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

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: «Інформаційна технологія визначення впливу погодних умов на продуктивність альтернативних джерел енергії»

за напрямом підготовки 6.050101 «Комп'ютерні науки»

Виконавець роботи: студент групи ІТ-51 Казлаускайте Анастасії Сергіївни

Кваліфікаційна робота бакалавра
захищена на засіданні ЕК
з оцінкою _____ «___»
2019 р.

Науковий керівник _____ к.т.н., доц., Шендрик В.В
(підпис) (науковий ступінь, вчене звання, прізвище та ініціали)

Голова комісії _____ Шифрін Д.М.
(підпис) (науковий ступінь, вчене звання, прізвище та ініціали)

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

Студент _____
(підпис)

Суми-2019

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук
Секція інформаційних технологій проектування
Напрям підготовки – 6.050101 «Комп'ютерні науки»

ЗАТВЕРДЖУЮ
Зав. секцією ІТП

_____ В. В. Шендрик
«__» _____ 2019 р.

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА СТУДЕНТУ

Казлаускайте Анастасія Сергіївна

1 Тема роботи *Інформаційна технологія визначення впливу погодних умов на продуктивність альтернативних джерел енергії*

керівник роботи *Шендрик Віра Вікторівна, к.т.н., доцент*,

затверджені наказом по університету від « 17 » травня 2019 р. № 0834 III

2 Строк подання студентом роботи «10» червня 2019 р.

3 Вхідні дані до роботи технічне завдання на розробку інформаційної технології, зібрані дані освітленості у форматі xlsx

4 Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) аналіз предметної області, постановка задачі та методи дослідження, проектування інформаційної технології, розробка інформаційної технології для визначення впливу погодних умов.

5 Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) постановка задачі, аналіз аналогів, моделювання інформаційної технології, аналіз технологій, етапи розробки інформаційної технології «HYBRID», висновки.

6. Консультанти розділів роботи:

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7.Дата видачі завдання_____

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз предметної області	04.03.19 – 06.03.19	
2	Пошук аналогів та визначення необхідності	07.03.19 – 07.03.19	
3	Ідентифікація ідеї проекту	08.03.19 – 08.03.19	
4	Аналіз документації Python	11.03.19 – 12.03.19	
5	Визначення інструментарію	13.03.19 – 14.03.19	
6	Планування WBS	15.03.19 – 15.03.19	
7	Планування OBS	15.03.19 – 15.03.19	
8	Складання календарного плану	18.03.19 – 18.03.19	
9	Визначення ризиків	18.03.19 – 18.03.19	
10	Розрахунок бюджету	19.03.19 – 19.03.19	
11	Розробка модулю парсінга даних	20.03.19 – 04.04.19	
12	Розробка модулю роботи з базою даних	05.04.19 – 13.04.19	
13	Розробка графічного інтерфейсу	25.04.19 – 30.04.19	
14	Розробка функціоналу базових операцій	01.05.19 – 19.05.19	
15	Тестування розробником	20.05.19 – 24.05.19	
16	Виправлення помилок	25.05.19 – 31.05.19	
17	Оформлення документації	11.03.19 – 06.06.19	
18	Введення в експлуатацію	03.06.19 – 10.06.19	

Студент _____ Казлаускайте А.С.
(підпис)

Керівник роботи _____ к.т.н., доц. Шендрик В.В.
(підпис)

РЕФЕРАТ

Тема дипломної роботи «Інформаційна технологія визначення впливу погодних умов на продуктивність альтернативних джерел енергії».

Дипломна робота складається з переліку умовних позначень, вступу, чотирьох розділів, висновка, списку використаної літератури та додатків.

Пояснювальна записка містить 133с., 70 рис., 9 табл., 5 додатків, 30 джерел.

У першому розділі досліджується актуальність проблеми, проводиться аналіз існуючих аналогів.

У другому розділі сформовано мету дипломної роботи та задачі проекту, вибір засобів реалізації та планування робіт.

Третій розділ присвячено проектуванню інформаційної технології, де представлено діаграми у нотації IDF0, Use Case, ER-діаграма бази даних та математична модель.

В останньому розділі представлено детальний опис практичної реалізації проекту: створення прототипу діалогового вікна, розробка бази даних, розробка парсера та створення головного модулю додатку з відповідним функціоналом.

Результатом проведеної роботи є розроблена інформаційна технологія, яка використовуючи погодні дані, автоматично визначає пріоритетне джерело електроенергії в гібридній електромережі

Ключові слова: PYTHON, ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ, ВДЕ, ГІБРИДНА ЕЛЕКТРОМЕРЕЖА, НЕЧІТКЕ ЧИСЛО

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ВДЕ – відновлювальні джерела енергії

ЕС – енергетична система

ЕМ – енергетична мережа

ГЕСВДЕ – гібридна енергетична система з відновлювальними джерелами енергії

АБ – акумуляторна батарея

СБ – сонячна батарея

ВЕУ – вітрогенератор

БД – база даних

ВСТУП

Сучасне життя неможливо уявити без енергії. Щорічно люди споживають енергію все більше і більше, адже стабільний розвиток економіки безпосередньо залежить від постійного розвитку енергетики. На сьогоднішній день суспільство активно використовує органічне паливо, таке як газ, вугілля або нафта, задля видобування енергії. Але, це всім відомий факт, що запаси їх в природі обмежені. І через певний час настане такий момент, коли вони всі вичерпаються. Нераціональне та неефективне споживання енергетичних ресурсів призводить до енергетичної кризи і, маючи всі передумови, загрожує навіть енергетичній безпеці. Тому гостро постає питання застосування необхідних дій та заходів для запобігання енергетичної кризи. Ефективне рішення даної проблеми є єдине і це залучення альтернативних джерел енергії, тих, що мають властивість відновлюватись.

На сьогодні електроенергія виробляється переважно централізовано. Попри те, що великі електростанції мають високі енергетичні показники, вони змушені передавати її на великі відстані, тому що електроенергію не можна накопичувати та зберігати, її потрібно одразу споживати. А відповідно споживачі розташовані далеко. Як наслідок, передача енергії на великі відстані несе значні її втрати. Тому все більше і більше країн застосовують концепцію розподільного виробництва енергії, яка полягає в розбудові системи енергогенерації. Тобто мається на увазі виробництво енергії кінцевими споживачами для власного споживання, та передача надлишку енергії, якщо такий з’являється, до загальної мережі. Об’єктами розподіленої генерації використовуються відновлювані локальні джерела (вітер, сонце та ін.). Якщо виробляють енергію одразу декілька джерел, то енергетична система стає гібридною.

В зв’язку з відмінностями метеорологічних умов між регіонами, існує необхідність завчасної оцінки продуктивності відновлювальних джерел енергії (ВДЕ). Вирішення даної проблеми потребує дослідження результативного інструментарію з метою управління енергією, також за умови застосування інформаційних технологій. Однак, на початковому етапі необхідно реалізувати аналіз впливу погодних умов на продуктивність альтернативних джерел енергії.

