НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Інститут прикладного системного аналізу
Кафедра штучного інтелекту

До захисту допущено:
В. О. завідувача кафедри
_____ Олена Чумаченко
«___» _____ 20__ р.

Дипломна робота
на здобуття ступеня бакалавра
за освітньо-професійною програмою «Системи і методи штучного інтелекту»
спеціальності 122 «Комп'ютерні науки»
на тему: «Прогнозування електрогенерації у MicroGrid»

Виконала:
студентка IV курсу, групи КІ-92
Соколова Дарина Анатоліївна _____

Керівник:
Старший викладач, Гуськова В. Г. _____

Консультант з економічного розділу:
доцент, к.е.н. Рощина Н. В. _____

Консультант з нормконтролю:
доцент, к.т.н. Гончарук М. М. _____

Рецензент:
професор кафедри ММСА д.т.н. Бідюк П. І. _____

Засвідчую, що у цьому дипломному проекті немає запозичень з праць інших авторів без відповідних посилань
Студент _____

Київ – 2023 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Інститут прикладного системного аналізу
Кафедра штучного інтелекту

Рівень вищої освіти – перший (бакалаврський)
Спеціальність – 122 «Комп'ютерні науки»
Освітня програма «Системи і методи штучного інтелекту»

ЗАТВЕРДЖУЮ
В. О. завідувача кафедри
_____ Олена ЧУМАЧЕНКО
«___» _____ 20___ р.

ЗАВДАННЯ
на дипломну роботу студенту
Соколовій Дарині Анатоліївні

1. Тема роботи «Прогнозування електрогенерації у MicroGrid», керівник роботи Гуськова Віра Геннадіївна, старший викладач кафедри ШІ, затверджені наказом по університету від «30» травня 2023 р. № 2065-с
2. Термін подання студентом роботи _____
3. Вихідні дані до роботи: технічна документація, теоретичні дані. Моделі сонячних панелей: LEAPTON LP-M-72-H-400W HALF-CELL – ККД 20%, максимальна потужність 400 Вт; Amerisolar AS-6P30 285W – ККД 17%, максимальна потужність 280 Вт; JA SOLAR JAM60S09-320W PR 5BB – ККД 19%, максимальна потужність 320 Вт; Моделі вітрогенераторів: Flamingo aero FA - 4. 4 – стартова швидкість 2,5 м/с, номінальна швидкість 8 м/с, максимальна потужність 1,6 кВт; Шторм 3 кВт –стартова швидкість 1,5 м/с, номінальна швидкість 8 м/с, максимальна потужність 3,4 кВт ; EW 2000 – стартова швидкість 2,5 м/с, номінальна швидкість 12 м/с, максимальна потужність 2 кВт
4. Зміст пояснювальної записки (перелік завдань, які потрібно розробити):
Розробка програми прогнозування електрогенерації в елементах системи Microgrid(сонячних панелей, вітрогенераторах).
5. Перелік ілюстративного матеріалу: презентація до захисту роботи, 3 теоретичні креслення.

6. Консультанти розділів роботи:

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Рощина Н. В.		

7. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1.	Консультація з дипломним керівником та огляд літератури	1.02.2023 – 1.04.2023	
2.	Дослідження системи MicroGrid	1.02.2023 – 16.04.2023	
3.	Дослідження роботи сонячних панелей	16.04.2023 – 30.04.2023	
4.	Дослідження роботи вітрогенераторів	30.04.2023 – 14.05.2023	
5.	Створювання графічного інтерфейсу програми	14.05.2023 – 26.05.2023	
6.	Вибір моделей альтернативних джерел енергії і розрахунок потужностей при різних погодних умовах	26.05.2023 – 30.05.2023	
7.	Написання коду для графічного інтерфейсу та підготовка економічної частини	30.05.2023 – 05.06.2023	
8.	Підготовка пояснювальної записки та презентації доповіді	05.06.2023 – 09.06.2023	

Студент ____ Дарина СОКОЛОВА
Керівник ____ Віра ГУСЬКОВА

А Н О Т А Ц І Я

Даний проект є актуальним оскільки було враховано проблему розподіленого електропостачання, яка виникає через нестабільність погодних умов. Було розроблено програму, що прогнозує електрогенерацію пристроїв на основі метеорологічних даних з використанням різних підходів аналізу даних. Аналіз роботи та характеристик вітрогенераторів та сонячних панелей дає можливість розрахувати вихідні коефіцієнти корисної дії при різних погодних умовах з використанням технічної документації альтернативних джерел енергії.

Зростаюча популярність альтернативної енергетики та прагнення до автономії та енергоефективності робить проект важливим. Альтернативні джерела енергії, такі як вітрогенератори та сонячні панелі, можуть допомогти зменшити залежність від традиційних джерел енергії і сприяти сталому розвитку. Прогнозування електрогенерації на основі метеорологічних даних дозволить ефективно керувати енергетичними потоками і забезпечувати стабільне електропостачання навіть при змінних погодних умовах.

Дана робота вирішує важливу проблему енергетики, і має потенціал стати важливим кроком до створення більш стійких, ефективних та екологічних систем енергопостачання.

Ключові слова: Прогнозування; Нейронні мережі; Математичні моделі; MicroGrid; Сонячна панель; Вітрогенератор; Енергоефективність;.

A N N O T A T I O N

During the course of my work, I have extensively explored various models and types of alternative renewable energy sources, the detailed study of wind generators and solar panels has provided valuable insights into their characteristics and performance. By analyzing the possible solutions to address the issue of energy subsidence caused by weather instability, you have developed software that incorporates meteorological data to predict the power generation of these devices. The calculation of initial efficiency factors using the technical documentation of power generation devices further enhances the accuracy of the predictions under different weather conditions.

My project holds great relevance in light of the increasing popularity of alternative energy sources and the growing emphasis on autonomy and energy efficiency. By harnessing renewable energy, I contribute to a more sustainable and environmentally friendly energy landscape. The development of software that enables accurate forecasting of power generation based on meteorological data is instrumental in efficient energy management and can help optimize the utilization of alternative energy sources. Overall, my project demonstrates a proactive approach towards meeting the evolving energy needs while reducing reliance on traditional energy sources.

Keywords : Forecasting; Neural Networks; Mathematical models; Microgrid; Solar panel; Wind power; Energy efficiency;.

ЗМІСТ

ВСТУП

Швидке зростання населення та промисловий розвиток, збільшення споживання електроенергії та водночас обмеженість джерел енергії є однією з центральних проблем сучасної енергетики.

Інноваційною відповіддю на нові виклики електроенергетики стала концепція MicroGrid, відносно невеликих систем децентралізованого виробництва електроенергії. Основним фактором створення та популяризації MicroGrid стало завдання забезпечення енергоефективності.

Завдання проекту полягало у розробці програми прогнозування вироблення електроенергії від вітрогенераторів та сонячних батарей. Завдання актуальне при розробці та проектуванні систем електропостачання з альтернативними та відновлюваними джерелами енергії.

Основним недоліком відновлюваної енергетики є те, що вона не є постійною, а залежить від кліматичних факторів та погодних умов. У безвітряні дні, наприклад, значно знижується енергія від вітрогенераторів, а дощові чи похмурі дні можна спостерігати значне зниження енергії від фотобатарей. Прогнозування обсягів генерації з урахуванням різних факторів дозволяє підвищити гнучкість системи управління та забезпечити безперебійне електропостачання відповідальних споживачів у мікромережі.

1. ЕЛЕКТРОГЕНЕРАЦІЯ У АЛЬТЕРНАТИВНИХ ДЖЕРЕЛАХ ЕЛЕКТРИЧНОЇ ЕНЕРГІЇ СИСТЕМИ MICROGRID

1.1 Опис системи MicroGrid

Вимоги до енергетики в 21-му столітті включають доступність, надійність, економічність, ефективність, екологічну гармонію та безпеку. Для розподілених енергетичних систем з низьким коефіцієнтом корисної дії особливо важливо підвищення якості та зменшення втрат електроенергії при розподіленій генерації та передачі до кінцевих споживачів.

Такі фактори, як подальше економічне зростання, збільшення обсягу електропотреби, високі вимоги до якості та надійності енергопостачання, негативний вплив електроенергетики на навколишнє середовище через використання традиційних технологій і проблеми зі створенням потужного енергетичного обладнання, призводять до необхідності розробки інтелектуальних енергетичних систем з розподіленою генерацією. Ці системи повністю інтегровані, саморегульовані та самовідновлювані, мають мережеву топологію і об'єднують джерела генерації, магістральні та розподільні мережі, а також різноманітних споживачів, що керуються єдиним системним управлінням у реальному часі.

Основні цілі, які досягаються при впровадженні інтелектуальних мереж, включають енергетичну безпеку, безперебійне постачання електроенергії, якість електричної енергії, енергоощадність та доступну ціну на електроенергію, а також мінімальний негативний вплив на навколишнє середовище [22].

Мікросистеми (MicroGrids) є локалізованими електромережами, що дозволяють інтегрувати розподілені ресурси генерації для задоволення електричних навантажень частково або повністю в об'єктах, що належать до мережі [1].

MicroGrid можуть функціонувати в режимі централізованого або децентралізованого управління. У централізованому режимі ключову роль в оптимізації MicroGrid відіграє центральний контролер MicroGrid (MGCC - MicroGrid Central Controler). За допомогою ціни на електроенергію, ціни на газ та інформації про безпеку, MGCC визначає, скільки енергії необхідно імпортувати зі спільної електричної мережі та скільки некритичних навантажень слід додати в критичних ситуаціях.

У децентралізованому режимі основною метою є максимізація виробництва електроенергії для задоволення потреб у навантаженні та експорту надлишкової електроенергії до спільної електричної мережі (рис. 1.1) [2, 3].

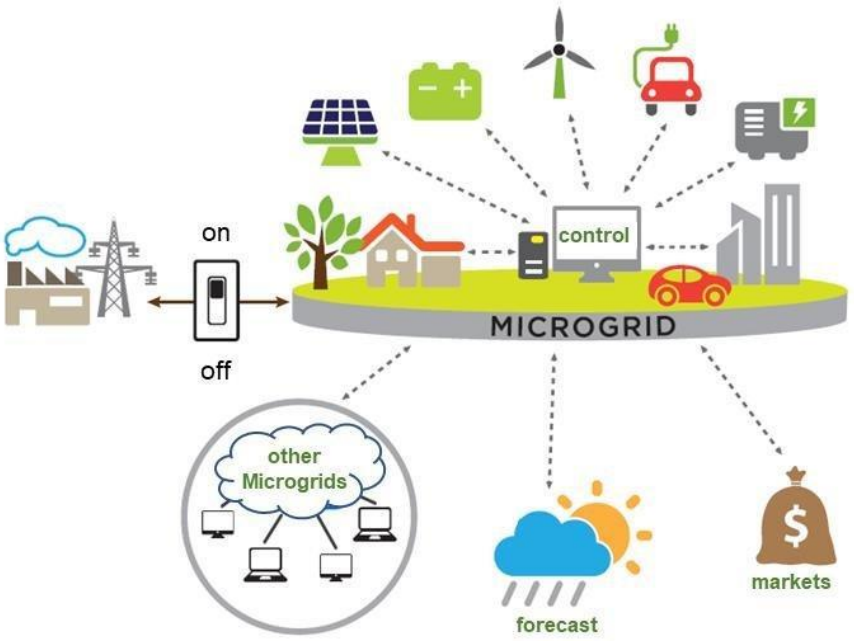


Рис.1.1. Концепція MicroGrid

На рис.1.2 показано протікання різних типів енергій в системі MicroGrid [3].

Червоним кольором позначено теплову енергію, вона генерується тепловими генераторами, такими як геліосистеми (Solar Heat на Рис.1.2), теплові насоси (Heat Exchanger на Рис.1.2) та інші.

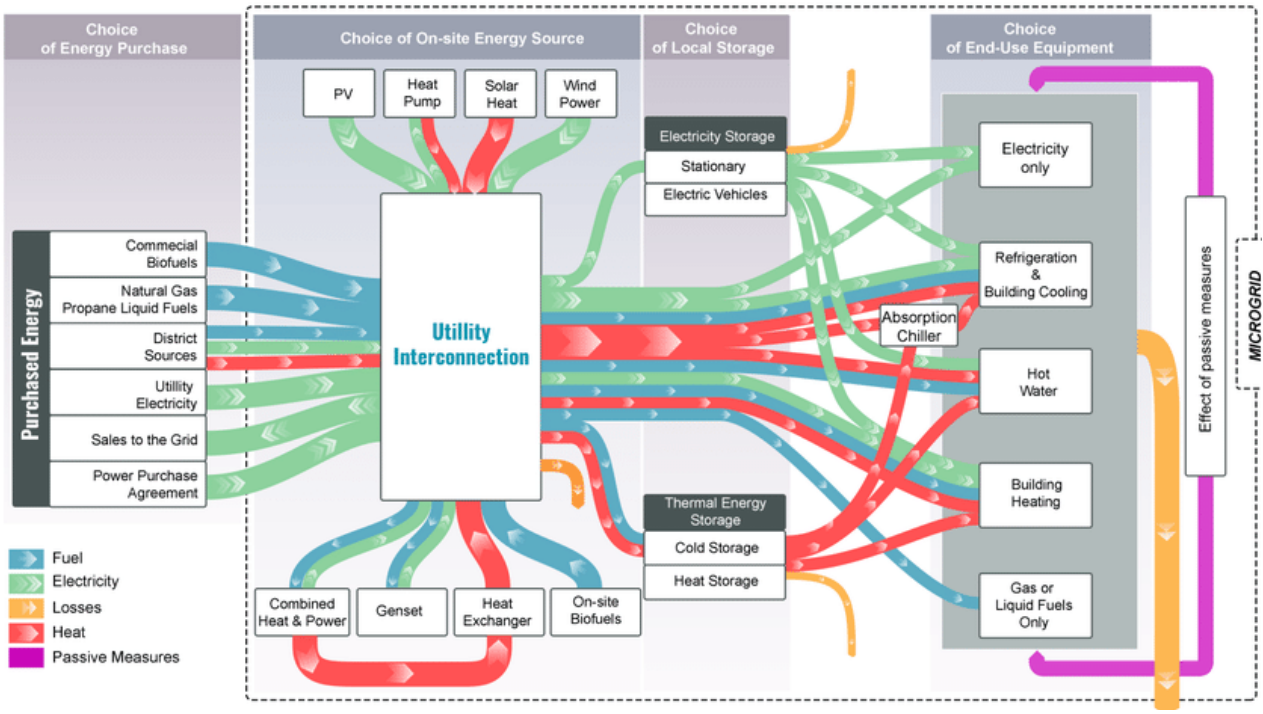


Рис.1.2. Протікання енергій в MicroGrid

Геліосистеми, або геліотермальні системи, використовують сонячну енергію для перетворення її на теплову. Вони здатні постачати тепло для опалення та нагріву води в побутових і промислових приміщеннях. Геліосистеми особливо ефективні для систем гарячого водопостачання в побутових будинках і мають широкі застосування у різних галузях, включаючи сільське господарство та харчову промисловість [4].

Теплові насоси використовуються для опалення, гарячого водопостачання та кондиціонування приміщень. Вони забирають тепло з навколишнього середовища, такого як ґрунт, вода або повітря, і перетворюють його в енергію для опалення. Тепловий насос є екологічно чистою альтернативою традиційним системам опалення і може суттєво знизити витрати на опалення та гарячу воду [5].

Обидва ці джерела енергії, геліосистеми і теплові насоси, можуть бути ефективно інтегровані в MicroGrid для генерації теплової енергії, що використовується для задоволення потреб у навантаженні та забезпечення комфорту в будинках і промислових приміщеннях.

Синім кольором позначена енергія, вироблена за рахунок генераторів, які працюють на біопаливі, такому як вугілля, нафта, торф та інші.

Біопаливо є відновлювальним джерелом енергії, оскільки воно походить від живих організмів. Використання біопалива у MicroGrid допомагає зменшити залежність від традиційних джерел енергії та зменшити викиди шкідливих речовин[6].

Помаранчевим кольором позначено протікання втрат енергії, які можуть статися під час переносу, зберігання та використання енергії в MicroGrid. Це включає ефективність конверсії енергії, втрати при передачі електричної енергії по проводам та інші фактори, які можуть спричинити зниження загальної ефективності системи.

Зеленим кольором позначено електроенергію, яка генерується за допомогою сонячних панелей (Photovoltaic (PV) на рис.1.2) і

вітрогенераторів(Wind Power на рис. 1.2). Сонячні панелі перетворюють сонячне випромінювання на електричну енергію, а вітрогенератори використовують силу вітру для генерації електроенергії. Обидва ці джерела є відновлювальними та незабруднюючими, що сприяє створенню екологічно сталих систем енергопостачання.

На рисунку 1.3 наведений приклад системи MicroGrid, яка використовує різнотипні відновлювальні джерела енергії та традиційні генератори. Ця система може функціонувати як частина більшої SmartGrid або як ізольована система [22]. Вона спроектована таким чином, щоб забезпечити якість напруги на навантаженні необхідного рівня та забезпечити надійне енергопостачання.

Для керування та забезпечення стійкої роботи MicroGrid використовуються різні джерела енергії та компоненти. Двигун внутрішнього згорання (ДВЗ) разом із синхронним генератором (СГ) відповідають за задавання частоти та рівня напруги в системі. Вітрогенератор (ВГ) та сонячна батарея (СБ) генерують певний рівень потужності в залежності від параметрів потоків первинної енергії. Паливний елемент (ПЕ) служить джерелом постійного струму, а акумуляторна батарея (АБ) виконує функцію накопичення електроенергії.

Для забезпечення оптимальної роботи системи і максимального використання відновлюваних джерел енергії, умова рівності еквівалентного внутрішнього опору джерел опору навантаження виконується за допомогою акумуляторної батареї (АБ). Це дозволяє використовувати максимальну енергію від відновлюваних джерел залежно від зовнішніх умов та навантаження[22].

Така структура MicroGrid дозволяє забезпечити електричне та теплове енергопостачання, а також забезпечує можливість керування та оптимізації системи в залежності від вхідних параметрів та вимог.

Ефективне використання електроенергії у MicroGrid досягається шляхом застосування різних стратегій та оптимізаційних підходів. Наведені

пункти вказують на основні методи та принципи, які допомагають забезпечити оптимальну роботу системи. Основні аспекти включають:

1. Оптимізація передачі електроенергії: Це включає мінімізацію втрат енергії при передачі від джерел генерації до навантажень, а також оптимізацію обміну енергією між вузлами розподіленої генерації. Це може включати використання ефективних трансформаторів та кабелів, розташування джерел генерації ближче до навантажень та інші стратегії передачі.

2. Вибір оптимальних режимів роботи генераторів: Кожне джерело генерації може мати різні режими роботи, які відповідають потребам системи. Це включає вибір оптимальної напруги, струму та потужності, які максимізують ефективність та задовольняють потреби навантажень.

3. Використання відновлюваних джерел енергії: Вітрогенератори та сонячні батареї можуть генерувати енергію залежно від зовнішніх умов, таких як швидкість вітру та сонячне випромінювання. Максимальна енергія повинна бути відібрана з цих джерел враховуючи поточні умови та витрати палива двигунів.

4. Ефективне управління накопичувачами електроенергії: Використання акумуляторних батарей дозволяє зберігати надлишкову електроенергію та використовувати її в періоди пікового навантаження. Оптимальний режим роботи накопичувача може бути обраний для мінімізації втрат енергії під час процесу заряду та розряду.

5. Забезпечення якості електроенергії: Важливо, щоб якість електроенергії відповідала вимогам споживачів. Це означає забезпечення стабільного рівня напруги, коефіцієнта гармонік та інших параметрів, які впливають на роботу споживачів та генераторів.

6. Ефективне управління та керування системою: Необхідно розробити систему керування, яка враховує зміну конфігурації мережі та забезпечує узгодженість між режимами роботи генераторів та навантажень.

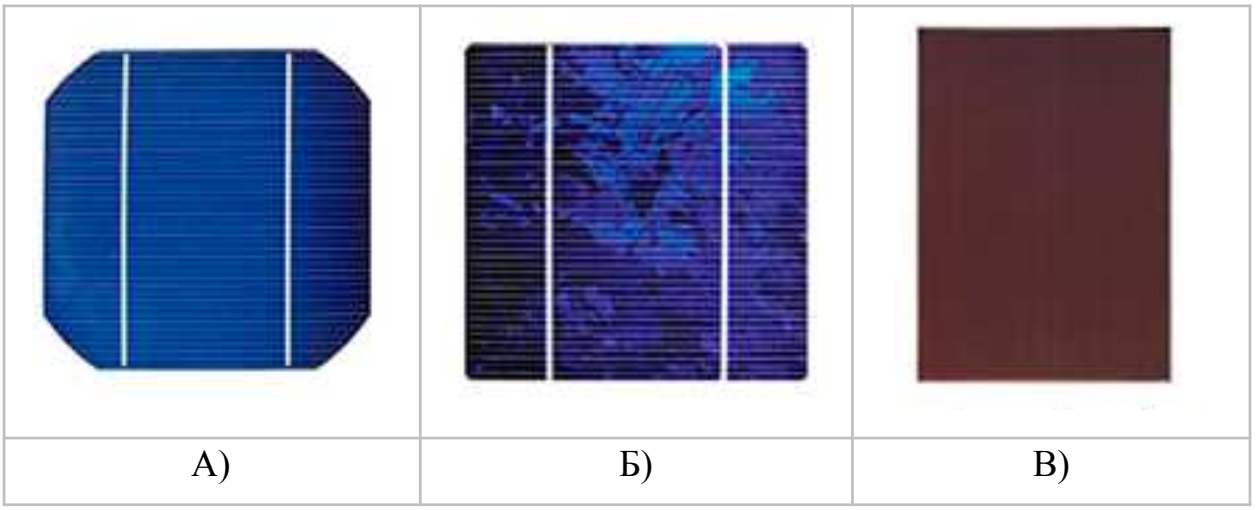
Це допомагає досягти максимальної ефективності та стійкості системи в цілому[22].

Ці підходи допомагають мінімізувати втрати електроенергії, забезпечити ефективне використання відновлюваних джерел енергії та забезпечити надійну та стабільну роботу системи MicroGrid.

1.2 Альтернативні джерела енергії

1.2.1 Сонячні панелі. Типи та режими роботи

Фотоелектричні перетворювачі існують різні (рис.1.5) [19].



А) Б) В)

Рис.1.3. Зовнішній вигляд комірки фотоелектричних перетворювачів:
А)Монокристалічний- для виготовлення пластини використаний один монокристал;

Б) Полікристалічний - для виготовлення пластини використано полікристал;

В)Аморфний(тонкоплівковий) – порошкоподібну форму елементу напиляють на панель.

Різноманітність сонячних перетворювачів залежить від матеріалів, використовуваних для їх виготовлення, а також від технологій, які використовуються. Продуктивність цих перетворювачів безпосередньо

залежить від цих факторів. Деякі фотоелементи мають коефіцієнт корисної дії (ККД) від 5% до 7%, тоді як нові розробки показують ККД 44% і вище. Звичайно, розробка цих технологій для побутового використання вимагає значних зусиль і коштів. Проте, майбутнє обіцяє багато нових можливостей. Для поліпшення характеристик сонячних перетворювачів використовуються інші рідкоземельні метали, але це призводить до зростання вартості. Загалом, середня продуктивність доступних сонячних перетворювачів становить приблизно 20-25%.

Найпоширенішим типом сонячних батарей є кремнієві батареї, які виробляються з використанням недорогого напівпровідника і мають довгу історію виробництва. Проте їх ККД не є найвищим і становить від 20% до 25%. Тому на сьогоднішній день переважно використовуються три типи сонячних перетворювачів:

1. Тонкоплівкові батареї - найдешевший варіант, які представляють собою тонкий шар кремнію на несучому матеріалі з захисною плівкою. Вони працюють навіть при розсіяному світлі, що дозволяє встановлювати їх на стінах будівель. Однак їх ефективність становить лише 7-10%, і вони піддаються деградації з часом, незважаючи на захисний шар. Тим не менш, завдяки їх великій площі можна отримувати електрику навіть в хмарну погоду.

2. Полікристалічні сонячні батареї виготовляються з розплавленого кремнію, який повільно охолоджується. Їх можна впізнати за яскраво-синім кольором. Ці батареї мають кращу продуктивність з ККД від 17% до 20%, але менш ефективні при розсіяному світлі.

3. Монокристалічні сонячні батареї є найдорожчими, але вони широко поширені. Вони виготовляються шляхом розділення одного кристала кремнію на пластини і мають характерну геометрію зі скошеними кутами. ККД цих батарей становить від 20% до 25%[19].

Сучасні тонкоплівкові панелі на основі різноманітних хімічних елементів краще сприймають промені, що падають під великим кутом, а

також розсіяне світло. Вплив погодних умов, таких як дощ, також знижує ефективність сонячних батарей. У похмуру погоду, ККД зменшується на 10-25%, залежно від щільності хмар. Тінь від об'єктів, таких як дерева, вежі, стовпи або сусідні будівлі, також негативно впливає на продуктивність. У зимовий період, ККД сонячних батарей залежить від рівня освітленості більше, ніж від температури повітря або кількості снігу. Єдиним вирішальним фактором є нічний час, коли без сонячної енергії виробництво електрики припиняється, і навіть сонячні панелі з високим ККД залишаються без роботи, тоді власники змушені підключатися до електричної мережі або використовувати акумулятори [7].

Залежно від рівня енергетичного забезпечення приміщення, можна виділити наступні режими використання енергосистеми (рис.1.6): повний, комфортний, помірний, базовий і аварійний [20].

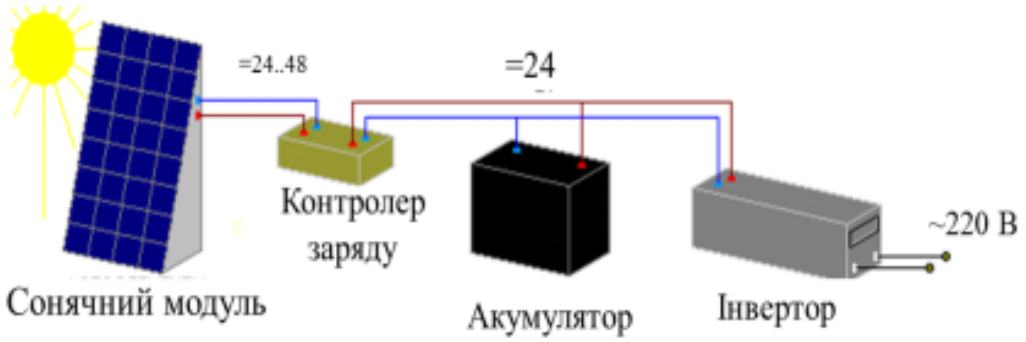


Рис.1.4. Схематичний вигляд фотоелектричної системи автономного електрозабезпечення

Кожен з вказаних режимів має вимоги, що стосуються миттєвої потужності та енергетичного резерву системи, що впливають на вартість її встановлення.

Повне енергозабезпечення передбачає заміну постачання електроенергії від електричних мереж на енергію, що генерується сонячними батареями. Це досягається шляхом вибору відповідної потужності фотоелектричної системи, яка може задовольнити максимальні потреби господарства в енергії та виключити необхідність підключення до зовнішньої електричної мережі.

Для повного відключення від мережі, при цьому не змінюючи спосіб життя сім'ї, потрібна система, яка може виробляти не менше 600 кВт-год електроенергії протягом місяця з тривалою потужністю не менше 5 кВт, а споживання енергії протягом доби може сягати 50 кВт-год з середньодобовим значенням від 10 до 20 кВт-год.

У комфортному режимі енергозабезпечення енергетична система повинна забезпечувати живлення низькопотужних і середньопотужних приладів (<4 кВт), тоді як енергозатратні прилади (електричні плити, духовки, конвектори та підігрівачі великих площ) мають отримувати електроенергію з зовнішньої електричної мережі.

