енергії.

Аналіз предметної області показав, що існуючий загальний метод для прийняття рішення щодо кращого джерела електроенергії, заснований на дереві рішень, не є ефективним, тому що в силу своєї обмеженості не враховує нечіткі фактори метеорологічних даних. Тому було вирішено розробити інші підходи прийняття рішень, а також порівняти їх за певним переліком кількісних та якісних критеріїв.

Для порівняння було обрано методи дерева рішень, кластеризації та нечітких правил. Визначено щонайменше алгоритмічну складність кожного з методів, корисність, структурованість та інше. Визначено, що метод, заснований на використанні нечіткої логики враховує більшу кількість факторів неточності та невизначеності, що несуть в собі погодні показники та прогнозні дані споживання, ніж інші з досліджуваних.

Далі було проведено проектування проекту «Аналіз методів визначення впливу погодних умов на продуктивність альтернативних джерел енергії» за методологією SADT нотації IDEF0, а також виконано побудову діаграми варіантів використання. Діаграми нотації IDEF0 демонструють складність проекту, для чого було виконано декомпозицію концептуальної моделі. Дані діаграми демонструють взаємодію підзадач, а також результати, що отримуються після виконання кожного етапу роботи. Діаграма варіантів використання в свою чергу надає повний перелік можливостей, які надаються акторам під час використання проекту, формуючи сценарій взаємодії з проектом.

На освнові розроблених структурно-функціональних діаграм проекту було виконано практичну реалізацію у вигляді програмного додатку «Hybrid» з використанням технологій Python 3.7 та MySQL на основі створених математичних моделей, на який було отримано авторське свідоцтво № 94425. Також було проведено перевірку надійності результатів вибору ефективного джерела АДЕ, що задовольняє потреби користувача в споживанні електроенергії за певних погодних та технічних умов.

Отже, розроблений дипломний проект дозволить власникам домашніх господарств та гібридних енергосистем використовувати передові інформаційні технології для швидкої обробки неявної інформації та вибору найбільш ефективного джерела електроенергії, що задовольняє всім вимогам, без докладання додаткових зусиль.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. 12 Best Benefits of Gantt Charts for Project Management [Електронний ресурс] – режим доступу: https://bitly.su/3nRn 7

2. A historical review of evolutionary learning methods for Mamdani-type fuzzy rule-based systems: Designing interpretable genetic fuzzy systems [Електронний ресурс] – режим доступу: https://bitly.su/1h5wk7C4

3. Dai Wei. A mapreduce implementation of C4.5 decision tree algorithm [Text] / Dai Wei, Ji, Wei // International Journal of Database Theory and Application. – 2014. – С. 49–60.

4. Decision-Making Model at the Management of Hybrid Power Grid Tymchuk, S. , Shendryk, S. , Shendryk, V., Kazlauskaite, A.,Levytska, T. Communications in Computer and Information Science, 2020, 1283 CCIS, pp. 60-71

5. Functional and Information Modeling of Production Using IDEF Methods [Електронний ресурс] – режим доступу: https://bitly.su/WgeNZ 11

6. Gantt Charts [Електронний ресурс] – режим доступу: https://bitly.su/7Ix9 6

7. J. C. Bezdek, R. Ehrilich, W Fill, “FCM: The Fuzzy C-means Clustering Algorithm”, Computers & Geosciences, vol. 10, no. 2-3, pp. 191-203, 1984.

8. Know Thy Complexities! [Електроний ресурс] – режим доступу: https://bitly.su/U1N1SSrs

9. Parameswari, D. Intrusion detection system using modified J48 decision tree algorithm [Text] / D. Parameswari, V. Khanaa // Journal of Critical Reviews. – 2020. – С. 730–734.

10. Project Risk Management: How not to Fail Your Idea [Електронний ресурс] – режим доступу: https://bitly.su/Dhu7XnV 10

11. The IDEF Process Modeling Methodology [Електронний ресурс] – режим доступу: https://bitly.su/RzJlXk 12

12. The Methodology of Obtaining Power Consumption Fuzzy Predictive Model for Enterprises

13. Tymchuk, S., Shendryk, S. , Shendryk, V. , Abramenko, I. , Kazlauskaite, A. Lecture Notes in Mechanical Engineering, 2020, pp. 210-219

14. The RACI matrix: Your blueprint for project success [Електронний ресурс] – режим доступу: https://bitly.su/HQyk7iV 5

15. The Risk Management Process in Project Management [Електронний ресурс] – режим доступу: https://bitly.su/FsiS 9

16. Use case specifications: How complete are they? [Електронний ресурс] – режим доступу: https://bitly.su/XY3g 13

17. What Is a Network Diagram in Project Management? [Електронний ресурс] – режим доступу: https://bitly.su/PEYn 8

18. What is a Work Breakdown Structure? [Електронний ресурс] – режим доступу: https://www.workbreakdownstructure.com/ 2

19. What Is an Organizational Breakdown Structure (OBS)? [Електронний ресурс] – режим доступу: https://bitly.su/565ZGr 3

20. What is Use Case Diagram? [Електронний ресурс] – режим доступу: https://bitly.su/nwTM 14

21. Алгоритмы кластеризации на службе Data Mining [Електронний ресурс] – режим доступу: https://bitly.su/GDhrkBs

22. Грабченко А.І., Федорович В.О., Гаращенко Я.М. Методи наукових досліджень: Навч. посібник. – Х.: НТУ "ХПІ", 2009. – 142 с.

23. Енергоефективне керування вітроустановками малої потужності для генерування електричної і теплової енергії [Електронний ресурс] – режим доступу: https://bitly.su/nF4hvAn

24. Казлаускайте А. С. Прогнозування рівня електрогенерації сонячних батарей при управлінні гібридною електромережею [Текст] / А. С. Казлаускайте, С. О. Шендрик // Інформатика, математика, автоматика – 2019 – С. 138.

25. Кластерный анализ [Електроний ресурс] – режим доступу: https://bitly.su/GaDAVa

26. Лапицкая, О. В. Принятие решений в маркетинге [Текст] / О. В. Лапицкая, А. В. Шах // Экономика и управление народным хозяйством. – 2018. – С. 62–69.

27. Интеллектуальный анализ данных [Електронний ресурс] – режим доступу: https://bitly.su/zpj1

28. Нечіткі моделі в медичних експертних системах [Електронний ресурс] – режим доступу: https://bitly.su/7Zorr6

29. Организационная структура компании [Електронний ресурс] – режим доступу: https://bitly.su/p3ClovOK 4

30. Оценка сложности алгоритмов [Електроний ресурс] – режим доступу: https://bitly.su/RqAy

31. Преимущества и недостатки метода нечетких множеств [Електронний ресурс] – режим доступу: https://bitly.su/9FFvQZn

32. Розподілене виробництво енергії [Електронний ресурс] – режим доступу: https://bitly.su/9qdPvhUE

33. Розподілене виробництво енергії [Електронний ресурс] – режим доступу: https://bitly.su/SSYV96B

34. Складність алгоритмів [Електроний ресурс] – режим доступу: https://bitly.su/EUal4X

35. Тихонов, Є.С. Аналіз існуючих алгоритмів кластеризації даних. Переваги та недоліки [Текст] / Є. С. Тихонов, К. В. Тихонова // Проблеми розвитку та вдосконалення єдиної національної системи зв’язку. – 2020. – С. 17–20.

36. Українська Асоціація Відновлюваної Енергетики [Електронний ресурс] – режим доступу: https://bitly.su/wLQoj

37. Чому в Україні слід розвивати децентралізовану енергетику вже сьогодні? [Електронний ресурс] – режим доступу: https://bitly.su/Iuz3is

38. Что такое дерево решений и где его используют? [Електронний ресурс] – режим доступу https://bitly.su/Ysu0AFBj

39. Шендрик, В. В. Актуальность моделирования распределенных энергосистем эффективного использования возобновляемых источников энергии [Текст] / В. В. Шендрик, С. М. Ващенко, О. В. Шулима, К. А. Омеляненко // Восточно-Европейский журнал передовых технологий. – 2013. – № 5/8(65). – С. 4–8. – Режим доступа: URL: http://journals.uran.ua/ eejet/article/view/18118/15866.

Додаток А

Планування робіт

Ідентифікація мети ІТ-проекту. Метою дипломної роботи є створення інформаційної технології, яка буде виконувати аналіз методів дослідження впливу постійно-змінних погодних умов на продуктивність альтернативних джерел енергії. Для більш докладної деталізації мети було застосовано метод SMART, отримані результати представлено в таблиці А.1.

Таблиця А.1 – Деталізація мети методом SMART

Specific (конкретна)	Провести аналіз методів відповідно до визначених критеріїв та створити практичну реалізацію кращого з методів.
Measurable (вимірювання)	Результатом роботи проекту є оцінка самого замовника, оскільки проект – не комерційний.
Achievable (досяжна, узгоджена)	Мета даного проекту вважається досяжною, оскільки розробник володіє необхідними знаннями та кваліфікацією для проведення такого роду аналізу, а також необхідними навичками у створенні програмних продуктів засобами Python.
Relevant (реалістична)	Для практичної реалізації проекту наявними є всі необхідні технічні та програмні засоби: веб-сервер Open Server, універсальний інтерпретатор Notepad++, open source дистрибутив для мови програмування Python Anaconda, а також доступ до мережі інтернет.

Продовження таблиці А.1 – Деталізація мети методом SMART

Time-framed (обмежена в часі)	Аналіз методів, а також його практична реалізація розробляються з обмеженням у часі, ґрунтуючись на сформований календарний план та матрицю відповідальності.

Планування змісту структури робіт ІТ-проекту. Наступним кроком у плануванні роботи з ІТ-проектом є розробка діаграм WBS та OBS.

Ієрархічна структура робіт (WBS) - це ключовий результат проекту, який об'єднує роботу групи в керовані розділи. У Зводі знань з управління проектами (PMBOK) структура декомпозиції робіт визначається як «ієрархічна декомпозиція роботи, орієнтованої на кінцевий результат, яка повинна бути виконана командою проекту». Структурна декомпозиція робіт візуально визначає обсяг робіт, необхідний для виконання керованих фрагментів, які є зрозумілими для команди проекту, оскільки кожен рівень ієрархічної структури робіт забезпечує подальше визначення і деталізацію [11]. Діаграму даного дипломного проекту «Аналіз методів визначення впливу погодних умов на продуктивність альтернативних джерел енергії» зображено на рис. А.1

Організаційна структура проекту (OBS) – це документ, який бізнес може використовувати разом із графіком робочого циклу та розподілом ресурсів для організації діяльності людей, які будуть працювати над певним проектом. OBS деталізує підпорядкування, подробиці ієрархії і структури звітності [15].

До головних характеристик організаційної структури проекту можна віднести:

– створюється на рівні підприємства;

– елементи призначаються на проекти, пакети робіт WBS;

– надає можливість контролю доступу користувачів до інформації на відповідному рівні;

– елемент OBS, призначений відповідному елементу, є керівником проекту або відповідальним за всю роботу, що міститься в цьому елементі [15].

У таблиці А.2 представлено учасників дипломного проекту та відповідні обов’язки кожного з учасників. Діаграму OBS даного дипломного проекту представлено на рис. А.2.

Таблиця А.2 – Виконавці проекту

Роль	Ім’я	Проектна роль
Розробник, тестувальник	Казлаускайте А.С.	Виконує збір та аналіз даних, розробку математичних моделей, основного функціоналу проекту та його інтерфейсу, відповідає за тестування
Керівник проекту	Шендрик В.В.	Відповідає за контроль термінів, формує завдання на розробку проекту