Обов’язковою умовою для коректної роботи інформаційної технології є наявність поточних даних та прогнозних показників функціонування сонячних батарей (СБ) та вітроустановок. Поточні дані збираються за допомогою різноманітних датчиків, тоді як прогнозні можливо отримати лише на основі математичних моделей.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Актуальність проблеми

Зростання світової економіки супроводжується неухильним збільшенням споживання енергоресурсів. Як вже було сказано раніше, основну частину складають невідновлювані вуглеводні запаси (вугілля, нафта, газ), яких в світі обмежена кількість. При середньорічному споживанні нафти близько 85 млн. барелів на день її повинно вистачити на 50 років, газу - на 70, вугілля - на 230. Крім того, вуглеводневе паливо при спалюванні в промисловості, транспорті, та побуті виділяє значні обсяги вуглекислого газу, що забруднює атмосферу і посилює «парниковим ефектом» потепління клімату, що чревате масштабними природними катастрофами[30].

На сьогоднішній день є досить актуальним питання переходу до альтернативних джерел енергії, які є менш небезпечними для навколишнього середовища ніж традиційні. Перш за все треба зазначити світлову енергію Сонця, обсяг якої поступає на поверхню Землі в 14-20 разів більше, аніж виробляють всі техногенні джерела планети. Деякі вчені прогнозують, що до 2020 року сонячна енергія замінить приблизно 2,5 млрд т палива і складатиме близько 8% від всієї виробленої електроенергії в світі[10].

Згідно зі статистикою, 40% первинної електроенергії споживається саме житловими спорудами. Тому для таких споживачів альтернативою стане розподільне виробництво енергії. Це концепція розбудови системи енергогенерації, яка має на увазі видобуток енергії самим споживачем для власних потреб[22].

Спираючись на міжнародний досвід, то вже у 2009 році в Данії 43% всієї електроенергії вироблено з розподіленої генерації, у 2014 році 20% нових

потужностей у Німеччині склали малі системи генерації. До таких змін призвело багато поштовхів, а саме:

- з розвитком технологій ВДЕ відбувається зниження їх вартості;
- забруднення навколишнього середовища та зміни в кліматі;
- державні фінансові механізми підтримки для громад , які роблять доступними малі енергетичні установки;
- дозвіл на самостійне споживання відновлюваних джерел енергії та децентралізоване акумулювання

Насамперед, розподілена генерація має багато переваг, серед яких :

- зменшення витрат електроенергії;
- швидке будівництво систем порівняно з традиційними централізованими станціями;
- легка масштабованість;
- віддалений контроль та моніторинг;
- підвищення безпеки енергопосточання[26].

Що стосується України, то використання саме такої генерації електроенергії призведе до збільшення інвестицій у громади та створення нових робочих місць. Існують також державні програми підтримки, які дозволяють отримати прибуток за «Зеленим тарифом», у разі продажу надлишкової енергії до загальної мережі. Також через ОСББ, кооперативи або інші організаційні громади можна ставати співвласниками малих систем виробництва енергії[28].

Для громадян розподілена генерація значить вироблення електроенергії для власних потреб та споживання. До того ж зменшується залежність від застарілих енергосистем, тобто не потрібно переплачувати за газ чи інше викопне паливо, зважаючи на нестабільні ціни саме на ці енергоресурси[19]. Порівняно з попередніми енергосистемами, витрати на системи розподіленої генерації на відновлюваних джерелах енергії – це лише витрати на обслуговування, що є значно меншими і стабільними[21].

Даний проект буде нести економічну, енергетичну, екологічну та, навіть, соціальну цінність, адже буде мати попит серед споживачів енергії – громад, підприємств та індивідуальних господарств. Дана інформаційна технологія допомагає не тільки обрати кращий варіант джерела енергії або їх комбінацію в даних погодних і експлуатаційних умовах, але й за допомогою експорта надлишкової електроенергії заощаджувати.

1.2 Аналіз існуючих аналогів

Перед початком роботи над проектом було проведено аналіз існуючих аналогів. Адже використання альтернативних джерел енергії є досить актуальною темою, яка набирає популярності серед українців та в світі в цілому.

Перший аналог – MAN Enerqy Solution. Це продукт німецької компанії, яка розробляє для сучасних заводів різні рішення зі застосування гібридних електромереж. MAN Energy Solutions – оптимізоване гібридне енергетичне рішення для комунальних служб, муніципалітетів та незалежних виробників електроенергії, які хочуть знизити викиди CO2.

Енергія вітру і сонця може бути підвищена за рахунок накопичення надлишкової потужності і миттєвого поповнення потужності двигуна і турбіни, що працюють на газі або навіть на біопаливі. Системи відновлюваної енергії можуть навіть бути додані до електростанцій, щоб діяти як економія палива і гібридні острівні енергосистеми. Засоби управління MAN Microgrid (рис. 1.1) забезпечують оптимізовану і надійну роботу гібридних систем харчування в будь-якому місці і в будь-який час. Рішення MAN Hybrid Power доступні в якості економії палива, стабілізаторів сітки або гібридних острівних систем харчування.

Ці гібридні електростанції об'єднують відновлювані джерела енергії, системи виробництва теплової енергії та системи зберігання енергії в мікромережі. Компанія пропонує рішення з можливістю швидкого запуску, зупинки та лінійної зміни навантаження, включаючи високоефективні багатопаливні генератори для надійного, гнучкого і ефективного виробництва електроенергії. Вони також можуть бути включені в різні системи для забезпечення стабільності і ефективності сітки[3].