Помірне енергозабезпечення передбачає комфортний режим енергозабезпечення з більш раціональним використанням великопотужного обладнання. Таким чином, енергозатратні роботи слід проводити в періоди максимального сонячного випромінювання до фотоелектричної системи, а живлення їх здійснюється за гібридною схемою з використанням як зовнішньої електричної мережі, так і накопиченої енергії у системі акумуляторів.

Базовий та аварійний режими характеризуються постійним живленням від зовнішньої електричної мережі з використанням фотоелектричної системи для живлення низькопотужних приладів або в ситуаціях аварії, коли зовнішня електрична мережа відключена[20].

Відбір максимальної енергії від сонячної батареї. При задоволенні умов вибору максимальної енергії, сонячна батарея(СБ) працює на своїй точці максимальної потужності.

Зміна зовнішніх умов та режиму навантаження вимагає вибору середнього значення струму широтно-імпульсного перетворювача, який підключений до виходу сонячної батареї, та амплітуди пульсації його змінної складової. Значення цих параметрів регулюються струмом короткого замикання сонячної батареї.

Наявність несиметричної пульсації потужності вимагає зміни значень струму та амплітуди пульсацій, щоб регулювати потужність сонячної батареї в діапазоні від нуля до одиниці. При цьому існує безліч режимів роботи сонячної батареї, які забезпечують однакову вихідну потужність, шляхом комбінування величини струму та амплітуди пульсацій навколо точки вибору максимальної потужності.

Існують інші методи вибору максимальної енергії з сонячної батареї, які базуються на використанні відомих схем включення та стабілізації напруги при послідовному або паралельному з'єднанні. Незалежно від використаного методу, важливим компонентом системи є широтно-імпульсний перетворювач, який виконує роль регулятора або стабілізатора напруги, струму або потужності[22].

Відомо, що режим відбору максимальної електричної потужності з джерела енергії досягається, коли його вихідний опір $r_{вих}$ відповідає опору навантаження R_H

$r_{вих} = R_H$.	(1.1)

При зміні умов навколишнього середовища та режиму роботи навантаження, ця умова (1.1) порушується, і енергія, отримана від джерела, стає меншою, ніж максимально можлива. У таких випадках навантаження підключається до джерела електричної енергії за допомогою узгоджувального пристрою, який часто є імпульсним регулятором постійної напруги. Оскільки вихідна потужність відновлювальних джерел електричної енергії(ВДЕЕ) залежить від умов навколишнього середовища, для забезпечення потрібного обсягу енергії до навантаження використовується енергетичний акумулятор.

За заданих умов навколишнього середовища ВДЕЕ можуть забезпечити певну максимально можливу вихідну потужність P_{MM} . Проте, при передачі електричної енергії до навантаження в системі живлення виникають втрати

енергії. Крім того, ці компоненти впливають на режими роботи відновлювальних джерел електричної енергії, що призводить до зміни вихідної потужності відносно максимально можливої точки потужності (ММП). У результаті, енергія, що надходить до навантаження, стає меншою, ніж максимально можлива $W_{MM} = P_{MM} \cdot t$. Оскільки головною метою таких систем є отримання максимальної кількості енергії в навантаженні, є доцільним проаналізувати джерела додаткових втрат, оцінити їх розмір та запропонувати способи їх зменшення. Давайте розглянемо ці питання на прикладі відновлювального джерела електричної енергії, такого як сонячна батарея.

Відомо, що при виконанні умови (1.1) сонячна батарея (СБ) працює в точці максимальної потужності, з координатами $I_{МП}$, $U_{МП}$ (рис. 1.7), де $I_{кз}$ - струм короткого замикання; U_{xx} - напруга холостого ходу [29].

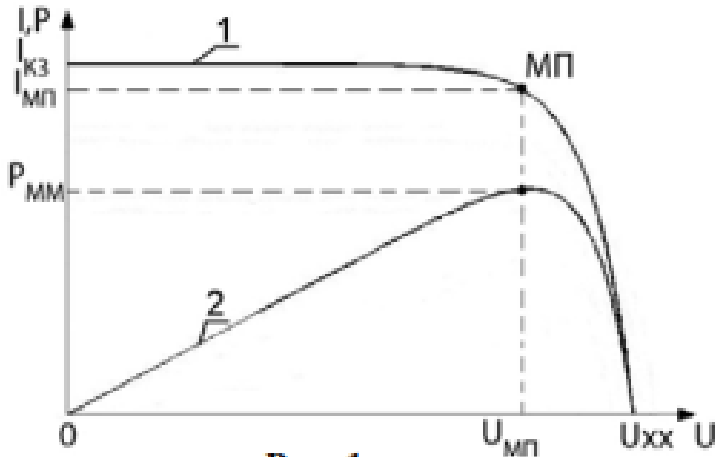


Рис. 1.5. Вольт-амперна характеристика (ВАХ) сонячної батареї

При заданих умовах можна одержати ММП P_{MM} та максимально можливу енергію. Однак, при зміні зовнішніх умов та режиму роботи навантаження для забезпечення виконання умови (1.1) застосовується узгоджувальний імпульсний регулятор (ІР).

Оскільки ІР працює в імпульсному режимі, його вхідний струм (який є вихідним струмом СБ) має пульсуючий характер. Це призводить до коливань робочої точки СБ навколо точки максимальної потужності (МП). В результаті енергія, отримана від СБ, буде меншою, ніж W_{MM}. Очевидно, для мінімізації недоотриманої енергії необхідно зменшувати пульсацію струму ΔI, що споживається від СБ.

Введемо поняття коефіцієнт використання електричної енергії СБ: 1 – ВАХ СБ, 2 – крива потужності СБ :

$$\eta = W_{вих} / W_{MM},$$

де $W_{MM} = P_{MM} \cdot t$ – кількість електричної енергії, яку може віддати СБ протягом одного періода Т роботи ІР при споживанні від неї постійного струму $I_{МП}$; $W_{вих}$ – реальна кількість одержаної електричної енергії на виході СБ [29].

1.2.2 Вітрогенератор. Типи та принцип дії

Використання вітроенергії стає все більш важливим елементом енергетичних систем багатьох країн, і зростає кількість автономних вітрових систем електропостачання. Однак виробництво електроенергії з вітрових установок супроводжується рядом проблем, пов'язаних зі змінністю вітру та невизначеністю значень [21].

Сучасні вітрогенератори обертаються з оптимальною частотою, щоб максимально використовувати енергію вітру, і використовують спеціальні перетворювачі або генератори з подвійним збудженням, щоб отримати електричну енергію при номінальній частоті змінного струму. Зміна кута лопастей може впливати на зібрану вітрогенератором потужність, проте, для ефективного використання доступної енергії, кут лопастей зазвичай регулюється для досягнення максимальної виходу потужності. Таким чином,

вітрогенератори, ймовірно, не є оптимальними для регулювання частоти в періоди великого навантаження системи.

Крім того, вітрогенератори можуть бути неефективними для регулювання частоти під час високого навантаження системи, оскільки їх потужність залежить від поривів вітру, що створює додаткове навантаження для джерел, які регулюють свою потужність для збалансування навантаження та регулювання електричної частоти. Отже, встановлена потужність вітряної турбіни в MicroGrid може бути обмежена або потребувати великих систем накопичування енергії, наприклад, гідроакумуляторів, для забезпечення балансу між навантаженням і виробництвом електроенергії та регулювання частоти системи. Крім того, можна використовувати компактні постійного струму конденсатори у перетворювачах, щоб накопичувати енергію між періодами зміни сили вітру і сприяти обмеженню відхилень частоти після виникнення збоїв у системі, до того моменту, коли інші джерела енергії зможуть реагувати [23]. Генератори вітроенергетичних установок мають схожу конструкцію з генераторами, що використовуються на традиційних електростанціях, які працюють на основі спалювання копалинного палива. На даний момент існують два типи вітрогенераторів (див. рис. 1.8) [10].

	
Горизонтальний	Вертикальний
Рис.1.6. Типи вітроелектроустановок	

У ряді країн з 1940-х початку 1960-х років активно проводилися дослідження у сфері вітроенергетики. Однак зниження вартості викопного

палива призвело до того, що вітроелектричні установки втратили конкурентоспроможність порівняно з тепловими електростанціями з точки зору вартості виробленої електроенергії. В цей період комерційний розвиток вітроенергетики був мінімальним. Було створено різні експериментальні прототипи [26].

Початок 1970-х років приніс нову хвилю розвитку вітроенергетики, спричинену енергетичною кризою. У цей період уряди багатьох країн запустили масштабні програми для створення вітроелектростанцій. Відповідно до цих програм, розробка технологій, проектні роботи та експериментальні дослідження проводилися в постійному взаємодії. Плани передбачали створення трьох поколінь експериментальних систем. Перше покоління дало базові даних та інформацію для формування чіткої концепції вітроелектростанцій. Друге покоління було спрямоване на набуття практичного досвіду у створенні вітроелектростанцій. Нарешті, третє покоління повинно було підтвердити надійність та ефективність вітроелектричних установок для комерційного використання. Ці серії систем мали вдосконалити вітротехнології до такого рівня, коли технічний ризик стане достатньо невеликим, щоб привернути значні комерційні інвестиції. Дослідження були розподілені на два напрямки: вітроелектростанції з горизонтальною віссю обертання та вітроелектростанції з вертикальною віссю обертання [25].

Вітроелектричний генератор з горизонтальною віссю обертання, найпоширеніший тип вітроустановок, має дві або три лопаті, розташовані на вершині вежі. В основі такої турбіни з горизонтальною віссю обертання знаходиться горизонтально розташований ведучий вал ротора. Залежно від напрямку повітряного потоку, ротор турбіни може перебувати перед опорою, в цьому випадку його називають "навітряним ротором", або за опорою, коли його називають "підвітряним ротором". Зазвичай турбіни з горизонтальною віссю обертання мають дві або три лопаті, але існують моделі з більшою

кількістю лопатей. Останні вітрогенератори представляють собою диск з численними лопатями і отримали назву "монолітних" установок [10].

Зазначене обладнання (див. рис.1.9) [9] працює за принципом, коли лопаті встановлюються у рух під впливом сили вітру.

Рис.1.7. Розріз корпусу вітрогенератора

Цей принцип вітряної електростанції спричиняє початок обертання турбіни у вітрогенераторі. Коли турбіна починає рухатися, вона генерує енергію, яка залежить від сили вітру. Зі зростанням вітрової енергії збільшується і механічна енергія, що виробляється турбіною.

У вітрогенераторах можуть бути різні конфігурації, залежно від наявності мультиплікатора на роторі. Якщо мультиплікатор передбачений, то енергія передається йому від турбіни. Функція мультиплікатора полягає в прискоренні обертання вала. Установки без цього обладнання є більш ефективними, оскільки не потребують додаткової енергії для прискорення обертання вала. Таким обладнанням достатньо вітрової енергії для повноцінної роботи.

Принцип роботи вітроелектростанції дозволяє отримувати електроенергію альтернативним шляхом та забезпечує автономність об'єктів. Потужність вітрогенератора напряму залежить від розмірів його лопатей.

Більша площа лопатей дозволяє отримати вищу потужність, використовуючи принцип роботи вітроустановки [9].

У вітрогенератора вектор швидкості вітру, спрямований на вісь обертання, залежить від косинуса кута атаки, і за малих змін кута при постійній швидкості обертання внутрішній опір вітрогенератора можна вважати лінійною функцією часу:

$$r(t) = k_R \cdot t,$$

де k_R – коефіцієнт масштабування, який має розмірність $[\frac{Ом}{c}]$.

Оскільки для роботи вітрогенератора в режимі відбору максимальної потужності необхідно, щоб виконувалася умова :

$$r(t) = R_{EKB}(t),$$

то еквівалентний опір навантаження має змінюватися за таким же лінійним законом, що й внутрішній:

$$R_{EKB}(t) = k_R \cdot t$$

У цьому випадку, нехтуючи внутрішнім опором транзистора, пульсаціями струму в індуктивності $L_ф$ та пульсаціями напруги на ємності $C_ф$, рівняння RL -кола спрощеної еквівалентної схеми запишеться у вигляді:

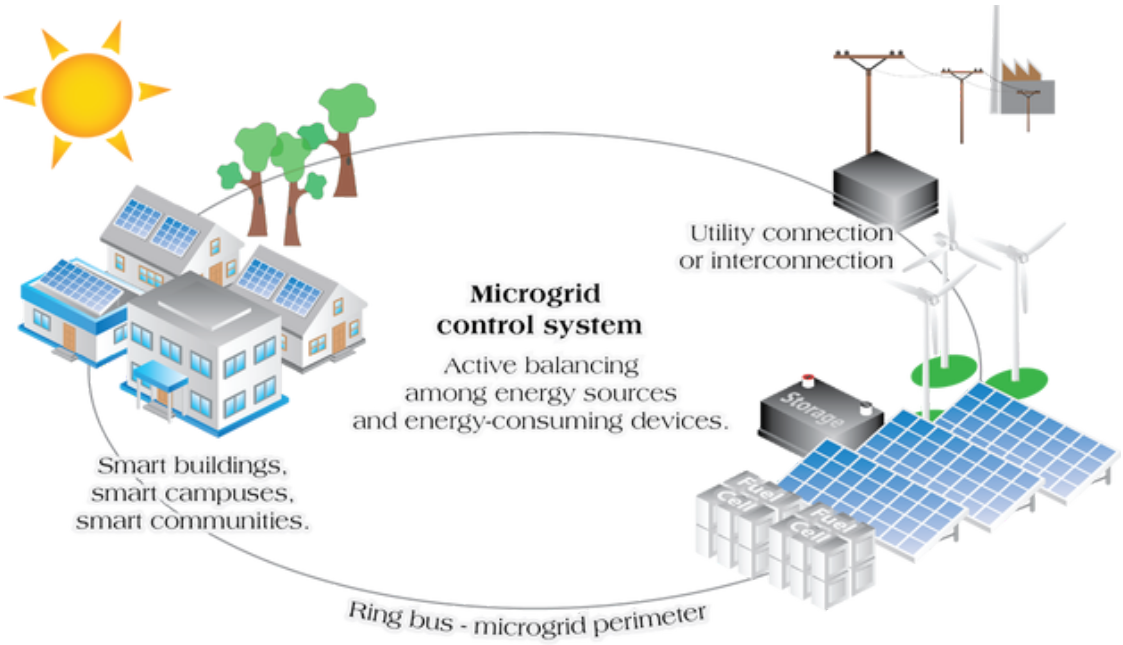
$L\frac{di(t)}{dt} + r(t) \cdot i(t) + R_H(t) \cdot i(t) = E_m \sin \omega t$.	(1.2)

Для визначення струму перехідного процесу у вихідному колі вітрогенератора необхідно розв’язати рівняння (1.2), права частина якого є синусоїдальною функцією напруги. Однак у вітроенергетичних установках

потужністю до 2 кВт [27] швидкість обертання лопатей становить 4,6...5 °/с, а швидкість обертання гондоли – 0,5...1 °/с, при цьому точність орієнтації на напрямок вітру має бути не меншою 4...5 °, оскільки при відхиленні в 5 ° втрати потужності можуть становити до 10 %. Час повороту лопатей та гондоли дорівнює приблизно 10 с. Період синусоїдальної напруги з частотою 50 Гц становить 0,02 с, що набагато менше, ніж час повороту. Вказана особливість дає змогу перейти від синусоїдальної напруги до її середнього значення за період $E_m \sin \omega t \to E_{CEP}$, що значно спрощує розрахунок [22].

Розрахунок потужності вітряного обладнання здійснюється на основі кубічної залежності швидкості вітрового потоку. Кубічна залежність означає, що якщо вітровий потік зі швидкістю, умовно 6 м/с, забезпечує потужність установки 100 Вт, то збільшення потоку до 12 м/с призведе до зростання потужності у вісім разів – до 800 Вт. Якщо турбіна характеризується невеликими розмірами, для отримання високої потужності буде потрібен дуже сильний вітер.

Якщо ж турбіна велика, вона здатна і за незначної вітрової швидкості видавати необхідну потужність. [9]



Максимальна енергія, що вилучається з вітрогенератора, залежить від режиму максимального вилучення енергії з кожного компонента, зокрема синхронного генератора та вітроколеса. Синхронний генератор підключається до навантаження через перетворювач частоти з вставкою постійного струму або безпосередньо до мережі як генератор змінної напруги. У системі генерації з використанням постійного струму для узгодження між синхронним генератором та навантаженням встановлений випрямляч.

Використання випрямляча при роботі синхронного генератора на постійному струмі призводить до появи вищих гармонік струму, зміни кута навантаження та зменшення рівня передаваної енергії. Для забезпечення максимального рівня передачі енергії до навантаження використовується компенсатор К, який відповідає за активне навантаження та формування задавального струму.

Залежність потужності Р вітроколеса від швидкості вітру та інших параметрів може бути описана формулою [24]

$$P = C_p \frac{\rho \pi R^2 v^3}{2},$$

де v – швидкість вітру; ρ – густина повітря; R – радіус вітроколеса; C_p – коефіцієнт потужності.

Оскільки швидкість вітру впливає на швидкість обертання вітроколеса, при якій досягається максимальна потужність на валу, необхідно враховувати зміни в характеристиках вітроколеса та електричного генератора в залежності від швидкості обертання. Цей аспект ілюструється на діаграмі 1.10 [22].

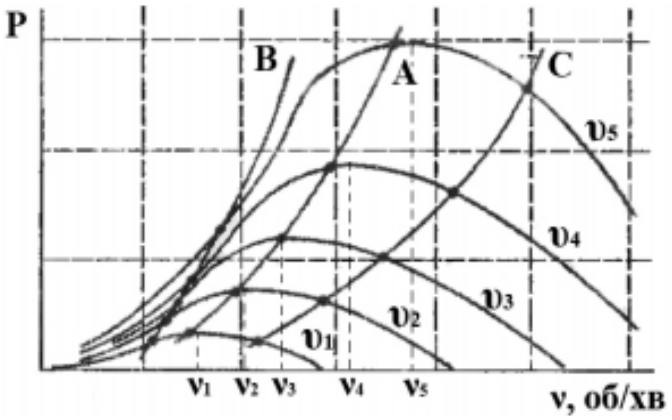


Рис.1.8. Характеристики потужності вітроколеса та електричного генератора

За аналізом діаграми 1.10 можна зробити висновок, що оптимальний режим роботи системи вітроколесо-генератор досягається при використанні кривої потужності А, яка відповідає максимальним значенням потужності для різних швидкостей вітру $v_1 \ldots v_5$ [22].

1.3 Відновлювальні джерела в Україні

Країни надають особливого значення виробництву енергії за допомогою відновлюваних джерел енергії, щоб усунути негативних економічних, екологічних та соціальних наслідків, що спричинені використанням ресурсів у виробництві енергії. Відновлювані джерела енергії є внутрішніми і не мають жодного негативного впливу, такого як зовнішня залежність в енергетиці та парникових газах, спричинених викопними ресурсами та які становлять загрозу для сталого економічного розвитку. У цьому відношенні для країн дуже важливий прогноз кількості енергії, яка буде вироблена відновлювальною енергією.

Енергетика є одним із показників рівня розвитку країн світу. Це важливий фактор, особливо для зменшення бідності та підвищення рівня життя. В останні роки споживання енергії значно зросло через фундаментальні зміни в секторах та економіках у світі. Енергія має життєво важливе значення для сталого розвитку нації в соціальному, економічному та

екологічному аспектах. За останнє десятиліття споживання енергії в глобальному масштабі поступово зросло. У зв'язку з цим точне прогнозування попиту має важливе значення для тих, хто приймає рішення, щоб розробити оптимальну стратегію не лише для зменшення ризику, але й для покращення економіки та суспільства в цілому, а енергопостачання є одним із найважливіших питань. у глобальному масштабі. Прогнозування попиту на енергію на ринках, що розвиваються, є одним із найважливіших політичних інструментів, які використовують особи, які приймають рішення в усьому світі. Енергетичне прогнозування, що стосується прогнозування попиту на електроенергію, використовується в усіх відділах сектору громадських послуг, включаючи передачу, генерацію, розподіл і роздрібний продаж.

Відновлювальна енергетична система України матиме важливе значення як для ширшої реконструкції, так і для відновлення економічної діяльності. Є два принципи, якими мають керуватися зусилля з відновлення енергетичної системи України: (1) забезпечення енергетичної безпеки та незалежності та (2) поглиблення зв'язків та економічних відносин між Україною та Європейським Союзом. Щоб досягти обох, Україна повинна розвивати свої потенційні відновлювані джерела енергії. Відновлювані джерела енергії, в тому числі вітрова, сонячна, в достатку в Україні. Розвиток цих ресурсів підтримуватиме виробництво електроенергії всередині країни, тим самим зміцнюючи енергетичну безпеку та незалежність України. Україна має достатній потенціал, щоб стати експортером енергії після війни, підтримуючи таким чином декарбонізацію Європейського Союзу та цілі енергетичної безпеки.

Як і у випадку з ширшими зусиллями з реконструкції, сектор відновлюваних джерел енергії потребуватиме значної фінансової підтримки як з боку державних, так і приватних партнерів. Створення ринку, здатного залучати капітал і конкурувати з іншими країнами, які прагнуть використовувати відновлювану енергетику, вимагатиме вдосконалення

політичних рамок України щодо відновлюваної енергетики. Розширення виробництва енергії з відновлюваних джерел залежатиме від значних зусиль з модернізації мережі та збільшення потужності зберігання та експорту для управління змінною генерацією в системі.

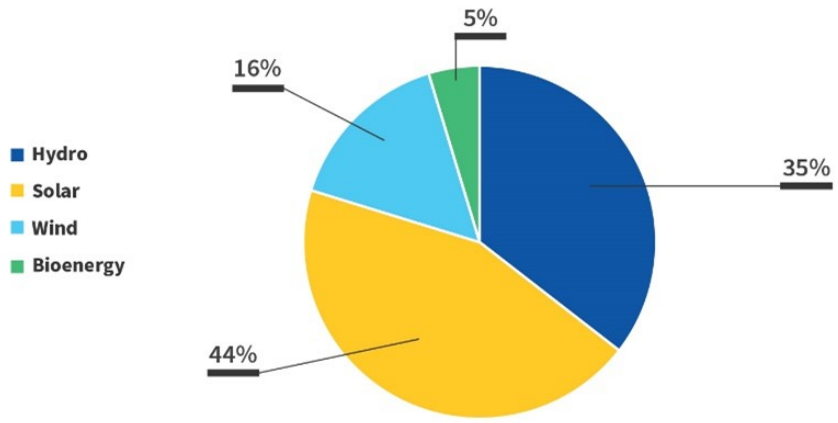


Рисунок 1.9. Відновлювані джерела енергетики в Україні

Система Microgrid, незалежно від того, підключена вона до комунальної мережі або незалежна система, зазвичай складається з суміші генерації - відновлювальної та невідновлюваної; навантаження - контрольовані чи неконтрольовані. Через періодичний характер відновлюваних ресурсів, таких як вітрова або сонячна енергія, важко точно спрогнозувати результат. Ці прогнози дуже залежать від прогнозів погоди, від хмарності та оточення. Очевидно, що прогноз будь-яких даних на основі прогнозу інших параметрів призведе до подальшої неточності, навіть якщо зв’язок між входом та виходом може бути заздалегідь визначений методами регресії. Тому ця робота ілюструє підхід до використання історичних даних про потужність замість числових прогнозів погоди для отримання результатів короткострокового прогнозу. Оскільки прогнози залежать від історичних середніх даних у «недалекому» минулому, точність обернено пропорційна зміні потужності між історичними та фактичними даними.

Основною функцією алгоритму прогнозування в мікромережі є прогнозування попиту на навантаження в мережі мікромережі або

електроенергії, виробленої відновлюваними джерелами енергії, підключеними до мережі, на найближче майбутнє. Це необхідно для визначення того, скільки енергії використовується з контрольованих ресурсів, таких як системи накопичення енергії, дизельні генератори, мікротурбіни або газові турбіни. Іншими словами, алгоритм оптимізації мікромережі використовує прогнози навантаження та відновлюваної енергії, щоб заздалегідь планувати електроенергію, вироблену розподіленими генераторами або заряджену/розряджену пристроями зберігання, оптимальним чином.

2. ОГЛЯД МЕТОДІВ ТА ПІДХОДІВ

Як було зазначено в попередньому розділі, зі швидким розвитком глобальної індустріалізації було визнано, що надмірне споживання викопного палива не тільки прискорить скорочення запасів викопного палива, але й матиме негативний вплив на навколишнє середовище. Ці впливи призведуть до збільшення ризиків для здоров'я та загроз глобальної зміни клімату. На додаток до викопного палива та ядерної енергії, відновлювана енергетика наразі є найшвидше зростаючим джерелом енергії. Відновлювана енергія – це повторно використовувана енергія, яка може бути відновлена в природі, наприклад сонячна енергія, енергія вітру, гідроенергія, енергія біомаси, хвилі, припливи та геотермальна енергія. Питання відновлюваних джерел енергії привернуло увагу завдяки характеристикам стійкості та низького рівня забруднення навколишнього середовища, і останнім часом було проведено багато відповідних досліджень. Одним із найважливіших викликів відновлюваної енергетики в найближчому майбутньому є енергопостачання. Відновлюване джерело енергії – це інтеграція відновлюваних джерел енергії в існуючі чи майбутні структури енергопостачання [5]. Розвиток систем відновлюваної енергетики зможе впоратися з основними питаннями поточних енергетичних проблем, такими як підвищення надійності енергопостачання та вирішення регіонального дефіциту енергії. Однак через величезну нестабільність і періодичний і випадковий характер відновлюваної енергії це генерування різних джерел енергії є переривчастим і хаотичним. Тому точна робота з випадковістю даних про відновлювану енергетику все ще є проблемою. Високоточний енергетичний моніторинг може підвищити ефективність енергосистеми. Технологія енергетичного прогнозування відіграє життєво важливу роль у розвитку, управлінні та виробленні політики енергетичних систем. У міру збільшення способів забезпечення електроенергією з відновлюваних джерел енергії дуже важливо розробити відповідні технології зберігання відновлюваної енергії [6]. Багато досліджень

виявили, що різні моделі машинного навчання використовувалися для прогнозування використання відновлюваної енергії. Моделі на основі даних надають практичні способи прогнозування відновлюваної енергії. Крім того, гібридні моделі машинного навчання були розроблені для підвищення точності прогнозування відновлюваної енергії. Різні часові інтервали, такі як хвилини, години, дні та тижні, використовувалися для прогнозування відновлюваної енергії відповідно до різних цілей прогнозів. Точність і ефективність прогнозування зазвичай використовувалися для оцінки продуктивності моделей машинного навчання в прогнозах відновлюваної енергії [7].

2.1. Алгоритми прогнозування.

Для застосування Microgrid необхідні прогнози генерації та навантаження зазвичай є короткостроковими. Будь-який прогноз, виконаний за кілька годин або днів наперед, можна класифікувати як короткострокове прогнозування. Оскільки навантаження та генерація залежать від кількох погодних умов, історичних вимірювань, особливих подій, часу доби, вхідними даними, що використовуються для короткострокових даних про навантаження та генерацію, є числове прогнозування погоди, дані минулого навантаження чи генерації та час- пов'язані дані (таблиця 2.1.). Проблема цього методу полягає в тому, що прогнози генерації базуються на прогнозах погоди, що може призвести до подальшої неточності.