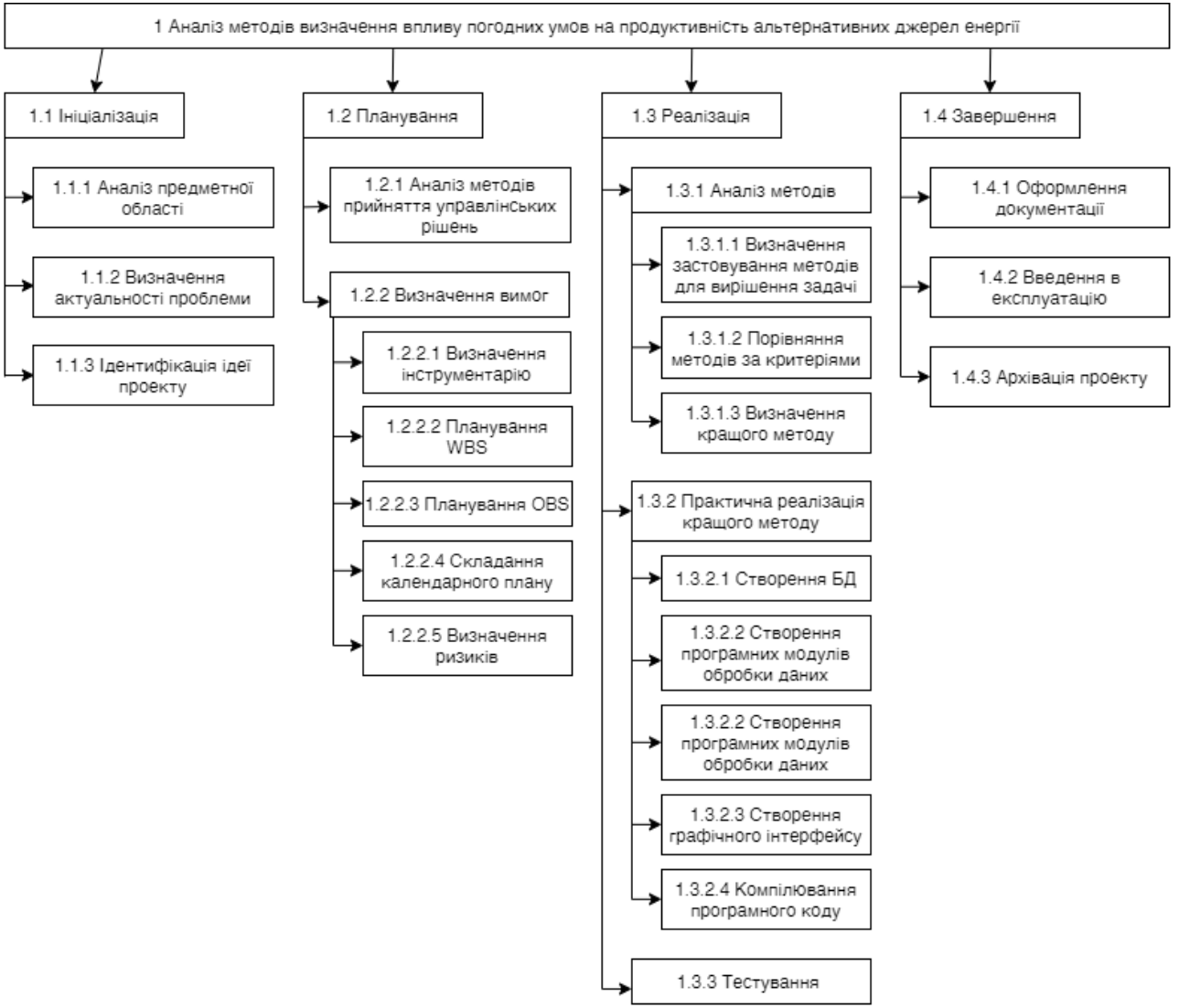

Рисунок А.1 − WBS. Структура робіт проекту

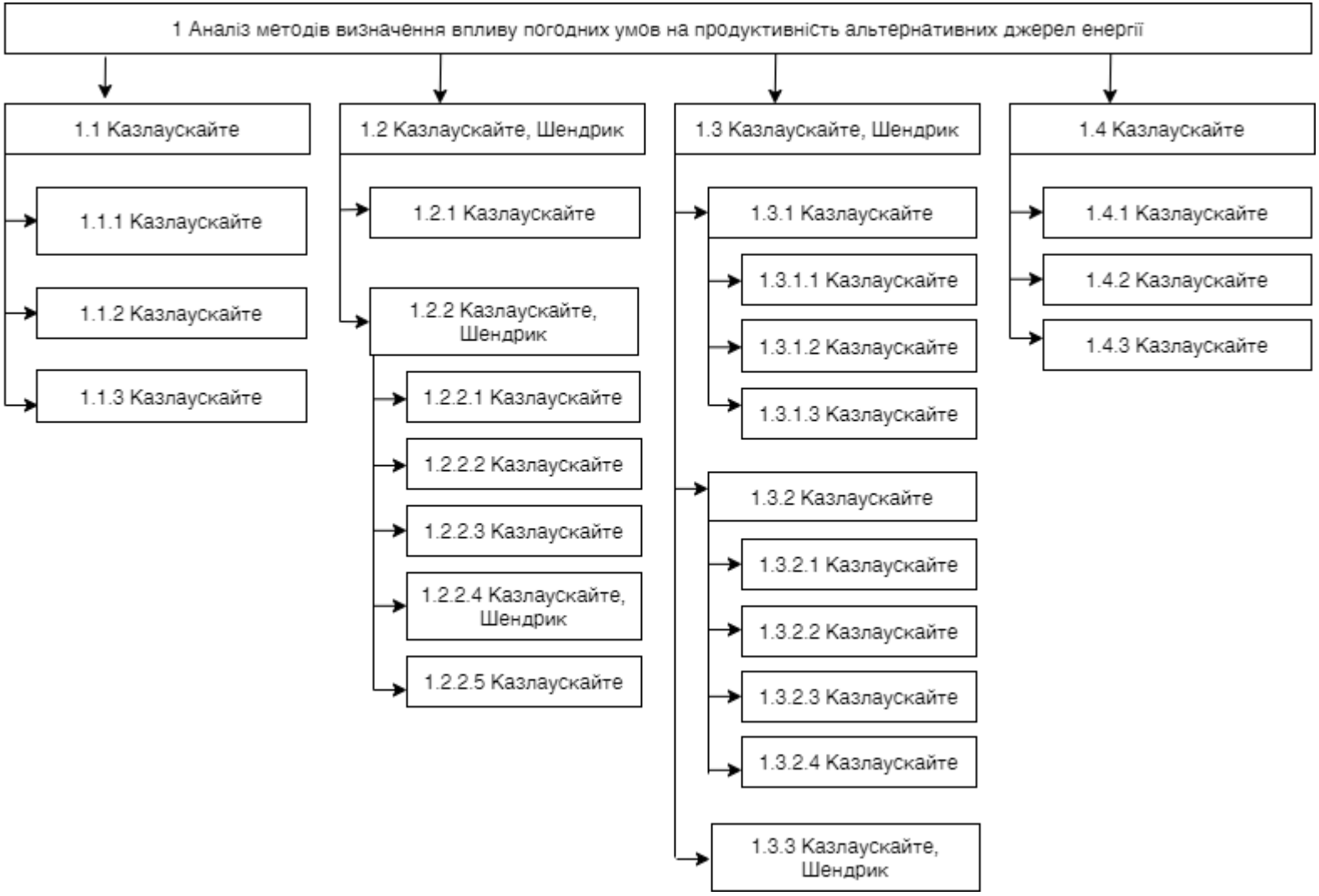

Рисунок А.2 − OBS. Організаційна структура проекту

Після розробки діаграм WBS та OBS необхідно сформувати матрицю відповідальності.

Матриця відповідальності (RACI) є найпростішим, найефективнішим засобом для визначення та документування проектних ролей та відповідальності. Точне знання того, хто відповідальний, з ким потрібно проконсультуватися, а хто повинен бути поінформований на кожному кроці, значно підвищує шанси на успіх проекту [7].

У таблиці А.3 представлено матрицю відповідальності дипломного проекту «Аналіз методів визначення впливу погодних умов на продуктивність альтернативних джерел енергії».

Таблиця А.3 – Матриця відповідальності

WBS\OBS	Казлаускайте	Шендрик
Аналіз предметної області	+	
Визначення в потребі системи	+	
Ідентифікація ідеї проекту	+	
Аналіз методів прийняття рішень	+	
Визначення інструментарію	+	
Планування WBS	+	
Планування OBS	+	
Складання календарного плану	+	+
Визначення ризиків	+	
Визначення застосування методів для вирішення задачі	+	
Порівняння методів за критеріями	+	
Визначення кращого методу	+	
Створення бази даних	+	

Продовження таблиці А.3 – Матриця відповідальності

Створення програмних модулів обробки даних	+	
Створення графічного інтерфейсу	+	
Компілювання програмного коду	+	
Тестування	+	+
Оформлення документації	+	
Введення в експлуатацію	+	
Архівація проекту	+	

Побудова календарного графіку виконання ІТ-проекту. По закінченню визначення мети дипломного проекту за допомогою методології SMART, а також створення діаграм WBS та OBS для детермінації структури проекту, існує необхідність розробки діаграми Ганта та PDM-мережі.

Діаграма Ганта – це інформативне графічне представлення дій, які необхідно виконати відповідно до плану. Діаграма Ганта дає графічний огляд дій в проекті і дає пряме уявлення про статус проекту щодо часу виконання, залежностей і прогресу. Детальне продумування допомагає забезпечити робочий графік, призначити потрібних людей для кожного завдання та мати обхідні шляхи для потенційних проблем перед початком роботи [3].

До переваг побудови діаграми Ганта можна віднести наступне:

– підвищення продуктивності роботи команди;
– можливість керувати складною інформацією;
– відстежування прогресу проекту;
– легкість керування віддаленими командами;
– планування ресурсів [1].

Діаграму Ганта проекту «Аналіз методів визначення впливу погодних умов на продуктивність альтернативних джерел енергії» представлено на рисунку А.3. На рисунку А.4 представлено перелік етапів проекту з термінами їх виконання.

Як доповнення до діаграми Ганта було розроблено мережевий графік. Мережевий графік для проекту є також важливим інструментом, оскільки допомагає проектним командам візуалізувати всі заходи, які необхідно виконати протягом проекту. Це також дає важливий контекст, такий як тривалість завдання, послідовність та залежність між етапами робот. Сам мережевий графік проекту представляє собою граф, що демонструє етапи роботи, їх тривалість та взаємозалежність [11]. На рис. А.5 – А.10 представлено мережевий графік дипломного проекту «Аналіз методів визначення впливу погодних умов на продуктивність альтернативних джерел енергії».

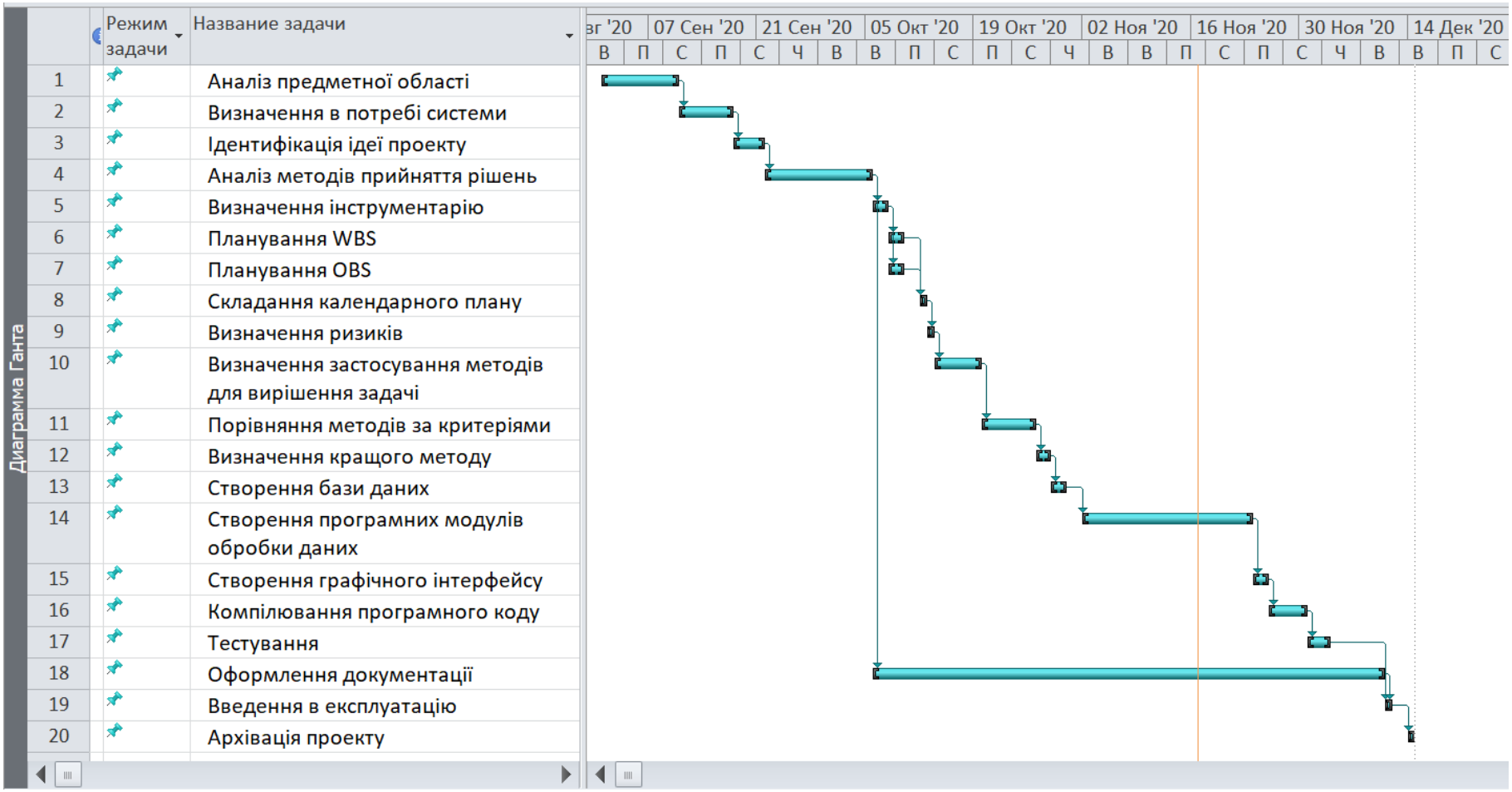

Рисунок А.3 – Діаграма Ганта для проекту «Аналіз методів визначення впливу погодних умов на продуктивність альтернативних джерел енергії»

	Режим задачи	Название задачи	Длительно	Начало	Окончание
1		Аналіз предметної області	8 дней	Вт 01.09.20	Чт 10.09.20
2		Визначення в потребі системи	5 дней	Пт 11.09.20	Чт 17.09.20
3		Ідентифікація ідеї проекту	2 дней	Пт 18.09.20	Пн 21.09.20
4		Аналіз методів прийняття рішень	10 дней	Вт 22.09.20	Пн 05.10.20
5		Визначення інструментарію	2 дней	Вт 06.10.20	Ср 07.10.20
6		Планування WBS	2 дней	Чт 08.10.20	Пт 09.10.20
7		Планування OBS	2 дней	Чт 08.10.20	Пт 09.10.20
8		Складання календарного плану	1 день	Пн 12.10.20	Пн 12.10.20
9		Визначення ризиків	1 день	Вт 13.10.20	Вт 13.10.20
10		Визначення застосування методів для вирішення задачі	4 дней	Ср 14.10.20	Пн 19.10.20
11		Порівняння методів за критеріями	5 дней	Вт 20.10.20	Пн 26.10.20
12		Визначення кращого методу	2 дней	Вт 27.10.20	Ср 28.10.20
13		Створення бази даних	2 дней	Чт 29.10.20	Пт 30.10.20
14		Створення програмних модулів обробки даних	16 дней	Пн 02.11.20	Пн 23.11.20
15		Створення графічного інтерфейсу	2 дней	Вт 24.11.20	Ср 25.11.20
16		Компілювання програмного коду	3 дней	Чт 26.11.20	Пн 30.11.20
17		Тестування	3 дней	Вт 01.12.20	Чт 03.12.20
18		Оформлення документації	48 дней	Вт 06.10.20	Чт 10.12.20
19		Введення в експлуатацію	1 день	Пт 11.12.20	Пт 11.12.20
20		Архівація проекту	1 день	Пн 14.12.20	Пн 14.12.20