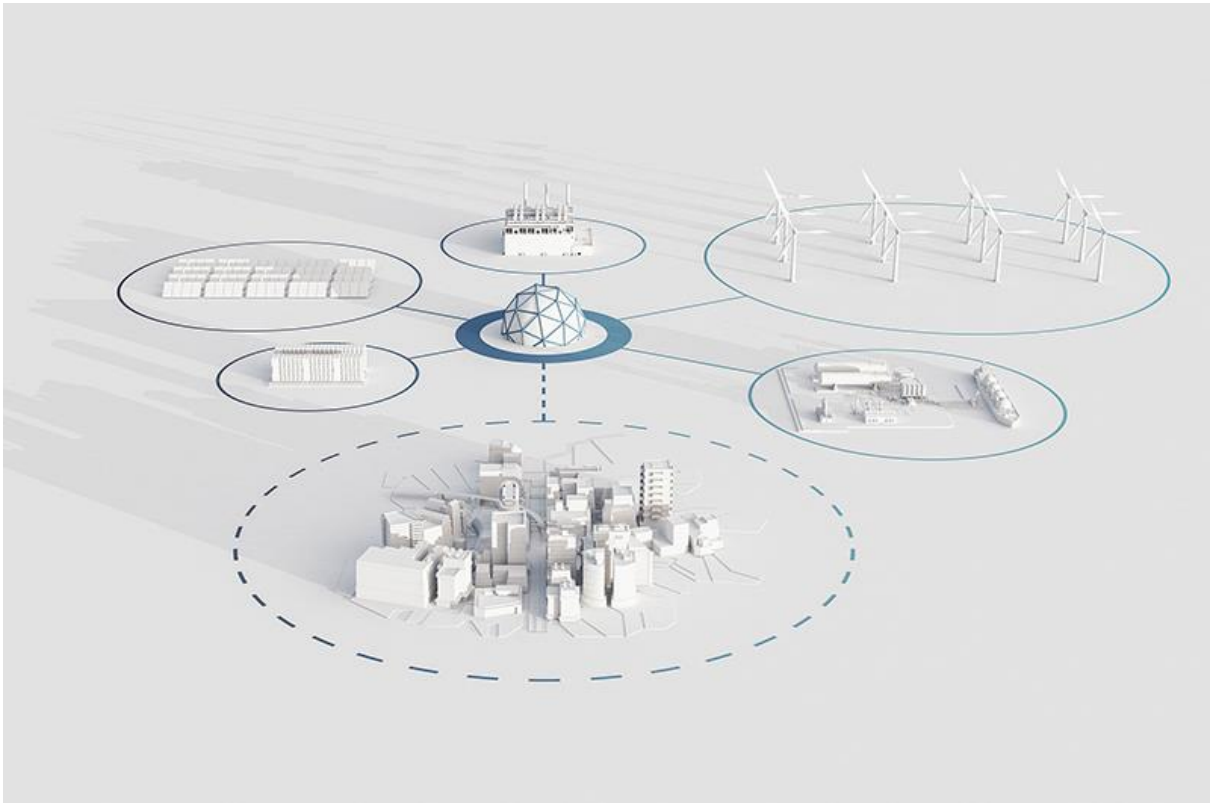

Рисунок 1.1 – MAN Microgrid

Наступний аналог – це гібридна енергетична система GE Power (рис. 1.2). Це універсальний програмний контролер, який надає можливість керувати сонячною енергією та резервним дизельним паливом, що і забезпечує надійне електропостачання в автономних або слабких мережах. Він автоматично визначає пріоритетність джерела енергії, що скорочує затрати, час та ресурси.

Рисунок 1.2 – GE Power

Незалежно від того, чи має користувач на меті зміцнити поточне енергозабезпечення з надійним поєднанням енергії чи вирішити проблему електрозабезпечення сільської місцевості поза електромережею, гібридний контролер GE є енергетичним рішенням для тих регіонів, що не входять до мережі або мають досить слабку електромережу, навіть, щоб увімкнути світло.

До переваг GE контролера можна віднести більш низькі експлуатаційні витрати, ніж у порівнянні зі звичайними дизельними системами, віддалений моніторинг і контроль, що дозволяє не прив’язуватись до відповідного місцезнаходження, та легка масштабованість, завдяки паралельному з’єднанню декількох пристроїв в залежності від рівня попиту[2].

Проаналізувавши наявні аналоги, можна зробити висноски, що вони не зовсім задовольняють умовам поставленої задачі, адже GE контроллер використовує у якості додаткового джерела живлення дизельне паливо, а MAN Microgrid хоч і використовує ВЕУ та СБ для живлення, проте задовольняє потреби

комунальних служб або незалежних виробників енергії. Найбільшим їх недоліком є відсутність систем керування через наявні складності таких систем, неповноти та невизначеності інформації. Тому пропонується інформаційна технологія саме для домашніх господарств, що використовують альтернативні джерела енергії, наприклад для встановлення стабільного забезпечення електроенергії.

2 ПОСТАНОВКА ЗАДАЧІ ТА МЕТОДИ ДОСЛІДЖЕННЯ

2.1 Мета та задачі

Метою даної роботи – створення інформаційної технології для дослідження впливу погодних умов на продуктивність альтернативних джерел енергії зі створенням програмного продукту в якості практичної реалізації інформаційної технології. Розроблений продукт може мати попит серед споживачів електроенергії, що використовують ВДЕ, від індивідуальних господарств до підприємств малого бізнесу, які мають на меті заощадження коштів та збереження навколишнього середовища.

Розроблений програмний продукт повинен виконувати такі функції:

- додавання або зміна місця для збору необхідних даних;
- збереження та пошук даних;
- доступ до метеорологічного сайту;
- визначення впливу погодних умов на ВДЕ;
- визначення кращого джерела електроенагії або їх поєднання в даних погодних та експлуатаційних умовах ;

Створена інформаційна технологія повина бути інтуїтивно зрозумілою та містити зручний для користувача інтерфейс, та працювати не залежно від місцезнаходження користувача.

Для досягнення поставленої мети потрібно вирішити відповідні задачі:

- проаналізувати предметну область;
- обрати мову програмування і відповідно технологію для створення віконних додатків;
- розробити математичну модель на основі нечіткої логіки;

- розробити функціонал програмного продукту, базових маніпуляцій та, відповідно, інтерфейсу;
- провести тестування програмного продукту

Наявність ухваленого технічного завдання, що наведене у Додатку А, є необхідною умовою для коректної реалізації даного проекту. Було вирішено створювати інформаційну технологію саме для гібридних електромереж, бо саме вони є головними тенденціями сьогодення в електроенергетиці.

2.2 Вибір засобів реалізації

Для реалізації поставленої задачі, а саме – програмного продукту у вигляді віконного додатку, було обрано такі засоби реалізації: мова програмування Python 3.7 з технологіями PyQt5, Python GUI та система управління базами даних MySQL з мовою запитів SQL.

Python є інтерпретованою мовою програмування високого рівня загального призначення[6]. Існує досить багато різних мов програмування, тому вибір конкретної мови залежить від унікальних особливостей засобів розробки та виконується на індивідуальний розсуд. Python має певні переваги, серед яких розширюваність мови, що надає можливість удосконалювати мову. Інтерпретатор Python реализований практично на всіх платформах та операційних системах. Крім того, Python підтримує найсучасніші механізми багаторазового використання програмного коду, яким є об'єктно-орієнтоване програмування (ООП). Також можна сказати , що Python має досить високу швидкість розробки. Наприклад, код, написаний на даній мові програмування у тричі, а подекуди у п’ять разів, має менший обсяг, ніж порівняно з такими строго типізованими мовами , як C, C++ або Java, що підвищує продуктивність програміста. Окрім всього цього сценарії Python легко можуть взаємодіяти з іншими частинами програми завдяки різним механізмам інтеграції. Ця інтеграція дозволяє використовувати Python для настройки і розширення функціональних можливостей програмних продуктів. На сьогоднішній день програмний код на мові Python має можливість викликати функції з бібліотек на мові C / C ++, сам викликатися з програм, написаних на мові C / C ++, інтегруватися з програмними компонентами на мові Java, взаємодіяти з такими платформами, як COM і .NET , і проводити обмін даними через послідовний порт або через мережу за допомогою таких протоколів, як SOAP, XML-RPC і CORBA. Тому можна сказати, що Python – це не відокремлений інструмент[15].