Таблиця 2.1. - Джерела, моделі і методи прогнозування відновлюваної енергії

Sources of Energy	Models	Techniques
Wind	Artificial intelligence	Gaussian process regression(GPR), Support vector regression(SVR), Artificial neural networks(ANN); xGBoost regression, SVR, Random forest(RF); Least squares support vector machine(SVM); SVR, ANN, Gradient boosting(GB), RF; RF; GB trees;

		Multi-layer perceptron(MLP); Deep neural network(DNN)-principal component analysis; Feedforward ANN; Efficient deep convolution neural network; Linear regression, neural networks, SVR; Convolutional neural networks(CNN); DNN; Efficient deep CNN; Stacked auto-encoders, back propagation; Predictive deep CNN; Improved radial basis function neural network-based model with an error feedback scheme; ANN and genetic programming; Long short-term memory(LSTM); Improved LSTM-enhanced forget-gate network; LSTM-ANN; Auto-LSTM; Shared weight LSTM network; Ensem-LSTM; Instance-based transfer-GB decision trees; Extreme learning machine(ELM); Empirical mode decomposition(EMD)-stacked auto-encodersELM; Pattern sequence-based forecasting
	Hybrid	Adaptive neuro-fuzzy inference system, Particle swarm optimization(PSO), Genetic algorithm; Improved dragonfly algorithm-SVM; Deep belief network with genetic algorithms; Type-2 fuzzy neural network-PSO; Multi-objective ant Lion algorithm-Least squares SVM; Complete ensemble EMDmulti-Objective grey wolf optimization-ELM; Variational Mode decomposition(VMD)-Backtracking Search-regularized ELM; Coral reefs optimization algorithm with substrate layer, ELM; Stacked extreme-learning machine; VMD-singular spectrum analysis-LSTM-ELM; Ensemble EMD-deep Boltzmann machine; ELM-Improved complementary ensemble EMD with Adaptive noise-Autoregressive integrated moving average(ARIMA); Bayesian model averaging and Ensemble learning; Sparse Bayesian-based robust

		functional regression; Kernel principal component analysis-Core vector regression-Competition over resource; Wavelet packet decomposition-LSTM; Empirical wavelet transformation, Recurrent neural network(RNN)
Solar	Artificial intelligence	Gated recurrent units; RF, SVR, RF; RF; RF, gradient boosted regression, extreme GB; Linear regression, decision trees, SVM, ANN; ANN, SVM, GB, RF [70]; ANN; CNN [76]; DNN; DNN, RNN, LSTM; LSTM, auto-LSTM, gate recurrent unit(GRU), machine learning and statistical hybrid model; LSTM; LSTM, GRU; Copula-base nonlinear quantile regression; Multi-method ; Smart persistence ; K-nearest-neighbors, GB; Knearest-neighbors, SVM; Angstrom-Prescott; Multilayer feed-forward neural network; Support vector classification; GPR; Regime-dependent ANN; ELM; Adaptive forward–backward greedy algorithm, leapForward, spikeslab, Cubist and bagEarthGCV; Static and dynamic ensembles
	Hybrid	Wavelet decomposition-Hybrid; Improve moth-flame optimization algorithm-SVM; SVM-PSO; Cluster-based approach, ANN, SVM; SVM, Horizon, General; ANN, Principle component analysis; Auto regressive mobile average, MLP, Regression trees; Ensemble EMD-least square SVR; Least absolute shrinkage and Selection operator, LSTM; RF, SVR, ARIMA, k-nearest neighbors; MycielskiMarkov; VMD-deep CNN; PSO-ELM; Multiobjective PSO; Artificial bee colony-empirical models; Gated recurrent unit and Attention mechanism
	Statistical	ARIMA

Огляд алгоритмів. Деякі з широко використовуваних методів для короткострокового прогнозування потужності:

Авторегресійне інтегроване ковзне середнє (ARIMA) – це стохастичний метод, основний принцип якого полягає в тому, що прогнози на наступні години базуватимуться на прогнозах для попередніх одиниці. Незважаючи на те, що це проста модель, вона базується на наближенні, і справжня модель прогнозу невідома.

Модель *ARIMA* – це досить поширений клас статистичних моделей для аналізу та прогнозування даних часових рядів. Вона добре працює із стандартними структурами в даних у формі часових рядів, і таким чином забезпечує простий, але потужний спосіб для якісних прогнозів часових рядів.

ARIMA – це абревіатура, що означає AutoRegressive Integrated Moving Average. Це узагальнення простішої моделі AutoRegressive Moving Average, що додає поняття інтегрованості. Дана абревіатура є описовою, що фіксує ключові аспекти самої моделі. До моделі *ARIMA* входять наступні складові:

- *AR*: авторегресія. Модель, яка використовує зв'язок між деяким спостереженням та певною кількістю спостережень із затримкою;
- *I*: інтегрованість. Застосування методу різниць до спостережень (наприклад, віднімання спостереження від спостереження на попередньому кроці), щоб зробити часовий ряд стаціонарним;
- *MA*: ковзне середнє. Модель, яка використовує залежність між спостереженням та залишковою помилкою від моделі ковзного середнього, застосованої до спостережень із лагами.

Кожен із цих компонентів явно вказаний у моделі як параметр. Стандартне позначення моделі – *ARIMA*(*p*, *d*, *q*), де параметри замінюються цілими значеннями, щоб швидко вказати конкретну модель *ARIMA*, яка використовується.

Параметри моделі *ARIMA* визначаються наступним чином:

- *p*: кількість спостережень із затримкою, включених у модель, також називається порядком затримки;
- *d*: кількість разів, коли до спостережень застосовують метод різниць, також називають порядком різниць;
- *q*: розмір вікна ковзного середнього, також називається порядком ковзного середнього.

Далі будується лінійна регресійна модель, враховуючи вказані вище параметри, а до часового ряду також застосовують метод різниць, щоб зробити його стаціонарним, тобто видалити тренди та сезонні компоненти, які негативно впливають на регресійну модель.

Значення 0 також може використовуватися для певного параметра, причому це вказує на те, що конкретний компонент не слід використовувати в даній моделі. Таким чином, модель *ARIMA* може бути налаштована на виконання функцій моделі *ARMA*, і навіть простіших моделей *AR*, *I* або *MA*. Застосування моделі *ARIMA* до часового ряду передбачає, що основним процесом, який генерує спостереження, є процес *ARIMA*. Це може здатися очевидним, але допомагає обґрунтовувати необхідність підтвердження припущень моделі для початкових спостережень та залишкових помилок прогнозів моделі.

Тепер пояснимо в чому саме полягає відмінність моделі *ARIMA* від моделі *ARMA*. Розглянемо спочатку рівняння для моделі *ARMA*.

Рівняння виду:

$$y(k) = a_0 + \sum_{i=1}^{p} a_i\, y(k-i) + \sum_{j=1}^{q} b_j\, \varepsilon(k-j) + \varepsilon(k) =$$

$$= a_0 + \sum_{i=1}^{p} a_i\, y(k-i) + \sum_{j=0}^{q} b_j\, \varepsilon(k-j),$$

де $b_0 = 1,\ \sum_{j=1}^{q} b_j = 1,$

називають моделлю авторегресії та ковзного середнього (*ARMA*), якщо корені його характеристичного рівняння лежать всередині одиничного кола на комплексній площині, а $\{\varepsilon(k)\}$ – випадковий процес.

При цьому, якщо один або більше коренів характеристичного рівняння, записаного для (2.16), дорівнюють одиниці, то послідовність $\{y(k)\}$ називають інтегрованою або процесом з одиничними коренями, а рівняння (2.16) називають авторегресією з інтегрованим ковзним середнім (*ARIMA*). Процеси з одиничними коренями відносяться до класу нестаціонарних процесів. Оскільки такі процеси є досить характерними для виробничих технологій, економіки, фінансів, екології та інших галузей, то їм необхідно приділяти значну увагу.

• Множинна лінійна регресія (MLR) — це проста техніка часових рядів для моделювання набору даних про навантаження або генерацію у вигляді лінійної кривої. Але, через переривчастий характер даних, насправді моделі є нелінійними.

• Розширена нейронна мережа (ANN) — це математичний інструмент, спочатку створений на основі того, як людський мозок обробляє інформацію [3]. Нейрон отримує інформацію (погоду, час та історичні дані) через низку вхідних вузлів, обробляє її внутрішньо та видає відповідь. В основі моделі лежить чорний ящик підхід і може призвести до переобладнання. Нечіткі нейронні моделі — це комбінація нечіткої логіки та концепції ШНМ, де нечітка попередня обробка допомагає зменшити кількість вхідних даних для алгоритму ШНМ.

Штучні нейронні мережі - це методи прогнозування, які базуються на простих математичних моделях нейрона. Вони допускають складні нелінійні взаємозв'язки між змінною відповіді та її предикторами. [7] Нейронну мережу (NN) можна розглядати як мережу «нейронів», які організовані в шари. Прогнози (або вхідні дані) утворюють нижній шар, а прогнози (або результати) - верхній шар. Також можуть бути проміжні шари, що містять «приховані нейрони». [12] Найпростіші мережі не містять прихованих шарів і

еквівалентні лінійним регресіям. Коефіцієнти, прикріплені до цих предикторів, називаються "вагами". Прогнози отримують за допомогою лінійної комбінації вхідних даних. Ваги вибираються в рамках нейронної мережі за допомогою «алгоритму навчання», який мінімізує «функцію витрат», таку як MSE. Звичайно, у цьому простому прикладі можна використати лінійну регресію, яка є набагато ефективнішим методом навчання моделі.

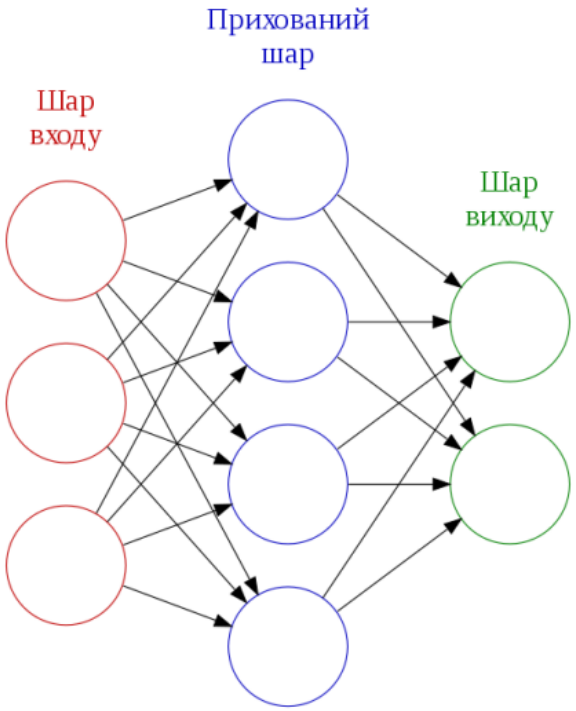


Рис. 2.3 – Загальний вигляд нейронної мережі

Як тільки додається проміжний шар із прихованими нейронами, нейронна мережа стає нелінійною (рис 2.3). [12] Це відомо як багатошарова мережа прямого пересилання, де кожен рівень вузлів отримує вхідні дані від попередніх шарів. Виходи вузлів в одному шарі є входами для наступного шару. Входи кожного вузла об'єднуються за допомогою зваженої лінійної комбінації. Потім результат модифікується нелінійною функцією перед виведенням. Наприклад, вхідні дані у прихований нейрон j (рис 2.3) об'єднані лінійно, щоб отримати:

$$z_j = b_j + \sum_{i=1}^{n} w_{i,j} x_i.$$

У прихованому шарі модифікується за допомогою нелінійної функції, такої як сигмоїда:

$$s(z) = \frac{1}{1 + e^{-z}},$$

щоб дати вхід для наступного шару. Це має тенденцію до зменшення ефекту екстремальних вхідних значень, роблячи таким чином мережу надійнішою до вищого рівня. [12] Параметри *bj* , *wi* залежать від даних. Значення ваг *wi* часто обмежують, щоб запобігти їх надмірному зростанню. Параметр, який обмежує ваги, відомий як "параметр затухання", і часто встановлюється рівним 0,1. Для початку ваги приймають випадкові значення, які потім оновлюються з використанням спостережуваних даних. У прогнозуваннях, створених нейронною мережею, є елемент випадковості. Тому мережу навчають кілька разів, використовуючи різні випадкові вихідні точки, і результати усереднюють.

Кількість прихованих шарів та кількість вузлів у кожному прихованому шарі повинні бути вказані заздалегідь. Найбільш важливою здатністю нейронних мереж порівняно з іншими нелінійними моделями є їх гнучкість у моделюванні будь-якого типу нелінійних шаблонів без попереднього припущення про основний процес генерації даних [16]. Найпопулярнішою моделлю нейронної мережі для прогнозування часових рядів є тришарова модель прямої мережі, яку можна записати як:

$$y_t = \alpha_0 + \sum_{j=1}^{n} \alpha_j f\left(\sum_{i=1}^{m} \beta_{ij} y_{t-i} + \beta_{0j}\right) + \varepsilon_t,$$

де m - кількість вхідних вузлів, n - кількість прихованих вузлів, f - сигмоїдна передавальна функція, {αj, j=0,1,…, jn =…} - це вектор ваг від прихованих до вихідних вузлів, α0 та β 0 j - це ваги дуг, що ведуть із зміщених членів. Для проблеми прогнозування часових рядів побудова моделі NN еквівалентна визначенню як кількості вхідних вузлів, так і кількості

прихованих вузлів [17]. Вхідні вузли – це минулі спостереження, за допомогою яких може бути отримана основна структура автокореляції даних. Хоча теорія універсального наближення NN вказує на хороше наближення може знадобитися велика кількість прихованих вузлів, лише невелика кількість часто потрібна в реальних додатках [16].

Типи прогнозу вітроенергетики. Існують різні способи прогнозування потужності вітру. Ці методи класифікуються відповідно до тимчасових шкал і відповідно до різних методологій, доступних у літературі та наведених вище.

Тимчасові масштаби та методи прогнозування сили або енергії вітру, поєднуючи літературу, можна розділити на чотири основні категорії:

- Ультракороткостроковий прогноз: від кількох хвилин до 1 години наперед.
- Короткостроковий прогноз: від однієї години до кількох годин наперед.
- Середньостроковий прогноз: від кількох годин до одного тижня наперед.
- Довгостроковий прогноз: від одного тижня до одного року та більше вперед.

Але також треба звертати увагу на інтервали в початкових даних. Якщо початкові дані представлені з інтервалом в годину, то ми не можемо прогнозувати похвилинно часовий ряд. Так само і навпаки, якщо початкові дані представлені похвилинно, ми можемо прогнозувати тільки на хвилини, і у разі необхідності будувати прогноз на 60 кроків вперед для отримання значення через годину.

2.2. Оцінка якості моделей та прогнозу

2.2.1. Аналіз якості моделі

Адекватність регресійних моделей може бути встановлена на основі аналізу послідовності залишків (похибок моделі); при цьому розрахункові значення знаходять підстановкою в модель фактичних значень всіх включених у модель факторів. Залишкова послідовність перевіряється на

виконання властивостей випадкової компоненти економічного часового ряду: близькість нулю математичного очікування, випадковий характер відхилень, відсутність автокореляції та нормальність закону розподілу.

Аналіз адекватності моделі виконується на основі відповідних параметрів адекватності, до яких відносяться подані нижче.

- Сума квадратів залишків – сума квадратів величин розбіжності між змодельованими і фактичними значеннями пояснюючої змінної на період і ідентифікації, яка розраховується за такою формулою:

$$R^2 = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

Коефіцієнт детермінації — статистичний параметр, щовикористовуєтьсявстатистичнихмоделяхякміраінформативностімоделістосов ноданих. Він показує на скільки наявні спостереження відповідають моделі:

$$R^2 = 1 - \frac{V(y|x)}{V(y)} = 1 - \frac{\sigma^2}{\sigma_y^2}$$

де $V(y|x) = \sigma^2$ – дисперсія основної змінної, оціненої за моделлю.

Критерій Дарбіна-Уотсона (або DW критерій) – статистичний критерій, який використовується для тестування автокореляції першого порядку елементів досліджуваної послідовності. Найбільш часто застосовується при аналізі часових рядів і залишків регресійних моделей:

$$d = \frac{\sum_{t=2}^{n} (e_t - e_{t-1})^2}{\sum_{t=1}^{n} e_t^2} \approx 2(1 - \rho_1)$$

Інформаційний критерій Акаіке (ІКА, англ. Akaike information criterion, AIC) – це міра відносної якості статистичних моделей для заданого набору даних. Маючи сукупність моделей для наявних даних, ІКА оцінює якість кожної з моделей відносно кожної з інших моделей. ІКА заснований на теорії інформації: він характеризує відносні оцінки втраченої інформації при застосуванні даної моделі для представлення процесу, що породжує дані.

Таким чином, він вказує на компроміс між ступенем узгодженості моделі та її складністю. Припустімо, що ми маємо статистичну модель якихось даних. Нехай L – максимальне значення функції правдоподібності для цієї моделі; і нехай k буде числом оцінюваних параметрів у цій моделі. Тоді значення ІКА для цієї моделі обчислюється так:

$$AIC = 2k - 2\ln(L)$$

Аналіз якості прогнозу Важливим моментом процесу прогнозування є об'єктивне визначення якості отриманого прогнозу. Оскільки прогнозовані значення – випадкові величини, то для оцінювання їх якості необхідно використовувати декілька статистичних критеріїв. Рис. 2.1 ілюструє часову вісь та відрізки часу, на яких виконується оцінювання моделі і перевірка якості прогнозу.

Рис. 2.1. Види прогнозування за часовим рядом

Наявну вибірку даних доцільно розділити на навчальну та перевірочну. На навчальній вибірці виконується оцінювання параметрів моделі процесу і реалізується так званий „історичний" прогноз, який дає змогу встановити якість однокрокового прогнозу на цьому участку ряду. Прогноз на перевірочній частині вибірки даних в науковій літературі називають ще прогнозом ex post. В різних емпіричних дослідженнях рекомендують залишати для перевірки (5 – 40) % значень ряду даних. Хоча при аналізі коротких рядів доцільно значно більшу частину ряду використовувати для оцінювання параметрів моделі. Прогнозування значень поза вибіркою даних називають прогнозом ex ante (рис. 4.1). Як правило, для оцінювання якості прогнозів використовують множину взаємно доповнюючих статистичних

критеріїв. Наприклад, значення середньоквадратичної похибки залежить від масштабу даних, а тому недостатньо використовувати тільки цей статистичний параметр для аналізу якості прогнозу. Розглянемо деякі статистичні критерії якості прогнозу та їх призначення.

Обов'язковим етапом прогнозування є точність та обґрунтованість прогнозів. На цьому етапі використовується сукупність критеріїв, підходів і процедур, які можуть оцінити якість прогнозу.

2.2.2 Аналіз якості прогнозу

- Середній квадрат похибок моделі. Для обчислення середнього квадрату похибок (MSE) все окремі залишки регресії зводяться в квадрат,

підсумовуються, а сума ділиться на загальне число похибок:

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$$

Квадратний корінь з цієї величини позначається як RMSE (Root Mean Squared Error):

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(y_i - \hat{y}_i)^2}{n}}$$

- Абсолютна помилка прогнозу може бути визначена як різниця між фактичним значенням (y) і прогнозом (y*):

$$\Delta_{pr} = y_t - y^*$$

Коефіцієнт нерівності Тейла. Коефіцієнт нерівності Тейла – це важливий індикатор якості моделі і прогнозу; за означенням, . Якщо , то модель має практично нульові (неприйнятні) прогнозуючі властивості, що випливає з формули для обчислення :

$$U = \frac{\sqrt{\frac{1}{N}\sum_{k=1}^{N}[y(k) - \hat{y}(k)]^2}}{\sqrt{\frac{1}{N}\sum_{k=1}^{N} y^2(k)} + \sqrt{\frac{1}{N}\sum_{k=1}^{N} \hat{y}^2(k)}}.$$

– Середня абсолютна похибка в процентах (САПП) – це середнє абсолютних значень похибок оцінок прогнозу в процентах відносно фактичного значення показника:

$$САПП = \frac{1}{N}\sum_{k=1}^{N}\frac{|y(k)-\hat{y}(k)|}{|y(k)|}\times 100\% = \frac{1}{N}\sum_{k=1}^{N}\frac{|e(k)|}{|y(k)|}\times 100\%,$$

бо у випадку прогнозування на кроків відносно го моменту:

$$САПП = \frac{1}{s}\sum_{i=1}^{s}\frac{|y(k+i)-\hat{y}(k+i,k)|}{|y(k+i)|}\times 100\% = \frac{1}{s}\sum_{i=1}^{s}\frac{|e(k+i)|}{|y(k+i)|}\times 100\%$$

Оскільки ця міра характеризує відносну якість прогнозу, то її використовують, в основному, для порівняння точності прогнозів різнорідних об’єктів (процесів) прогнозування. Однак, вона є завжди корисною при виконанні порівняльного аналізу якості прогнозування одного й того ж процесу різними методами, оскільки відносна міра є чіткою і зрозумілою для дослідника і практичного користувача. Типові значення САПП та їх пропонована інтерпретація наведені в таблиці 2.1.

Таблиця 2.1 Інтерпретація типових значень критерію САПП

САПП, %	Інтерпретація
< 10	Висока точність
10 – 20	Хороша точність
20 – 50	Задовільна точність
> 50	Незадовільна (неприйнятна) точність

2.3 Обгрунтування вибору моделей для прогнозування

Статистичні моделі прості у використанні та дешевші у розробці порівняно з іншими моделями. По суті, статистичні методи використовують попередню історію даних про вітер для виконання прогнозу на наступні кілька годин, вони є хорошими для коротких періодів часу. Недоліком цього є

те, що помилка передбачення збільшується зі збільшенням часу прогнозування, тобто статистичні часові ряди і методи нейронних мереж насамперед призначені для короткострокових прогнозів. Статистичні моделі, найбільш поширені дослідниками, включають: авторегресійну (AR), авторегресійну ковзну середню (ARMA) і авторегресійну інтегровану ковзну середню (ARIMA).

Статистичні методи у багатьох прогнозах використовують різницю між прогнозованою та фактичною швидкістю вітру для налаштування параметрів моделі. Перевага штучних нейронних мереж (ШНМ) полягає в тому, що вони дозволяють дізнатися про співвідношення між вхідними та вихідними даними за допомогою нестатистичного підходу. Відповідно до [14], нейронні мережі можуть легко вчитися на вхідному і вихідному відображенні на етапі навчання.

2.4 Розробка програмного продукту

Програма, яка використовується для прогнозування електрогенерації у системі MicroGrid, розраховує потужність для сонячних панелей (СП) та вітрогенераторів. Ці генератори потребують відповідних погодних умов для оптимальної роботи. Для вітрогенератора важлива швидкість вітру, яка повинна бути більше певного значення, а найкраще наближатися до номінальної швидкості, щоб забезпечити максимальну вихідну потужність. Щодо сонячних панелей, важливими є відсоток хмарності (високий відсоток знижує вихідну потужність) та наявність опадів, зокрема дощу, який також знижує ККД панелей через каплі на їх поверхні.

Дані для розрахунків отримуються з метеорологічного сайту шляхом парсингу. Існує обмежена кількість метеорологічних сайтів, які дозволяють отримати необхідну інформацію, але я обрала підходящий сайт, який надавав потрібні дані та дозволяв їх збирати.

Програма реалізована на мові програмування Python. Python - це інтерпретована об'єктно-орієнтована мова програмування високого рівня, яка

призначена для вирішення різноманітних завдань. За допомогою Python можна обробляти дані, створювати зображення, працювати з базами даних, розробляти веб-сайти та додатки з графічним інтерфейсом.

Програма також показує графік потужності у окремому вікні і зберігає всі дані у відповідному файлі.

2.1.1 Інтерфейс та його функції

Інтерфейс програми простий і одновіконний, як показано на рис. 2.2.

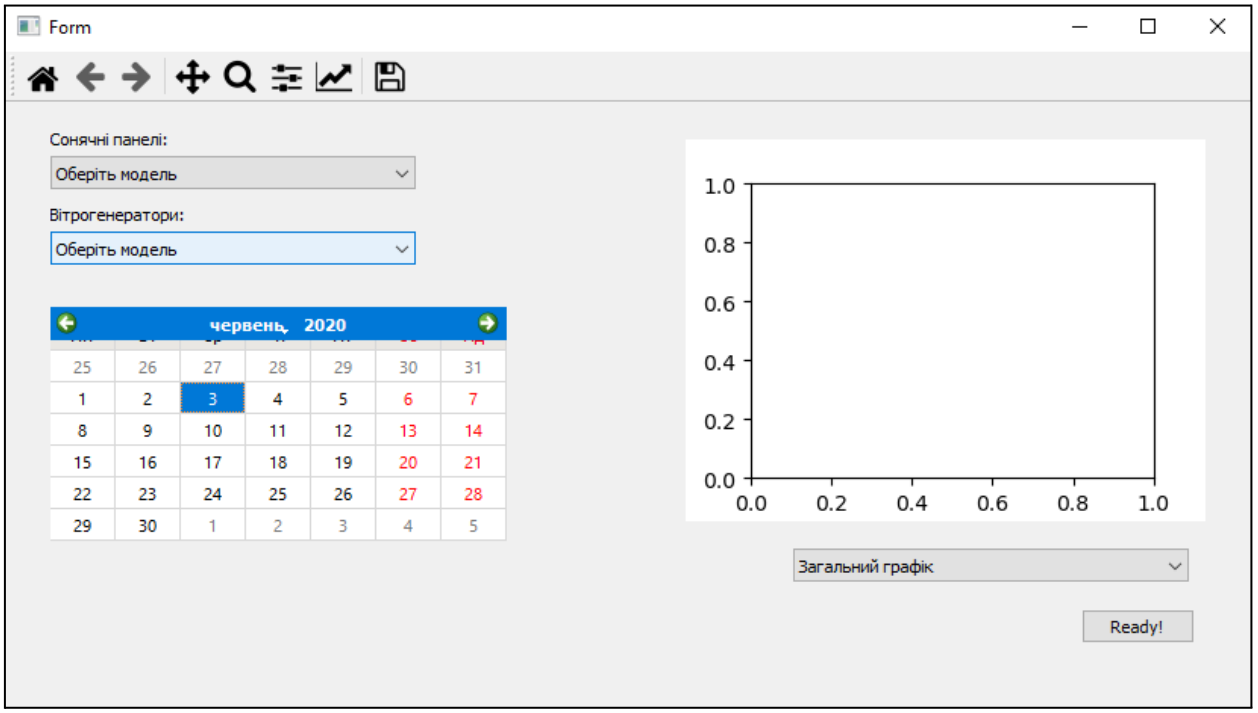
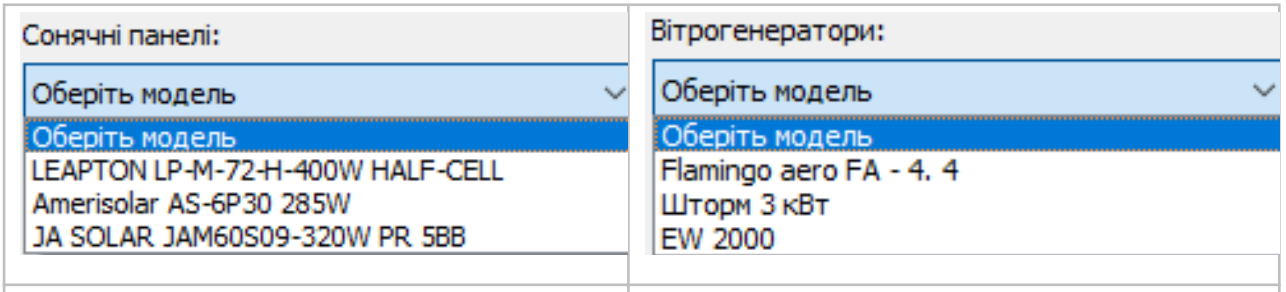


Рис.2.2. Інтерфейс програми

У вікні знаходяться елементи керування програмою:

- меню вибору моделі СП або вітрогенератора
- календар для вибору дати
- меню вибору типу графіків
- панель керування
- область графіка – місце, де буде будуватись графік
- керуюча кнопка «Ready» – вона запускає алгоритм побудови.

Розкрите меню вибору моделі альтернативного джерела енергії виглядає так, як показано на рис.2.3.



Меню моделей СП | Меню моделей вітрогенератора

Рис.2.3. Меню моделей альтернативного джерела енергії

В кожному меню міститься по 3 моделі кожного альтернативного джерела енергії. Важливою деталлю інтерфейсу є календар (рис.2.4).