Рисунок А.4 – Список робіт для діаграми Ганта

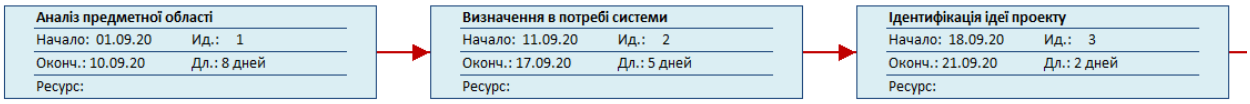

Рисунок А.5 – PDM-мережа проекту

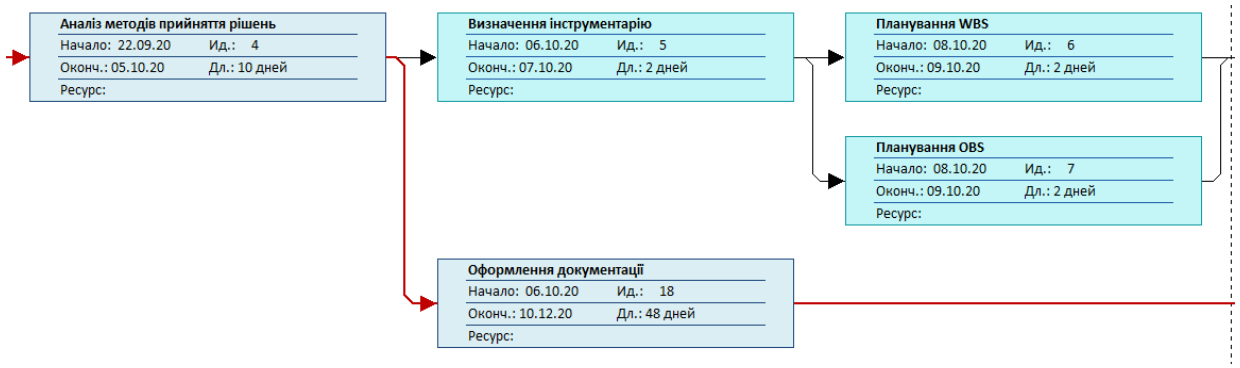

Рисунок А.6 – Продовження PDM-мережі проекту

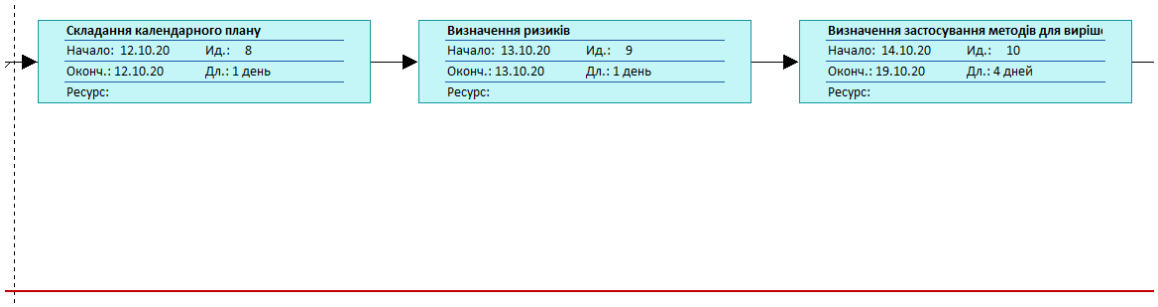

Рисунок А.7 – Продовження PDM-мережі проекту

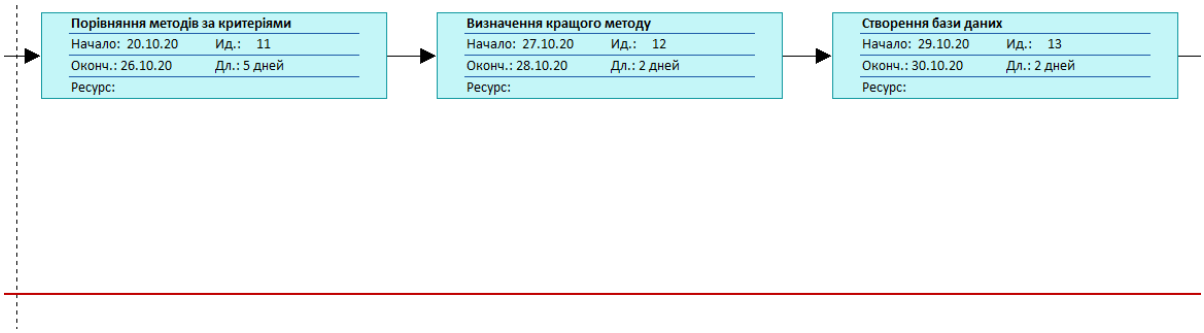

Рисунок А.8 – Продовження PDM-мережі проекту

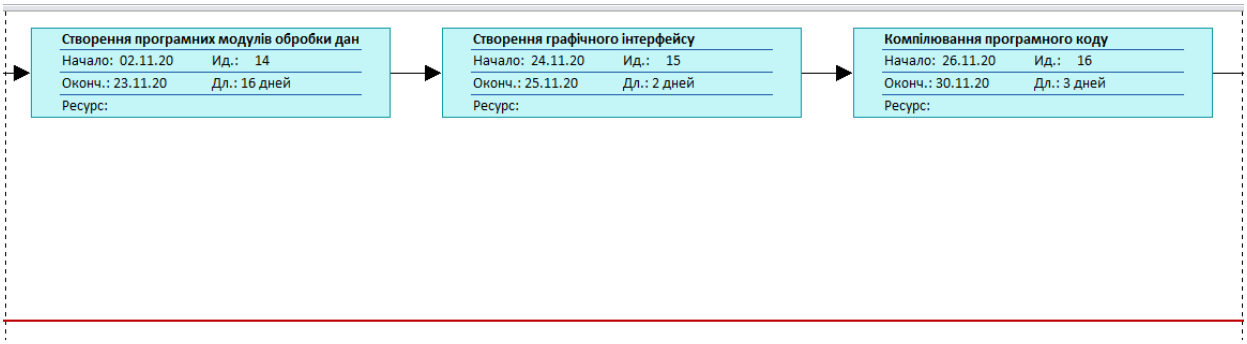

Рисунок А.9 – Продовження PDM-мережі проекту

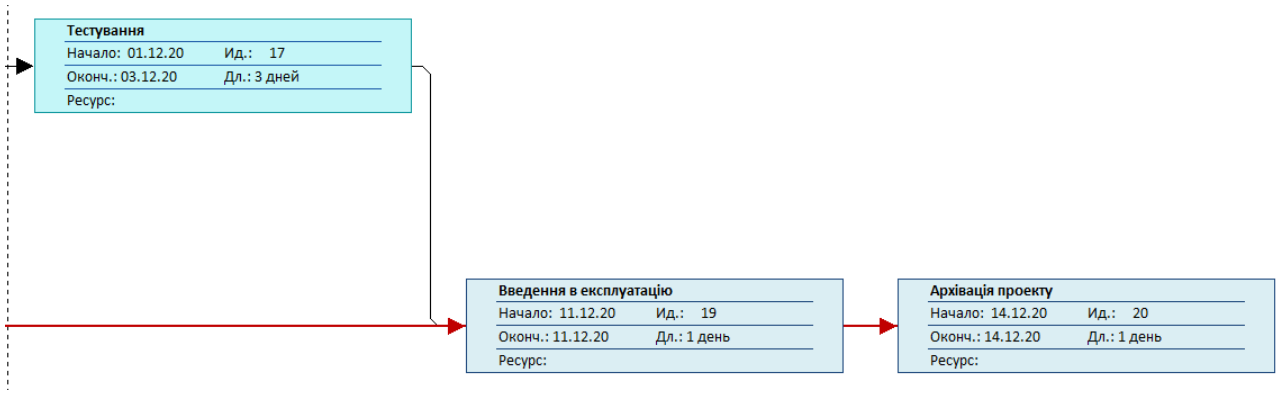

Рисунок А.10 – Продовження PDM-мережі проекту

Планування ризиків проекту. Управління ризиками проекту – це процес виявлення, аналізу та подальшого реагування на будь-який ризик, що виникає протягом життєвого циклу проекту, щоб допомогти проекту досягнути поставленої мети. Управління ризиками – це не тільки реагування; цей процес повинен початися ще на стадії планування, щоб мати можливість ідентифікувати ризик, який може статися в проекті, і контролювати цей ризик, якщо він дійсно виникає.

Ризик – це все, що потенційно може вплинути на графік виконання робіт, результативність чи бюджет проекту. Ризики – це потенціальні можливості, і в контексті управління проектами, якщо вони перетворюються на реальність, класифікуються як “проблеми”, які необхідно вирішити. Отже, управління ризиками – це процес виявлення, категоризації, розстановки пріоритетів та планування ризиків до того, як вони стануть проблемами.

Управління ризиками може означати різні речі для різних типів проектів. Для великомасштабних проектів стратегії управління ризиками включають детальне планування кожного ризику, щоб забезпечити наявність стратегій пом'якшення, якщо виникають проблеми. Для менших проектів управління ризиками означає більш простий, пріоритетний перелік ризиків з високим, середнім та низьким пріоритетом [7].

Найпоширеніші ризики в проекті розробки програмного забезпечення пов'язані з:

– період часу (досить часто проекти схильні до невідповідності встановленим термінам);

– вартість (ще одним важливим питанням для проектів є дотримання попередньо узгоджених кошторисів);

– продуктивність (ризик того, що кінцеве рішення не буде працювати належним чином і цілі проекту не будуть досягнуті) [4].

Виконавши аналіз можливих ризиків даного дипломного проекту було сформовано таблицю А.4, що відображає класифікацію ймовірних ризиків.

Таблиця А.4 – Класифікація рисків

№	Назва ризику	Ймовірність	Величина втрат
1	Некоректно складене ТЗ	2	4
2	Недотримання календарного плану	1	3
3	Некоректна робота програмного забезпечення	4	4
4	Некоректна робота супроводжуючого устаткування	4	5
5	Хвороба розробника	1	2
6	Некоректне тестування	2	1

За допомогою даної класифікації будується матриця ризику, що представлено на рис. А.11.

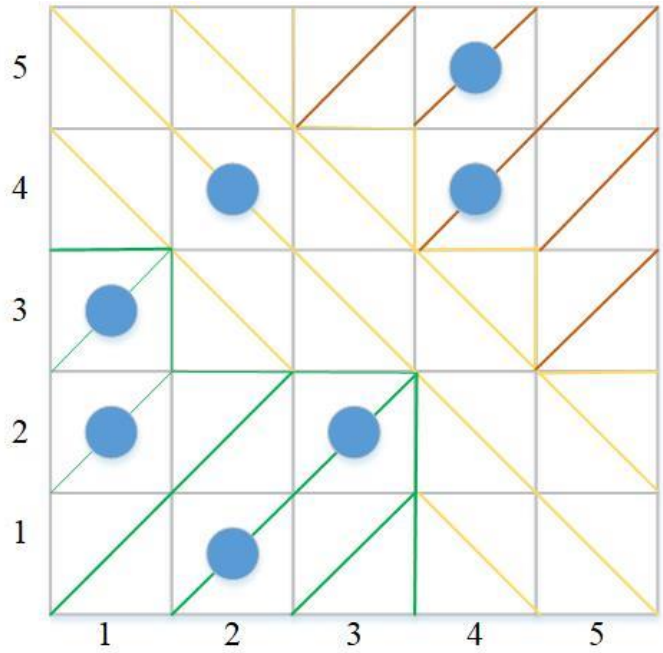

Рисунок А.11 – Матриця ризиків

Далі визначено рівень ризиків та рівень їх впливу за відповідною формулою для розрахунку: R = P * L, де R – ранг ризику; P – ймовірність виникнення; L – величина збитків.

Рівні ризиків можуть бути:

– допустимі 1<R<4;
– оправдані 5<R<10;
– недопустимі 11<R<25.

Ступінь впливу ризиків:

– ті, що можна проігнорувати 1<R<4;
– незначні 5<R<8;
– помірні 9<R<10;
– істотні 11<R<16;
– критичні 17<R<25.

На підставі вищезазначених даних було оцінено ступінь та рівень кожного ризику проекту. Результати наведені в таблиці А.5.

Таблиця А.5 – Визначення ступенів та рівнів ризиків

№	Назва ризику	Ймовірність ризику	Ранг ризику	Рівень ризику	Ступінь дії
1	Некоректно складене ТЗ	2	8	Виправданий	Незначний
2	Недотримання календарного плану	1	3	Допустимий	Проігнорувати
3	Некоректна робота програмного забезпечення	4	16	Недопустимий	Істотний
4	Некоректна робота супроводжуючого устаткування	4	20	Недопустимий	Критичний
5	Хвороба розробника	1	2	Допустимий	Проігнорувати
6	Некоректне тестування	2	2	Допустимий	Проігнорувати

Додаток Б

Тестування методу кластеризації

Таблиця Б.1 – Прогноз погоди

	21.03.2019							
	2.00	5.00	8.00	11.00	14.00	17.00	20.00	23.00
t	0	-1	1	4	4	4	3	3
хмарність	ясно	нев. хмар	пом. хмар	хмарно	хмарно	пом. хмар	хмарно	пом. хмар
шв.вітру	1	2	3	5	5	5	3	3
вітер	з	юз	з	з	з	сз	сз	з
	21.06.2019							
	2.00	5.00	8.00	11.00	14.00	17.00	20.00	23.00
t	21	19	20	25	29	31	30	26
хмарність	нев. хмар	пом. хмар	пом. хмар	нев. хмар	пом. хмар	пом. хмар	нев. хмар	нев. хмар
шв.вітру	2	2	2	3	3	1	3	1
вітер	в	в	в	ю	юв	юв	юв	в
	22.09.2019							
	2.00	5.00	8.00	11.00	14.00	17.00	20.00	23.00
t	7	7	10	12	11	15	15	10
хмарність	пом. хмар	хмарно	хмарно	хмарно	хмарно	пом. хмар	пом. хмар	хмарно
шв.вітру	4	5	5	7	8	9	6	4
вітер	юз	юз	з	з	сз	з	з	сз
	22.12.2019							
	2.00	5.00	8.00	11.00	14.00	17.00	20.00	23.00
t	5	7	7	8	9	9	10	9
хмарність	хмарно	хмарно	хмарно	хмарно	хмарно	хмарно	хмарно	хмарно
шв.вітру	5	5	6	6	7	6	7	6
вітер	юв	ю	юв	ю	юв	ю	юв	ю