В даній мові програмування для створення GUI додатків найчастіше використовують модуль PyQt5. PyQt5 об'єднує платформу Qt C++ для міжплатформних додатків і крос-платформний інтерфейс Python. Не можна сказати, що Qt – це лише інструментарій для створення графічного інтерфейсу. Цей модуль також включає в себе Unicode, бази даних SQL, SVG, OpenGL, XML, веб-браузер з повним функціоналом та багато іншого. Досить значна частина можливостей Python надає широке різноманіття функцій, що включають HTTP-сервери, XML-парсери, засоби стиснення даних та доступ до баз даних, а також, графічні інтерфейси користувача[8].

MySQL – найпопулярніша база даних у світі з відкритим кодом. Найчастіше при виборі баз даних обирають саме MySQL завдяки її надійності, продуктивності та простоті використання. Необхідною умовою для коректної роботи з базами даних є використання SQL, що й надає відповідний функціонал. SQL (Structured Query Language) представляє собою структуровану мову запитів (переклад з англійської), яка орієнтована на роботу саме з реляційними базами даних. За своєю суттю мова досить проста і складається з команд, за допомогою яких розробник має можливість працювати з великими об’ємами даних, реалізуючи додавання, видалення, зміну інформації і здійснення зручного пошуку[5].

Говорячи про реалізацію даного програмного продукту, то можна побачити широкі можливості мови програмування Python завдяки її інтегрованості, а створена бази даних MySQL робить створений продукт універсальним.

2.3 Планування робіт

Наступний етап, який слідує після формування мети проекту, визначення переліку задач та обирання необхідного інструментарію, є планування робіт. Перш за все було проведено деталізацію мети за допомогою методології SMART. Далі для відображення ієрархічної структури плану робіт було розроблено діаграму OBS, а для визначення учасників, які виконують той чи інший етап створення програмного продукту, було розроблено діаграму WBS. Після виконання попередніх кроків виконується формування діаграми Ганта та мережі PDM, які є невід'ємними частинами контролю строків реалізації проекту. За допомогою мережі PDM можна з легкістю визначити тривалість будь-якого етапу робіт, критичний шлях та при бажанні змінити терміни виконання окремих процесів для оптимізації тривалості проекту. Наступним етапом було визначення всіх можливих ризиків, які можуть виникнути в процесі розробки та дії для їх запобігання. І врешті було розраховано бюджет проекту, який складається з заробітної плати учасників проекту. Повна інформація щодо діаграм, мереж, ризиків, тощо наведена у Додатку Б.

3 МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ВИЗНАЧЕННЯ ВПЛИВУ ПОГОДНИХ УМОВ

3.1 Структурно-функціональне моделювання діяльності інформаційної технології

Для функціонального моделювання об’єкта будь-якої предметної області використовують методологію SADT. Функціональна модель SADT відображає функціональну структуру об’єкта, тобто вироблені їм дії та зв’язки між ними.

У результаті застосування методології SADT маємо модель, що складається з діаграм, фрагментів текстів і глосарію, які мають посилання один на одного. Відповідно до даної методології, головними компонентами моделі – є діаграми, а всі функції і інтерфейси на них представлені блоками і дугами. Тип інтерфейсу визначається місцем з’єднання дуги з блоком. Якщо інформація входить в блок зверху, то вона – керуюча, тоді як вхідна інформація, що піддається обробці, записується зліва, а результати її обробки – справа. Механізмом, що здійснює операцію, може виступати як людина, так і автоматизована система, він представляється дугою, що входить в блок знизу.(29)

Для створення структурно-функціональної моделі біло використано програмний продукт AllFusion Process Modeler r7.

Розглянемо процес роботи програмного продукту «Hybrid». Контекстну діаграму даного процесу представлено на рис. 3.1.

Рисунок 3.1 – Контекстна діаграма

Вхідними даними до функції «Визначення впливу погодних умов на продуктивність альтернативних джерел енергії» є:

- Список міст;
- Погодні дані Gismeteo;
- Технічні характеристики ВЕУ та СБ;
- Дані Excel-таблиць.

На виході ми отримуємо визначене джерело електроенергії.

Визначення впливу погодних умов керується вимогами користувача, функціональними обмеженнями сайту Gismeteo, документацією phpMyAdmin та математичною моделлю.

Механізмами або ресурсами є Інтернет, БД та програмний продукт.

У зв’язку з тим, що контекстна діаграма надає лише загальний опис системи, то необхідно декомпозувати її для деталізації структури. Цей процесс дозволяє детально ознайомитись з послідовністю виконання робіт для досягнення потрібного результату.

Дана діаграма має 2 рівні деталізації. На першому рівні деталізації моделі батьківська діаграма декомпозується на 3 блоки, що представлені на рис. 3.2:

– Заповнення Бази Даних;
– Розрахунок потужностей;
– Визначення пріоритетності.

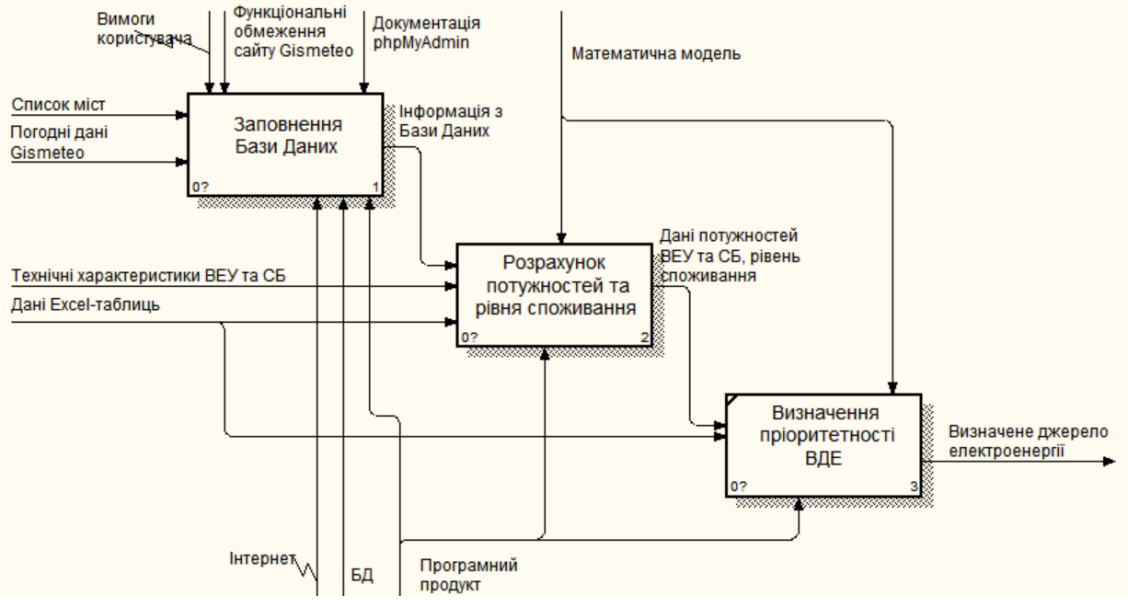

Рисунок 3.2 – Діаграма декомпозиції IDEF0

Вхідними стрілками до діяльності «Заповнення Бази Даних» є «Список міст» та «Погодні дані Gismeteo»; вихідними – «Інформація з заповненої Бази Даних»; стрілками контролю – «Вимоги користувача», «Функціональні обмеження сайту Gismeteo» та «Документація phpMyAdmin»; стрілками механізмів – «Інтернет», «БД» та «Програмний продукт».