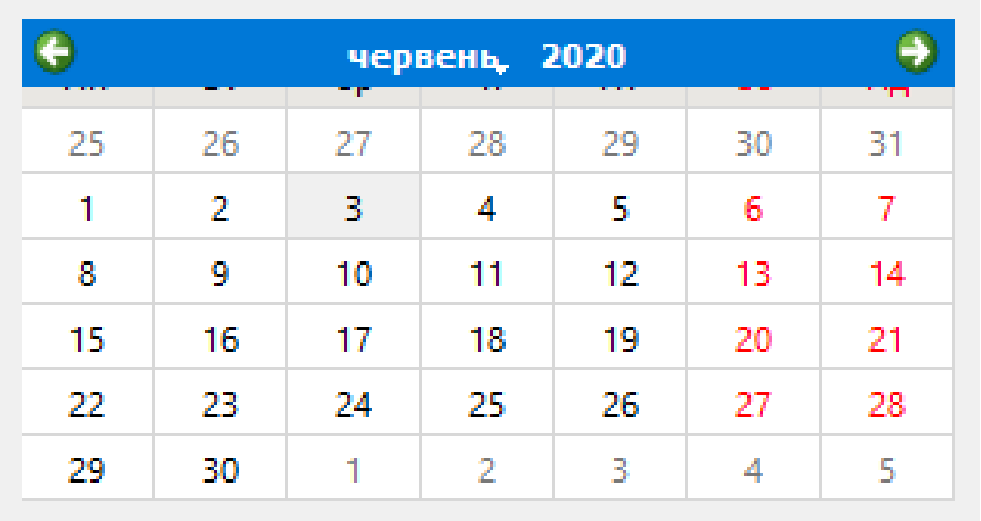


Рис.2.4. Календар програми

Він представляє собою путівник по даним за певний день і дозволяє обирати в певному проміжку (на даний момент цей проміжок 7 днів), на більшу кількість часу отримати даних неможливо.

Меню вибору графіків дозволяє обрати тип графіків, котрий користувач захоче побачити (рис.2.5) :

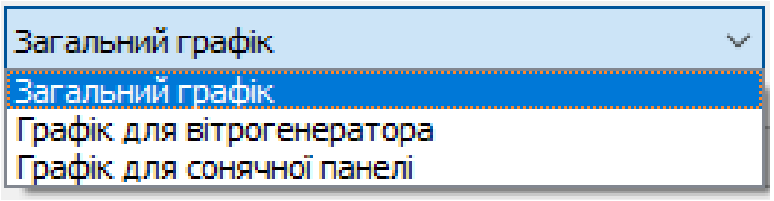


Рис.2.5. Меню вибору типу графіка

- загальний графік – будує графік суми потужностей двох альтернативних джерел енергії (СП та вітрогенератора);
- графік для сонячної панелі – будує графік потужностей СП;
- графік для вітрогенератора – будує графік потужносте для вітрогенератора.

Панель керування (рис.2.6) дозволяє пересуватись по вікна (функції будуть корисні в більш просунутій версії програми), керувати інтерфейсом або графіком.

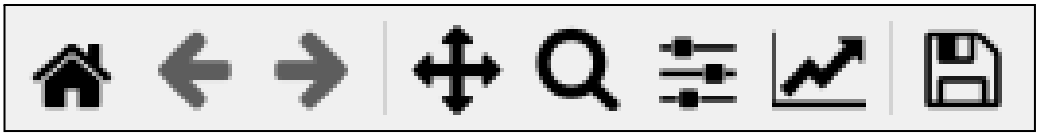


Рис.2.6. Панель керування

Кнопки керування на панелі (зліва на право):

- повернення на головне вікно (функція буде корисною для швидкого повернення на стартове вікно програми);
- кнопка повернення на одне вікно назад (корисна, якщо потрібно зробити повернення на одну дію назад);
- кнопка повернення вперед (дозволяє повернутись на одну дію вперед);
- кнопка дозволяє керувати пересуванням по графіку (корисна при збільшені графіка);
- кнопка дозволяє збільшити окрему область на графіку (якщо потрібно більш чітко роздивитись графік);

- кнопка дозволяє настроїти інтерфейс (доступній лише для адміністратора);
- керує відображенням графіка (можна налаштувати осі і надписи);
- дозволяє обрати місце куди зберігати дані потужностей.

Область графіка це графічний дисплей, на якому будується графік потужностей (рис.2.7).

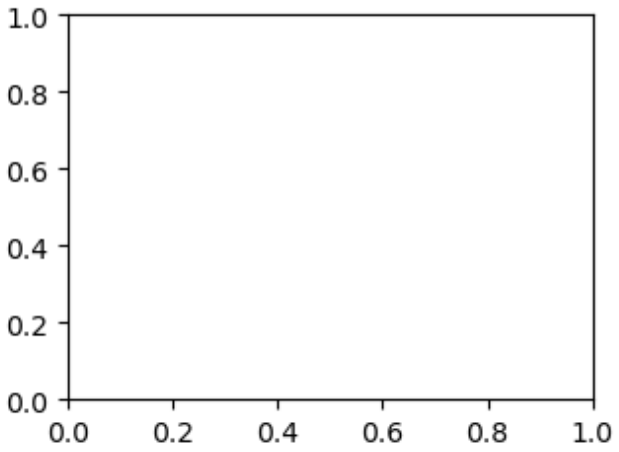


Рис.2.7. Дисплей відображення графіка потужності

2.1.2 Розрахунок потужностей та побудова графіків

Спираючись на отримані дані погодних умов, а саме хмарності, дощу були підібрані 3 моделі сонячних панелей (табл.2.1) [12-14].

Таблиця 2.1. Основні параметри сонячних панелей

Модель СП	**ККД, %**	**Максимальна потужність, Вт**
LEAPTON LP-M-72-H-400W HALF-CELL	20	400
Amerisolar AS-6P30 285W	17	280
JA SOLAR JAM60S09-320W PR 5BB	19	320

Оскільки дані подані не чітко, а саме відсутні відсоткові значення хмарності, була розроблена таблиця ККД альтернативних джерел енергії відносно погодного стану для СП (табл.2.2).

Таблиця 2.2. Зменшення вихідного ККД відносно погодних умов

Погодні умови	**Зменшення вихідного ККД на, %**
Ясна погода	0
Мінлива хмарність	10
Значна хмарність	20
Пахмурна погода	35
Невеликий дощ	45
Сильний дощ	65

Такий грубий розподіл збільшить похибку в розрахунках, але спростить обрахунок потужності.

Формула для визначення вихідної потужності СП при ясній погоді :

$$P_{вих} = P_{\max} * \eta ,$$

де P_{max} – максимальна потужність альтернативного джерела енергії;

η – ефективність перетворення сонячної енергії.

Формула вихідної потужності при наявності негативних погодних умов:

$$P_{вих} = (P_{\max} * \eta) * \eta_{ну} ,$$

де $\eta_{ну}$ – ефективність перетворення сонячної енергії при негативних погодних умовах:

$$\eta_{ну} = 100\% - \eta_{зв} ,$$

де $\eta_{зв}$ – зменшення вихідного ККД.

Для вітрогенераторів також були підібрані 3 моделі альтернативних джерел енергії (табл.2.3) [15-17].

Таблиця 2.3. Параметри вітрогенераторів

Модель	**Стартова швидкість, м/с**	**Номінальна швидкість, м/с**	**Максимальна потужність, кВт**
Flamingo aero FA - 4. 4	2,5	8	1,6
Шторм 3 кВт	1,5	8	3,4
EW 2000	2,5	12	2

Основні параметри для роботи – це стартова та номінальна швидкість вітру при якій працює вітряк.

Для більш чіткого результату були додані додаткові проміжні ККД між стартовою і номінальною швидкістю (табл.2.4).

Таблиця 2.4. Вихідний ККД відносно швидкості вітру

Модель	**ККД, %**	**Швидкість вітру, м/с**
Flamingo aero FA - 4. 4	60	4,5
	80	6,5
	100	8
Шторм 3 кВт	60	3,5
	80	6,5
	100	8
EW 2000	60	6,5
	80	9,5
	100	12

Якщо ж швидкість вітру менша стартової, то вихідна потужність дорівнює 0.

Формула вихідної потужності вітрогенератора дорівнює:

$$P_{вих} = P_{\max} * \eta \ ,$$

де η – ефективність перетворення сили вітру.

Після збирання даних і підстановки їх у формули будуються графіки в області графіків.

Для отримання графіку потужності СП потрібно обрати модель альтернативного джерела енергії в програмі (рис.2.8).

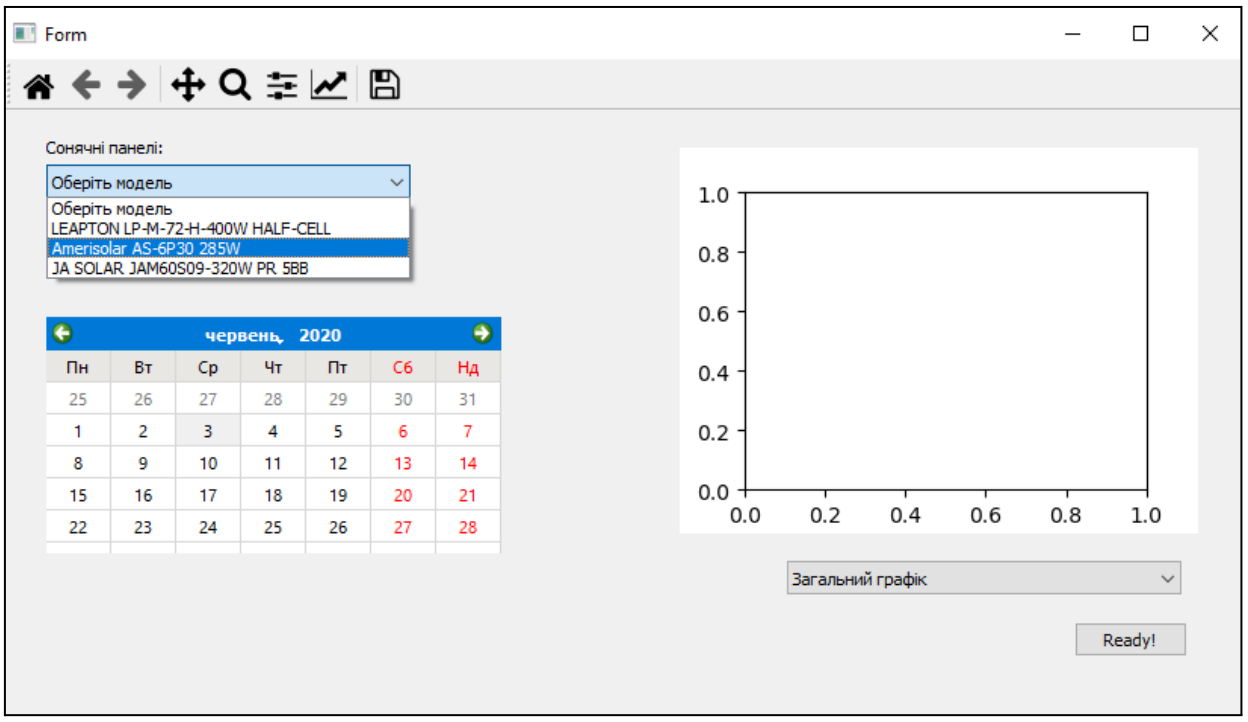


Рис.2.8. Вибір моделі СП

Наступний крок – вибір періоду, за який надається графік, але не більше допустимого (не більше 7 днів).

Діапазон вибирається в календарі (рис.2.9).

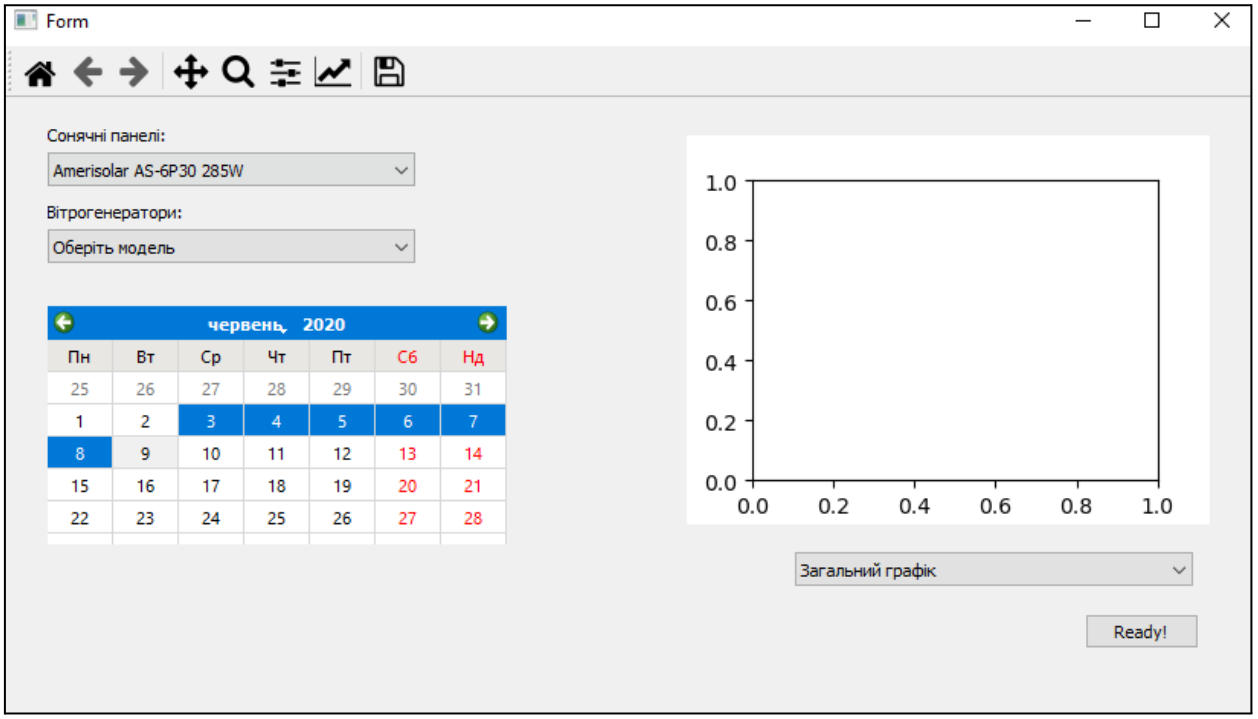


Рис.2.9. Вибір діапазонудля побудови графіка

Для побудови графіку обирається тип графіку, котрий підходить для альтернативного джерела енергії з меню графіків або залишається «Загальний графік» – підійде для будь якого альтернативного джерела енергії. Графік потужності наведено на рис.2.10.

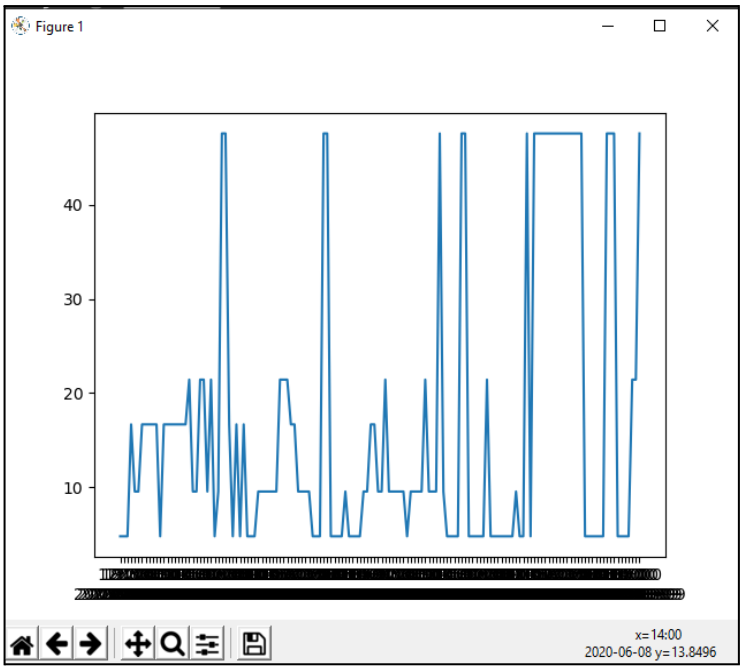


Рис.2.10 Графік потужності СП

Графік залежності потужності від часу можна спостерігати в окремому (рис.2.9) або ж на головному вікні програми (рис.2.11).

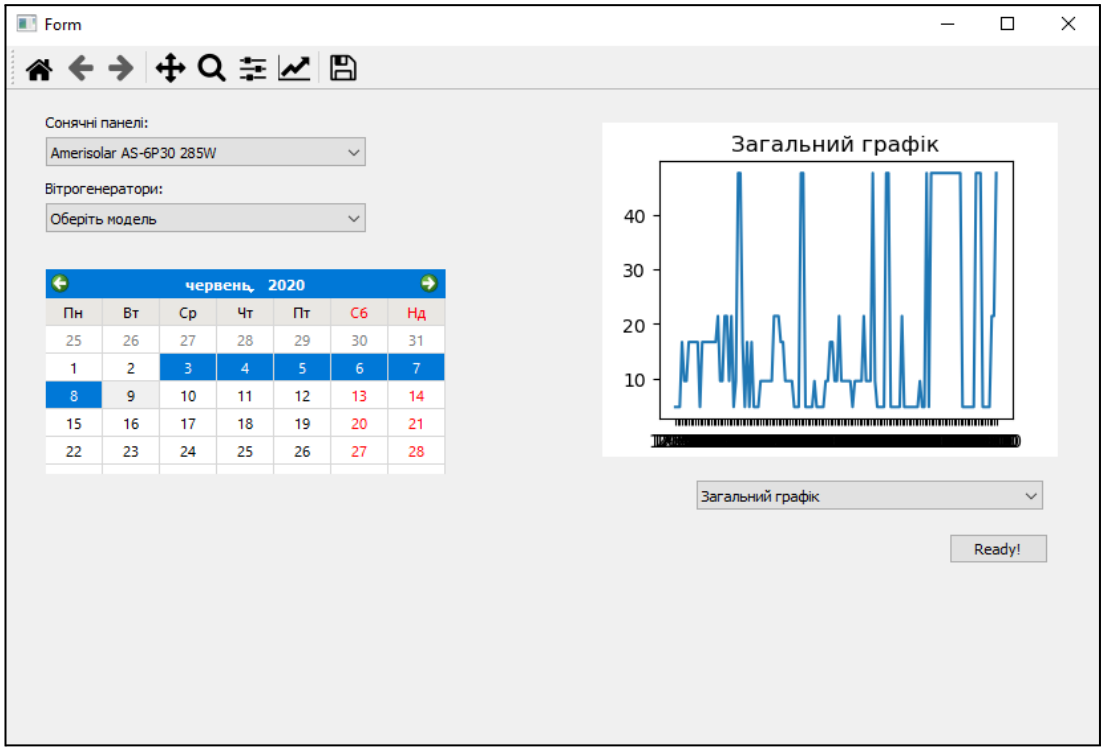


Рис.2.11. Інтерфейс з побудованим графіком

Аналогічно робиться побудова графіку прогнозування вихідної потужності вітрогенератора (рис.2.12).

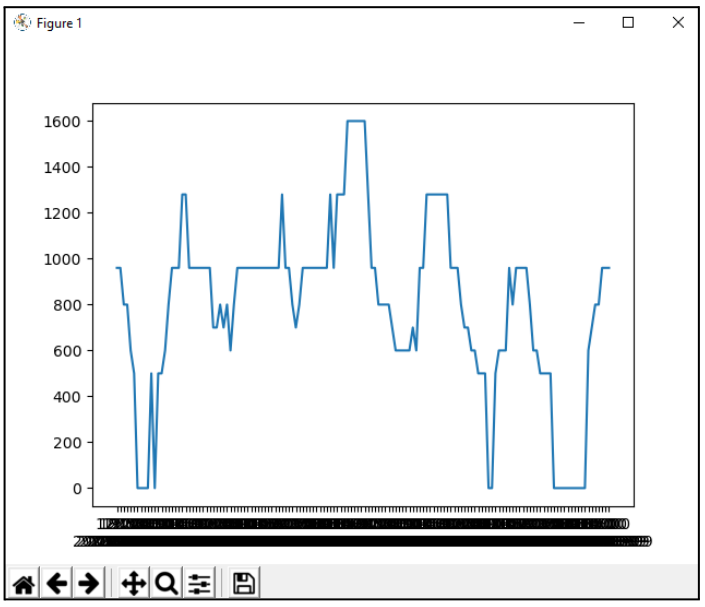


Рис.2.12. Графік вихідної потужності для вітрогенератора

1) обирається модель вітрогенератора;
2) задаємо діапазон днів;
3) обираємо тип побудови графіка;
4) натискаємо кнопку «Ready!».

Але, якщо залишили в меню графіків пункт «Загальний графік», то потрібно прибрати модель СП або перемкнути на пункт «Графік для вітрогенератора».

Через великий діапазон часу, на котрий не вистачає місця на осях графіку, було знайдено рішення заносити дані в документ (рис.2.13), що полегшить не тільки сприймання, а й прибере необхідність списувати дані з графіку.

A	B	C	D	E	F	G	H	I	J	K	L	M
17:00 2020-06-03	18:00 2020-06-03	19:00 2020-06-03	20:00 2020-06-03	21:00 2020-06-03	22:00 2020-06-03	23:00 06.0	0:00 2020-06-04	1:00 2020-06-04	2:00 2020-06-04	3:00 2020-06-04	4:00 2020-06-04	5:00 2020-06-04
960	960	800	800	600	500	0	0	0	0	500	0	500

Рис.2.13. Фрагмент файлу даних вітрогенератора

3. ОБЧИСЛЮВАЛЬНІ ЕКСПЕРИМЕНТИ ТА РЕЗУЛЬТАТИ РОБОТИ ПРОГРАМИ

Для роботи було взято початкові дані з сайту https://www.meteoservice.ru/weather/hourly/kiev# та kaggle, всі дані знаходяться у відкритому доступі. Перший набір даних характеризується такими атрибутами як: час, хмарність, вітер. Другий набір даних - температура повітря, вологість повітря, атмосферний тиск, середня швидкість вітру, напрям вітру. Прогнозування будемо проводити на для короткострокового та середньострокових проміжків часу. Наприклад, на 1 день, на 3 дні, на 7 днів.

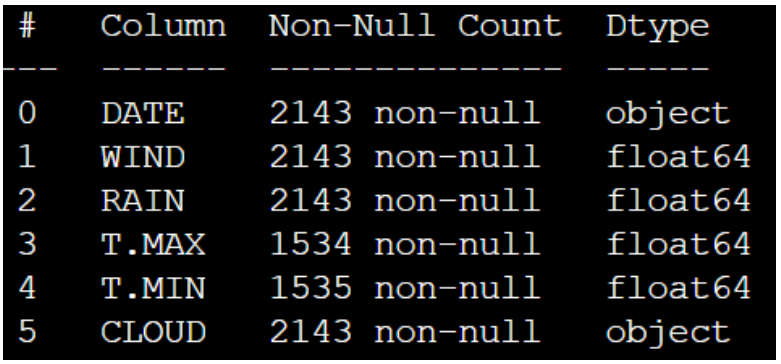
3.1 Підготовка даних

Для початку наш існуючий набір даних (рисунок 3.1) необхідно перевірити на наявність пропусків і у разі, якщо вони виявляться, заповнити їх або видалити.

1	DATE	WIND	RAIN	T.MAX	T.MIN	CLOUD
2	2001-01-01	13.67	0.2	9.5	3.7	0,3
3	2001-01-02	11.5	5.1	7.2	4.2	0,0
4	2001-01-03	11.25	0.4	5.5	0.5	0,7
5	2001-01-04	8.63	0.2	5.6	0.4	0,1
6	2001-01-05	11.92	10.4	7.2	-1.5	0,2
7	2001-01-06	10.67	0	6.5	1.2	0,9
8	2001-01-07	9.17	1.9	9.2	-2.4	0,1
9	2001-01-08	14.29	0	6.6	3.1	1,0
10	2001-01-09	08.04	8.3	6.8	2.4	0,1
11	2001-01-10	11.42	0	6.5	3	0,8
12	2001-01-11	7.54	0.5	10	-3.1	0,4
13	2001-01-12	15.54	6	12.2	-3.3	0,1
14	2001-01-13	4.63	0	13.1	5.5	0,0
15	2001-01-14	03.08	0	5.6	-3.2	0,9

Рисунок 3.1. - Початкові дані для роботи з вітроенергетикою

Після перевірки можна впевнитись, що пропусків в даних немає, всі записи містять в собі значення (рисунок 3.2).



#	Column	Non-Null Count	Dtype
0	DATE	2143 non-null	object
1	WIND	2143 non-null	float64
2	RAIN	2143 non-null	float64
3	T.MAX	1534 non-null	float64
4	T.MIN	1535 non-null	float64
5	CLOUD	2143 non-null	object

Рисунок 3.2. - Результати перевірки пропусків

3.2 Побудова математичної моделі ARIMA та нейронної мережі

Після перевірки початкових даних на пропуски ми можемо приступати до нашого алгоритму. Першим кроком запропонованого алгоритму, є модель ARIMA (автоматична інтегрована регресія ковзного середнього), яка є результатом комбінації трьох фільтрів: компонента AR, інтеграційного фільтра (I) і компонента MA. Подання цієї моделі здійснюється через позначення ARIMA порядку (p, d, q). Подання ARIMA (1, 2, 0) вказує порядок 1 для компонента AR (саморегресивний), порядок 2 для компонента I (інтеграція або диференціація) та останній 0 для MA, де: p - кількість сезонних авторегресивних термінів; d - кількість сезонних відмінностей; q - кількість сезонних переміщень середовищ.

Моделі ARIMA (p, d, q), отримані з використанням програмного продукту SPSS для кожного з часових рядів, представлені відповідно на кожному етапі застосування:

Таблиця 3.1. Порядок моделі ARIMA для короткострокового та середньострокового періоду прогнозування

Short Term	(2, 0, 0)
Medium Term	(1, 0, 1)
Long Term	(2, 0, 1)

На цьому етапі змінна, яка залежить від швидкості, і незалежні змінні, описані вище, такі як вологість, тиск, температура та напрямок, генерують незалежні один від одного результати. Вихідними значеннями, знайденими з представленою моделлю ARIMA, є мінімальна помилка, максимальна помилка, середня помилка, стандартне відхилення та лінійна кореляція. Середня швидкість вітру (СШВ), середня абсолютна помилка (MAE), середньоквадратична помилка (RMSE) та середня абсолютна відсоткова помилка (MAPE) використовувалися для оцінки точності прогнозування змінної швидкості та наведені в таблиці 3.2.

Таблиця 3.2. Результати прогнозування з використанням ARIMA

Model	**Показники якості моделі**		**Показники якості прогнозу**			
	R2	DW	СШВ (m/s)	MAE	RMSE	MAPE (%)
ARIMA (короткостроковий)	0,99	2,15	5.290	0.795	1.777	**15.024**
ARIMA (середньостроковий)	0,99	2,18	6.428	1.132	3.579	17.607
ARIMA (довгостроковий)	0,99	**2,13**	5.739	1.182	5.285	20.591

Наступний крок моделі, виконується з використанням нейронної мережі. Цей крок виконується для прогнозування незалежних змінних, і він використовує результати ARIMA як вхідні змінні, які скорочуються за допомогою аналізу основних компонентів (PCA), який знаходить лінійні комбінації вхідних полів, що скорочують компоненти для використання основних змінних.

Нейронна мережа представляє вісім нейронів у вхідних шарах, два нейрони у прихованому шарі та один нейрон у вихідному шарі, конфігурація використовується для всіх змінних та представлена на зображенні 3.1. Було використано алгоритм навчання помилок зворотного поширення, який коригує ваги мережі, щоб мінімізувати помилку між фактичними значеннями та прогнозованими вихідними даними.