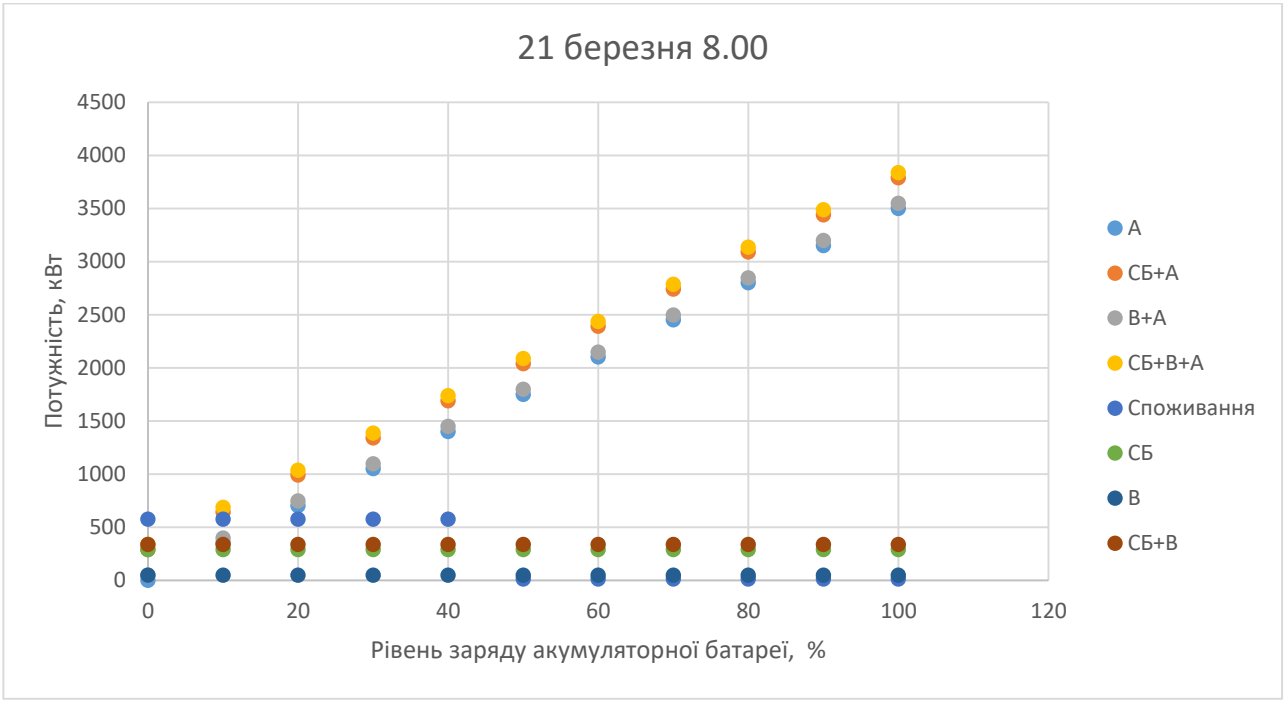

Рисунок Б.1 – Приклад рівня енергії в мережі на 21 березня 8.00

Рисунок Б.2 – Приклад рівня енергії в мережі на 21 березня 17.00

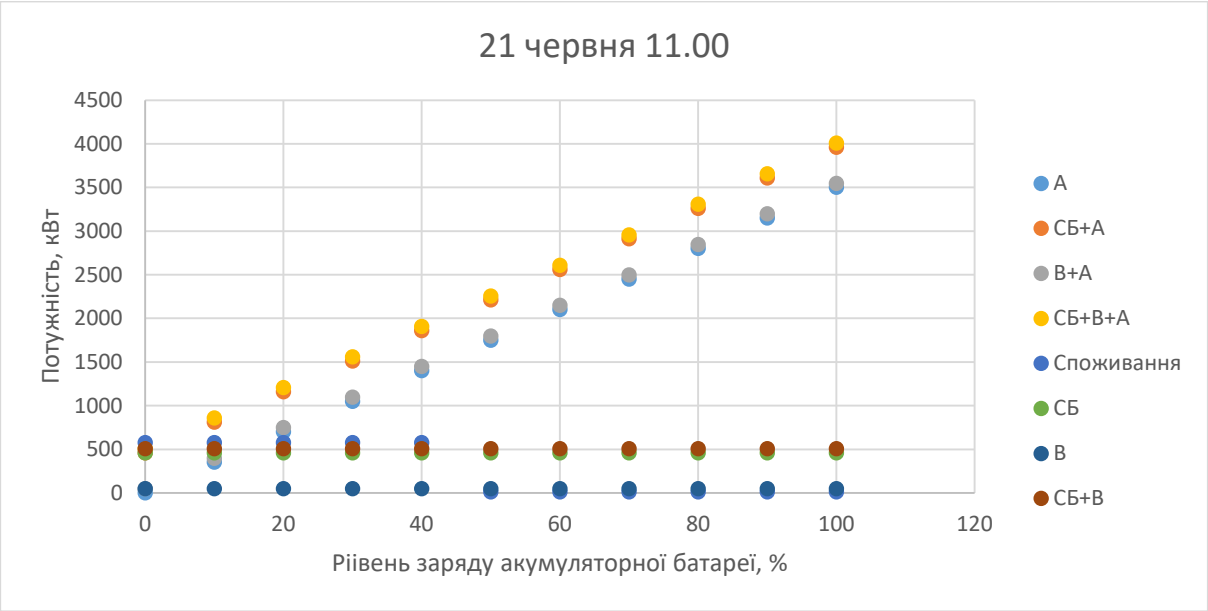

Рисунок Б.3 – Приклад рівня енергії в мережі на 21 червня 11.00

Рисунок Б.4 – Приклад рівня енергії в мережі на 21 червня 17.00

Рисунок Б.5 – Приклад рівня енергії в мережі на 22 вересня 14.00

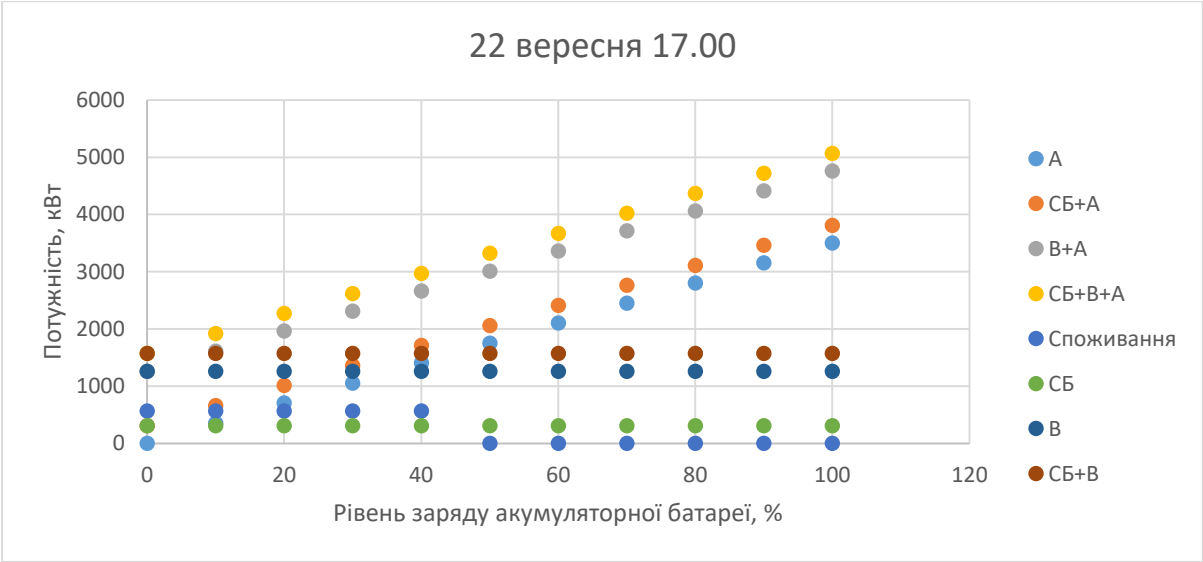

Рисунок Б.6 – Приклад рівня енергії в мережі на 22 вересня 17.00

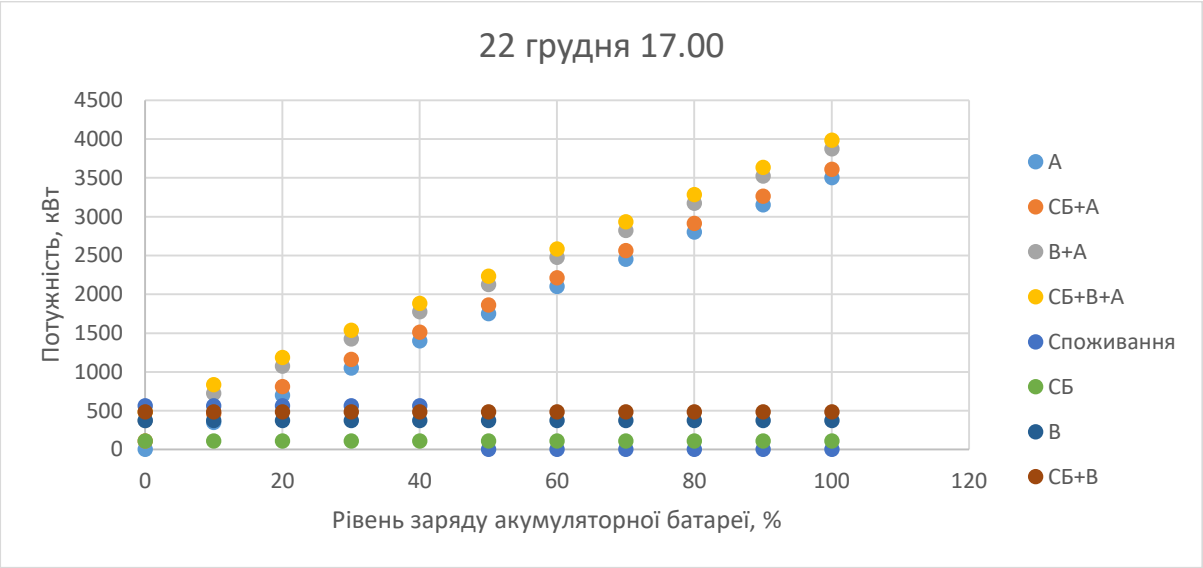

Рисунок Б.7 – Приклад рівня енергії в мережі на 22 грудня 17.00

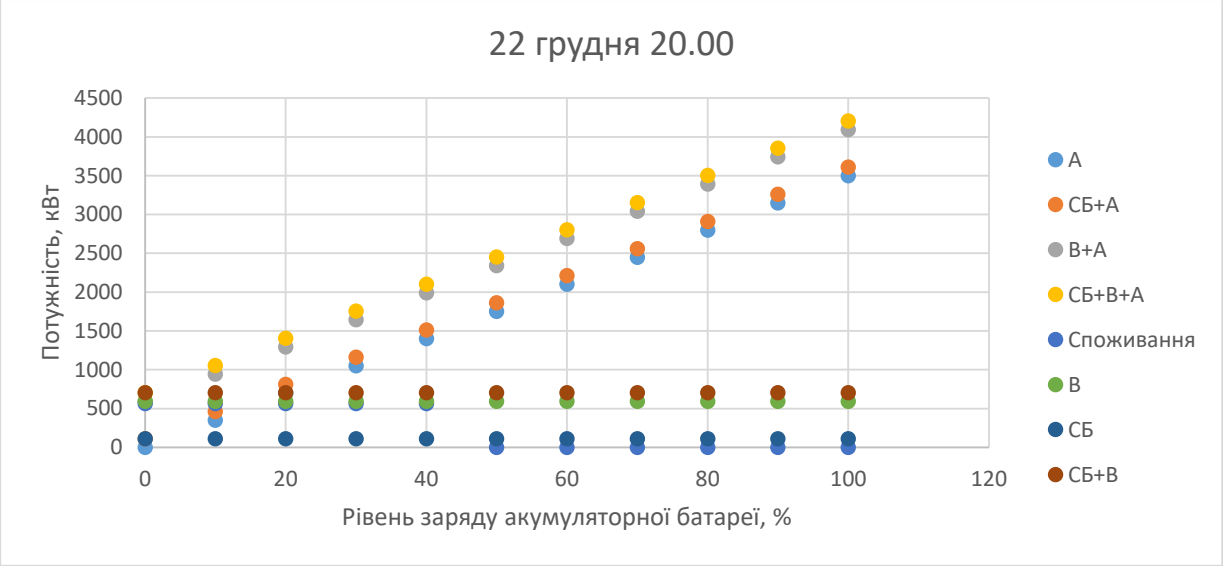

Рисунок Б.8 – Приклад рівня енергії в мережі на 22 грудня 20.00

Додаток В

Інструкція користувача до програмного продукту «Hybrid»

1. Запуск програмного продукту

Для налаштування даного програмного продукту необхідно виконати наступні кроки:

1. Встановити веб-сервер, створити базу даних «hybrid» та імпортувати до неї наявну базу даних «hybrid.sql».

2. Запустити exe-файл «hybrid.exe», що має підключення до мережі Інтернет, а також до створеної бази даних, та використовувати програмний продукт за призначенням.

3. Вимоги до апаратного забезпечення:

– 1 Гбайт оперативної пам’яті;

– 4 і більше ядер центрального процесора;

– ARM32/ARM64/x86/x64 архітектури центрального процесора;

– операційна система Windows 7 та вище

2. Взаємодія з програмним продуктом

При запуску програмного продукту користувачеві відкривається головне діалогове вікно (рис. В.1).

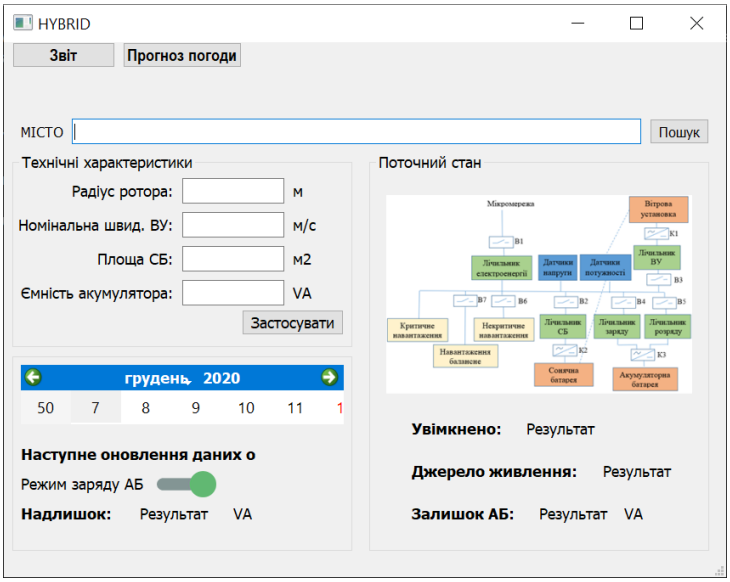

Рисунок В.1 – Головне вікно програмного продукту

Щоб здійснити пошук потрібного міста, необхідно ввести у поле введення коротке (лише назва) або повне (назва, регіон, країна) наіменування міста та натиснути кнопку «Пошук». Як результат, угорі відображатиметься повне наіменування бажаного міста (рис. В.2).

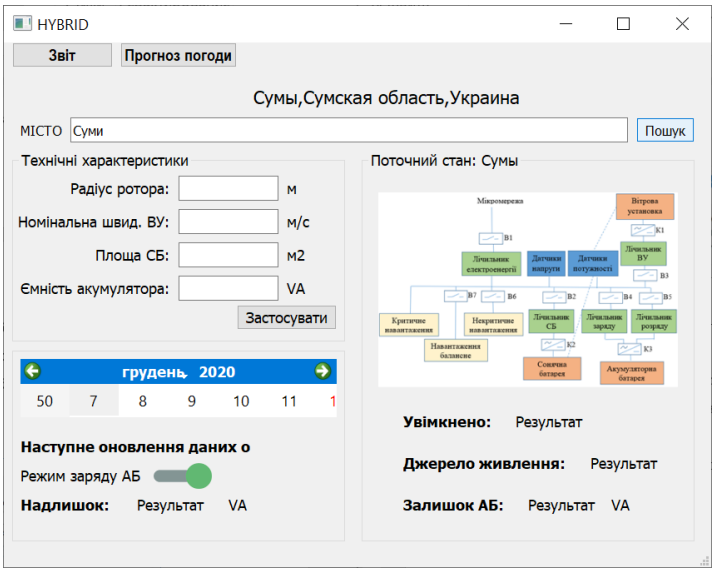

Рисунок В.2 – Результат виконання пошуку необхідного міста

Якщо такого міста не існує, або дані, введені користувачем, є некорректними, користувач отримає повідомлення про помилку (рис. В.3).

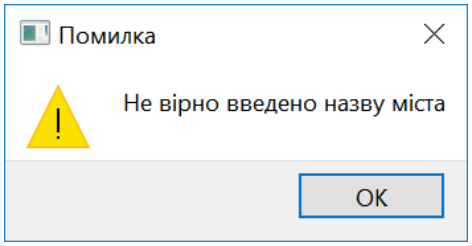

Рисунок В.3 – Виведення повідомлення про помилку пошуку міста

Далі для того, щоб виконати необхідні розрахунки та отримати результати, необхідно ввести у поля введення технічні характеристики складових гібридної енергосистеми: радіус вітроротора (або довжину лопаті), номінальну швидкість вітроротора та загальну площину встановлених сонячних панелей, а також рівень заряду акумулятора. Ці дані можна знайти в технічному паспорті кожного з виробів.

Далі після заповнення необхідних полей введення користувачеві потрібно натиснути кнопку «Застосувати», результат виконання дії показано на рисунку В.4. Якщо користувач вводить неправильні дані, він отримає відповідне повідомлення про помилку (рис. В.5) і повинен ввести виправити дані.