Вхідними стрілками до діяльності «Розрахунок потужностей» є «Інформація з заповненої Бази Даних», «Технічні характеристики ВЕУ та СБ» та «Дані Excel-таблиць»; вихідними – «Дані потужностей ВЕУ та СБ, рівень споживання»; стрілкою контролю – «Математична модель»; стрілкою механізмів – «Програмний продукт».

Вхідними стрілками до діяльності «Визначення пріоритетності ВДЕ» є «Дані потужностей ВЕУ та СБ, рівень споживання», «Дані Excel-таблиць»;

вихідною – «Визначене джерело електроенергії»; стрілкою контролю – «Математична модель»; стрілкою механізмів – «Програмний продукт».

Як було зазначено раніше, контекстна діаграма має 2 рівні декомпозиції. Діаграму декомпозиції процесу «Заповнення Бази Даних» зображено на рис. 3.3.

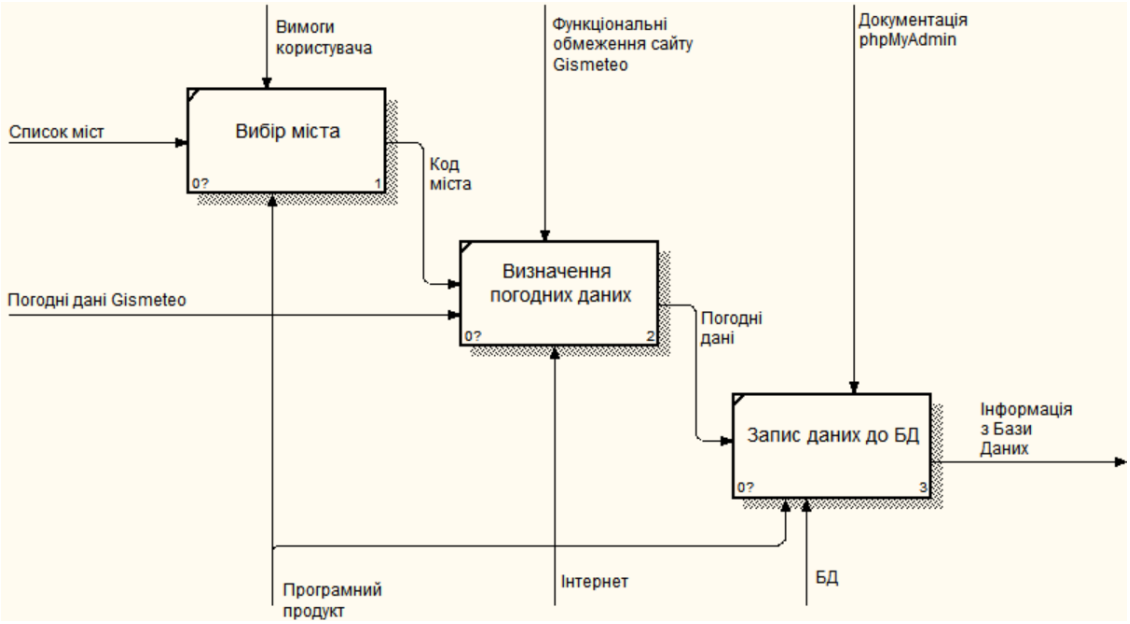

Рисунок 3.3 – Діаграма декомпозиції IDEF0 «Заповнення Бази Даних»

Вхідною стрілкою до діяльності «Вибір міста» є «Список міст»; вихідною – «Код міста»; стрілкою контролю – «Вимоги користувача»; стрілкою механізмів – «Програмний продукт».

Вхідними стрілками до діяльності «Визначення погодних даних» є «Код міста» та «Погодні дані Gismeteo»; вихідною – «Погодні дані»; стрілкою керування – «Функціональні обмеження сайту Gismeteo»; стрілкою механізмів – «Інтернет».

Вхідними стрілками до діяльності «Запис даних до БД» є «Погодні дані»; вихідними – «Заповнена База Даних»; стрілкою керування – «Документація phpMyAdmin»; стрілками механізмів – «БД» та «Програмний продукт».

На рисунку 3.4 зображено діаграму декомпозиції процесу «Розрахунок потужностей та рівня споживання».

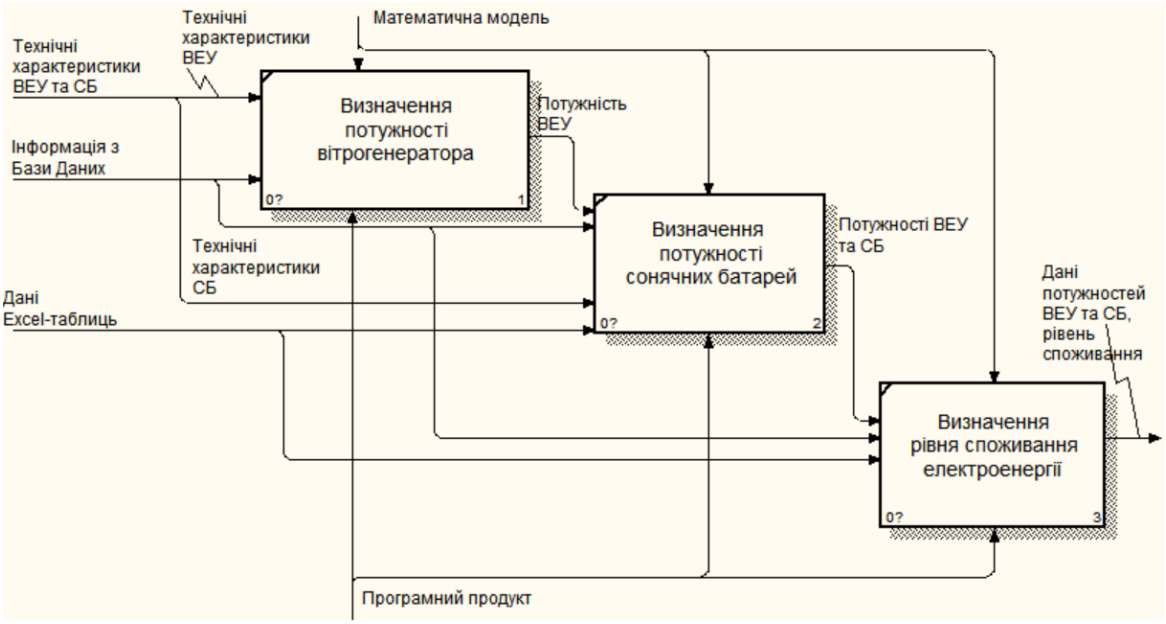

Рисунок 3.4 – Діаграма декомпозиції IDEF0 «Розрахунок потужностей та рівня споживання»

Вхідними стрілками до діяльності «Визначення потужності вітрогенератора» є «Технічні характеристики ВЕУ» та «Заповнена База Даних»; вихідними – «Потужність ВЕУ»; стрілками контролю – «Математична модель»; стрілками механізмів – «Програмний продукт».