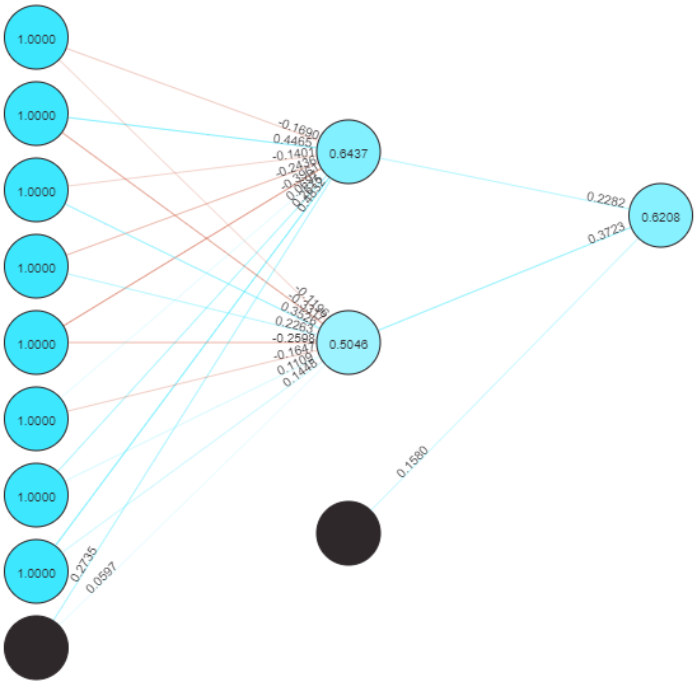


Рисунок 3.3. - Архітектура нейронної мережі

Поділ даних був 80% для навчання та 20% для тестування. Як критерій зупинки використовується максимальний час навчання на модель. Мережа навчалася з допомогою сигмоїдальної дотичної функції активації всім нейронів.

Для оцінки якості використовували середню швидкість вітру (VMED), середню абсолютну помилку (MAE), середньоквадратичну помилку (RMSE) та середню абсолютну помилку у відсотках (MAPE), результати чого наведені в таблиці 3.3.

Таблиця 3.3. Результати прогнозування з використанням APIMA + HM

Model	**Показники якості моделі**		**Показники якості прогнозу**			
	R2	DW	СШВ (m/s)	MAE	RMSE	MAPE (%)
ARIMA + HM (короткостроковий)	0,99	2,08	5.290	0.397	0.889	7.512
ARIMA + HM (середньостроковий)	0,99	2,07	6.428	0.566	1.790	8.804
ARIMA + HM (довгостроковий)	0,99	2,05	5.689	0.616	2.754	10.825

Додатковим кроком можна зробити комбінацію моделі ARIMA та нейронної мережі, запускати результати прогнозу на входи мережі нескінченну кількість разів.

Також окремо було проведено експеримент з використанням тільки нейронної мережі, без використання результатів прогнозування за допомогою APIMA. Для такої ж конфігурації нейронної мережі було взято вхідні початкові дані та пропущено їх через два прихованих шари НМ з використанням алгоритму зворотного розповсюдження помилки та сигномідальну функцію належності. Результати прогнозу представлені в таблиці 3.4.

Таблиця 3.4. Результати прогнозування з використанням НМ

Model	**Показники якості моделі**		**Показники якості прогнозу**			
	R2	DW	СШВ (m/s)	MAE	RMSE	MAPE (%)
НМ (короткостроковий)	0,99	2,07	5.555	0,265	0,592	4.769
НМ (середньостроковий)	0,99	2,05	6.317	0,377	1.193	5.972
НМ (довгостроковий)	0,99	2,02	5.843	0,411	1,836	7.027

В результаті отримаємо зведену таблицю результатів прогнозування швидкості вітру, представлену в таблиці 3.5.

Таблиця 3.5. Зведені результати прогнозування швидкості вітру

Model	**Показники якості моделі**		**Показники якості прогнозу**			
	R2	DW	СШВ (m/s)	MAE	RMSE	MAPE (%)
ARIMA (короткостроковий)	0,99	2,15	5.290	0.795	1.777	15.024
ARIMA (середньостроковий)	0,99	2,18	6.428	1.132	3.579	17.607
ARIMA (довгостроковий)	0,99	2,13	5.739	1.182	5.285	20.591
ARIMA + НМ (короткостроковий)	0,99	2,08	5.290	0.397	0.889	7.512

ARIMA + НМ (середньостроковий)	0,99	2,07	6.428	0.566	1.790	8.804
ARIMA + НМ (довгостроковий)	0,99	2,05	5.689	0.616	2.754	10.825
НМ (короткостроковий)	0,99	2,07	5.555	0,265	0,592	4.769
НМ (середньостроковий)	0,99	2,05	6.317	0,377	1.193	5.972
НМ (довгостроковий)	0,99	2,02	5.843	0,411	1,836	7.027

Як ми бачимо з таблиці, найкращі результати показників якості моделі та прогнозу були отримані з використанням підходу АРІМА + НН. Але необхідно зазначити, що результати, отримані іншими підходами, а саме АРІМА та НМ, також надали якісні результати.

3.3. Прогнозування швидкості вітру та генерованої потужності

Кінцевою метою даної роботи є прогнозування швидкості вітру для можливості прогнозування потужності, що генерується. При заданій швидкості вітру потужність, що виробляється, залежить від типу використовуваного генератора. В якості вітряної турбіни для дослідження була обрана модель Ветрогенератора 48 В 3/4 кВт STORM USE потужністю 4.00 кВт, здатна генерувати 4.00 кВт за середньої швидкості вітру (11 м/с). На рис. 3.2 показано криву потужності вітряної турбіни, де по осі у розташована потужність у кВТ, а по осі х - швидкість у м/с.

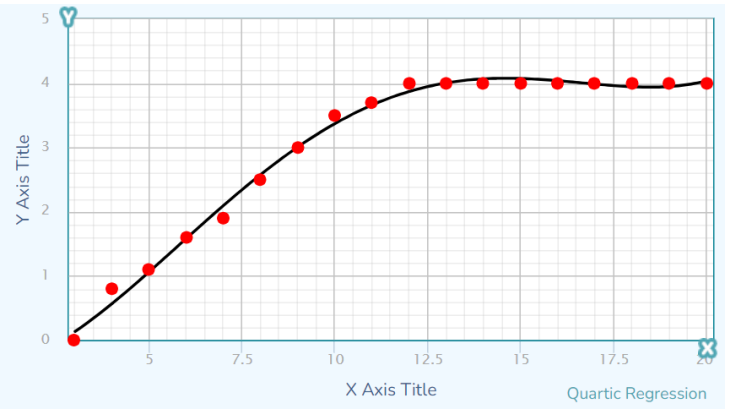


Рисунок 3.4 - Крива потужності вітрової турбіни STORM

Для отримання рівняння кривої потужності генератора використовувалося програмне забезпечення Curve Expert (https://mycurvefit.com/) на основі даних потужності та швидкості. Рівняння, що представляє криву генерації, має вигляд: $P = a + b\ x + cx2 + dx3 + ex4$ та описується коефіцієнтами, представленими в таблиці 3.5.

Таблиця 3.5. - Значення коефіцієнтів кривої потужності

Коефіцієнт	Значення
a	−0.3790678
b	−0.1285376
c	0.1278947
d	−0.009977223
e	0.0002228164

Наступним кроком додаємо отримані коефіцієнти та значення швидкості вітру до математичної моделі і отримаємо графік, представлений на рисунку 3.5 Де синім кольором представлені значення потужності, побудоване на реальних показниках (існуючих), червоним - побудоване на прогнозному значенні швидкості вітру з використанням ARIMA, жовтий колір - побудоване з прогнозним значенням ARIMA + НН, зелений колір - на основі даних НН.

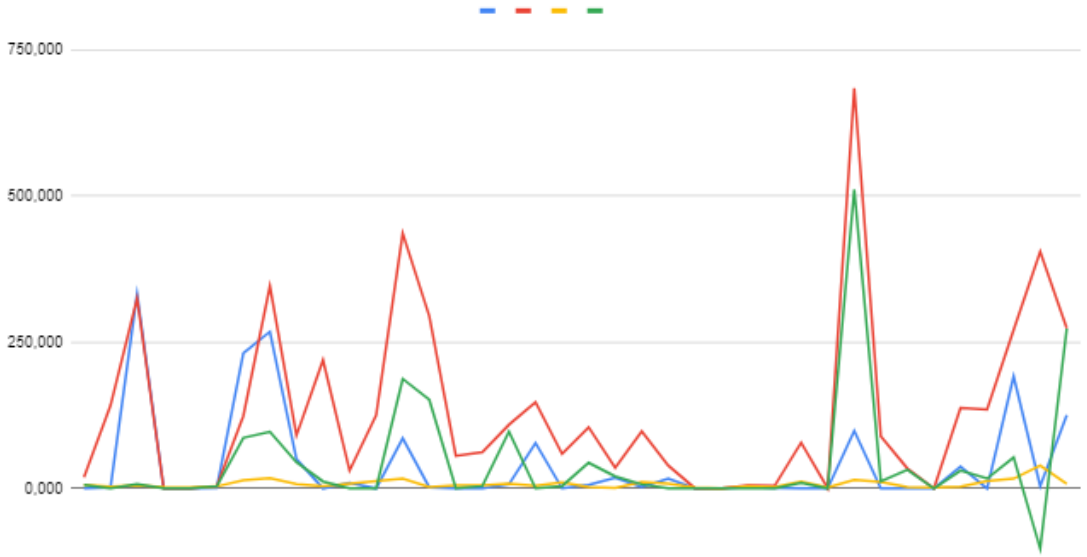


Рисунок 3.5 - Значення потужності на основі прогнозованих значень швидкості вітру

На рис. 3.5 представлений графік річного вироблення вітрогенератором енергетичного рішення. В якості осі x виступає середня швидкість вітру, в якості осі y виступає кількість енергії.

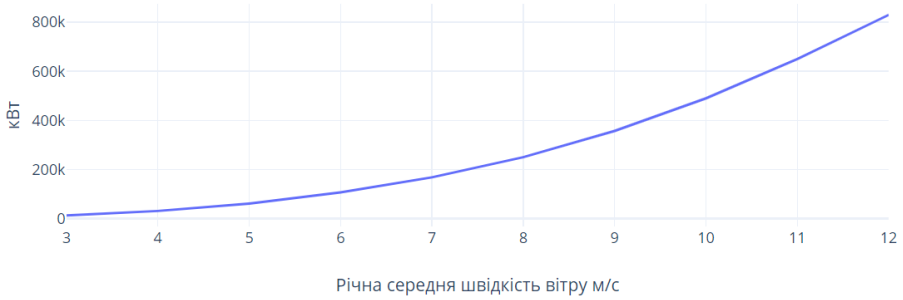


Рисунок 3.3. - Річна виробка енергії в кВт

Щоб отримати рівняння кривої енергії та коефіцієнти, отримане від генератора, використовувалося програмне забезпечення Curve Expert на основі даних енергії та швидкості. Рівняння річної енергії, отримане залежно від швидкості вітру та роботи цієї турбіни, виглядає так:

$$E = a + b \times v + c \times v\text{^}2 + d + e$$

где E = генеруєма енергія;

v = швидкість вітру;

a = −5.359091 × 10^5 ;

в = 3.840793 × 10^5 ;

c = −9.636218 × 10^4 ;

d = 1.500874 × 10^3 ;

e = −3.77331 × 10^2 .

В якості швидкості вітру будемо використовувати значення без прогнозу, тобто реальне, прогнозоване значення за допомогою ARIMA, за допомогою APIMA + HM, а також просто з HM.

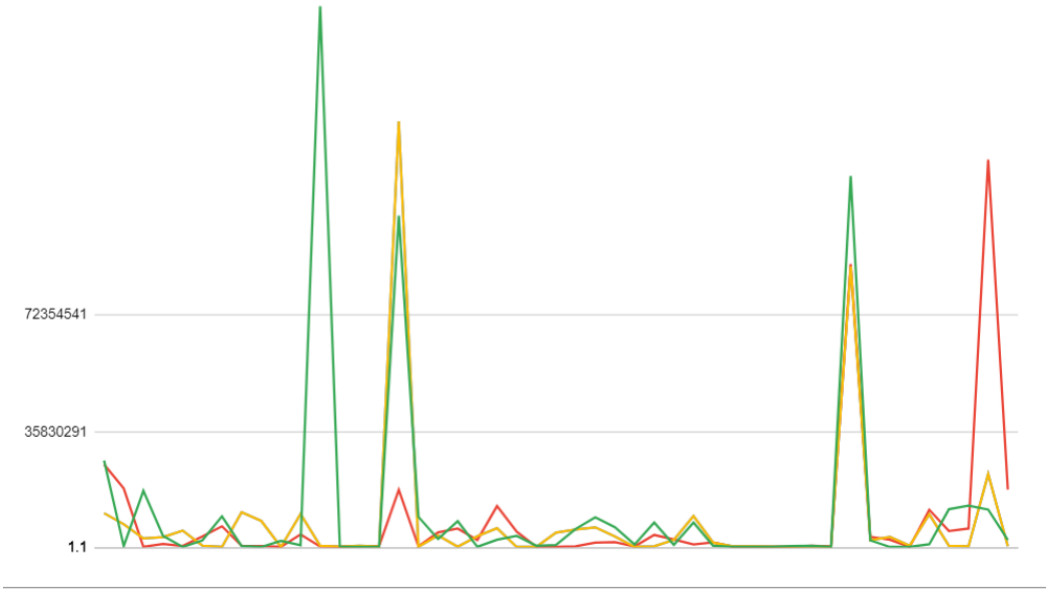


Рисунок 3.6 Значення генеруємой енергії, отриманої за результатами прогнозу

3.4. Реалізація блоків коду програмного продукту

Давайте розглянемо кілька кодових блоків програми, які виявляють функціональні аспекти програми з програмування, повний код якого можна знайти у Додатку А. Оскільки стандартні методи PyQT (бібліотека для побудови інтерфейсу на мові Python) не надають можливості вибирати та виділяти діапазони дат, цей код перевіряє, чи натиснута клавіша Shift, і якщо так, то він виконує зафарбовування дат між початковою та кінцевою:

```
def date_is_clicked(self, date):
    self.format_range(QTextCharFormat())
    if QApplication.instance().keyboardModifiers() & Qt.ShiftModifier and self.b
egin_date:
        self.end_date = date
        self.format_range(self.highlight_format)
    else:
        self.begin_date = date
        self.end_date = None
```

У випадку, коли користувач вибирає одну дату замість діапазону, у нас є блок, який відображає помилку вибору дати і виводить спливаюче повідомлення (див. рис. 3.7):

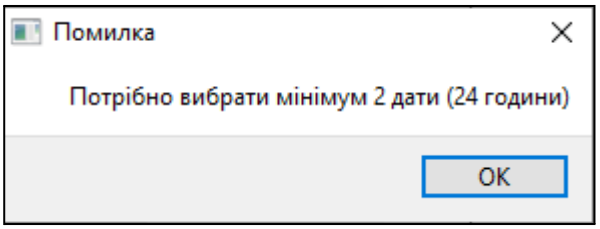


Рис.3.7 Помилка «обраний діапазон в один день»

```
def get_data(self):
    #отримуємо вибрані в календарі дати, та переводимо їх у формат рррр-
мм-дд (рік-місяць-день)
    try:
        s_date = QtCore.QDate.getDate(self.start)
        e_date = QtCore.QDate.getDate(self.end)
    except:
        QMessageBox.about(self, "Помилка", "Потрібно вибрати мінімум 2 дат
и (24 години)")
        Return
```

Ця функція відповідає за побудову графіків. Вона отримує назви та значення, а потім малює діаграму. Наприклад, це може бути використано в меню типу графіків під назвою "Загальний графік":

```
def draw_diagram(self, a_names, w_values, s_values, type_diagram):
        names = a_names
        wind_value = w_values
        sun_value = s_values
        if type_diagram == 'general':
            get_sun_len = len(sun_value)
            all_value = []
            for i in range(get_sun_len):
                all_value.append(wind_value[i] + sun_value[i])
                i += 1
            self.MplWidget.canvas.axes.clear()
            self.MplWidget.canvas.axes.plot(names, all_value)
            self.MplWidget.canvas.axes.set_title('Загальний графік')
            self.MplWidget.canvas.draw()
            self.save_files('Загальний графік', names, all_value) #створюємо таблицю
            plt.clf()
            plt.plot(names, all_value)
            plt.show()
```

Ці блоки коду, які відносяться до розділу "Помилки діапазону дат", дозволяють відображати вікно помилки, якщо обраний діапазон дат не містить наявної інформації про погодні умови.

1) Наприклад, якщо ви вибираєте дні, які вже минули і для яких немає даних про погоду (див. рис. 3.8), цей код викликає вікно помилки:

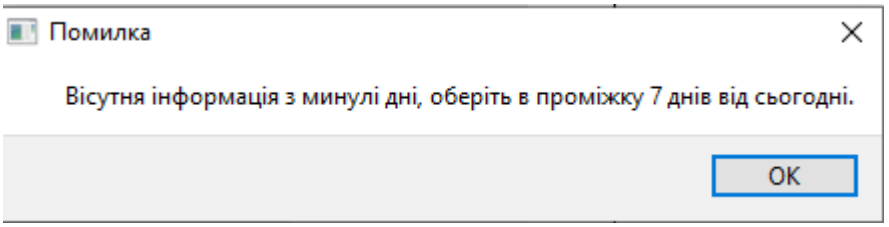


Рис.3.8 Помилка «обраний минулий день»

```
elif diff_range < 0:
        QMessageBox.about(self, "Помилка", "Вісутня інформація з мину
ді дні, оберіть в проміжку 7 днів від сьогодні")
        return
```

2) У разі, коли обраний діапазон дат виходить за межі доступних 7 днів на сайті (див. рис. 3.5), ці блоки коду спричиняють виведення вікна помилки. Таким чином, користувач буде повідомлений про неприпустимість такого вибору та відображено відповідне повідомлення:

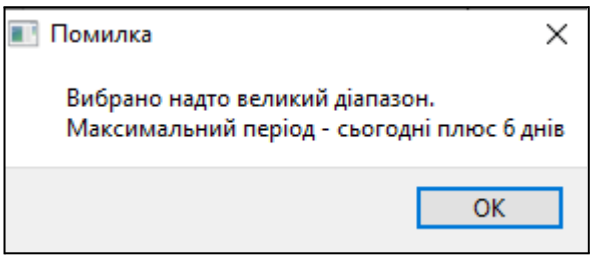


Рис.3.5 Помилка «обраний період більше 7 днів»

```
if diff_range > 6:
        QMessageBox.about(self, "Помилка", "Вибрано надто великий діа
пазон.\nМаксимальний період - сьогодні плюс 6 днів")
        return
```

Ця функція виконує розрахунок вихідної потужності для сонячної панелі моделі. Вона приймає необхідні параметри, такі як площа панелі, ефективність моделі та інші вхідні дані, і повертає обчислену вихідну потужність сонячної панелі «LEAPTON LP-M-72-H-400W HALF-CELL» :

```
def sun_panel_kpd(self, s_name, generate_data):
    #Вираховуємо ККД сонячних панелей
    LEAPTON = [0.20, 400]
    AMERISOLAR = [0.17, 280]
    JA_SOLAR = [0.19, 320]
    timenow = datetime.datetime.now().time().strftime('%H')
    time_index = int(timenow)


self.day_sun = [] #список, який містить значення точок на графіку
if s_name == 'LEAPTON LP-M-72-H-400W HALF-CELL':
  for i in range(generate_data):
    if float(self.cloud_list[time_index + i]) == -1.0:
      get_kpd = float(LEAPTON[1] * float(LEAPTON[0]))

      self.day_sun.append(get_kpd)
```

```
            else:
                get_kpd = float(LEAPTON[1]) * float(LEAPTON[0]) * float(self.clou
d_list[time_index + i])
                self.day_sun.append(get_kpd)
```

Наступні блоки коду відповідають за сповіщення користувача у випадку:

1) Якщо користувач намагається побудувати графік для вітрогенератора, але не вибрано жодної моделі вітрогенератора (див. рис. 3.9), код виведе сповіщення користувачу:

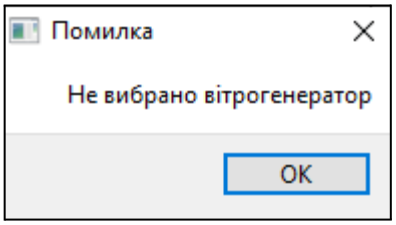


Рис.3.9 Помилка «не було обрано вітрогенератора»

```
if graphics_type == 'Графік для вітрогенератора':
    if wind_generator == 'Оберіть модель':
        QMessageBox.about(self, 'Помилка', 'Не вибрано вітрогенератор')
        Return
```

Наступні блоки коду відповідають за сповіщення користувача у випадку:

2) Якщо користувач намагається побудувати графік для сонячної панелі, але не вибрано жодної моделі СП (див. рис. 3.10), код виведе сповіщення користувачу:

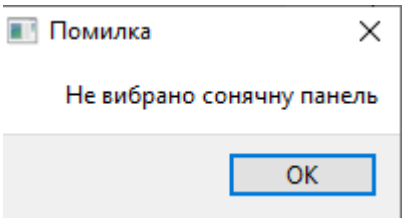


Рис.3.10 Помилка «не було обрано сонячної панелі»

```
    else:
        self.draw_diagram(days, self.day_wind, self.day_sun, 'wind')
elif graphics_type == 'Графік для сонячної панелі':
    if sun_panel == 'Оберіть модель':
        QMessageBox.about(self, 'Помилка', 'Не вибрано сонячну панель')
        Return
```

Наступні блоки коду відповідають за сповіщення користувача у випадку:

3) Якщо не було обрано жодної моделі альтернативного джерела енергії (див. рис. 3.11), код виведе сповіщення користувачу:

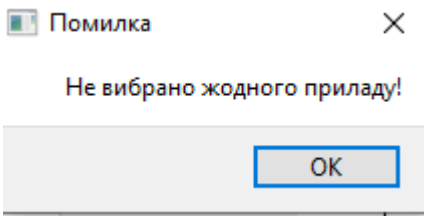


Рис.3.11 Помилка «не було обрано приладу»

```
    else:
        self.draw_diagram(days, self.day_wind, self.day_sun, 'sun')
elif graphics_type == 'Загальний графік':
    if sun_panel == 'Оберіть модель' and wind_generator == 'Оберіть модель':
        QMessageBox.about(self, 'Помилка', 'Не вибрано жодного приладу!')
    else:
        self.draw_diagram(days, self.day_wind, self.day_sun, 'general')
```

Ця функція виконує розрахунок вихідної потужності для вітрогенератора моделі "Flamingo aero FA - 4.4". Вона приймає необхідні параметри, які включають в себе характеристики даної моделі вітрогенератора, і обчислює вихідну потужність, яку цей вітрогенератор може забезпечити:

```
def wind_generator_kpd(self, g_name, generate_data):
    #Функція підрахунку значень для ККД вітрогенераторів
    self.day_wind = []#список який містить значення точок на графіку
    timenow = datetime.datetime.now().time().strftime('%H')
    time_index = int(timenow)
    self.wind = []
    if g_name == 'Flamingo aero FA - 4. 4 ':
        for i in range(generate_data):
            inc = float(self.wind_list[i + time_index])
            if inc >= 2.5:
                if inc >= 8:
                    self.day_wind.append(1600.0)
                elif inc < 8 and inc >= 6.5:
                    self.day_wind.append(1600.0 * 0.8)
                elif inc < 6.5 and inc >= 4.5:
                    self.day_wind.append(1600.0 * 0.6)
                else:
                    get_kpd = (float(self.wind_list[i + time_index]) / 8.0) * 1600.0
                    self.day_wind.append(get_kpd)
            else:

self.day_wind.append(0)
            i += 1
```

Цей блок коду відповідає за зміну текстового значення хмарності, яке знаходиться на сайті, на числовий формат зменшення ККД (коефіцієнт корисної дії) сонячної батареї. Це дозволяє користувачеві ввести числове значення, яке відображатиме рівень хмарності і впливатиме на ККД сонячної батареї:

```
def cloud_converter(self, c_name):
        if 'Малооблачно' in c_name:
            result = 0.1
        elif 'Значительная облачность' in c_name:
            result = 0.2
        elif 'Пасмурно' in c_name:
            result = 0.35
        elif 'Небольшой дождь' in c_name:
            result = 0.45
        elif 'Ливневый дождь' in c_name:
            result = 0.65
        else:
            result = -1.0
```

Функція збереження даних в таблицю (excel) :

```
def save_files(self, diagram_name, hourly, data_1):
    name_generation = str(diagram_name) + str(datetime.datetime.now().date())
+ '.csv'
    with open(name_generation, mode='w') as employee_file:
        employee_writer = csv.writer(employee_file)
        employee_writer.writerow(hourly) #День та година, верхній рядок
        employee_writer.writerow(data_1) #Значення, нижній рядок
```

Цей блок коду включає функцію, яка збирає інформацію з метеорологічного сайту. Ця функція виконує запит до метеорологічного API (інтерфейсу програмування додатків), отримує відповідь з поточними метеорологічними даними, такими як температура, вологість, швидкість вітру та інші параметри. Потім ці дані можуть бути використані для подальшого аналізу або відображення на веб-сторінці:

```
def parse_weather(self):
    site_url = 'https://www.meteoservice.ru/weather/hourly/kiev#' #основна частина адреси сайту для парсингу
    #списки для зберігання інформації про час, хмарність, температуру, вітер
    self.time_list = []
    self.cloud_list = []
    self.wind_list = []
    self.day_list = []
    #Збираємо дані з сайту
    final_url = site_url + str(datetime.datetime.now().today().strftime('%Y%m%d'))
    urlPage = urlopen(final_url).read()
    weatherPage = BeautifulSoup(urlPage, 'lxml')
    for day in weatherPage.findAll('div', {'class':'small-12 columns forecast days-6'}):
        for time in day.findAll('span', {'class':'h3'}):
            self.time_list.append(time.text + ':00')
        for cloud in day.findAll('div', {'class':'column value show-for-large text-left font-smaller font-condensed'}):
            self.cloud_list.append(self.cloud_converter(cloud.text))
        for wind in day.findAll('span', {'class':'font-smaller'}):
            wind_speed = str(wind.text).replace(' м/с', '')
            if wind_speed[0].isdigit():
                if len(wind_speed) > 1:
                    new_wind = wind_speed.split('–')
                    result = (int(new_wind[0]) + int(new_wind[1])) / 2
                    self.wind_list.append(result)
                else:
                    self.wind_list.append(wind_speed)
```

Звучить як великий обсяг коду для програми прогнозування електрогенерації у альтернативних джерелах енергії. З 487 рядків коду, можна припустити, що в програмі реалізовано широкий спектр функціональності та логіки. Таке стисле рішення вказує на ефективну організацію коду та його оптимізацію. Це дозволяє забезпечити прогнозування електрогенерації з використанням альтернативних джерел енергії з високою точністю і ефективністю.

4. ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ

В заданому розділі буде проведено оцінювання основних характеристик для майбутнього програмного продукту, що спеціалізується на дослідженні демографічного стану.

Дана реалізація буде сприяти проведенню усіх необхідних досліджень, що дасть змогу якісно дослідити питання не лише в Україні, проте у всьому світі.

Також в даному дослідженні показано різні варіанти реалізації для забезпечення найбільш коректної та оптимальної стратегії вибору, що має вплив на економічні фактори та сумісність з майбутнім програмним продуктом. Для цього застосовувався апарат функціонально-вартісного аналізу.

Функціонально-вартісний аналіз (ФВА) передбачає собою технологію, що дозволяє оцінити реальну вартість продукту або послуги незалежно від організаційної структури компанії. ФВА проводиться з метою виявлення резервів зниження витрат за рахунок ефективніших варіантів виробництва, кращого співвідношення між споживчою вартістю виробу та витратами на його виготовлення. Для проведення аналізу використовується економічна, технічна та конструкторська інформація.

Алгоритм функціонально-вартісного аналізу включає в себе визначення послідовності етапів розробки продукту, визначення повних витрат (річних) та кількості робочих часів, визначення джерел витрат та кінцевий розрахунок вартості програмного продукту.