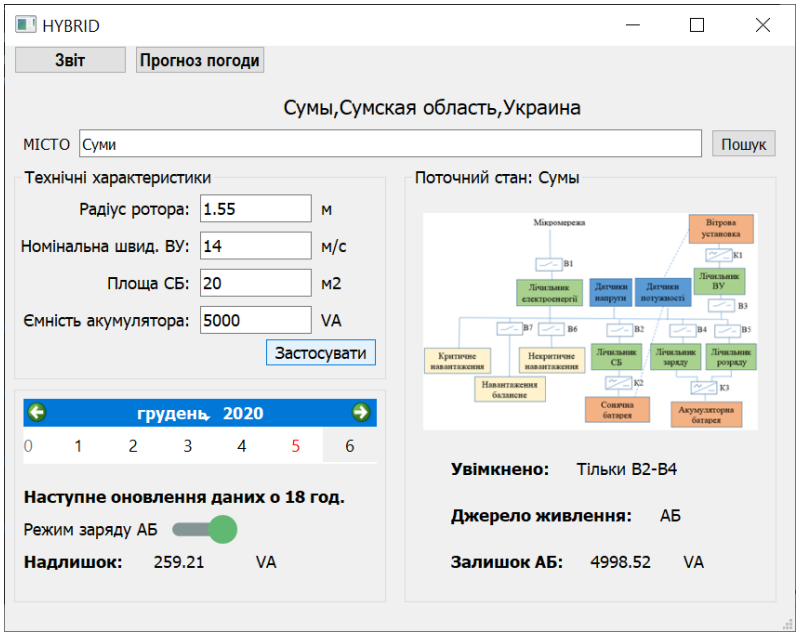

Рисунок В.4 – Результат виконання розрахунків

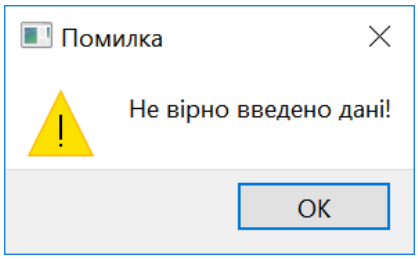

Рисунок В.5 – Виведення повідомлення про некоректно введені дані

Як результат користувачу відображається перелік увімкнутих перемикачів, залишок в акумуляторній батареї, джерело живлення:

– СБ – живлення від сонячних батарей;
– ВЕУ – живлення від вітрогенератора;
– АБ – живлення від акумуляторної батареї;
– СБ + ВЕУ – живлення від сонячних батарей та вітрогенератора;

– СБ + ВЕУ + АБ – живлення від сонячних батарей, вітрогенератора та акумуляторної батареї.

Якщо користувачу необхідно переглянути дані про погоду, таку потребу можна задовольнити, натиснувши кнопку "Прогноз погоди". В результаті цієї операції буде відкрито веб-сайт "Gismeteo.com", що містить дані про погоду, найближчі до місцезнаходження користувача (рис. В.6).

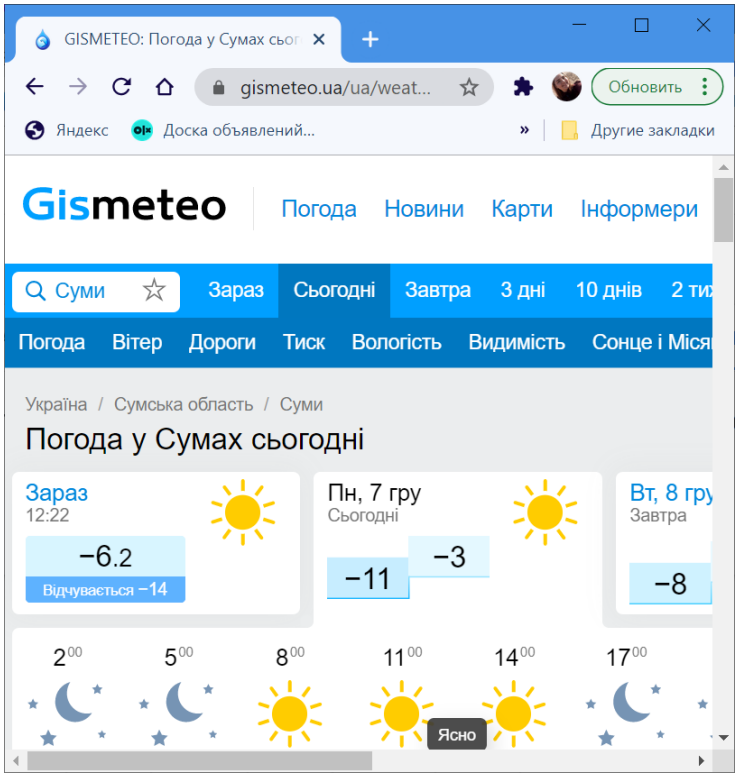

Рисунок В.6 – Веб-сайт Gismeteo.com

Отримані в результаті виконання розрахунків дані зберігаються в базі даних, якщо існує необхідність збереження даних в іншій формі, користувач має можливість зберегти дані у вигляді звіту, натиснувши кнопку «Звіт» (рис. В.7). У випадку успішного виконання даної операції, користувач отримує відповідне повідомлення (рис. В.8). Звіт буде збережено в форматі .docx в папці програмного продукту.

РЕЗУЛЬТАТИ·ЕЛЕКТРОСПОЖИВАННЯ¶

¶

Потужність·ВЕУ¤	10¤
Потужність·СБ¤	268¤
Ємність·акумулятора¤	4999¤
Рівень·споживання·електроенергії¤	149¤
Джерело·енергії¤	АБ¤
Передано·до·загальної·мережі¤	0¤

¶

Дата:2020-12-07¶

Рисунок В.7 – Збережений звіт з результатами рознахунків

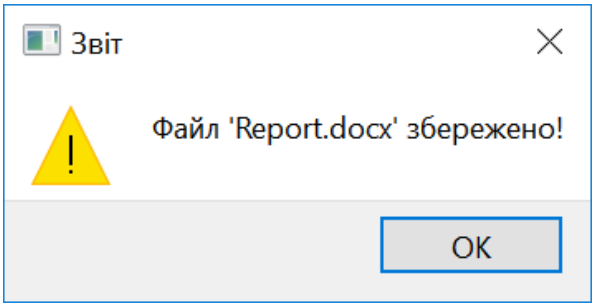

Рисунок В.8 – Виведення повідомлення про збереження звіту

Додаток С

Файли коду реалізації

Файл коду gparcer.py

```
import logging
import datetime

import requests
from lxml import html
import json

BASE_URL = 'https://www.gismeteo.ua/'

BASE_HEADERS = { 'User-Agent' : 'Mozilla/5.0 (X11; Linux x86_64; rv:64.0) Gecko/20100101 Firefox/64.0',
                 'Accept-Encoding' : 'gzip, deflate, br',
                 'Accept' : 'text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8',
                 'Accept-Language': 'en-US,en;q=0.5',
                 'Connection': 'keep-alive',}


def split_todatetime(row):
    arr = row.strip().split()
    if(arr[0] == 'Фактические'):
        return [arr[-2], arr[-1].replace('(UTC)', '')]
    else:
        d = arr[-2].split(':')[-1]
        t = arr[-1]
        return [d, t]

# extract full date
# return list contains 8 el (every 3h of day)  [['YYYY-mm-dd', 'hh:mm:ss(UTC)'], ...  x8 el]
def extract_datetimes(row):
    dates = []
```

```
            for el in row.xpath('./div/div/@title'):
                dates.append(split_todatetime(el))
            return dates


        # extract text description of weather ('Ясно')
        # return list contains 8 el (every 3h of day)
        def extract_weather(row):
            weathers = []
            for el in row.xpath('.//span/@data-text'):
                weathers.append(el.split(', ')[0])
            return weathers


        # extract temperature in C and F
        # return list contains 8 el (every 3h of day) [['temper C', 'temp F'], ... x8
el]
        def extract_temperature(row):
            temperatures = []
            for el in row.xpath(".//div[@class='value']"):
                temperatures.append(el.xpath('./span/text()'))
            return temperatures


        # extract temperature in m/s, m/h and km/h
        # return list contains 8 el (every 3h of day) [['m/s', 'm/h', 'km/h'], ... x8
el]
        def extract_wind(row):
            winds = []
            for wind in row:
                w = [ el.strip() for el in wind.xpath(".//span/text()")]
                w.append(wind.xpath(".//*[contains(@class,
'gray')]/text()")[0].strip())
                winds.append(w)
            return winds


        # parse html page to get info about weather
        def extract_winfo(page):
```

```
        widget_info_rows = page.xpath("//*[@class='forecast_frame
hw_wrap']//*[@class='widget__container']/*")[:3]
        dates = extract_datetimes(widget_info_rows[0])
        weathers = extract_weather(widget_info_rows[1])
        temperatures = extract_temperature(widget_info_rows[2])
        info = []
        widget_wind = page.xpath("//*[@data-widget-
id='wind']//*[contains(@class,'widget__row_wind')]/*")
        winds = extract_wind(widget_wind)
        info.extend(dates)
        for i in range(len(info)):
            info[i].append(weathers[i])
            info[i].extend(temperatures[i])
            info[i].extend(winds[i])

        return info


    # request info from gismeteo search
    # return dict with keys ('city', 'subDistrict', 'district', 'country', 'url')
    # raise Exception if cann`t find city
    def req_city_info(city, lang):
        url = 'https://www.gismeteo.ua/api/v2/search/searchresultforsuggest/'
        params = '/?lang' + lang + "&domain=ua"

        response = requests.get(url + city + params, headers=BASE_HEADERS)

        data = json.loads(response.text)
        if(data['total'] == 0): raise Exception('Error : no such city ' + suggest)

        res = {}
        res['id'] = data['items'][0]['id']
        res['city'] = data['items'][0]['name']
        res['district'] = data['items'][0]['district']['name']
        res['country'] = data['items'][0]['country']['name']
        res['url'] = data['items'][0]['url']

        return res
```

```
    # make reqest to url
    # return string with html
    def req_html(url, headers):
        response = requests.get(url, headers=headers)

        logging.debug('response status ' + str(response.status_code) + ' url: ' +
url)

        page = html.fromstring(response.text) # get html from response to parser
object
        return page

    def clean_data(data):
        for row in data:
            curr_date = datetime.datetime.strptime(row['date'], '%Y-%m-%d')
            wk_number = str(curr_date.isocalendar()[1])

            row['wkday'] = str(curr_date.weekday()+1)
            row['wknumber'] = wk_number

            del row['temperature_f']
            del row['wind_mh']
            del row['wind_kmh']



    # utils method
    def to_list_of_dict(keys, val_list):
        k = []
        for row in val_list:
            k.append(dict(zip(keys,row)))
        return k

    def weather(city):
        city_info = req_city_info(city, 'ua') # get info about city from gismeteo
search
        today_page = req_html(BASE_URL + city_info['url'], BASE_HEADERS)
        tomorrow_page = req_html(BASE_URL + city_info['url'] + '/tomorrow/',
BASE_HEADERS)
```

```
        weather_info = extract_winfo(today_page) # extract all info to list, each element is info about some period 00:00, 00:03, ... 21:00
        weather_info.pop(0) # remove measurement for previous day from dataset
        weather_info.append(extract_winfo(tomorrow_page)[0]) # add last measurement of day to dataset

        keys = ['date', 'time', 'weather', 'temperature_c', 'temperature_f', 'wind_ms', 'wind_mh', 'wind_kmh', 'wind_dir']
        data = to_list_of_dict(keys, weather_info) # put data to readable format as list of dictionaries
        clean_data(data)
        return data
```

Файл коду myconnutils.py

```
import logging

import pymysql.cursors
from PyQt5.QtWidgets import QMessageBox

logging.getLogger().setLevel(logging.DEBUG)

#the function returns connection
def get_connection():
    connection = pymysql.connect(host='127.0.0.1',
                                 user='mysql',
                                 password='mysql',
                                 db='hybrid',
                                 charset='utf8mb4',
                                 cursorclass= pymysql.cursors.DictCursor)
    return connection

def insert_city(connection, place):
    # Prepare SQL query to INSERT a record into the database.
    id = str(place['id'])
    sql_check =("SELECT number FROM place")
    cursor = execute_select(connection, sql_check)
    result = cursor.fetchall()
```

```
            n=0
            print(place)
            print(result)
            for x in range(len(result)):
                i=result[x]
                ID = i['number']
                if id == str(ID):
                    n+=1
            if n==0 :
                sql = """INSERT INTO place (city_name, country, number, region)
                                VALUES ('%s', '%s', '%s', '%s');"""
                data     =     (place['city'],     place['country'],     place['id'],
place['district'])
                execute_query(connection, sql, data)


        def insert_power(connection, power):
            # Prepare SQL query to INSERT a record into the database.
            sql = """INSERT INTO power (Ps, Pw, Pb, type_P, consumption, overflow,
weather_id)
                            VALUES ('%s', '%s', '%s', '%s', '%s', '%s', '%s');"""
            execute_query(connection, sql, power)


        def insert_weather(connection, weather, place_id):
            # Prepare SQL query to INSERT a record into the database.
            sql = """INSERT INTO weather (w_speed, w_direction, temperature, overcast,
dat_e, tim_e, place_id)
                            VALUES ('%s', '%s', '%s', '%s', '%s', '%s', '%s');"""
            data = (weather['wind_ms'], weather['wind_dir'], weather['temperature_c'],
weather['weather'], weather['date'], weather['time'], place_id)
            execute_query(connection, sql, data)


        def execute_query(connection, sql, values):
            cursor = connection.cursor()
            try:
               # Execute the SQL command
               cursor.execute(sql % values)
               # Commit changes in the database
```

```
        connection.commit()
    except:
        # Rollback in case there is any error
        #except Exception as e: print(e)
        connection.rollback()


def execute_select(connection, sql):
    cursor = connection.cursor()
    try:
        # Execute the SQL command
        cursor.execute(sql)
        # Commit your changes in the database
        connection.commit()
    except:
        # Rollback in case there is any error
        connection.rollback()
    return cursor
```