Вхідними стрілками до діяльності «Визначення потужності сонячних батарей» є «Заповнена База Даних», «Технічні характеристики СБ» та «Дані Excel-таблиць»; вихідними – «Потужності ВЕУ та СБ»; стрілками контролю – «Математична модель»; стрілками механізмів – «Програмний продукт».

Вхідними стрілками до діяльності «Визначення рівня споживання електроенергії» є «Заповнена База Даних», «Дані Excel-таблиць» та «Потужності ВЕУ та СБ»; вихідними – «Потужності ВЕУ та СБ, рівень споживання»; стрілками контролю – «Математична модель»; стрілками механізмів – «Програмний продукт».

3.2 Моделювання бази

Реляційна модель являє собою певну сукупність даних, що складається з набору двомірних таблиць. Найбільш звичною та зручною для представлення даних є саме реляційна модель.

Таблиця є базовою одиницею даних реляційної моделі, яка називається «відношенням» (relation). Таблиці складаються зі стовпців (columns), які визначають типи даних, та рядків (rows), що містять множину значень стовпців.

Найчастіше для моделювання баз даних використовують модель «сутність-зв’язок». Така модель описує концептуальні схеми, використовуючи узагальнені конструкції блоків. Діаграми «сутність-зв’язок» перш за все визначають дані і відносини між ними, використовуючи які виконується деталізація сховищ даних системи, що проектується.(30)

ER-діаграми має перевагу, перш за все, в тому, що процес виділення сутностей, зв’язків та атрибутів є ітераційним. Розробляючи таку інформаційну модель ми повинні отримати список сутностей предметної області, опис взаємозв’язків між ними та список атрибутів сутностей.

У ході аналізу поставленої задачі було визначено перелік функцій, що повинна виконувати змодельована система:

- зберігати інформацію прогнозу погоди;
- зберігати інформацію про перелік міст;
- зберігати інформацію про потужності ВЕУ, СБ та ін.;

Тож виходячи зі списку визначених функцій виділимо перелік сутностей:

- погода (Weather) – явний кандидат на сутність;
- місто (Place) – явний кандидат на сутність;
- потужність (Power) – явний кандидат на сутність;

Виникає очевидний зв’язок між сутностями – «один прогноз погоди повинен визначатися лише для одного міста». Розглядаючи цей зв’язок з іншої сторони, то в одному місті повинно бути один або більше прогнозів погоди. Перший варіант діаграми має вигляд:

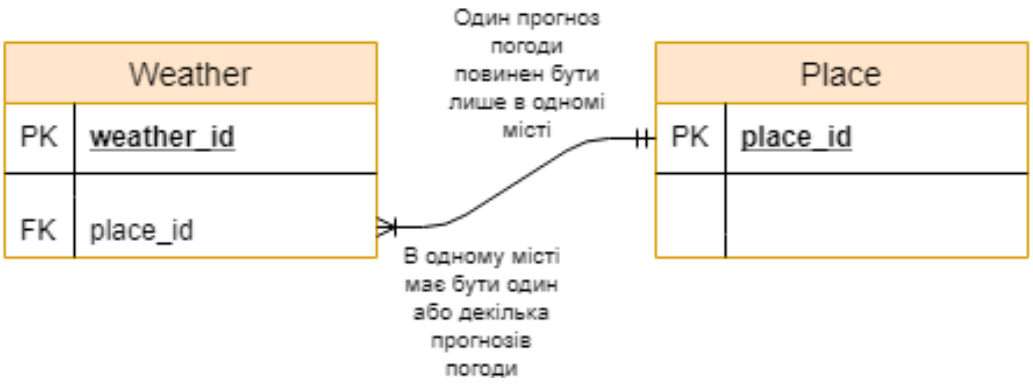

Рисунок 3.5 – Зв’язок між сутностями Weather та Place

Далі визначаємо зв’язок між сутностями Погода та Потужність. На основі одних погодних даних виконуються розрахунки потужностей складових гібридної електромережі. Та для кожної потужності потрібні дані лише одного прогнозу погоди. Тож після даного уточнення діаграма набуває такого вигляду:

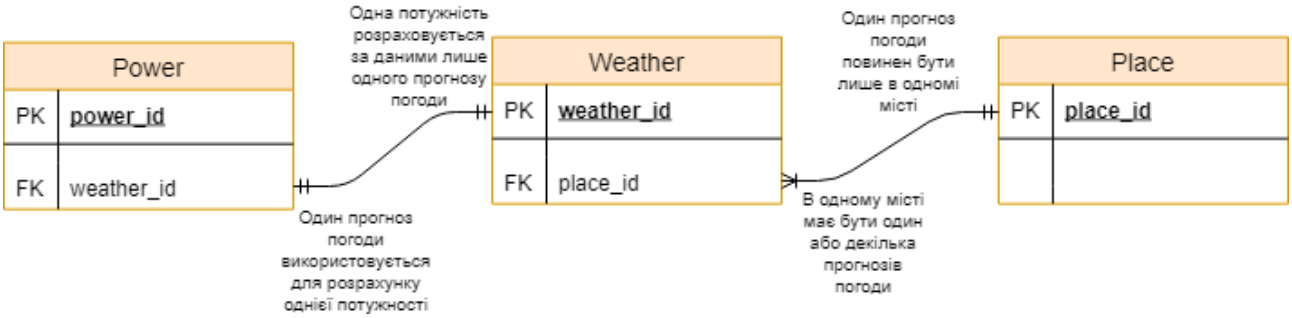

Рисунок 3.6 – Відображення зв’язків між сутностями

Наступним етапом розглядаємо властивості сутностей (кандидати в атрибути):

Погода

Погода має такі атрибути: номер прогнозу погоди, швидкість вітру, напрямок вітру, температура, хмарність, дата, час, місто.

Місто

Місто має атрибути: номер міста, назва міста, країна, порядковий номер міста на сайті.

Потужність

Потужність має атрибути: номер розрахунку потужностей, потужність СБ, потужність ВЕУ, ємність АБ, тип підключеного ВДЕ, рівень споживання електроенергії, погоні дані, на основі яких були виконані розрахунки потужностей.

Далі доповнимо створену діаграму атрибутами. На рисунку 3.7 зображено остаточний варіант ER-діаграми.