4.1 Постановка задачі проектування

У роботі застосовується метод ФВА для проведення техніко-економічного аналізу розробки системи прогнозу стійкості

фінансових показників. Оскільки рішення стосовно проектування та реалізації компонентів, що розробляється, впливають на всю систему, кожна окрема підсистема має її задовольняти. Тому фактичний аналіз представляє собою аналіз функцій програмного продукту, призначеного для збору, обробки та проведення аналізу даних по компанії.

Технічні вимоги до програмного продукту є наступні:

- функціонування на персональних комп'ютерах із стандартним набором компонентів;
- зручність та зрозумілість для користувача;
- швидкість обробки даних та доступ до інформації в реальному часі;
- можливість зручного масштабування та обслуговування;
- мінімальні витрати на впровадження програмного продукту.

4.2 Обґрунтування функцій програмного продукту

Головна функція F_0 – розробка можливого програмного продукту, яка дозволяє аналізувати різні характеристики, що безпосередньо впливають на стійкість підприємства. Беручи за основу цю функцію, можна виділити наступні:

F_1 – вибір самої програми.

F_2 – якісний аналіз даних.

F_3 – графічні показники.

Кожна з цих функцій має декілька варіантів реалізації:

Функція F_1:

а) Python.

б) Дані погодних умов.

Функція F_2.

а) Застосування штучного інтелекту.

б) Створення обчислень значень.

Функція F_3:

а) Використання вхідних даних.

б) Створення графіків.

Варіанти реалізації основних функцій наведені у морфологічній карті системи (рис. 4.1).

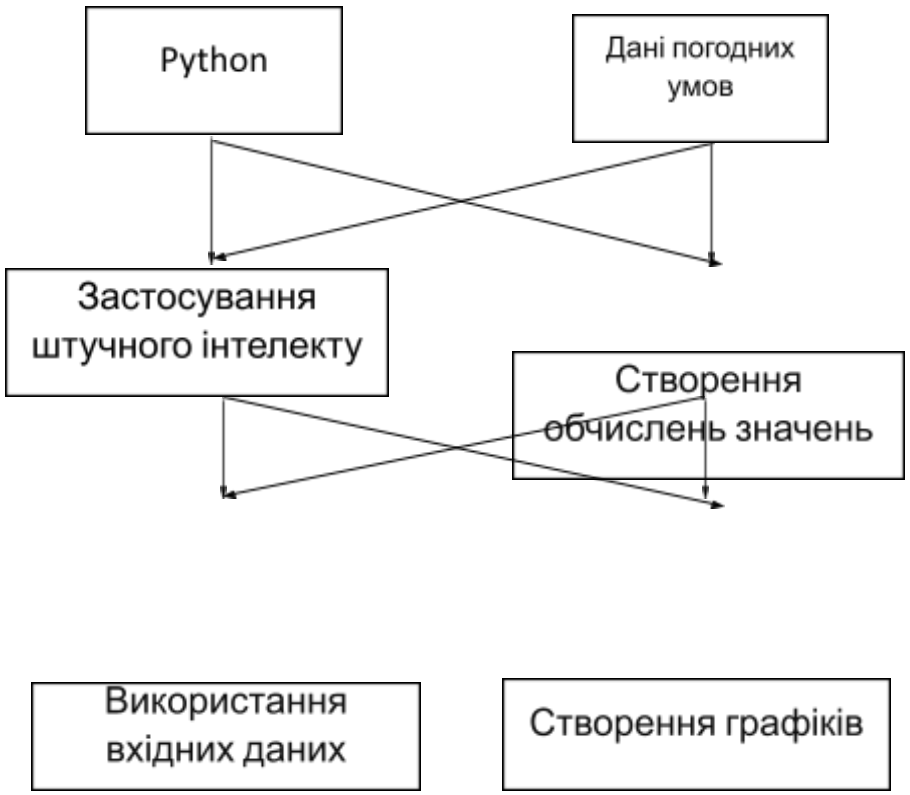


Рисунок 4.1 – Морфологічна карта

Морфологічна карта відображає множину всіх можливих варіантиів основних функцій. Позитивно-негативна матриця показана в таблиці 4.1.

Таблиця 4.1 - Позитивно-негативна матриця

Функції	Варіанти реалізації	Переваги	Недоліки
F_1	*А*	Загальнодоступна програма, доступність багатьох бібліотек	Необхідність повної реалізації алгоритму
	Б	Доступна в реалізації програма для різних обчислень	На написання коду необхідно мати базові навички та вміння
F_2	*А*	Доступність та легкість при написанні	Іноді не відповідає задачі яку треба розв’язати
	Б	Ідеально описують усі необхідні характеристики	Достатньо затратно реалізовувати свої алгоритми для подальшої реалізації
F_3	*А*	Загально прийнята реалізація	Іноді не відповідає очікуваним значенням
	Б	При виконанні власних досліджень краще може передавати висновки	Необхідно достатньо багато часу для написання програми для побудови та знаходження всього необхідного в задачі.

На основі аналізу позитивно-негативної матриці робимо висновок, що при розробці програмного продукту деякі варіанти реалізації функцій варто відкинути, тому, що вони не відповідають поставленим перед програмним продуктом задачам. Ці варіанти відзначені у морфологічній карті.

Функція F_1:

Перевагу даємо багатофункціональності та широкому вибору бібліотек. Для спрощення роботи по написанню коду варіант Б має бути відкинутий.

Функція F_2:

Програма допускає обрання обох варіантів. Можливо використати варіанти А чи Б.

Функція F_3:

Реалізація першого варіанту є сприйнятливою для програми. Це варіант А.

Таким чином, будемо розглядати такий варіанти реалізації ПП:

F_1а – F_2а – F_3а

F_1а – F_2б – F_3а

Для оцінювання якості розглянутих функцій обрана система параметрів, описана нижче.

4.3 Обґрунтування системи параметрів програмного продукту

На основі даних, розглянутих вище, визначаються основні параметри вибору, які будуть використані для розрахунку коефіцієнта технічного рівня.

Для того, щоб охарактеризувати програмний продукт, будемо використовувати наступні параметри:

- *X1* – швидкодія мови програмування;
- *X2* – об’єм пам’яті для обчислень та збереження даних;
- *X3* – час навчання даних;
- *X4* – потенційний об’єм програмного коду.

Гірші, середні і кращі значення параметрів вибираються на основі вимог замовника й умов, що характеризують експлуатацію програмного продукту, як показано у таблиці 4.2.

Таблиця 4.2 - Основні параметри програмного продукту

Назва Параметра	Умовні позначення	Одиниці виміру	Значення параметра		
			гірші	середні	кращі
Швидкодія мови програмування	X1	оп/мс	910	11010	15010
Об’єм пам’яті	X2	Мб	256	128	64
Час попередньої обробки даних	X3	мс	905	605	405
Потенційний об’єм програмного коду	X4	кількість рядків коду	3010	1510	1010

За даними таблиці 4.3 будуються графічні характеристики параметрів – рис. 4.2 – рис. 4.5.

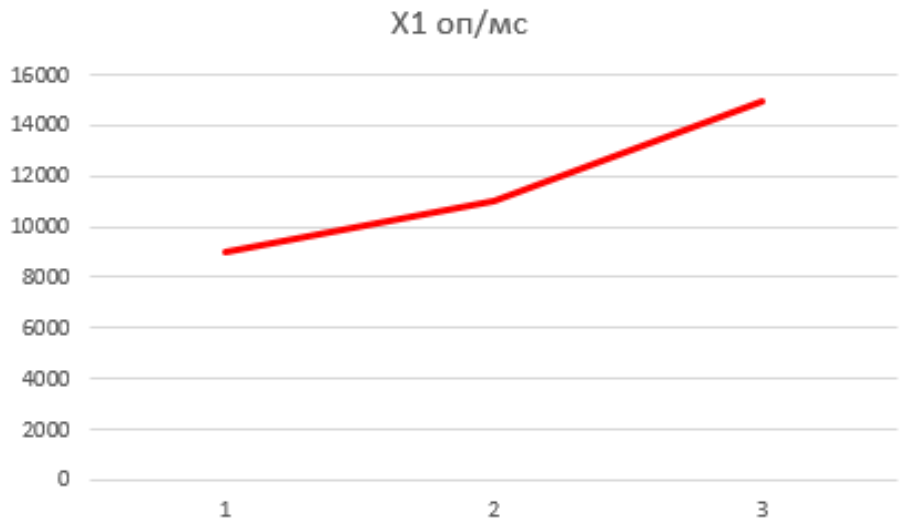


Рисунок 4.2 – X1, швидкодія мови програмування

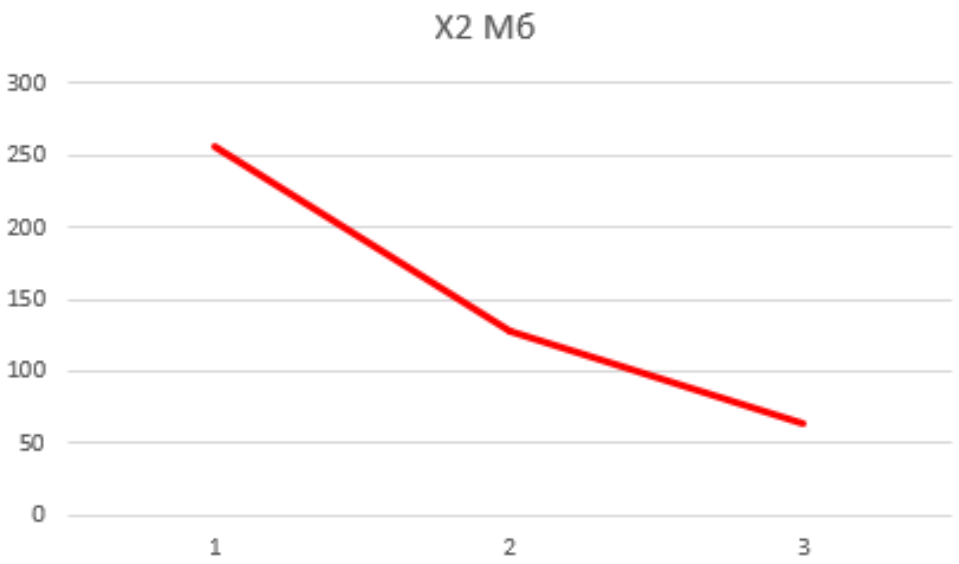


Рисунок 4.3 – X2, об’єм пам’яті

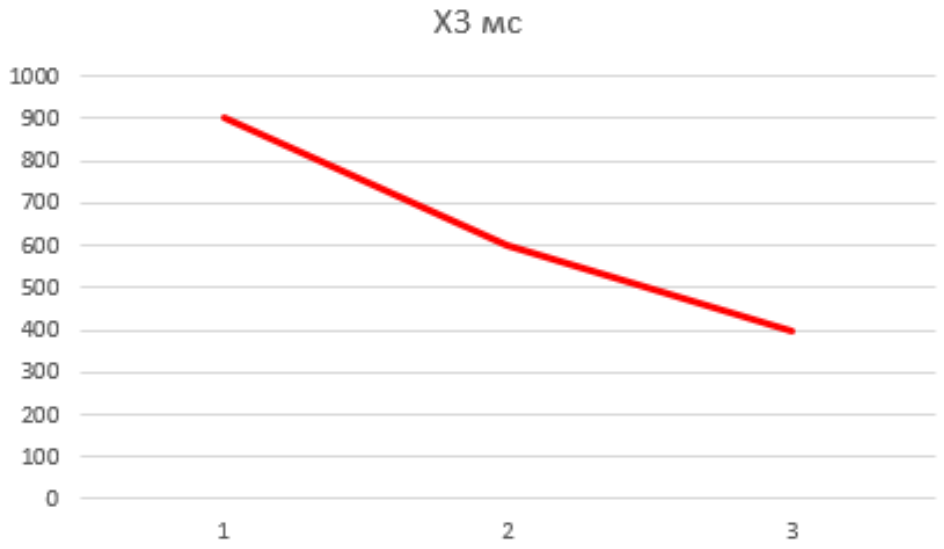


Рисунок 4.4 – X3, час попередньої обробки даних

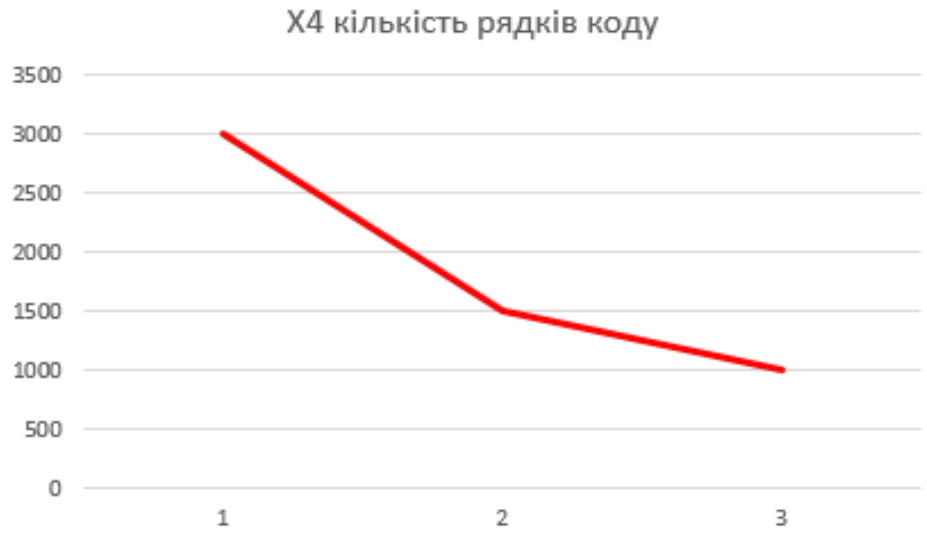


Рисунок 4.5 – X4, потенційний об’єм програмного коду

4.4 Аналіз експертного оцінювання параметрів

Після детального обговорення й аналізу кожний експерт оцінює ступінь важливості кожного параметру для конкретно поставленої цілі – розробка програмного продукту, який дає найбільш точні результати при знаходженні параметрів моделей адаптивного прогнозування і обчислення прогнозних значень.

Значимість кожного параметра визначається методом попарного порівняння. Оцінку проводить експертна комісія із 7 людей. Визначення коефіцієнтів значимості передбачає:

- визначення рівня значимості параметра шляхом присвоєння різних рангів;
- перевірку придатності експертних оцінок для подальшого використання;
- визначення оцінки попарного пріоритету параметрів;
- обробку результатів та визначення коефіцієнту значимості.

Результати експертного ранжування наведені у таблиці 4.3.

Таблиця 4.3 - Результати ранжування параметрів

Позначення параметра	Назва параметра	Одиниці виміру	Ранг параметра за оцінкою експерта							Сума рангів R_i	Відхилення Δ_i	Δ_i^2
			1	2	3	4	5	6	7			
X1	Швидкодія мови програмування	Оп/мс	1	3	3	1	2	1	2	13	-8	64
X2	Об’єм пам’яті	Mb	3	5	3	4	4	3	5	27	6	36
X3	Час попередньої обробки даних	мс	3	1	2	2	3	3	2	16	-5	25
X4	Потенційний об’єм програмного коду	Кількість рядків коду	5	3	4	5	3	5	3	28	7	49
	Разом		12	12	12	12	12	12	12	84	0	174

Для перевірки степені достовірності експертних оцінок, визначимо наступні параметри:

а) сума рангів кожного з параметрів і загальна сума рангів:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 84, \ \#(4.1)$$

де N – число експертів,

n – кількість параметрів;

б) середня сума рангів:

$$T = \frac{1}{n} R_{ij} = 21 \ \#(4.2)$$

в) відхилення суми рангів кожного параметра від середньої суми рангів:

$$\Delta_i = R_i - T. \ \#(4.3)$$

Сума відхилень по всім параметрам повинна дорівнювати 0;

г)загальна сума квадратів відхилення:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 174. \ \#(4.4)$$

Порахуємо коефіцієнт узгодженості:

$$W = \frac{12S}{N^2\left(n^3 - n\right)} = \frac{12 \cdot 174}{7^2\left(4^3 - 4\right)} = 0,710 > W_k = 0,67. \ \#(4.5)$$

Ранжування можна вважати достовірним, тому що знайдений коефіцієнт узгодженості перевищує нормативний, котрий дорівнює 0,67.

Скориставшись результатами ранжирування, проведемо попарне порівняння всіх параметрів і результати занесемо у таблицю 4.4.

Таблиця 4.4 - Попарне порівняння параметрів.

Параметри	Експерти							Кінцева оцінка	Числове значення
	1	2	3	4	5	6	7		
X1 і X2	<	<	=	<	<	<	<	<	0,5
X1 і X3	<	>	>	<	<	<	=	<	0,5
X1 і X4	<	=	<	<	<	<	<	<	0,5
X2 і X3	=	>	>	>	>	=	>	>	1,5
X2 і X4	<	>	<	<	>	<	>	<	0,5
X3 і X4	<	<	<	<	=	<	<	<	0,5

Числове значення, що визначає ступінь переваги *i*–го параметра над *j*–тим, a_{ij} визначається по формулі:

$$a_{ij} = \{1.5 \text{ при } X_i > X_j \ 1.0 \text{ при } X_i = X_j \ 0.5 \text{ при } X_i < X_j \ . \#(4.6)$$

З отриманих числових оцінок переваги складемо матрицю А=$\left\| a_{ij} \right\|$.

Для кожного параметра зробимо розрахунок вагомості $K_{вi}$ за наступними формулами:

$$K_{\text{вi}} = \frac{b_i}{\sum_{i=1}^{n} b_i} \#(4.7)$$

$$b_i = \sum_{i=1}^{N} a_{ij} \#(4.8)$$

Відносні оцінки розраховуються декілька разів доти, поки наступні значення не будуть незначно відрізнятися від попередніх (менше 2%). На другому і наступних кроках відносні оцінки розраховуються за наступними формулами:

$$K_{\text{вi}} = \frac{b_i^{'}}{\sum_{i=1}^{n} b_i^{'}}, \#(4.9)$$

$$b_i^{'} = \sum_{i=1}^{N} a_{ij} b_j \#(4.10)$$

Як видно з таблиці 4.5, різниця значень коефіцієнтів вагомості не перевищує 2%, тому більшої кількості ітерацій не потрібно.

Таблиця 4.5 - Розрахунок вагомості параметрів

Параметри x_i	Параметри x_j				Перша ітер.		Друга ітер.		Третя ітер	
	X1	X2	X3	X4	b_i	$K_{\text{вi}}$	b_i^1	$K_{\text{вi}}^1$	b_i^2	$K_{\text{вi}}^2$
X1	1	0,5	0,5	0,5	2,5	0,16	9,25	0,16	34,125	0,16
X2	1,5	1	1,5	0,5	4,5	0,28	16,25	0,28	59,125	0,28
X3	1,5	0,5	1	0,5	3,5	0,22	12,25	0,21	41,875	0,2
X4	1,5	1,5	1,5	1	5,5	0,34	21,25	0,35	77,875	0,36
Всього:					16	1	59	1	213	1

4.5 Аналіз рівня якості варіантів реалізації функцій

Визначаємо рівень якості кожного варіанту виконання основних функцій окремо.

Абсолютні значення параметрів *X2* (Об'єм пам'яті), *X3* (час попередньої обробки даних) та *X4* (потенційний об'єм програмного коду) відповідають технічним вимогам умов функціонування даного ПП.

Абсолютне значення параметра *X1* (швидкість роботи мови програмування) обрано не найгіршим.

Коефіцієнт технічного рівня для кожного варіанта реалізації ПП розраховується так (таблиця 4.6):

$$K_K(j) = \sum_{i=1}^{n} K_{\text{в}i,j} B_{i,j}, \ \#(4.11)$$

де n – кількість параметрів;

$K_{\text{в}i}$ – коефіцієнт вагомості i–го параметра;

B_i – оцінка i–го параметра в балах.

Таблиця 4.6 - Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основні функції	Варіант реалізації функції	Параметри	Абсолютне значення параметра	Бальна оцінка параметра	Коефіцієнт вагомості параметра	Коефіцієнт рівня якості
F1	А	X1	11010	22	0,16	4,7
F3	А	X2	64	18	0,28	6,2
	Б	X3	128	19	0,2	3,8
F4	А	X4	1010	23	0,36	8,28

За даними з таблиці 4.6 за формулою:

$$K_K = K_{\text{ТУ}}\left[F_{1k}\right] + K_{\text{ТУ}}\left[F_{2k}\right] + \ldots + K_{\text{ТУ}}\left[F_{zk}\right], \ \#(4.12)$$

визначаємо рівень якості кожного з варіантів:

$$K_{K1} = 4{,}7 + 6{,}2 + 8{,}28 = 19{,}18;$$
$$K_{K2} = 4{,}7 + 3{,}8 + 8{,}28 = 16{,}78.$$

Як видно з розрахунків, кращим є 2 варіант, для якого коефіцієнт технічного рівня має найбільше значення.

4.6 Економічний аналіз варіантів розробки ПП

Для визначення вартості розробки ПП спочатку проведемо розрахунок трудомісткості.

Всі варіанти включають в себе два окремих завдання:

1. Розробка проекту програмного продукту;

2. Розробка програмної оболонки;

Завдання 1 за ступенем новизни відноситься до групи А, завдання 2 – до групи Б. За складністю алгоритми, які використовуються в завданні 1 належать до групи 1; а в завданні 2 – до групи 3.

Для реалізації завдання 1 використовується довідкова інформація, а завдання 2 використовує інформацію у вигляді даних.

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань.

Загальна трудомісткість обчислюється як:

$$\mathrm{Т_О} = \mathrm{Т_Р} \cdot \mathrm{К_П} \cdot \mathrm{К_{СК}} \cdot \mathrm{К_М} \cdot \mathrm{К_{СТ}} \cdot \mathrm{К_{СТ.М}}, \#(4.13)$$

де $\mathrm{Т_Р}$ – трудомісткість розробки ПП;

$\mathrm{К_П}$ – поправочний коефіцієнт;

$\mathrm{К_{СК}}$ – коефіцієнт на складність вхідної інформації;

$\mathrm{К_М}$ – коефіцієнт рівня мови програмування;

$К_{СТ}$ – коефіцієнт використання стандартних модулів і прикладних програм;

$К_{СТ.М}$ – коефіцієнт стандартного математичного забезпечення

Для першого завдання, виходячи із норм часу для завдань розрахункового характеру степеню новизни А та групи складності алгоритму 1, трудомісткість дорівнює: $Т_Р$ =33 людино-днів. Поправочний коефіцієнт, який враховує вид нормативно-довідкової інформації для першого завдання: $К_П$ = 1.9. Поправочний коефіцієнт, який враховує складність контролю вхідної та вихідної інформації для всіх семи завдань рівний 1: $К_{СК}$ = 1. Оскільки при розробці першого завдання використовуються стандартні модулі, врахуємо це за допомогою коефіцієнта $К_{СТ}$ = 0.7. Тоді загальна трудомісткість програмування першого завдання дорівнює:

$$Т_1 = 33 \cdot 1.9 \cdot 0.7 = 43,89 \text{ людино-днів.}$$

Проведемо аналогічні розрахунки для подальших завдань.

Для другого завдання (використовується алгоритм третьої групи складності, степінь новизни Б), тобто $Т_Р$ =27 людино-днів, $К_П$ =1.2, $К_{СК}$ = 1, $К_{СТ}$ =0.7:

$$Т_2 = 27 \cdot 1.2 \cdot 0.7 = 22,68 \text{ людино-днів.}$$

Складаємо трудомісткість відповідних завдань для кожного з обраних варіантів реалізації програми, щоб отримати їх трудомісткість:

$$Т_I = (43,89 + 22,68 + 4,8 + 22,68) \cdot 8 = 752,4 \text{ людино-годин.}$$

$$Т_{II} = (43,89 + 22,68 + 6.91 + 22,68) \cdot 8 = 769,28 \text{ людино-годин.}$$

Найбільш високу трудомісткість має варіант II.

В розробці беруть участь два програмісти з окладом 19000 грн., один аналітик в області даних з окладом 22000. Визначимо середню зарплату за годину за формулою:

$$\text{СЧ} = \frac{\text{М}}{T_m \cdot t} \text{грн.}, \ \#(4.14)$$

де М – місячний оклад працівників;

T_m – кількість робочих днів тиждень;

t – кількість робочих годин в день.

$$\text{СЧ} = \frac{19000+19000+22000}{3 \cdot 21 \cdot 8} = 119,05 \ \text{грн.} \ \#(4.15)$$

Тоді, розрахуємо заробітну плату за формулою:

$$\text{СЗП} = \text{С}_\text{ч} \cdot \text{Т}_i \cdot \text{КД}, \#(4.16)$$

де $С_ч$– величина погодинної оплати праці програміста;

$Т_i$ – трудомісткість відповідного завдання;

$К_Д$ – норматив, який враховує додаткову заробітну плату.

Зарплата розробників за варіантами становить:

I. $С_{ЗП} = 119,05 \cdot 752,4 \cdot 1.2 = 107\,487,86$ грн.

II. $С_{ЗП} = 119,05 \cdot 769,28 \cdot 1.2 = 109\,899,34$ грн.

Відрахування на єдиний соціальний внесок становить 22%:

I. $С_{ВІД} = С_{ЗП} \cdot 0.22 = 107\,487,86 \cdot 0.22 = 23\,647,33$ грн.

II. $С_{ВІД} = С_{ЗП} \cdot 0.22 = 109\,899,34 \cdot 0.22 = 24177,85$ грн.

Тепер визначимо витрати на оплату однієї машино-години. ($С_М$)

Так як одна ЕОМ обслуговує одного програміста з окладом 18000 грн., з коефіцієнтом зайнятості 0,2 то для однієї машини отримаємо:

$$С_Г = 12 \cdot М \cdot К_З = 12 \cdot 19000 \cdot 0,2 = 45600 \text{ грн.}$$

З урахуванням додаткової заробітної плати:

$$С_{ЗП} = С_Г \cdot (1 + К_З) = 45600 \cdot (1 + 0.2) = 54720 \text{ грн.}$$

Відрахування на соціальний внесок:

$$С_{ВІД} = С_{ЗП} \cdot 0.22 = 54720 \cdot 0,22 = 12038,4 \text{ грн.}$$

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 20000 грн.

$$С_А = К_{ТМ} \cdot К_А \cdot Ц_{ПР} = 1.2 \cdot 0.25 \cdot 20000 = 6000 \text{ грн.,}$$

де $К_{ТМ}$– коефіцієнт, який враховує витрати на транспортування та монтаж приладу у користувача;

$К_А$– річна норма амортизації;

$Ц_{ПР}$– договірна ціна приладу.

Витрати на ремонт та профілактику розраховуємо як:

$$С_Р = К_{ТМ} \cdot Ц_{ПР} \cdot К_Р = 1.2 \cdot 20000 \cdot 0.06 = 1440 \text{ грн.,}$$

де $К_Р$– відсоток витрат на поточні ремонти.

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$$Т_{ЕФ} = (Д_К – Д_В – Д_С – Д_Р) \cdot t_З \cdot К_В = (365 – 104 – 12 – 16) \cdot 8 \cdot 0.7 =$$

$$= 1304,8 \text{ години,}$$

де $Д_К$ – календарна кількість днів у році;

$Д_В$, $Д_С$ – відповідно кількість вихідних та святкових днів;

$Д_Р$ – кількість днів планових ремонтів устаткування;

t –кількість робочих годин в день;

$К_В$– коефіцієнт використання приладу у часі протягом зміни.

Витрати на оплату електроенергії розраховуємо за формулою:

$$C_{ЕЛ} = T_{ЕФ} \cdot N_C \cdot K_З \cdot Ц_{ЕН} = 1304,8 \cdot 0,2 \cdot 0,3 \cdot 4,79 = 1000 \text{ грн.},$$

де N_C – середньо-споживча потужність приладу;

$K_З$– коефіцієнтом зайнятості приладу;

$Ц_{ЕН}$ – тариф за 1 КВт-годин електроенергії.