Файл коду mydesign.py

```
from PyQt5 import QtCore, QtGui, QtWidgets
from PyQt5.QtWidgets import QLCDNumber
from PyQt5.QtGui import QPixmap, QIcon
from PyQt5.QtCore import QSize, Qt, QTimer, QTime
from settings import Ui_Dialog
import webbrowser
import pymysql.cursors
import myconnutils
import urllib.request

class Ui_MainWindow(object):

    def openWindow(self):
        self.window = QtWidgets.QMainWindow()
        self.ui = Ui_Dialog()
        self.ui.setupUi(self.window)
        self.window.show()
```

```
def openWeb(self):
    webbrowser.open('https://www.gismeteo.ua/')

def showTime(self):
    time = QTime.currentTime()
    text = time.toString('hh:mm')
    if (time.second() % 2) == 0:
        text = text[:2] + ' ' + text[3:]
    self.display(text)

def setupUi(self, MainWindow):
    MainWindow.setObjectName("MainWindow")
    MainWindow.resize(850, 630)
    MainWindow.setMinimumSize(QtCore.QSize(850, 630))
    MainWindow.setBaseSize(QtCore.QSize(850, 630))

    self.centralwidget = QtWidgets.QWidget(MainWindow)
    self.centralwidget.setObjectName("centralwidget")

    self.lbl_main_city = QtWidgets.QLabel(MainWindow)
    self.lbl_main_city.setGeometry(QtCore.QRect(300, 50, 550, 30))
    self.lbl_main_city.setObjectName("lbl_main_city")
    font = QtGui.QFont()
    font.setPointSize(11)
    self.lbl_main_city.setFont(font)

    self.lbl_city = QtWidgets.QLabel(MainWindow)
    self.lbl_city.setGeometry(QtCore.QRect(20, 90, 50, 30))
    self.lbl_city.setObjectName("lbl_city")
    self.input_city = QtWidgets.QLineEdit(MainWindow)
    self.input_city.setGeometry(QtCore.QRect(80, 90, 670, 30))
    self.input_city.setObjectName("input_city")
    self.search_btn = QtWidgets.QPushButton(MainWindow)
    self.search_btn.setGeometry(QtCore.QRect(760, 90, 70, 30))
    self.search_btn.setObjectName("search_btn")

    self.groupBox1 = QtWidgets.QGroupBox(self.centralwidget)
    self.groupBox1.setGeometry(QtCore.QRect(10, 130, 400, 230))
    font = QtGui.QFont()
```

```
font.setPointSize(9)
self.groupBox1.setFont(font)
self.groupBox1.setObjectName("groupBox1")

self.lbl_veu_r = QtWidgets.QLabel(self.groupBox1)
self.lbl_veu_r.setGeometry(QtCore.QRect(70, 30, 120, 30))
self.lbl_veu_r.setObjectName("lbl_veu_r")
self.lbl_veu_n = QtWidgets.QLabel(self.groupBox1)
self.lbl_veu_n.setGeometry(QtCore.QRect(7, 70, 300, 30))
self.lbl_veu_n.setObjectName("lbl_veu_n")
self.lbl_sb_s = QtWidgets.QLabel(self.groupBox1)
self.lbl_sb_s.setGeometry(QtCore.QRect(100, 110, 120, 30))
self.lbl_sb_s.setObjectName("lbl_sb_s")
self.lbl_bt_e = QtWidgets.QLabel(self.groupBox1)
self.lbl_bt_e.setGeometry(QtCore.QRect(10, 150, 300, 30))
self.lbl_bt_e.setObjectName("lbl_bt_e")

self.input_veu_r = QtWidgets.QLineEdit(self.groupBox1)
self.input_veu_r.setGeometry(QtCore.QRect(200, 30, 120, 30))
self.input_veu_r.setObjectName("input_veu_r")
self.input_veu_n = QtWidgets.QLineEdit(self.groupBox1)
self.input_veu_n.setGeometry(QtCore.QRect(200, 70, 120, 30))
self.input_veu_n.setObjectName("input_veu_n")
self.input_sb_s = QtWidgets.QLineEdit(self.groupBox1)
self.input_sb_s.setGeometry(QtCore.QRect(200, 110, 120, 30))
self.input_sb_s.setObjectName("input_sb_s")
self.input_bt_e = QtWidgets.QLineEdit(self.groupBox1)
self.input_bt_e.setGeometry(QtCore.QRect(200, 150, 120, 30))
self.input_bt_e.setObjectName("input_bt_e")

self.lbl_m = QtWidgets.QLabel(self.groupBox1)
self.lbl_m.setGeometry(QtCore.QRect(330, 30, 16, 30))
self.lbl_m.setObjectName("lbl_m")
self.lbl_n = QtWidgets.QLabel(self.groupBox1)
self.lbl_n.setGeometry(QtCore.QRect(330, 70, 35, 30))
self.lbl_n.setObjectName("lbl_n")
self.lbl_m2 = QtWidgets.QLabel(self.groupBox1)
self.lbl_m2.setGeometry(QtCore.QRect(330, 110, 35, 30))
self.lbl_m2.setObjectName("lbl_m2")
self.lbl_wt = QtWidgets.QLabel(self.groupBox1)
```

```
self.lbl_wt.setGeometry(QtCore.QRect(330, 150, 70, 30))
self.lbl_wt.setObjectName("lbl_wt")

self.apply_btn = QtWidgets.QPushButton(self.groupBox1)
self.apply_btn.setGeometry(QtCore.QRect(270, 185, 120, 30))
self.apply_btn.setObjectName("apply_btn")
self.groupBox2 = QtWidgets.QGroupBox(self.centralwidget)
self.groupBox2.setGeometry(QtCore.QRect(10, 370, 400, 230))
font = QtGui.QFont()
font.setPointSize(9)
self.groupBox2.setFont(font)
self.groupBox2.setObjectName("groupBox2")
self.calendarWidget = QtWidgets.QCalendarWidget(self.groupBox2)
self.calendarWidget.setGeometry(QtCore.QRect(10, 10, 380, 70))
self.calendarWidget.setObjectName("calendarWidget")

self.lbl_time = QtWidgets.QLabel(self.groupBox2)
self.lbl_time.setGeometry(QtCore.QRect(10, 100, 380, 30))
font = QtGui.QFont()
font.setPointSize(9)
font.setBold(True)
font.setWeight(75)
self.lbl_time.setFont(font)
self.lbl_time.setObjectName("lbl_time")

self.lbl_switch = QtWidgets.QLabel(self.groupBox2)
self.lbl_switch.setGeometry(QtCore.QRect(10, 135, 380, 30))
self.lbl_switch.setObjectName("lbl_switch")

self.img_switch = QtWidgets.QLabel(self.groupBox2)
self.img_switch.setGeometry(QtCore.QRect(170, 135, 70, 31))
self.img_switch.setObjectName("img_switch")
pixmap2 = QPixmap('E:\\Python-projects\\Hybrid\\switch.png')
self.img_switch.setPixmap(pixmap2)

self.img_switch2 = QtWidgets.QLabel(self.groupBox2)
self.img_switch2.setGeometry(QtCore.QRect(170, 135, 70, 31))
self.img_switch2.setObjectName("img_switch2")
pixmap2 = QPixmap('E:\\Python-projects\\Hybrid\\switch().png')
self.img_switch2.setPixmap(pixmap2)
```

```
        self.img_switch2.setEnabled(0)

        self.lbl_over = QtWidgets.QLabel(self.groupBox2)
        self.lbl_over.setGeometry(QtCore.QRect(10, 170, 200, 30))
        font = QtGui.QFont()
        font.setPointSize(9)
        font.setBold(True)
        font.setWeight(75)
        self.lbl_over.setFont(font)
        self.lbl_over.setObjectName("lbl_over")

        self.lbl_over_res = QtWidgets.QLabel(self.groupBox2)
        self.lbl_over_res.setGeometry(QtCore.QRect(150, 170, 200, 30))
        self.lbl_over_res.setObjectName("lbl_over_res")
        self.lbl = QtWidgets.QLabel(self.groupBox2)
        self.lbl.setGeometry(QtCore.QRect(260, 170, 200, 30))
        self.lbl.setObjectName("lbl")

        self.groupBox = QtWidgets.QGroupBox(self.centralwidget)
        self.groupBox.setGeometry(QtCore.QRect(430, 130, 400, 470))
        font = QtGui.QFont()
        font.setPointSize(9)
        self.groupBox.setFont(font)
        self.groupBox.setObjectName("groupBox")

    #kVt\god labels
        #self.label = QtWidgets.QLabel(self.groupBox)
        #self.label.setGeometry(QtCore.QRect(300, 50, 75, 30))
        #self.label.setObjectName("label")
        #self.label_3 = QtWidgets.QLabel(self.groupBox)
        #self.label_3.setGeometry(QtCore.QRect(300, 125, 75, 30))
        #self.label_3.setObjectName("label_3")
        #self.label_2 = QtWidgets.QLabel(self.groupBox)
        #self.label_2.setGeometry(QtCore.QRect(300, 200, 75, 30))
        #self.label_2.setObjectName("label_2")
        #self.label_4 = QtWidgets.QLabel(self.groupBox)
        #self.label_4.setGeometry(QtCore.QRect(300, 275, 75, 30))
        #self.label_4.setObjectName("label_4")

    #results labels
```

```
#self.label_veu = QtWidgets.QLabel(self.groupBox)
#self.label_veu.setGeometry(QtCore.QRect(150, 50, 100, 30))
#self.label_veu.setObjectName("label_veu")
#self.label_sb = QtWidgets.QLabel(self.groupBox)
#self.label_sb.setGeometry(QtCore.QRect(150, 125, 100, 30))
#self.label_sb.setObjectName("label_sb")
#self.label_bt = QtWidgets.QLabel(self.groupBox)
#self.label_bt.setGeometry(QtCore.QRect(150, 200, 100, 30))
#self.label_bt.setObjectName("label_bt")
#self.label_cons = QtWidgets.QLabel(self.groupBox)
#self.label_cons.setGeometry(QtCore.QRect(150, 275, 100, 30))
#self.label_cons.setObjectName("label_cons")

self.lbl_onn = QtWidgets.QLabel(self.groupBox)
self.lbl_onn.setGeometry(QtCore.QRect(50, 310, 200, 30))
font = QtGui.QFont()
font.setPointSize(9)
font.setBold(True)
font.setWeight(75)
self.lbl_onn.setFont(font)
self.lbl_onn.setObjectName("label_onn")

self.lbl_source = QtWidgets.QLabel(self.groupBox)
self.lbl_source.setGeometry(QtCore.QRect(50, 360, 200, 30))
font = QtGui.QFont()
font.setPointSize(9)
font.setBold(True)
font.setWeight(75)
self.lbl_source.setFont(font)
self.lbl_source.setObjectName("label_5")

self.lbl_overflow = QtWidgets.QLabel(self.groupBox)
self.lbl_overflow.setGeometry(QtCore.QRect(50, 410, 175,30))
font = QtGui.QFont()
font.setBold(True)
font.setWeight(75)
self.lbl_overflow.setFont(font)
self.lbl_overflow.setObjectName("label_6")

self.img_vey = QtWidgets.QLabel(self.groupBox)
```

```
                self.img_vey.setGeometry(QtCore.QRect(20, 50, 75, 75))
                self.img_vey.setObjectName("img_vey")
                pixmap = QPixmap('E:\\Python-projects\\Hybrid\\schema.png')
                self.img_vey.setPixmap(pixmap)
                self.img_vey.resize(360, 235)

                self.lbl_onn_value = QtWidgets.QLabel(self.groupBox)
                self.lbl_onn_value.setGeometry(QtCore.QRect(185, 310, 150, 30))
                self.lbl_onn_value.setObjectName("lbl_onn_value")

                self.lbl_source_value = QtWidgets.QLabel(self.groupBox)
                self.lbl_source_value.setGeometry(QtCore.QRect(275, 360, 100, 30))
                self.lbl_source_value.setObjectName("lbl_source")

                self.lbl_overflow_value = QtWidgets.QLabel(self.groupBox)
                self.lbl_overflow_value.setGeometry(QtCore.QRect(200, 410, 100,30))
                self.lbl_overflow_value.setObjectName("lbl_overflow_value")

                self.lbl_ov_value = QtWidgets.QLabel(self.groupBox)
                self.lbl_ov_value.setGeometry(QtCore.QRect(300, 410, 100,30))
                self.lbl_ov_value.setObjectName("lbl_ov_value")
                self.horizontalLayoutWidget = QtWidgets.QWidget(self.centralwidget)
                self.horizontalLayoutWidget.setGeometry(QtCore.QRect(10, 0, 270, 30))
                self.horizontalLayoutWidget.setObjectName("horizontalLayoutWidget")
                self.horizontalLayout                                                    =
QtWidgets.QHBoxLayout(self.horizontalLayoutWidget)
                self.horizontalLayout.setContentsMargins(0, 0, 0, 0)
                self.horizontalLayout.setObjectName("horizontalLayout")

                #self.settingBtn = QtWidgets.QPushButton(self.horizontalLayoutWidget)
                #self.settingBtn.setEnabled(True)
                #self.settingBtn.setBaseSize(QtCore.QSize(140, 50))
                #font = QtGui.QFont()
                #font.setFamily("Arial Narrow")
                #font.setPointSize(10)
                #font.setBold(True)
                #font.setWeight(75)
                #self.settingBtn.setFont(font)
                #self.settingBtn.setObjectName("settingBtn")
                #self.settingBtn.clicked.connect(self.openWindow)
```

```
            #self.horizontalLayout.addWidget(self.settingBtn)
            self.reportBtn = QtWidgets.QPushButton(self.horizontalLayoutWidget)
            self.reportBtn.setBaseSize(QtCore.QSize(140, 50))
            font = QtGui.QFont()
            font.setFamily("Arial Narrow")
            font.setPointSize(10)
            font.setBold(True)
            font.setWeight(75)
            self.reportBtn.setFont(font)
            self.reportBtn.setObjectName("reportBtn")
            self.horizontalLayout.addWidget(self.reportBtn)

            self.weatherBtn = QtWidgets.QPushButton(self.horizontalLayoutWidget)
            self.weatherBtn.setBaseSize(QtCore.QSize(140, 50))
            font = QtGui.QFont()
            font.setFamily("Arial Narrow")
            font.setPointSize(10)
            font.setBold(True)
            font.setWeight(75)
            self.weatherBtn.setFont(font)
            self.weatherBtn.setObjectName("weatherBtn")
            self.weatherBtn.clicked.connect(self.openWeb)

            self.horizontalLayout.addWidget(self.weatherBtn)

            self.horizontalLayoutWidget_2 = QtWidgets.QWidget(self.centralwidget)
            self.horizontalLayoutWidget_2.setGeometry(QtCore.QRect(750, 0, 81, 35))

self.horizontalLayoutWidget_2.setObjectName("horizontalLayoutWidget_2")
            self.horizontalLayout_2                                          =
QtWidgets.QHBoxLayout(self.horizontalLayoutWidget_2)
            self.horizontalLayout_2.setContentsMargins(0, 0, 0, 0)
            self.horizontalLayout_2.setObjectName("horizontalLayout_2")
        #    self.UA_btn = QtWidgets.QPushButton(self.horizontalLayoutWidget_2)
        #    self.UA_btn.setBaseSize(QtCore.QSize(35, 35))
        #    font = QtGui.QFont()
        #    font.setFamily("Arial")
        #    font.setPointSize(8)
        #    font.setBold(True)
        #    font.setWeight(75)
```

```
#     self.UA_btn.setFont(font)
#     self.UA_btn.setLayoutDirection(QtCore.Qt.RightToLeft)
#     self.UA_btn.setObjectName("pushButton")
#     self.horizontalLayout_2.addWidget(self.UA_btn)
#     self.ENG_btn = QtWidgets.QPushButton(self.horizontalLayoutWidget_2)
#     self.ENG_btn.setBaseSize(QtCore.QSize(35, 35))
#     font = QtGui.QFont()
#     font.setFamily("Arial")
#     font.setPointSize(8)
#     font.setBold(True)
#     font.setWeight(75)
#     self.ENG_btn.setFont(font)
#     self.ENG_btn.setObjectName("pushButton_2")
#     self.horizontalLayout_2.addWidget(self.ENG_btn)
        MainWindow.setCentralWidget(self.centralwidget)
        self.statusbar = QtWidgets.QStatusBar(MainWindow)
        self.statusbar.setObjectName("statusbar")
        MainWindow.setStatusBar(self.statusbar)

        self.retranslateUi(MainWindow)
        QtCore.QMetaObject.connectSlotsByName(MainWindow)

    def retranslateUi(self, MainWindow):
        _translate = QtCore.QCoreApplication.translate
        MainWindow.setWindowTitle(_translate("MainWindow", "HYBRID"))

        self.search_btn.setText(_translate("MainWindow", "Пошук"))

        self.groupBox.setTitle(_translate("MainWindow", "Поточний стан"))
        #self.label.setText(_translate("MainWindow", "Вт/год"))
        #self.label_3.setText(_translate("MainWindow", "Вт/год"))
        #self.label_2.setText(_translate("MainWindow", "Вт/год"))
        #self.label_4.setText(_translate("MainWindow", "Вт/год"))

        self.lbl_time.setText(_translate("MainWindow",   "Наступне   оновлення
даних о "))
        #self.label_sb.setText(_translate("MainWindow", "Результат"))
        #self.label_bt.setText(_translate("MainWindow", "Результат"))
        #self.label_cons.setText(_translate("MainWindow", "Результат"))
        self.lbl_switch.setText(_translate("MainWindow", "Режим заряду АБ "))
```

```
            self.lbl_over.setText(_translate("MainWindow", "Надлишок: "))
            self.lbl_over_res.setText(_translate("MainWindow", "Результат"))
            self.lbl.setText(_translate("MainWindow", "VA"))

            self.lbl_onn.setText(_translate("MainWindow", "Увімкнено:"))
            self.lbl_source.setText(_translate("MainWindow", "Джерело живлення:"))
            self.lbl_overflow.setText(_translate("MainWindow", "Залишок АБ:"))
            self.lbl_city.setText(_translate("MainWindow", "МІСТО"))
            self.lbl_onn_value.setText(_translate("MainWindow", "Результат"))
            self.lbl_source_value.setText(_translate("MainWindow", "Результат"))
            self.lbl_overflow_value.setText(_translate("MainWindow", "Результат"))
            self.lbl_ov_value.setText(_translate("MainWindow", "VA"))
            #self.settingBtn.setText(_translate("MainWindow", "Зберегти"))
            self.reportBtn.setText(_translate("MainWindow", "Звіт"))
            self.weatherBtn.setText(_translate("MainWindow", "Прогноз погоди"))
            #self.UA_btn.setText(_translate("MainWindow", "UA"))
            #self.ENG_btn.setText(_translate("MainWindow", "ENG"))

            self.groupBox1.setTitle(_translate("MainWindow",             "Технічні
характеристики"))
            self.lbl_veu_r.setText(_translate("MainWindow", "Радіус ротора:"))
            self.lbl_veu_n.setText(_translate("MainWindow",    "Номінальна    швид.
ВУ:"))
            self.lbl_sb_s.setText(_translate("MainWindow", "Площа СБ:"))
            self.lbl_bt_e.setText(_translate("MainWindow", "Ємність "
      "акумулятора:"))
            self.lbl_m.setText(_translate("MainWindow", "м"))
            self.lbl_n.setText(_translate("MainWindow", "м/с"))
            self.lbl_m2.setText(_translate("MainWindow", "м2"))
            self.lbl_wt.setText(_translate("MainWindow", "VA"))
            self.apply_btn.setText(_translate("MainWindow", "Застосувати"))
            #self.lbl_main_city.setText(_translate("MainWindow",   "Суми,   Сумська
область, Україна"))

      if __name__ == "__main__":
          import sys
          app = QtWidgets.QApplication(sys.argv)
          MainWindow = QtWidgets.QMainWindow()
          ui = Ui_MainWindow()
          ui.setupUi(MainWindow)
```

```
    MainWindow.show()
    sys.exit(app.exec_())
```