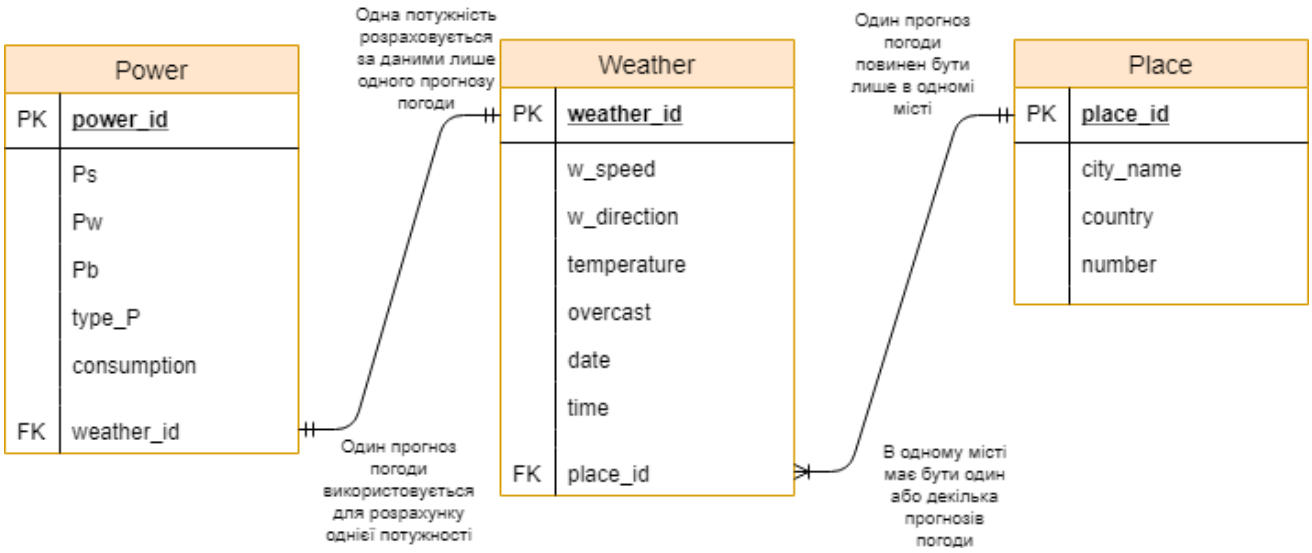

Рисунок 3.7 – Концептуальна модель бази даних

3.3 Моделювання варіантів використання

Для кращого розуміння роботи системи часто використовують опис функціональності через варіанти використання. Діаграми варіантів використання описують перелік функцій, які повинна робити система, для формування загальних вимог до поведінки системи, що проектується.

Для розробки моделі варіантів використання інформаційної технології було визначено відповідний перелік акторів:

- User – користувач, який має на меті настроїти гібридну мережу під власні потреби;
- Gismeteo.com – сайт прогнозу погоди, дані якого використовуються для подальших підрахунків;
- MS Excel – файли програми Microsoft Office Excel, які містять необхідні дані для коректних розрахунків;
- Data Base – База Даних, яка накопичує всі необхідні для роботи програмного продукту дані;
- MS Word – файл програми Microsoft Office Word для запису звіту;

Після визначення всіх можливих акторів, які взаємодіють з системою, формується перелік варіантів використання:

- Parsing – визначення погодних умов певного міста;
- Search – пошук міста;
- Calculation – обчислення потужностей та рівня споживання електроенергії;
- Priority – визначення пріоритетного джерела енергії;
- Save – збереження отриманого результату;

Використовуючи сформовані дані можливих варіантів використання та акторів було розроблено діаграму варіантів використання, що зображено на рисунку 3.8.

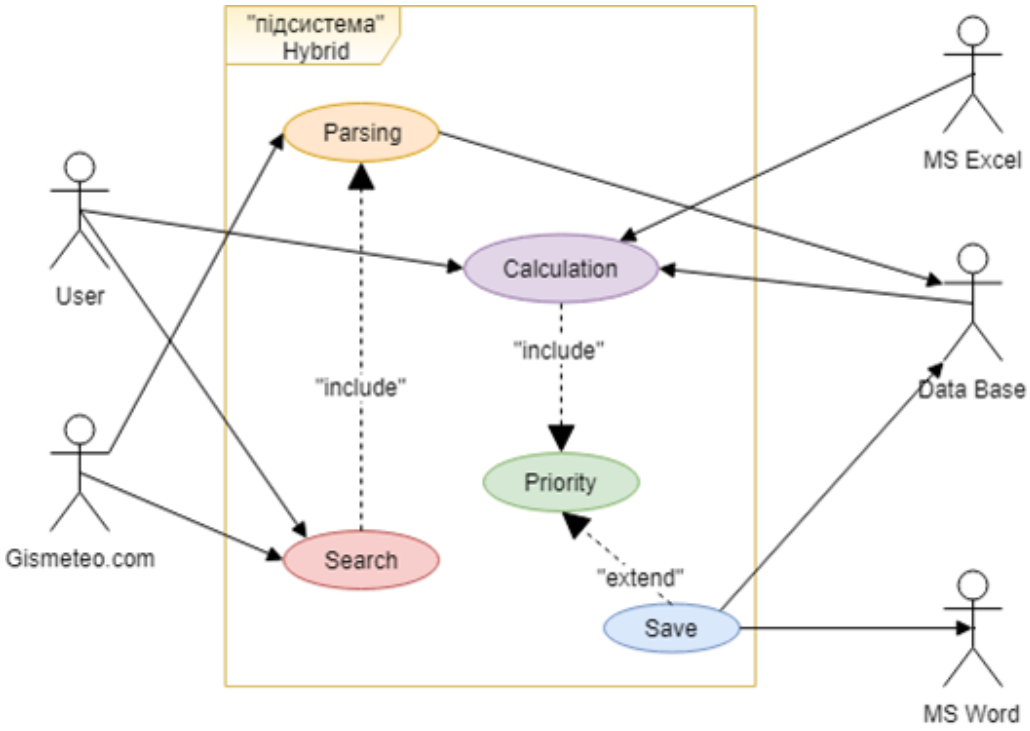

Рисунок 3.8 – Діаграма варіантів використання

3.4 Математична модель

3.4.1 Модель визначення технічних показників енергетичної системи

У даному проекті ми розглядаємо гібридну енергосистему, що може обслуговувати один або декілька будівель. Такі системи призначені для приватних

споживачів або для малого чи середнього бізнесу. Системи складаються з сонячних панелей, ряд вітряних генераторів та акумуляторні батареї для зберігання електроенергії. Крім того, гібридна мережа має зв’язок із зовнішньою мережею для передачі надлишково згенерованої електроенергії та додаткового споживання. Схема роботи такої системи зображено на рисунку 3.9.