Накладні витрати розраховуємо за формулою:

$$C_Н = Ц_{ПР} \cdot 0.67 = 20000 \cdot 0,67 = 13400 \text{ грн.}$$

Тоді, річні експлуатаційні витрати будуть:

$$C_{ЕКС} = C_{ЗП} + C_{ВІД} + C_А + C_Р + C_{ЕЛ} + C_Н, \#(4.17)$$

$$C_{ЕКС} = 54720 + 12038,4 + 6000 + 1440 + 1000 + 13400 = 88598,4 \text{ грн.}$$

Собівартість однієї машино-години ЕОМ дорівнюватиме:

$$C_{М\text{-}Г} = C_{ЕКС} / T_{ЕФ} = 88598,4 / 1304,8 = 67,9 \text{ грн/год.}$$

Оскільки в даному випадку всі роботи, які пов‘язані з розробкою програмного продукту ведуться на ЕОМ, витрати на оплату машинного часу, в залежності від обраного варіанта реалізації, складає:

$$C_М = C_{М-Г} \cdot T, \#(4.18)$$

I. $C_М = 67,9 \cdot 752,4 = 51087,96$ грн.

II. $C_М = 67,9 \cdot 769,28 = 52234,11$ грн.

Накладні витрати складають 67% від заробітної плати:

$$C_Н = C_{ЗП} \cdot 0,67, \#(4.19)$$

I. $C_Н = 107\,487,86 \cdot 0,67 = 72016,29$ грн.

II. $C_Н = 109\,899,34 \cdot 0,67 = 73632,56$ грн.

Отже, вартість розробки ПП за варіантами становить:

$$С_{ПП} = С_{ЗП} + С_{ВІД} + С_{М} + С_{Н}, \#(4.20)$$

I. $С_{ПП}$ = 107 487,86 + 23 647,33 + 51087,96 + 72016,29 = 254239,44 грн.

II. $С_{ПП}$ = 109 899,34 + 24177,85 + 52234,11 + 73632,56 = 259943,86 грн.

4.7 Вибір кращого варіанту ПП техніко-економічного рівня

Розрахуємо коефіцієнт техніко-економічного рівня за формулою:

$$К_{ТЕР j} = К_{К j} / С_{Ф j}, \#(4.21)$$

$$К_{ТЕР1} = 19,18 / 254239,44 = 7,544 \cdot 10^{-5},$$

$$К_{ТЕР2} = 16,78 / 259943,86 = 6,455 \cdot 10^{-5}.$$

Як бачимо, найбільш ефективним є перший варіант реалізації програми з коефіцієнтом техніко-економічного рівня $К_{ТЕР1} = 7,544 \cdot 10^{-5}$.

Після виконання функціонально-вартісного аналізу програмного комплексу що розроблюється, можна зробити висновок, що з альтернатив, що залишились після першого відбору двох варіантів виконання програмного комплексу оптимальним є перший варіант реалізації програмного продукту. У нього виявився найкращий показник техніко-економічного рівня якості $К_{ТЕР} = 7,544 \cdot 10^{-5}$.

Цей варіант реалізації програмного продукту має такі параметри:

- Вибір програмного продукту – Python;
- Реалізація важливої постановки з допомогою вбудованих функцій;
- Використання стандартного інтерфейсу для побудови значень.

Даний варіант виконання програмного комплексу дає користувачу зручний інтерфейс, швидку реалізацію програми та доступний функціонал для роботи.

4.8 Висновки до четвертого розділу

В даній частині було проведено повний функціонально-вартісний аналіз програмного продукту. Також було знайдено оцінку основних функцій програмного продукту.

В результаті виконання функціонально-вартісного аналізу програмного комплексу що розроблюється, було визначено та проведено оцінку основних функцій програмного продукту, а також знайдено параметри, які його характеризують.

На основі аналізу вибрано варіант реалізації програмного продукту.

ВИСНОВКИ

В ході даного проекту була проведена детальна робота з вивчення системи MicroGrid, виявлені її особливості та переваги. Також було проведено дослідження альтернативних джерел енергії, які використовуються в системі, і розглянуто їх потенціал у забезпеченні автономії мережі. В результаті цих досліджень отримані важливі висновки та рекомендації щодо оптимального використання системи MicroGrid та альтернативних джерел енергії для досягнення більшої ефективності та стійкості в енергетичних мережах.

Для прогнозування вітрогенерації були застосовані сучасні підходи інтелектуального аналізу даних, а саме авторегресійні моделі та нейронні мережі, а також їх комбінація. Даний підхід показав якісні результати показників якості моделі та прогнозу, за рахунок чого було побудовано прогнозне значення електроенергії та потужності вітрової станції.

Була розроблена програма, яка здійснює прогнозування генерації електроенергії від сонячних панелей та вітрогенератора. Програма має наступні можливості, а також передбачає майбутні модифікації. Вона отримує та візуалізує погодинні показники вихідної потужності альтернативних джерел енергії протягом 7 днів з 100% точністю для наявних даних у поточний момент. Рост попиту на відновлювальну енергію призводить до зростання можливостей енергоефективності та енергонезалежності. Це призводить до збільшення кількості підприємств, які постачають "зелену" енергію для будинків та міст. Люди, які мають маленькі сонячні або вітряні станції, нерідко не усвідомлюють потенційних доходів, які можна отримати від невикористаної енергії.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Moncef K. Optimal Design and Retrofit of Energy Efficient Buildings, Communities, and Urban Centers: Utility Rate Structures and Grid Integration. – Butterworth-Heinemann, 2018. – 189-245 с.
2. D.W. Gao Energy Storage for Sustainable Microgrid: Basic concepts and architecture of microgrid management. – Academic press, 2015. – 1-34 с.
3. Stadler M., Naslé A. Planning and implementation of bankable microgrids:Growing markets. – Magazine on electricity, 2019. – 2 с.
4. Геліосистеми. Блог компанії almosfera *[Електронний ресурс]* – Режим доступу: https://www.atmosfera.ua/uk/geliosistemi/ (дата звернення: 02.06.2023)
5. Типи теплових насосів: **Теплові насоси – огляд технологій.** Блог компанії almosfera *[Електронний ресурс]* – Режим доступу: https://www.atmosfera.ua/uk/teplovi-nasosi/tipi-teplovix-nasosiv/ (дата звернення: 02.06.2023)
6. Біопаливо. *[Електронний ресурс]* – Режим доступу: https://uk.wikipedia.org/wiki/Біопаливо#cite_ref-1 (12 травня 2023)
7. Як підрахувати ККД сонячних батарей за формулою? Рекорди показників в світі : Наскільки сильний вплив перерахованих факторів? Блог компанії GreenTechTrade *[Електронний ресурс]* – Режим доступу: https://greentechtrade.com.ua/kkd-sonyachnyh-batarej-rozrahunok/ (дата публікації 18.06.2022)
8. Покрокова інструкція по будівництву сонячної електростанції (частина 3) : Вибір технічного обладнання. Блог компанії «Сонячна Енергія» *[Електронний ресурс]* – Режим доступу: https://sun-energy.com.ua/articles_old/3 (дата звернення: 02.06.2023)

9. Принцип роботи вітрогенератора : Особливості пристрою вітрогенератора. Блог компанії VENCON *[Електронний ресурс]* – Режим доступу:https://vencon.ua/ua/articles/printsip-raboty-vetrogeneratora (дата звернення: 02.06.2023)

10. Типи вітрогенераторів. Блог компанії almosfera *[Електронний ресурс]* – Режим доступу: https://www.atmosfera.ua/uk/vitryani-elektrostancii/tipi-vitrogeneratoriv/ (дата звернення: 02.06.2023)

11. Прохоренко Н.А. Python найнеобхідніше: Вступ . – БХВ-Петербург, 2011. – 1с.

12. Сонячна батарея Leapton LP-M-72-H-400W Half-Cell монокристал. Інтернет магазин ECO TECH Ukraine *[Електронний ресурс]* – Режим доступу: https://eco-tech.com.ua/p1026671220-solnechnaya-batareya-leapton.html?utm_source=google&utm_medium=c&utm_campaign=9754107338&utm_content=428656134895&utm_term=&gclid=CjwKCAjw5cL2BRASEiwAENqAPnq5LnzgqGe-t65cOlY3Wj-pxVBn_STOFGZKyzPwvRcT1xs9lvm_WhoCCScQAvD_BwE (дата звернення: 03.06.2023)

13. Фотоелектрична сонячна панель Amerisolar AS-6P30 285W, (AS-6P30-285). Інтернет магазин Rozetka *[Електронний ресурс]* – Режим доступу: https://rozetka.com.ua/ua/202338745/p202338745/?gclid=CjwKCAjw5cL2BRASEiwAENqAPin5NmckwUQG2yGbAydwNGS-LVLT0XDlKg0xjBeGefROlhLzCaTKgxoCrvEQAvD_BwE (дата звернення: 03.06.2023)

14. Сонячна батарея JA Solar JAM60S09-320W PR 5BB монокристал. Інтернет магазин ECO TECH Ukraine *[Електронний ресурс]* – Режим доступу: https://eco-tech.com.ua/p916702061-solnechnaya-batareya-solar.html?utm_source=google&utm_medium=c&utm_campaign=9754107338&utm_content=42865613

4895&utm_term=&gclid=CjwKCAjw5cL2BRASEiwAENqAPt4X3VRZjXEJJIWNh6pDArDUi8WCB_hDXoEXatsFcQI5DfKujuNyHxoCTM4QAvD_BwE (дата звернення: 03.06.2023)

15. Вітрогенератор flamingo aero FA - 4. 4 (1,6 кВт). Інтернет магазин ZAKUPKA *[Електронний ресурс]* – Режим доступу: https://zakupka.com/p/671877078-vetrogenerator-flamingo-aero-fa-4-4-1-6-kvt/?e=1&i=glRQD7xj7VPQ3GSzysg-kZKfc1HrNsucfVw_raPpJHc-e5SHu2t2_x-TVCA5uhsBHTT14vWQZFBtB3XIEHiKitZLamjXWzHNxC0ZsRL_65rtwoE0DZFDnlwXg8Fhn9VPNtjuzejmlGPWPy1Aop7WMxXZNa4bAIQ5ZCPV4wulYuYRQDTZlChi4B0oCHFpEbklzGf9KY0rVZ0sOWPd8Cv7bg==&gclid=CjwKCAjw5cL2BRASEiwAENqAPmoXYLaGidQ9Nt01N8ThtS2bZ5uK9dbWSP1e-k_EG1ArBgmAIb3huxoCxO0QAvD_BwE (дата звернення: 03.06.2023)

16. Вертикальний вітрогенератор «Шторм» 3 кВт. Інтернет магазин ZAKUPKA *[Електронний ресурс]* – Режим доступу: https://zakupka.com/p/949533398-vertikalnyy-vetrogenerator-shtorm-3-kvt/?e=1&i=glRQD7xj7VPQ3GSzysg-kZKfc1HrNsucfVw_raPpJHc-e5SHu2t2_x-TVCA5uhsBHTT14vWQZFBtB3XIEHiKiux8G0GqDTJzvYiU8SZohTYjLuVXZKEaLokOIibhacAYYA9GUN_vKQiFTHGjWttFx4nltF1cUCyOiq-icBPFQGaDgj-00JBiaYjwsfgKCcuP7DTjBJ1GzIzAU80F6ZxtZQ==&gclid=CjwKCAjw5cL2BRASEiwAENqAPjEOWcMhi4IirXe-nMK6OUBI1hVHmBjdIPcp6e3lHAU63Q1LzAhBQhoCNbAQAvD_BwE (дата звернення: 03.06.2023)

17. Вітрогенератор EW 2000. Інтернет магазин ZAKUPKA *[Електронний ресурс]* – Режим доступу: https://zakupka.com/p/671843121-vetrogenerator-ew-2000/ (дата звернення: 03.06.2023)

18. Mashayekh S.,Stadler M.,Cardoso G.,Heleno M. A Mixed Integer Linear Programming Approach for Optimal DER Portfolio, Sizing, and Placement in Multi-Energy Microgrids : Microgrid Model. – Applied Energy, 2016. – 7 с.

19. Сонячні батареї для приватного будинку чи квартири : Види сонячних батарей. Блог компанії TEPLOWOOD *[Електронний ресурс]* – Режим доступу: https://teplowood.ru/solnechnye-batarei-dlya-chastnogo-doma-i-kvartiry.html (дата звернення: 03.06.2023)

20. Методичні вказівки та завдання до виконання розрахунково-графічної роботи з дисципліни «Фізико-технологічні основи перетворення сонячної енергії» / укладачі А. С. Опанасюк, О. А. Доброжан. – Суми : Сумський державний університет, 2016. – 18-24 с.

21. І науково-технічна конференція магістрантів ІЕЕ (за результатами дисертаційних досліджень магістрантів). Зб. наукових праць ІЕЕ, КПІ імені Ігоря Сікорського – Київ: ІЕЕ, 2018. – 52 с.

22. Інтелектуальні електричні мережі: елементи та режими: За заг. ред. акад. НАН України О.В. Кириленка / Інститут електродинаміки НАН України. – К.: Ін-т електродинаміки НАН України, 2016. – 324 -334 с.

23. Herman D. Investigation of the Technical and Economic Feasibility of Micro-Grid-Based Power Systems. – Final Report, December 2001. – 3-19 с.

24. Кривцов В.С., Олейников А.М., Яковлев А.И. Неисчерпаемая энергия. Кн. 1. Ветроэлектрогенераторы. – Национальный аэрокосмический университет «Харьковский авиационный институт», 2003. –400 с.

25. Кривцов В.С., Олейников А.М. Неисчерпаемая энергия. Кн. 2. Ветроэнергетика. – Национальный аэрокосмический университет «Харьковский авиационный институт», 2003. – 18 с.

26. Шефтер Я.И. Ветроэнергетические агрегаты. – Машиностроение, 1972. - 288 с.

27. Burton T., Sharpe D., Jenkins N. and Bossanyi E. Wind Energy Handbook. John Wiley and Sons Ltd. – West Sussex, England, 2001.

28. Волківський В.Б. Адаптивний метод заряду акумуляторних батарей. –Силова електроніка та енергоефективність,2005. – 40–43 с.

29. Ромашко В.Я., Вербицький Є.В., Киричик Є.І. Аналіз втрат енергії в системі відбору максимальної потужності сонячної батареї. – Техн. електродинаміка., 2014. – 55 с.

30. Tracking Solar Panel Schematic Circuit Diagram. Блог компанії Circuit Diagrams *[Електронний ресурс]* – Режим доступу: https://circuit-diagramz.com/tracking-solar-panel-schematic-circuit-diagram/https://teplowood.ru/solnechnye-batarei-dlya-chastnogo-doma-i-kvartiry.html (дата звернення: 09.06.2023)

31. Solar Battery Charger. Блог компанії Circuit Diagrams *[Електронний ресурс]* – Режим доступу: http://www.circuit-diagrams.net/page/solar-battery-charger https://teplowood.ru/solnechnye-batarei-dlya-chastnogo-doma-i-kvartiry.html (дата звернення: 09.06.2023)

32. Solar Inverter without a Buck Converter or MPPT. Блог компанії Homemade Circuits Project *[Електронний ресурс]* – Режим доступу: https://www.homemade-circuits.com/designing-solar-inverter-tutorial/(дата останньої зміни 01.08.2019)

33. Штучний інтелект для ДТЕК Енерго / ДТЕК. URL: https://dtek.com/mediacenter/news/iskusstvennyyintellekt-dlya-dtek-energo/

34. МХП: штучний інтелект покращує роботу енергетиків. Українська правда. 2020. 3 серп. URL: https://www.epravda.com.ua/news/2020/08/3/663660/

35. https://uk.wikipedia.org/wiki

Додаток А

```
from PyQt5 import QtCore, QtGui, QtWidgets
from PyQt5.QtGui import QPalette, QTextCharFormat
from PyQt5.QtCore import Qt
from PyQt5.QtWidgets import QApplication, QCalendarWidget, QMessageBox, QVBoxLayout
from bs4 import BeautifulSoup
from urllib.request import urlopen
import sys
import matplotlib as mpl
import matplotlib.patches as mpatches
import matplotlib.pyplot as plt
from matplotlib.figure import Figure
import matplotlib.dates as mdates
import datetime
import csv
import numpy as np
from matplotlib.backends.backend_qt5agg import (NavigationToolbar2QT as NavigationToolbar)
from mplwidget import MplWidget

#region FrontEnd
class Ui_Form(object):
    def setupUi(self, Form):
        Form.setObjectName("Form")
        Form.resize(823, 436)
        self.comboBox = QtWidgets.QComboBox(Form)
        self.comboBox.setGeometry(QtCore.QRect(30, 70, 241, 22))
        self.comboBox.setObjectName("comboBox")
        self.comboBox.addItem("")
        self.comboBox.addItem("")
        self.comboBox.addItem("")
        self.comboBox.addItem("")
        self.comboBox_2 = QtWidgets.QComboBox(Form)
        self.comboBox_2.setGeometry(QtCore.QRect(30, 120, 241, 22))
```

```
        self.comboBox_2.setObjectName("comboBox_2")
        self.comboBox_2.addItem("")
        self.comboBox_2.addItem("")
        self.comboBox_2.addItem("")
        self.comboBox_2.addItem("")
        self.calendarWidget = QtWidgets.QCalendarWidget(Form)
        self.calendarWidget.setGeometry(QtCore.QRect(30, 170, 301, 155))
        font = QtGui.QFont()
        font.setPointSize(8)
        self.calendarWidget.setFont(font)
        self.calendarWidget.setGridVisible(True)
        self.calendarWidget.setVerticalHeaderFormat(QtWidgets.QCalendarWidget.NoVerticalHeader)
        self.calendarWidget.setObjectName("calendarWidget")
        self.comboBox_3 = QtWidgets.QComboBox(Form)
        self.comboBox_3.setGeometry(QtCore.QRect(520, 330, 261, 22))
        self.comboBox_3.setObjectName("comboBox_3")
        self.comboBox_3.addItem("")
        self.comboBox_3.addItem("")
        self.comboBox_3.addItem("")
        self.pushButton = QtWidgets.QPushButton(Form)
        self.pushButton.setGeometry(QtCore.QRect(710, 370, 75, 23))
        self.pushButton.setObjectName("pushButton")
        self.label = QtWidgets.QLabel(Form)
        self.label.setGeometry(QtCore.QRect(30, 50, 141, 16))
        self.label.setObjectName("label")
        self.label_2 = QtWidgets.QLabel(Form)
        self.label_2.setGeometry(QtCore.QRect(30, 100, 151, 16))
        self.label_2.setObjectName("label_2")
        self.MplWidget = MplWidget(Form)
        self.MplWidget.setGeometry(QtCore.QRect(440, 50, 361, 271))
        self.MplWidget.setObjectName("MplWidget")

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

    def retranslateUi(self, Form):
```

```
        _translate = QtCore.QCoreApplication.translate
        Form.setWindowTitle(_translate("Form", "Form"))
        self.comboBox.setItemText(0, _translate("Form", "Оберіть модель"))
        self.comboBox.setItemText(1, _translate("Form", "LEAPTON LP-M-72-H-400W HALF-CELL"))
        self.comboBox.setItemText(2, _translate("Form", "Amerisolar AS-6P30 285W"))
        self.comboBox.setItemText(3, _translate("Form", "JA SOLAR JAM60S09-320W PR 5BB"))
        self.comboBox_2.setItemText(0, _translate("Form", "Оберіть модель"))
        self.comboBox_2.setItemText(1, _translate("Form", "Flamingo aero FA - 4. 4 "))
        self.comboBox_2.setItemText(2, _translate("Form", "Шторм 3 кВт"))
        self.comboBox_2.setItemText(3, _translate("Form", "EW 2000"))
        self.comboBox_3.setItemText(0, _translate("Form", "Загальний графік"))
        self.comboBox_3.setItemText(1, _translate("Form", "Графік для вітрогенератора"))
        self.comboBox_3.setItemText(2, _translate("Form", "Графік для сонячної панелі"))
        self.pushButton.setText(_translate("Form", "Ready!"))
        self.label.setText(_translate("Form", "Сонячні панелі:"))
        self.label_2.setText(_translate("Form", "Вітрогенератори:"))
#endregion

#region BackEnd
class Start_Window(QtWidgets.QMainWindow, Ui_Form):
    def __init__(self):
        super().__init__()
        self.setupUi(self)
        self.finished = False
        #після кліку на кнопку - відкриваємо функцію get_date
        self.pushButton.clicked.connect(self.get_data)

        #початкова і кінцева дата для подальшої взаємодії
        self.begin_date = None
        self.end_date = None

        self.highlight_format = QTextCharFormat()
        self.highlight_format.setBackground(self.palette().brush(QPalette.Highlight))
        self.highlight_format.setForeground(self.palette().color(QPalette.HighlightedText))

        self.calendarWidget.clicked.connect(self.date_is_clicked)
```

```
        #print(super().dateTextFormat())
        self.addToolBar(NavigationToolbar(self.MplWidget.canvas, self))

    def format_range(self, format):
        #Беремо максимальну і мінімальну серед вибраного регіону дат щоб визначити дату початку і кін
ця
        if self.begin_date and self.end_date:
            d0 = min(self.begin_date, self.end_date)
            self.start = d0
            d1 = max(self.begin_date, self.end_date)
            self.end = d1
            while d0 <= d1:
                self.calendarWidget.setDateTextFormat(d0, format)
                d0 = d0.addDays(1)

    def date_is_clicked(self, date):
        #Так як стандартні методи pyqt не дозволяють вибирати і замольовувати range дат
        #Код перевіряє чи натиснений Shift і якщо так то замальовує дати між початковою і кінцевою
        self.format_range(QTextCharFormat())
        if QApplication.instance().keyboardModifiers() & Qt.ShiftModifier and self.begin_date:
            self.end_date = date
            # set highilighting of currently selected date range
            self.format_range(self.highlight_format)
        else:
            self.begin_date = date
            self.end_date = None

    def save_files(self, diagram_name, hourly, data_1):
        #функція збереження даних в таблицю (excel)
        name_generation = str(diagram_name) + str(datetime.datetime.now().date()) + '.csv'
        with open(name_generation, mode='w') as employee_file:
            employee_writer = csv.writer(employee_file)

            employee_writer.writerow(hourly) #День та година, верхній рядок
            employee_writer.writerow(data_1) #Значення, нижній рядок
```

```
def draw_diagram(self, a_names, w_values, s_values, type_diagram):
    #функція побудови графіків. Отримує назви та значення після чого малює діаграму
    names = a_names
    wind_value = w_values
    sun_value = s_values

    if type_diagram == 'general':
        get_sun_len = len(sun_value)
        all_value = []
        for i in range(get_sun_len):
            all_value.append(wind_value[i] + sun_value[i])
            i += 1
        self.MplWidget.canvas.axes.clear()
        self.MplWidget.canvas.axes.plot(names, all_value)
        self.MplWidget.canvas.axes.set_title('Загальний графік')
        self.MplWidget.canvas.draw()
        self.save_files('Загальний графік', names, all_value) #створюємо таблицю
        plt.clf()
        plt.plot(names, all_value)
        plt.show()
    elif type_diagram == 'wind':
        self.MplWidget.canvas.axes.clear()
        self.MplWidget.canvas.axes.plot(names, wind_value)
        self.MplWidget.canvas.axes.set_title('Графік вітрогенераторів')
        self.MplWidget.canvas.draw()
        self.save_files('Графік вітрогенераторів', names, wind_value) #створюємо таблицю
        plt.clf()
        plt.plot(names, wind_value)
        plt.show()
    elif type_diagram == 'sun':
        get_sun_len = len(sun_value)
        for i in range(get_sun_len):
            element = str(names[i]).split(':')
            if int(element[0]) >= 21 or int(element[0]) <= 6:
                sun_value[i] = 0
        self.MplWidget.canvas.axes.clear()
```

```
            self.MplWidget.canvas.axes.plot(names, sun_value)
            self.MplWidget.canvas.axes.set_title('Графік сонячних панелей')
            self.MplWidget.canvas.draw()
            self.save_files('Графік сонячних панелей', names, sun_value) #створюємо таблицю
            plt.clf()
            plt.plot(names, sun_value)
            plt.show()

    def sun_panel_kpd(self, s_name, generate_data):
        #Вираховуємо кпд сонячних панелей
        LEAPTON = [0.20, 400]
        AMERISOLAR = [0.17, 280]
        JA_SOLAR = [0.19, 320]

        timenow = datetime.datetime.now().time().strftime('%H')
        time_index = int(timenow)

        self.day_sun = [] #ліст який містить значення точок на графіку
        if s_name == 'LEAPTON LP-M-72-H-400W HALF-CELL':
            for i in range(generate_data):
                if float(self.cloud_list[time_index + i]) == -1.0:
                    get_kpd = float(LEAPTON[1] * float(LEAPTON[0]))
                    self.day_sun.append(get_kpd)
                else:
                    get_kpd = float(LEAPTON[1]) * float(LEAPTON[0]) * float(self.cloud_list[time_index + i])
                    self.day_sun.append(get_kpd)
        elif s_name == 'Amerisolar AS-6P30 285W':
            for i in range(generate_data):
                if float(self.cloud_list[time_index + i]) == -1.0:
                    get_kpd = float(AMERISOLAR[1]) * float(AMERISOLAR[0])
                    self.day_sun.append(get_kpd)
                else:
                    get_kpd = float(AMERISOLAR[1]) * float(AMERISOLAR[0]) * float(self.cloud_list[time_index
+ i])
                    self.day_sun.append(get_kpd)
        elif s_name == 'JA SOLAR JAM60S09-320W PR 5BB':
```

```
        for i in range(generate_data):
            if float(self.cloud_list[time_index + i]) == -1.0:
                get_kpd = float(JA_SOLAR[1]) * float(JA_SOLAR[0])
                self.day_sun.append(get_kpd)
            else:
                get_kpd = float(JA_SOLAR[1]) * float(JA_SOLAR[0]) * float(self.cloud_list[time_index + i])
                self.day_sun.append(get_kpd)
    elif s_name == 'Оберіть модель':
        for i in range(generate_data):
            self.day_sun.append(0)



def wind_generator_kpd(self, g_name, generate_data):
    #self.parse_weather(date_end, '2020-06-01')
    #Функція підрахунку значень для кпд вітрогенераторів
    self.day_wind = []#ліст який містить значення точок на графіку

    timenow = datetime.datetime.now().time().strftime('%H')
    time_index = int(timenow)
    self.wind = []
    if g_name == 'Flamingo aero FA - 4. 4 ':
        for i in range(generate_data):
            inc = float(self.wind_list[i + time_index])
            if inc >= 2.5:
                if inc >= 8:
                    self.day_wind.append(1600.0)
                elif inc < 8 and inc >= 6.5:
                    self.day_wind.append(1600.0 * 0.8)
                elif inc < 6.5 and inc >= 4.5:
                    self.day_wind.append(1600.0 * 0.6)
                else:
                    get_kpd = (float(self.wind_list[i + time_index]) / 8.0) * 1600.0
                    self.day_wind.append(get_kpd)
            else:
                self.day_wind.append(0)
            i += 1
```

```
elif g_name == 'Шторм 3 кВт':
    for i in range(generate_data):
        inc = float(self.wind_list[i + time_index])
        if inc >= 1.5:
            if inc >= 8.0:
                self.day_wind.append(3400.0)
            elif inc < 8 and inc >= 6.5:
                self.day_wind.append(3400.0 * 0.8)
            elif inc < 6.5 and inc >= 3.5:
                self.day_wind.append(3400.0 * 0.6)
            else:
                get_kpd = (float(self.wind_list[i + time_index]) / 8.0) * 3400.0
                self.day_wind.append(get_kpd)
        else:
            self.day_wind.append(0)
        i += 1
elif g_name == 'EW 2000':
    for i in range(generate_data):
        inc = float(self.wind_list[i + time_index])
        if inc >= 2.5:
            if inc >= 12.0:
                self.day_wind.append(2000.0)
            elif inc < 12.0 and inc >= 9.5:
                self.day_wind.append(2000.0 * 0.8)
            elif inc < 9.5 and inc >= 6.5:
                self.day_wind.append(2000.0 * 0.6)
            elif inc < 6.5 and inc >= 4.5:
                self.day_wind.append(2000.0 * 0.4)
            else:
                get_kpd = (float(self.wind_list[i + time_index]) / 12.0) * 2000.0
                self.day_wind.append(get_kpd)
        else:
            self.day_wind.append(0)
        i += 1
elif g_name == 'Оберіть модель':
    for i in range(generate_data):
```

```
            self.day_wind.append(0)

    def cloud_converter(self, c_name):
        #тут ми текстові значення з сайту прирівнюємо до % значень
        if 'Малооблачно' in c_name:
            result = 0.1
        elif 'Значительная облачность' in c_name:
            result = 0.2
        elif 'Пасмурно' in c_name:
            result = 0.35
        elif 'Небольшой дождь' in c_name:
            result = 0.45
        elif 'Ливневый дождь' in c_name:
            result = 0.65
        else:
            result = -1.0
        return result

    def parse_weather(self):
        site_url = 'https://www.meteoservice.ru/weather/hourly/kiev#' #основна частина адреси сайту для пар
сингу

        #ліcти для зберігання інформації про час, хмарність, температуру, вітер
        self.time_list = []
        self.cloud_list = []
        self.wind_list = []
        self.day_list = []

        #Парсимо дані з сайту
        final_url = site_url + str(datetime.datetime.now().today().strftime('%Y%m%d'))
        urlPage = urlopen(final_url).read()
        weatherPage = BeautifulSoup(urlPage, 'lxml')
        for day in weatherPage.findAll('div', {'class':'small-12 columns forecast days-6'}):
            for time in day.findAll('span', {'class':'h3'}):
                self.time_list.append(time.text + ':00')
            for cloud in day.findAll('div', {'class':'column value show-for-large text-left font-smaller font-condense
d'}):
```

```
            self.cloud_list.append(self.cloud_converter(cloud.text))
        for wind in day.findAll('span', {'class':'font-smaller'}):
            wind_speed = str(wind.text).replace(' м/с', '')
            if wind_speed[0].isdigit():
                if len(wind_speed) > 1:
                    new_wind = wind_speed.split('–')
                    result = (int(new_wind[0]) + int(new_wind[1])) / 2
                    self.wind_list.append(result)
                else:
                    self.wind_list.append(wind_speed)

def get_data(self):
    #отримуємо вибрані в календарі дати, та переводимо їх у формат рррр-мм-дд (рік-місяць-день)
    try:
        s_date = QtCore.QDate.getDate(self.start)
        e_date = QtCore.QDate.getDate(self.end)
    except:
        QMessageBox.about(self, "Помилка", "Потрібно вибрати мінімум 2 дати (24 години)")
        return

    #Конвертуэмо з формату QDate в datetime.date
    date_start = datetime.date(year=s_date[0], month=s_date[1], day=s_date[2]).strftime('%Y-%m-%d')
    date_end = datetime.date(year=e_date[0], month=e_date[1], day=e_date[2]).strftime('%Y-%m-%d')

    print(date_start, date_end)

    #Різниця між сьогоднішньою датою та датою кінця періоду
    try:
        diff = datetime.date(year=e_date[0], month=e_date[1], day=e_date[2]) - datetime.datetime.now().date()
        diff_range = int(str(diff).split(' ')[0])
    except:
        QMessageBox.about(self, "Помилка", "Вісутня інформація з минулі дні, оберіть в проміжку 7 днів
від сьогодні. ")
        return

    print(diff)
```

```
        print(diff_range)


        if diff_range > 6:
            QMessageBox.about(self, "Помилка", "Вибрано надто великий діапазон.\nМаксимальний період - сьогодні плюс 6 днів")
            return
        elif diff_range < 0:
            QMessageBox.about(self, "Помилка", "Вісутня інформація з минулі дні, оберіть в проміжку 7 днів від сьогодні")
            return


        try:
            diff_start = datetime.date(year=s_date[0], month=s_date[1], day=s_date[2]) - datetime.datetime.now().date()
            diff_start_int = int(str(diff_start).split(' ')[0])
        except:
            diff_start_int = 0


        if diff_start_int == 0:
            #Отримуємо теперішній час, щоб в подальшому не виводити на графік години які вже минули
            timenow = datetime.datetime.now().time().strftime('%H')
            time_index = int(timenow)
            test_value = 23 - time_index
        else:
            time_index = diff_start_int * 24
            test_value = diff_start_int * 24


        #отримуємо інформацію про вибрані панелі/генератори
        sun_panel = self.comboBox.currentText()
        wind_generator = self.comboBox_2.currentText()


        #Отримуємо дані про тип графіку
        graphics_type = self.comboBox_3.currentText()



        #Кількість годин які потрібно згенерувати на графіку
        generate_data = (diff_range * 24)
```

```
#Викликаємо функцію парсингу погоди
try:
    self.parse_weather()
    #викликаємо функцію генерації даних для відповідного вітрового генератора
    self.wind_generator_kpd(wind_generator, generate_data)
    #викликаємо функцію генерації сонячних станцій
    self.sun_panel_kpd(sun_panel, generate_data)
except Exception as e:
    self.errorPrint(str(e))



days = [] #ліст із днями та часом по яких будується графік
date_now = datetime.datetime.today().date()
for i in range(generate_data):
    time_val = self.time_list[time_index + i]
    if diff_start_int == 0:
        if time_val == '23:00':
            date_print = str(date_now).replace('-', '.')
            days.append(str(time_val) + '\n' +date_print[5:-1])
            diff_start_int += 1
        else:
            days.append(str(time_val) + '\n' +str(date_now))
    elif diff_start_int == 1:
        if time_val == '23:00':
            days.append(str(time_val) + '\n' +str(date_now + datetime.timedelta(days=1)))
            diff_start_int += 1
        else:
            days.append(str(time_val) + '\n' +str(date_now + datetime.timedelta(days=1)))
    elif diff_start_int == 2:
        if time_val == '23:00':
            days.append(str(time_val) + '\n' +str(date_now + datetime.timedelta(days=2)))
            diff_start_int += 1
        else:
            days.append(str(time_val) + '\n' +str(date_now + datetime.timedelta(days=2)))
    elif diff_start_int == 3:
```

```
            if time_val == '23:00':
                days.append(str(time_val) + '\n' +str(date_now + datetime.timedelta(days=3)))
                diff_start_int += 1
            else:
                days.append(str(time_val) + '\n' +str(date_now + datetime.timedelta(days=3)))
        elif diff_start_int == 4:
            if time_val == '23:00':
                days.append(str(time_val) + '\n' +str(date_now + datetime.timedelta(days=4)))
                diff_start_int += 1
            else:
                days.append(str(time_val) + '\n' +str(date_now + datetime.timedelta(days=4)))
        elif diff_start_int == 5:
            if time_val == '23:00':
                days.append(str(time_val) + '\n' +str(date_now + datetime.timedelta(days=5)))
                diff_start_int += 1
            else:
                days.append(str(time_val) + '\n' +str(date_now + datetime.timedelta(days=5)))
        elif diff_start_int == 6:
            if time_val == '23:00':
                days.append(str(time_val) + '\n' +str(date_now + datetime.timedelta(days=6)))
                diff_start_int += 1
            else:
                days.append(str(time_val) + '\n' +str(date_now + datetime.timedelta(days=6)))
        i += 1

    print(len(days))

    #Обробка комбоБоксу під графіком, в залежності від вибраного - будуємо графіки
    if graphics_type == 'Графік для вітрогенератора':
        if wind_generator == 'Оберіть модель':
            QMessageBox.about(self, 'Помилка', 'Не вибрано вітрогенератор')
            return
        else:
            self.draw_diagram(days, self.day_wind, self.day_sun, 'wind')
    elif graphics_type == 'Графік для сонячної панелі':
        if sun_panel == 'Оберіть модель':
```

```
                QMessageBox.about(self, 'Помилка', 'Не вибрано сонячну панель')
                return
            else:
                self.draw_diagram(days, self.day_wind, self.day_sun, 'sun')
        elif graphics_type == 'Загальний графік':
            if sun_panel == 'Оберіть модель' and wind_generator == 'Оберіть модель':
                QMessageBox.about(self, 'Помилка', 'Не вибрано жодного приладу!')
            else:
                self.draw_diagram(days, self.day_wind, self.day_sun, 'general')

    def errorPrint(self, e_name):
        QMessageBox.about(self, 'Помилка', str(e_name))
#endregion

def main():
    app = QtWidgets.QApplication(sys.argv)
    window = Start_Window()
    window.show()
    app.exec_()

if __name__ == "__main__":
    try:
        main()
    except Exception as e:
        print(e)
```