Файл коду hybrid.py

```
import gparser
import myconnutils

import datetime
from time import sleep
import threading
import time

import xlrd
from openpyxl import load_workbook

from PyQt5 import QtGui,QtCore, QtWidgets
from PyQt5.QtGui import QPixmap
from PyQt5.QtWidgets import QMessageBox
from mydesign import *
from PyQt5 import uic

import sys

import pymysql.cursors
import pypyodbc
import webbrowser
from docxtpl import DocxTemplate

import numpy as np
import pandas as pd
import math

import os
import psutil


def report_create(Es, Ew, Eb, W, source, overflow):
```

```
            date = datetime.datetime.now().date()
            doc = DocxTemplate("Звіт.docx")
            context_Ew = { 'var_Ew' : str(Ew) , 'var_Es' : str(Es), 'var_Eb' : str(Eb)
, 'var_W' : str(W), 'var_Source' : str(source), 'var_Overflow' : str(overflow),
'var_date' : str(date)}
            doc.render(context_Ew)
            doc.save("Report.docx")

        def priority(Es, Ew, Eb, W, Eb_max):
            #start_time = time.time()

            result = ""
            if min(abs(W - Es), abs(W - Ew), abs(W - Eb_max), abs(W - Es - Ew), abs(W -
Es - Ew - Eb_max)) == abs(W - Es):
                Source = "СБ"
                Eb = Eb - W + Es
            elif  min(abs(W - Es), abs(W - Ew), abs(W - Eb_max), abs(W - Es - Ew), abs(W
- Es - Ew - Eb_max)) == abs(W - Ew):
                Source = "ВЕУ"
                Eb = Eb - W + Ew
                print (Eb)
            elif  min(abs(W - Es), abs(W - Ew), abs(W - Eb_max), abs(W - Es - Ew), abs(W
- Es - Ew - Eb_max)) == abs(W - Eb_max):
                Source = "АБ"
                Eb = Eb - W
            elif  min(abs(W - Es), abs(W - Ew), abs(W - Eb_max), abs(W - Es - Ew), abs(W
- Es - Ew - Eb_max)) == abs(W - Es - Ew):
                Source = "СБ + ВЕУ"
                Eb = Eb - W + Es + Ew
            else:
                Source = "СБ + ВЕУ + АБ"
                Eb = Eb - W + Es + Ew
            #print("--- %.9f seconds ---" % (time.time() - start_time))
            process = psutil.Process(os.getpid())
            print(process.memory_info().rss)  # in bytes
            return Source, Eb

        def Bon(s, P_sb, r, P_w, e, W):
            # розрахунок K2U, K0U визначаються за держстандартом з якості електроенергії
```

```
          UA = np.random.uniform(100.0, 320.0)
          UB = np.random.uniform(100.0, 320.0)
          UC = np.random.uniform(100.0, 320.0)

          UAB = np.random.uniform(223.0, 537.0)
          UAC = np.random.uniform(223.0, 537.0)
          UBC = np.random.uniform(223.0, 537.0)

          UMAX = max(UAB, UAC, UBC)
          UMIN = min(UAB, UAC, UBC)
          Umax2 = max(UA, UC, UB)
          Umin2 = min(UA, UC, UB)

          K2U = 0.62*(UMAX-UMIN)*100/380
          K0U = 0.62*(Umax2-Umin2)*100/220

          #потужність сонячної батареї Psb
          Psbmod = 34.237*s
          Psbmin = 11.495*s
          Psbmax = 64.793*s

          MsbL = max(0, min(1, ((Psbmod - P_sb)/(Psbmod - Psbmin))))
          MsbN = max(0, min(1, ((P_sb - Psbmin)/(Psbmod - Psbmin)), (Psbmod -
P_sb)/(Psbmax - Psbmod)))
          MsbH = max(0, min(1, (P_sb - Psbmod)/(Psbmax - Psbmod)))

          if max(MsbL, MsbN, MsbH) == MsbL:
               Msb = 1
          elif max(MsbL, MsbN, MsbH) == MsbN:
               Msb = 2
          else: Msb = 3

          #потужність вітроустановки PW
          Pwmin = 1.405*pow(r,2)
          Pwmod = 7.729*pow(r,2)
          Pwmax = 14.052*pow(r,2)

          MpwL = max(0, min(1, ((Pwmod - P_w)/(Pwmod - Pwmin))))
          MpwN = max(0, min(1, ((P_w - Pwmin)/(Pwmod - Pwmin)), (Pwmod - P_w)/(Pwmax
- Pwmod)))
```

```
MpwH = max(0, min(1, (P_w - Pwmod)/(Pwmax - Pwmod)))

if max(MpwL, MpwN, MpwH) == MpwL:
     Mpw = 1
elif max(MpwL, MpwN, MpwH) == MpwN:
     Mpw = 2
else: Mpw = 3

#ємність акумуляторної батареї PB
Pbmax = 5000

MpbL = max(0, min(1, (0.75*Pbmax - e)/(0.5*Pbmax)))
MpbH = max(0, min(1, (e - 0.25*Pbmax)/(0.5*Pbmax)))

if max(MpbL, MpbH) == MpwL:
     Mpb = 1
else: Mpb = 3

#поточна потужність електроспоживання W
Wmin = 0.54
Wmod = 6.49
Wmax = 31

MwL = max(0, min(1, ((Pwmod - W/100)/(Wmod - Wmin))))
MwN = max(0, min(1, ((W/100 - Wmin)/(Wmod - Wmin)), (Wmod - W/100)/(Wmax -
Wmod)))
MwH = max(0, min(1, (W/100 - Wmod)/(Wmax - Wmod)))

if max(MwL, MwN, MwH) == MwL:
     Mw = 1
elif max(MwL, MwN, MwH) == MwN:
     Mw = 2
else: Mw = 3

B1 = 2
B5 = 2
B6 = 2
B7 = 2
#вибір режиму роботи вимикачів B1, B5-B7
if UA<204 and UB<204 and UC<204:
```

```
            B1 = 1
            B5 = 1
            B6 = 1
            B7 = 1
        if UAB<351 and UAC<351 and UBC<351:
            B1 = 1
            B5 = 1
            B6 = 1
            B7 = 1
        if K2U>=3.5 and K0U>=3.5 and Msb == 1 and Mpw == 1 and Mpb == 1 and Mw ==
1:
            B1 = 1
            B5 = 1
            B6 = 1
            B7 = 1

        if UA>236 and UB>236 and UC>236:
            B1 = 1
            B5 = 1
            B6 = 1
            B7 = 1
        if UAB>409 and UAC>409 and UBC>409:
            B1 = 1
            B5 = 1
            B6 = 1
            B7 = 1
        if Msb == 3 and Mpw == 3 and Mpb == 3 and Mw == 3:
            B1 = 1
            B5 = 1
            B6 = 1
            B7 = 1

        if UA>=204 and UA<=236 and UB>=204 and UB<=236 and UC>=204 and UC<=236:
            B1 = 0
            B5 = 0
            B6 = 0
            B7 = 0
        if UAB>=351 and UAB<=409 and UAC>=351 and UAC<=409 and UBC>=351 and UBC<=409:
            B1 = 0
            B5 = 0
```

```
        B6 = 0
        B7 = 0
    if Msb == 2 and Mpw == 2 and Mpb == 2 and Mw == 2:
        B1 = 0
        B5 = 0
        B6 = 0
        B7 = 0

    B_on = ""
    if B1 == 1:
        B_on = B_on + " B1 "
    elif B5 == 1:
        B_on =  B_on + " B5 "
    elif B6 == 1:
        B_on =  B_on + " B6 "
    elif B7 == 1:
        B_on = B_on + " B7 "
    else: B_on = "Тільки B2-B4"

    process = psutil.Process(os.getpid())
    print(process.memory_info().rss)  # in bytes

    return B_on

A = {}
A[1] = np.array([[0,    0,    0],
                 [-0.058001325,     3.129664419,     -59.33981437],
                 [0.650827154, -37.25557963,     399.3108268]])
A[2] = np.array([[0,    0,    -16.48632757],
                 [0,    0,    84.01107552],
                 [0,    0,    531.0846902]])
A[3] = np.array([[-0.031114566,    2.021197585,     -90.76184864],
                 [0.068892225, -4.369347687,     355.3810179],
                 [0.812020337, -55.20939546,     1831.403134]])
A[4] = np.array([[0,    0,    0],
                 [0,    0,    0],
                 [0,    0,    0.00522]])
A[5] = np.array([[0,    0,    0],
                 [0,    0,    0],
                 [0,    0,    0.3966659]])
```

```
A[6] = np.array([[0.008553425,      -0.32813861,      -60.79002498],
                 [-0.235862408,     14.08462039,      125.6567786],
                 [1.054414961, -69.26651236,     1898.321959]])
A[7] = np.array([[0,    0,    0],
                 [0,    0,    0],
                 [0,    0,    0.015005432]])
A[8] = np.array([[0,    0,    0],
                 [0,    0,    0],
                 [0,    0,    0.55680151]])

B = {}
B[1] = np.array([[0,    0,    0],
                 [-0.058001325,     3.129664419,      -59.33981437],
                 [0.650827154, -37.25557963,     399.3108268]])
B[2] = np.array([[0,    0,    -16.48632757],
                 [0,    0,    84.01107552],
                 [0,    0,    276.9132438]])
B[3] = np.array([[-0.031114566,     2.021197585,      -90.76184864],
                 [0.068892225, -4.369347687,     355.3810179],
                 [0.812020337, -55.20939546,     1439.758948]])
B[4] = np.array([[0,    0,    0],
                 [0,    0,    0],
                 [0,    0,    0.00522]])
B[5] = np.array([[0,    0,    0],
                 [0,    0,    0],
                 [0,    0,    0.3966659]])
B[6] = np.array([[0.008553425,      -0.32813861,      -60.79002498],
                 [-0.235862408,     14.08462039,      125.6567786],
                 [1.054414961, -69.26651236,     1403.405653]])
B[7] = np.array([[0,    0,    0],
                 [0,    0,    0],
                 [0,    0,    0.015005432]])
B[8] = np.array([[0,    0,    0],
                 [0,    0,    0],
                 [0,    0,    0.55680151]])

C = {}
C[1] = np.array([[0,    0,    0],
                 [-0.058001325,     3.129664419,      -59.33981437],
                 [0.650827154, -37.25557963,     399.3108268]])
```

```
C[2] = np.array([[0,    0,    -16.48632757],
                 [0,    0,    84.01107552],
                 [0,    0,    770.4549488]])
C[3] = np.array([[-0.031114566,    2.021197585,     -90.76184864],
                 [0.068892225, -4.369347687,    355.3810179],
                 [0.812020337, -55.20939546,    2357.63803]])
C[4] = np.array([[0,    0,    0],
                 [0,    0,    0],
                 [0,    0,    0.00522]])
C[5] = np.array([[0,    0,    0],
                 [0,    0,    0],
                 [0,    0,    0.3966659]])
C[6] = np.array([[0.008553425,     -0.32813861,     -60.79002498],
                 [-0.235862408,     14.08462039,     125.6567786],
                 [1.054414961, -69.26651236,    2286.862064]])
C[7] = np.array([[0,    0,    0],
                 [0,    0,    0],
                 [0,    0,    0.015005432]])
C[8] = np.array([[0,    0,    0],
                 [0,    0,    0],
                 [0,    0,    0.55680151]])


def sourse(Vw, overcast, Eb, Ew, Es, W):
    if Vw < 5:
        V='Мала'
    elif Vw < 10:
        V = 'Середня'
    else: V = 'Велика'

    #---fuzzy rules
    if V == 'Мала' and overcast == 'Пасмурно':
        if Eb > W:
            Source = 'Eb'
        else: Source = 'Power Grid'

    if V == 'Середня' and overcast == 'Пасмурно':
        if Ew > W:
            Source = 'Ew'
        elif Eb > W:
```

```
        Source = 'Eb'
    else: Source = 'Power Grid'

if V == 'Велика' and overcast == 'Пасмурно':
    if Ew > W:
        Source = 'Ew'
    elif Eb > W:
        Source = 'Eb'
    else: Source = 'Power Grid'

if V == 'Мала' and overcast == 'Облачно':
    if Eb > W:
        Source = 'Eb'
    else: Source = 'Power Grid'

if V == 'Середня' and overcast == 'Облачно':
    if Ew + Es > W:
        Source = 'Ew + Es'
    elif Eb > W:
        Source = 'Eb'
    else: Source = 'Power Grid'

if V == 'Велика' and overcast == 'Облачно':
    if Ew > W:
        Source = 'Ew'
    elif Eb > W:
        Source = 'Eb'
    else: Source = 'Power Grid'

if V == 'Мала' and overcast == 'Малооблачно':
    if Ew + Es > W:
        Source = 'Ew + Es'
    elif Eb > W:
        Source = 'Eb'
    else: Source = 'Power Grid'

if V == 'Середня' and overcast == 'Малооблачно':
    if Ew + Es > W:
        Source = 'Ew + Es'
    elif Eb > W:
```

```
            Source = 'Eb'
        else: Source = 'Power Grid'

    if V == 'Велика' and overcast == 'Малооблачно':
        if Ew + Es > W:
            Source = 'Ew + Es'
        elif Eb > W:
            Source = 'Eb'
        else: Source = 'Power Grid'

    if V == 'Мала' and overcast == 'Ясно':
        if Es > W:
            Source = 'Es'
        elif Eb > W:
            Source = 'Eb'
        else: Source = 'Power Grid'

    if V == 'Середня' and overcast == 'Ясно':
        if Ew + Es > W:
            Source = 'Ew + Es'
        elif Eb > W:
            Source = 'Eb'
        else: Source = 'Power Grid'

    if V == 'Велика' and overcast == 'Ясно':
        if Ew + Es > W:
            Source = 'Ew + Es'
        elif Eb > W:
            Source = 'Eb'
        else: Source = 'Power Grid'
    #---------------
    return Source

def time_define():
    now = datetime.datetime.now().time()
    h = now.hour//3
    if h==0:
        t=0
    elif h==1:
        t=3
```

```
        elif h==2:
            t=6
        elif h==3:
            t=9
        elif h==4:
            t=12
        elif h==5:
            t=15
        elif h==6:
            t=18
        elif h==7:
            t=21
        return t

    def W_concumption(A, B, C, data):
        curr_date = datetime.datetime.strptime(str(data), '%Y-%m-%d')
        print (curr_date)
        n = float(curr_date.isocalendar()[1])
        d = float(curr_date.weekday()+1)
        #---------------
        k = time_define()
        t = k/24.0
        #---------------
        a1=A[1]
        a2=A[2]
        a3=A[3]
        a4=A[4]
        a5=A[5]
        a6=A[6]
        a7=A[7]
        a8=A[8]
        Wmod_f =
(a1[1,0]*d+a1[2,0])*n**2+(a1[1,1]*d+a1[2,1])*n+(a1[1,2]*d+a1[2,2])*t+a2[0,2]*d**2+a2
[1,2]*d+a2[2,2]
        Wmod_1 =
((a3[0,0]*d**2+a3[1,0]*d+a3[2,0])*n**2+(a3[0,1]*d**2+a3[1,1]*d+a3[2,1])*n+(a3[0,2]*d
**2+a3[1,2]*d+a3[2,2]))*math.exp(-((t-a5[2,2])**2)/a4[2,2])
        Wmod_2 =