Додаток Б

Графічна документація

На рисунку Д.2.1 показана схема інвертора для сонячної панелі, яка не використовує понижуючий перетворювач напруги або MPPT (максимальної потужності точки відбору) [32].

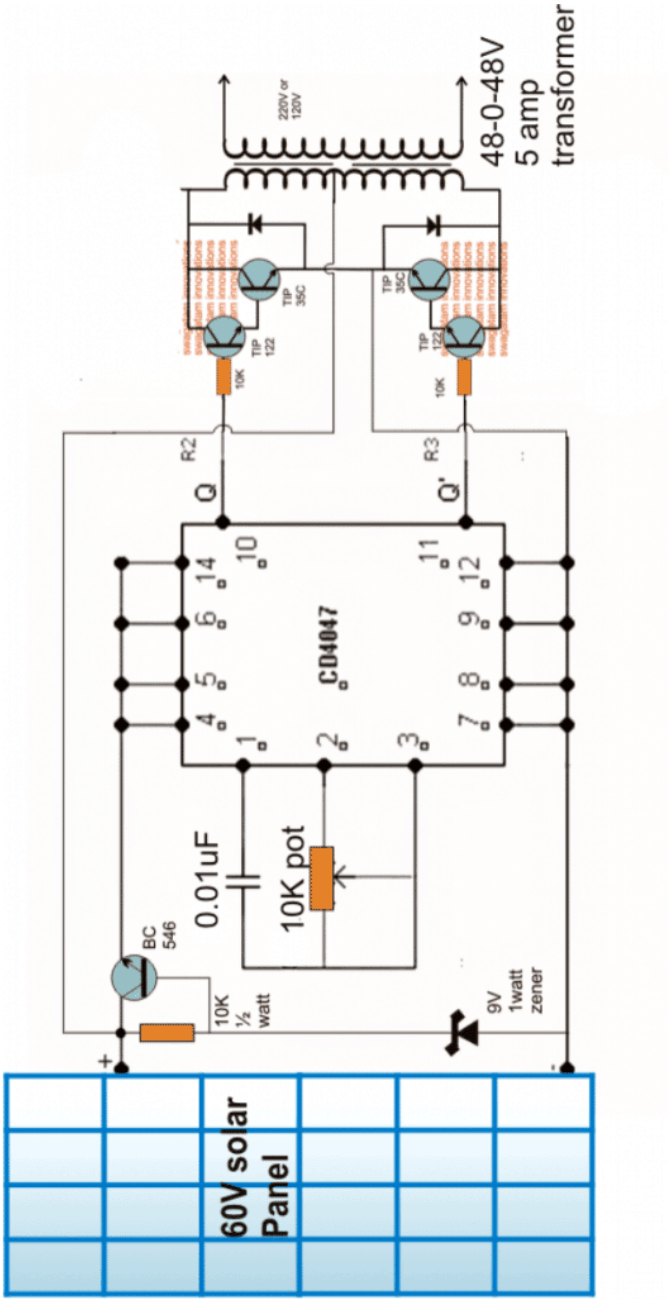


Рис.Д.2.1 Схема інвертора для сонячної панелі

На рисунку Д.2.2 зображено зарядний пристрій для сонячної батареї, який використовує регулятор шунту на базі TL497 для запобігання перезарядки [31].

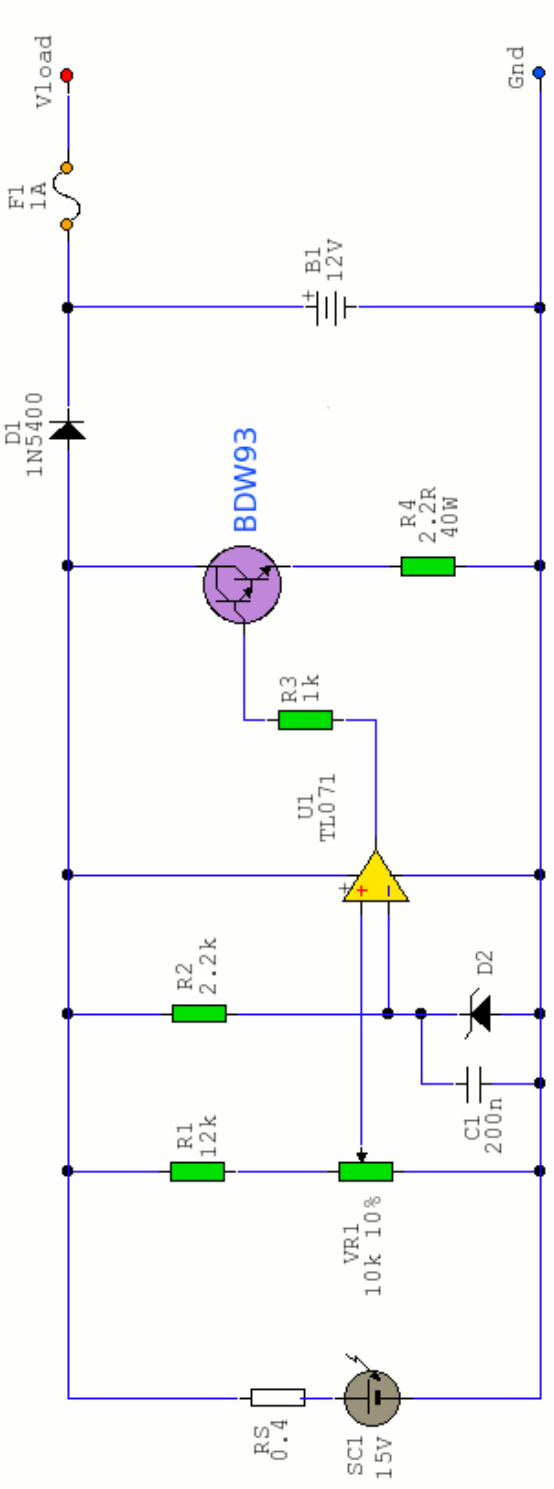


Рис.Д.2.2 Схема зарядного пристрою для сонячної батареї

На рисунку Д.2.3 показана схема відстеження напряму сонця для сонячної панелі [30].

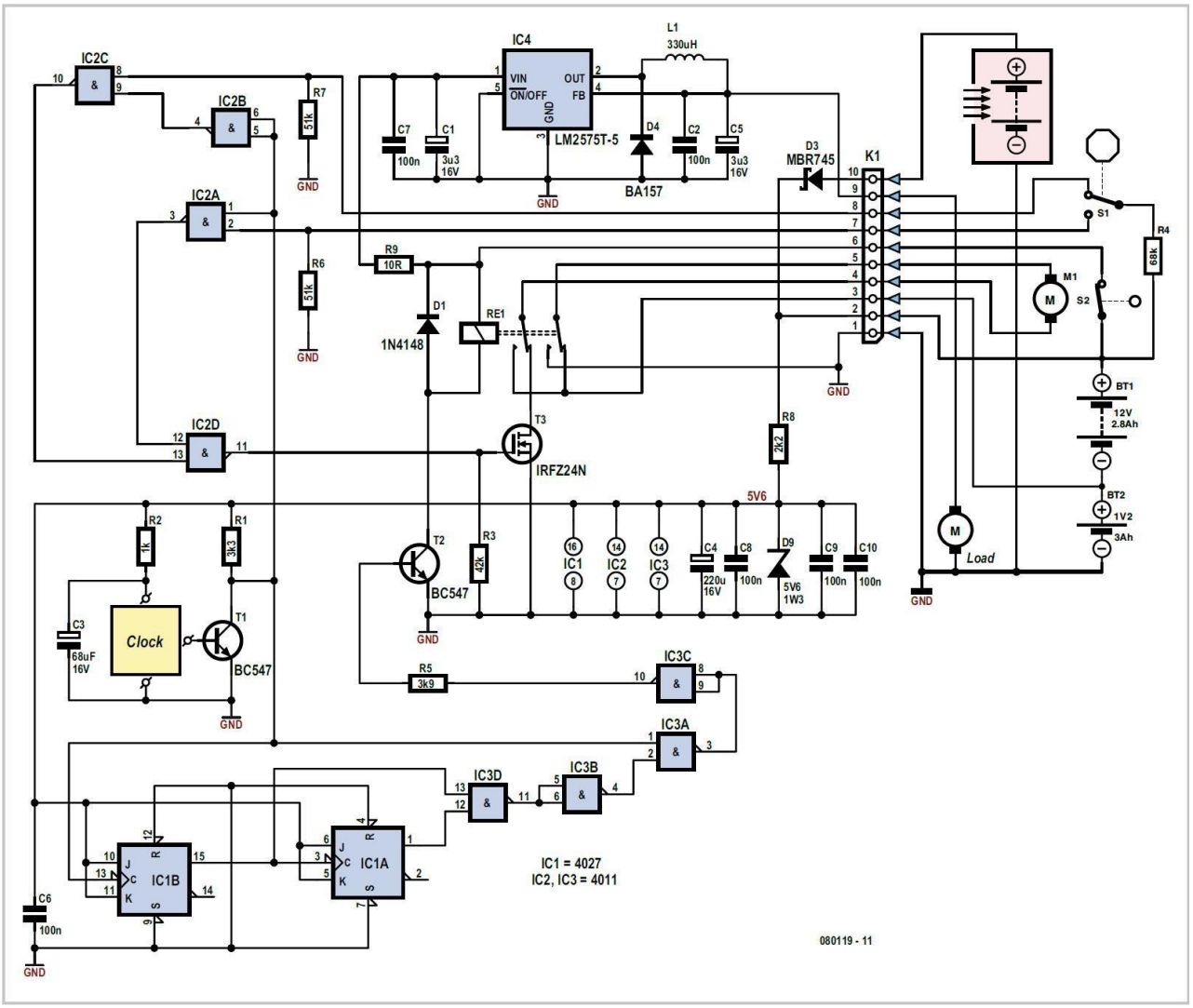


Рис.Д.2.3 Схема відстеження напряму сонця для сонячної панелі

Резюме

Дипломний проект першого освітнього рівня «Бакалавр» за спеціальністю 122 Комп’ютерні науки студента Соколової Дарини. Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського». Факультет Інститут системного прикладного аналізу, кафедра штучного інтелекту. Академічна група КІ-92.

Дипломний проект містить: 3 теоретичні малюнки; 56 сторінок пояснювальної записки, 4 розділи, 31 ілюстрацій, 4 таблиці.

Швидке зростання населення та розвиток промисловості, збільшення споживання електроенергії та водночас обмеженість джерел енергії є однією з ключових проблем сучасної енергетики.

Інноваційною відповіддю на нові виклики електроенергетики стала концепція MicroGrid – малої розподіленої енергії. Головним фактором появи та просування технології MicroGrid стало завдання забезпечення енергоефективності.

MicroGrids можуть працювати як у режимі централізованого керування, так і в режимі децентралізованого керування. У централізованому режимі центральний контролер MicroGrid відіграє найважливішу роль в оптимізації MicroGrid.

S U M M A R Y

Diploma project of the first educational level "Bachelor" in the specialty 122 Computer science of the student Sokolova Daryna. National Technical University of Ukraine "Ihor Sikorsky Kyiv Polytechnic Institute". Faculty, Institute of Systemic Applied Analysis, Department of Artificial Intelligence. Academic group KI-92.

The diploma project contains: 3 theoretical drawings; 56 explanatory note page, 4 sections, 31 illustrations, 4 tables.

Rapid population growth and industrial development, increasing electricity consumption and at the same time limited energy sources are one of the key problems of modern energy.

The concept of MicroGrid - low-distribution energy - was an innovative answer to the new challenges of the electric power industry. The main factor in the emergence and promotion of MicroGrid technology was the task of ensuring energy efficiency.

MicroGrids can operate both in centralized control mode and in decentralized control mode. In centralized mode, the central MicroGrid controller performs a single role in optimizing the MicroGrid.

Devices generation

Devices for electricity generation from renewable sources have various characteristics and efficiencies. The most commonly used type is silicon solar panels, which are inexpensive and widely produced, but have an efficiency range of 20-25%. In addition to silicon panels, three types of solar inverters are commonly used.

*Thin-film solar panels are cost-effective and consist of a thin layer of silicon on a carrier material. They can even generate electricity in diffused light and are suitable for installation on building walls. However, their efficiency is relatively low at 7-10%, and the silicon layer may degrade gradually over time. Despite this, they can still generate electricity in cloudy weather.

*Polycrystalline solar panels are made from molten silicon that slowly cools, resulting in a distinctive bright blue color. These panels have a better efficiency range of 17-20% but are less effective in diffuse light conditions.

*Monocrystalline solar panels are the most expensive among the three types but offer higher efficiency, ranging from 20% to 25%. They are produced by dividing single-crystal silicon plates and have a characteristic geometry with beveled corners.

The use of wind energy is also increasing in power systems worldwide. Wind turbines, or wind generators, are used to produce electricity from wind energy. However, wind power generation faces challenges due to the variability of wind speed and the uncertainties associated with instantaneous and average values.

The calculation of wind power equipment relies on the cubic dependence of wind flow velocity. This means that a doubling of wind speed from 6 m/s to 12 m/s results in an eight-fold increase in power generation, from 100 W to 800 W. For small-sized turbines, high power generation requires strong winds.

The software

The power generation forecasting software in the MicroGrid system is responsible for calculating the power output of solar panels and wind turbines. The accurate operation of these generators relies on specific weather conditions. For wind turbines, the wind speed needs to exceed the starting value and preferably be close to the nominal speed, allowing the turbine to operate at maximum power output. For solar panels, factors such as the percentage of cloud cover and the presence of rain are crucial. High cloud cover reduces the output power of solar panels, especially for small solar plants, while rain significantly affects their efficiency.

To obtain the necessary weather data, the software extracts information from a selected meteorological site by parsing its data. Although not all meteorological sites provide the required information, a suitable site was chosen for this task, which allows for the collection of precise data on power generation.

The software is implemented in Python, an interpreted, object-oriented, high-level programming language that offers a wide range of capabilities. Python enables data processing, image creation, database management, web development, and the creation of applications with a graphical user interface.

The power generation data is graphed and displayed in a separate window for ease of use. All the collected data is stored in a dedicated file.

The software features a user-friendly interface presented in a simple single-window form (Figure 1) to enhance usability.

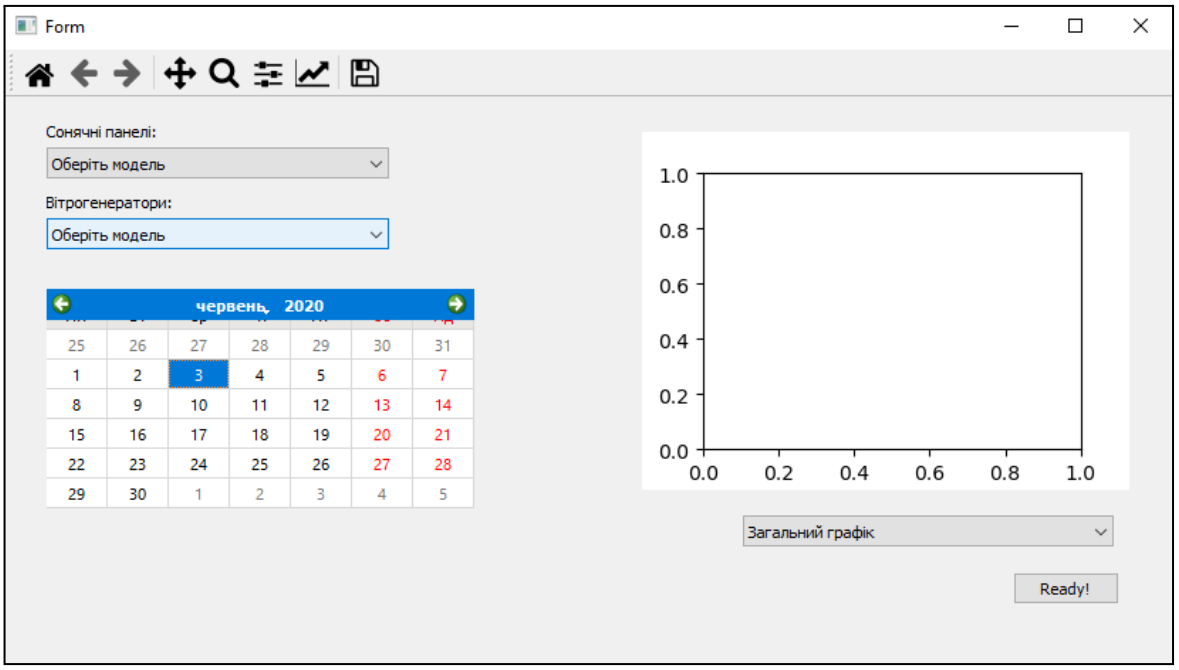


Fig.1. Software interface

The software interface includes the following controls:

*A menu for selecting the model of the solar panel or wind generator.

*A calendar for choosing the desired date.

*A menu for selecting the type of graphs to be displayed.

*A control panel for adjusting settings and options.

*An area designated for building the schedule.

*A "Ready" button that initiates the construction algorithm.

In addition, the software performs capacity calculation and plotting. Three models of solar panels have been selected, as specified in Table 1. The selection of different solar panel models allows for comparing their capacities and performances.

Table 1.

The main parameters of solar panels

Model	**Efficiency, %**	**Maximum power, Wt**

LEAPTON LP-M-72-H-400W HALF-CELL	20	400
Amerisolar AS-6P30 285W	17	280
JA SOLAR JAM60S09-320W PR 5BB	19	320

Based on the collected weather data, specifically regarding cloud cover and rain, a table of device efficiency relative to weather conditions has been developed. This table, referred to as Table 2, provides information on the efficiency of the devices under different weather conditions. Since the data regarding cloudiness percentages were not available, the efficiency values in the table serve as a reference for the corresponding weather conditions.

Table 2.

Reducing the initial efficiency relative to weather conditions

Weather conditions	**Reducing the output efficiency by, %**
Clear weather	0
Partly cloudy	10
Significant clouds	20
Cloudy weather	35
Light rain	45
Heavy rain	65

The formula for determining the output power of the joint venture in clear weather:

$$P_{вих} = P_{\max} * \eta ,$$

where P_{max} – maximum power of the device; η – solar energy conversion efficiency.

The formula of the output power in the presence of negative weather conditions:

$$P_{вих} = (P_{\max} * \eta) * \eta_{ny} ,$$

Where η_{ny} – the efficiency of solar energy conversion in adverse weather conditions:

$$\eta_{ny} = 100\% - \eta_{зв},$$

Where $\eta_{зв}$ – reducing the initial efficiency.

3 models of devices were also selected for wind generators (Table 3). The main parameters for operation are the starting and nominal wind speed at which the windmill operates.

Table 3

Parameters of wind generators

Model	**Starting speed, m/s**	**Rated speed, m/s**	**Maximum power, kWt**
Flamingo aero FA - 4. 4	2,5	8	1,6
Шторм 3 кВт	1,5	8	3,4
EW 2000	2,5	12	2

For a clearer result, additional intermediate efficiencies between starting and rated speed were added (Table 4). If the wind speed is less than the starting then the output power is 0.

Table 4

Output efficiency relative to wind speed

Model	Efficiency, %	Wind speed, m/s
Flamingo aero FA - 4. 4	60	4,5
	80	6,5
	100	8
Шторм 3 кВт	60	3,5
	80	6,5
	100	8
EW 2000	60	6,5
	80	9,5
	100	12

The formula for the output power of the wind turbine is equal to:

$$P_{вих} = P_{\max} * \eta \ ,$$

Where η – wind force conversion efficiency.

After collecting data and substituting them into formulas, graphs are built in the area of graphs.