((a6[0,0]*d**2+a6[1,0]*d+a6[2,0])*n**2+(a6[0,1]*d**2+a6[1,1]*d+a6[2,1])*n+(a6[0,2]*d
**2+a6[1,2]*d+a6[2,2]))*math.exp(-((t-a8[2,2])**2)/a7[2,2])
```

```
        b1=B[1]
        b2=B[2]
        b3=B[3]
        b4=B[4]
        b5=B[5]
        b6=B[6]
        b7=B[7]
        b8=B[8]
        Wmin_f                                                                                                  =
(b1[1,0]*d+b1[2,0])*n**2+(b1[1,1]*d+b1[2,1])*n+(b1[1,2]*d+b1[2,2])*t+b2[0,2]*d**2+b2
[1,2]*d+b2[2,2]
        Wmin_1                                                                                                  =
((b3[0,0]*d**2+b3[1,0]*d+b3[2,0])*n**2+(b3[0,1]*d**2+b3[1,1]*d+b3[2,1])*n+(b3[0,2]*d
**2+b3[1,2]*d+b3[2,2]))*math.exp(-((t-b5[2,2])**2)/b4[2,2])
        Wmin_2                                                                                                  =
((b6[0,0]*d**2+b6[1,0]*d+b6[2,0])*n**2+(b6[0,1]*d**2+b6[1,1]*d+b6[2,1])*n+(b6[0,2]*d
**2+b6[1,2]*d+b6[2,2]))*math.exp(-((t-b8[2,2])**2)/b7[2,2])

        c1=C[1]
        c2=C[2]
        c3=C[3]
        c4=C[4]
        c5=C[5]
        c6=C[6]
        c7=C[7]
        c8=C[8]
        Wmax_f                                                                                                  =
(c1[1,0]*d+c1[2,0])*n**2+(c1[1,1]*d+c1[2,1])*n+(c1[1,2]*d+c1[2,2])*t+c2[0,2]*d**2+c2
[1,2]*d+c2[2,2]
        Wmax_1                                                                                                  =
((c3[0,0]*d**2+c3[1,0]*d+c3[2,0])*n**2+(c3[0,1]*d**2+c3[1,1]*d+c3[2,1])*n+(c3[0,2]*d
**2+c3[1,2]*d+c3[2,2]))*math.exp(-((t-c5[2,2])**2)/c4[2,2])
        Wmax_2                                                                                                  =
((c6[0,0]*d**2+c6[1,0]*d+c6[2,0])*n**2+(c6[0,1]*d**2+c6[1,1]*d+c6[2,1])*n+(c6[0,2]*d
**2+c6[1,2]*d+c6[2,2]))*math.exp(-((t-c8[2,2])**2)/c7[2,2])

        #--------------
        W_mod=Wmod_f+Wmod_1+Wmod_2
        W_min=Wmin_f+Wmin_1+Wmin_2
```

```
    W_max=Wmax_f+Wmax_1+Wmax_2

    W=[W_min, W_mod, W_max]
    return W

def E_veu(tp, t, R, Vw, V):
    Q=0.00001661*tp**2-0.004764*tp+1.2924
    f=0.003869*Vw**2-0.128*Vw+6.650827154
    w=V/R
    Cp=0.351
    E=Q*Cp*Vw*math.pi*R**2
    return E

def E_sb(tp, Square, E, overcast, month):
    N1=0.0901 * E + 0.0873 * tp - 0.00032 * E * tp
    P_mod = Square*N1
    N2 = 0.0876*E + 0.0499*tp - 0.00027*E*tp
    P_min = N2*Square
    N3 = 0.0918*E + 0.1055*tp - 0.00035*E*tp
    P_max = N3*Square
    alfa = np.array([[0.6126, 0.8063,     0.0970,     0],
                     [0.6261,  0.8130,     0.0935,     0],
                     [0.5931,  0.7966,     0.1017,     0],
                     [0.6658,  0.8329,     0.0835,     0],
                     [0.7282,  0.8641,     0.0679,     0],
                     [0.7146,  0.8573,     0.0713,     0],
                     [0.7656,  0.8828,     0.0586,     0],
                     [0.7963,  0.8982,     0.0509,     0],
                     [0.7068,  0.8534,     0.0733,     0],
                     [0.6561,  0.8281,     0.0860,     0],
                     [0.5525,  0.7763,     0.1119,     0],
                     [0.5713,  0.7857,     0.1072,     0]])
    if overcast == 'Пасмурно':
        x = P_min + alfa[month, 0]*(P_mod - P_min)
    elif overcast == 'Облачно':
        x = P_min + alfa[month, 1]*(P_mod - P_min)
    elif overcast == 'Малооблачно':
        x = P_max - alfa[month, 2]*(P_max - P_mod)
    elif overcast == 'Ясно':
        x = P_max - alfa[month, 3]*(P_max - P_mod)
```

```
    return x

#||||||||||||||||||||||||||||||||||||||||||||||||

#windows works
class myMainWindow(QtWidgets.QMainWindow):

    def __init__(self, parent=None):
        QtWidgets.QWidget.__init__(self, parent)
        self.ui = Ui_MainWindow()
        self.ui.setupUi(self)
        self.ui.search_btn.clicked.connect(self.searchCity)
        self.ui.apply_btn.clicked.connect(self.applyData)
        self.ui.reportBtn.clicked.connect(self.create_report)
        #self.ui.UA_btn.clicked.connect(self.UA_text)
        #self.ui.ENG_btn.clicked.connect(self.ENG_text)

    def loaded_setData(self):
        city = self.ui.lbl_main_city.text()
        connection = myconnutils.get_connection()

    def create_report(self):
        e1=self.ui.label_veu.text()
        e2=self.ui.label_sb.text()
        e3=self.ui.label_bt.text()
        e4=self.ui.label_cons.text()
        e5=self.ui.lbl_source_value.text()
        e6=self.ui.lbl_overflow_value.text()
        report_create(e1, e2, e3, e4, e5, e6)
        msgBox = QMessageBox()
        msgBox.warning(self, 'Звіт ',"Файл 'Report.docx' збережено!")
    def searchCity(self):

        try:
            res=gparser.req_city_info(self.ui.input_city.text(), lang='ua')

            self.ui.groupBox.setTitle('Поточний стан: ' + res['city'])
            con = myconnutils.get_connection()
            myconnutils.insert_city(con, res)
```

```
                sql_check =("SELECT * FROM place WHERE number = %s")
                d = (str(res['id']), )
                cursor = con.cursor()
                cursor.execute(sql_check, d)
                l = cursor.fetchone()
                print(l)
                s =  ','.join([l['city_name'], l['region'], l['country']])
                self.ui.lbl_main_city.setText(s)

                weather_list = gparser.weather(l['city_name'])
                t = int(time_define()//3.0)
                weather = weather_list[t]

                myconnutils.insert_weather(con, weather, l['place_id'])
            except:
                msgBox = QMessageBox()
                msgBox.warning(self, 'Помилка ',"Не вірно введено назву міста")


        def applyData(self):

        #    try:
                connection = myconnutils.get_connection()
                cursor = connection.cursor()
                r = float(self.ui.input_veu_r.text())
                v = float(self.ui.input_veu_n.text())
                s = float(self.ui.input_sb_s.text())
                e = float(self.ui.input_bt_e.text())

                cit = self.ui.lbl_main_city.text()
                place = cit.split(',')
                city = place[0]
                t= time_define()
                if t<10:
                    time = '0'+str(t) + ':00:00'
                else: time = str(t) + ':00:00'
                sql  ="SELECT  w.w_speed,  w.temperature,  w.overcast,  w.dat_e,
w.tim_e, w.weather_id, p.city_name FROM weather AS w, place AS p WHERE w.place_id =
p.place_id AND p.city_name = '"
                a = sql+city+"' AND w.tim_e = '"+time+"' "
```

```
                print (time)
                cursor.execute(a)
                weth = cursor.fetchone()

                ins=pd.read_excel('ins.xlsx', index_col = 0)
                lst = list(ins.index)
                lst = [str(s) for s in lst]
                ins.index = lst

                n = int(t/3.0)
                dat = weth['dat_e']
                mounth = dat.strftime("%m")
                m = int(mounth)
                hour = weth['tim_e']
                h = str(hour) # '0'
                print (h, m)
                tp = int(weth['temperature'])
                over = weth['overcast']
                speed = float(weth['w_speed'])
                t_next = (t//3 + 1) * 3
                print(t_next)
                P_w = E_veu(tp, t, r, speed, v )
                P_sb = E_sb(tp, s, float(ins.loc[h,m]), over, m-1)# power of solar
pannels at current time
                W = W_concumption(A, B, C, dat)#electricity consumprion at current
time
                print(W)
                Prior = priority(P_sb, P_w, e, W[1]/100, 0)
                print (sys.getsizeof(Prior))
                source = Prior[0]
                pb = Prior[1]
                start_time = datetime.datetime.now()
                B_on = Bon(s, P_sb, r, P_w, e, W[1]/100)
                print(B_on)
                end_time = datetime.datetime.now()
                print('%.15f' %(end_time - start_time).seconds)
                over = P_sb + P_w - W[1]/100
                if over <= 0:
                     self.ui.img_switch2.setEnabled(1)
                     self.ui.img_switch.setEnabled(0)
```

```
                self.ui.lbl_onn_value.setText(B_on)
                self.ui.lbl_source_value.setText(source)
                self.ui.lbl_overflow_value.setText(str(round( pb, 2)))
                self.ui.lbl_over_res.setText(str(round( over, 2)))
                self.ui.lbl_time.setText("Наступне оновлення даних о " +
str(t_next) + " год.")

                w_id = weth['weather_id']
                power = (P_sb, P_w, pb, source, W[1], 0, w_id)
                myconnutils.insert_power(connection, power)

            except:
                msgBox = QMessageBox()
                msgBox.warning(self, 'Помилка ',"Не вірно введено дані!")
    class UpdateResults(threading.Thread):
        def __init__(self, label_w, label_v, label_s, label_e, label, label1,
label2, label3):
            super(UpdateResults, self).__init__()
            self.daemon = True
            self.labe_w = label_w
            self.label_v = label_v
            self.label_s = label_s
            self.label_e = label_e
            self.label = label
            self.label1 = label1
            self.label2 = label2
            self.label3 = label3
            self.start()

        def run(self):

            connection = myconnutils.get_connection()
            cursor = connection.cursor()
            cit = self.label.text()
            place = cit.split(',')
            city = place[0]
            print(city)
            t= time_define()-3.0
            s = str(t)
            s.split('.')[0]
```

```
            if t<10:
                time = '0'+s[0] + ':00:00'
            else: time = s[0] + ':00:00'
            sql ="SELECT p.Pb FROM weather AS w, power AS p, place AS pl WHERE
p.weather_id = w.weather_id  AND w.place_id = pl.place_id AND w.tim_e = '"
            a = sql+time+"' AND pl.city_name = '"+city+"' "
            print(a)
            cursor.execute(a)
            power = cursor.fetchone()
            print (power)
            t1= time_define()
            s1 = str(t1)
            s1.split('.')[0]

            if t1<10:
                time1 = '0'+ s1[0] + ':00:00'
            else: time1 = s1[0] + ':00:00'
            sql1 ="SELECT w.w_speed, w.temperature, w.overcast, w.dat_e, w.tim_e,
w.weather_id, p.city_name FROM weather AS w, place AS p WHERE w.place_id = p.place_id
AND p.city_name = '"
            a1 = sql1 + city + "' AND w.tim_e = '" + time1 + "' "
            cursor.execute(a1)
            weth = cursor.fetchone()

            ins=pd.read_excel('ins.xlsx', index_col = 0)
            lst = list(ins.index)
            lst = [str(s) for s in lst]
            ins.index = lst

            n = int(t1/3.0)
            dat = weth['dat_e']
            mounth = dat.strftime("%m")
            m = int(mounth)
            hour = weth['tim_e']
            h = str(hour) # '0'

            tp = int(weth['temperature'])
            over = weth['overcast']
            speed = float(weth['w_speed'])
```

```
            r = float(self.label_w.text())
            v = float(self.label_v.text())
            s = float(self.label_s.text())
            e = float(self.label_e.text())

            print(float(ins.loc[h,m]))
            P_w = E_veu(tp, t, r, speed, v )
            P_sb = E_sb(tp, s, float(ins.loc[h, m]), over, m-1)# power of solar
pannels at current time
            W = W_concumption(A, B, C, dat)#electricity consumprion at current time
            print (W)
            P_b = float(power['Pb'])

            Prior = priority(P_sb, P_w, P_b, W[1], P_b)
            source = Prior[0]
            pb = Prior[1]
            w_id = weth['weather_id']
            power = (P_sb, P_w, pb, source, W[1], 0, w_id)
            print (power)
            myconnutils.insert_power(connection, power)



            while True:

                #self.label1.setText(str(P_w))
                #self.label2.setText(str(P_sb))
                #self.label3.setText(str(P_b))
                self.label4.setText(str(W[1]))
                self.label5.setText(str(sour))
                self.label6.setText(str(P_send))
                self.lbl_over_res.setText(str(over))
                sleep(10800)


    if __name__ == '__main__':

        #open window
        app = QtWidgets.QApplication(sys.argv)
        Win = myMainWindow()
```

```
        up      =      UpdateResults(Win.ui.input_veu_r,      Win.ui.input_veu_n,
Win.ui.input_sb_s,  Win.ui.input_bt_e,  Win.ui.lbl_main_city,  Win.ui.lbl_onn_value,
Win.ui.lbl_source_value, Win.ui.lbl_overflow_value)
        Win.show()
        sys.exit(app.exec_())
```