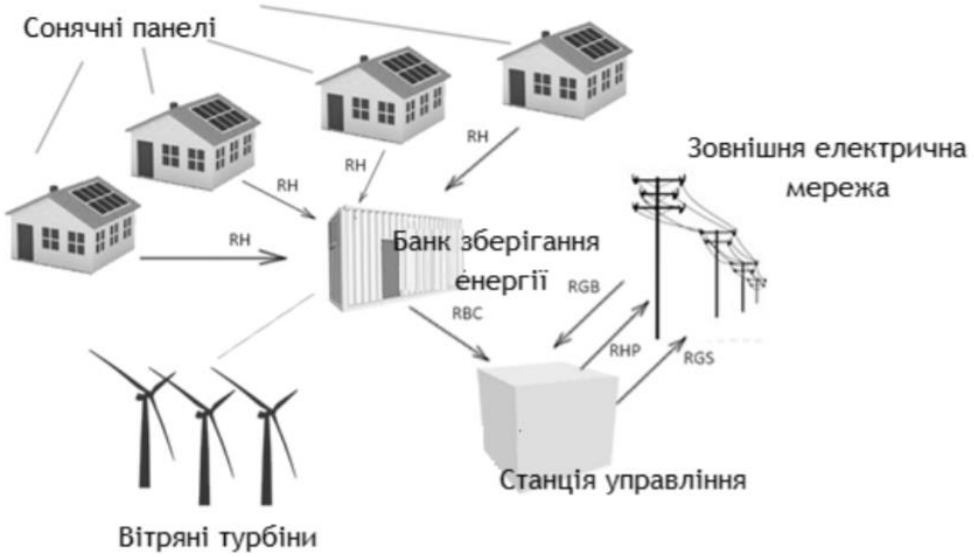

Рисунок 3.9 – Схема роботи гібридної електромережі

Така енергосистема виробляє електроенергію завдяки сонячним панелям (Es) та вітряним турбінам (Ew), яка використовується для власних потреб. У разі вироблення надлишкової електроенергії вона буде зберігатися в акумуляторних батареях (Eb) або передана до загальної електричної мережі. Зв’язок із зовнішньою електромережею також дає змогу отримувати додаткову енергію, якщо власна мережа виробляє недостатній рівень електроенергії. Алгоритм роботи, за яким працює гібридна енергетична система, представлено на рисунку 3.10.

На першому етапі визначається потужність від сонячних панелей Es, вітряних турбін Ew та ємність акумуляторної батареї за час t. Після чого визначаються запити від господарства на електроенергію RH. Далі визначається

джерело енергії, якщо згенерованої енергії недостатньо, то підключається або акумулятор, або зовнішня мережа. В іншому випадку надлишок енергії буде спрямовано до акумулятора чи загальної мережі.

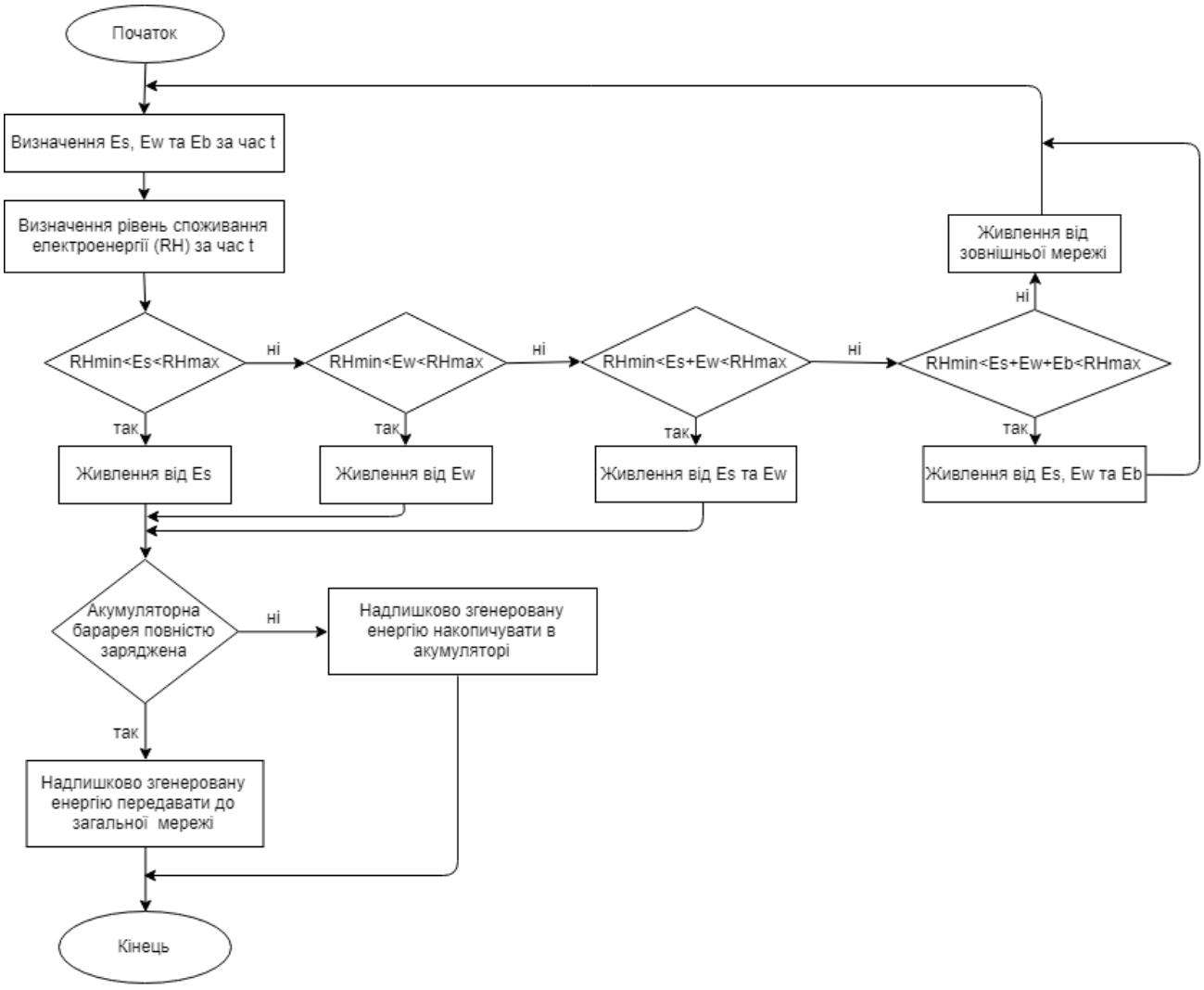

Рисунок 3.10 – Алгоритм вибору джерела електроенергії

При розробці математичної моделі ВЕУ було враховано такі фактори, як вплив температури повітря та залежність аеродинамічних характеристик вітроротора від швидкості вітру.

Згідно з висновками вчених останніх років, то аеродинамічна характеристика вітроротора $C_p(\lambda)$ не є константою та змінюється відповідно до швидкості вітру за певною закономірністю[14]:

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

Також визначимо залежність густини повітря від температури повітря $t_{\text{п}}$:

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

Для подальших розрахунків потрібно визначити потужність вітроелектроустановки наступним чином:

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

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

Якщо ми говоримо про потужність сонячних батарей, то потрібно визначити максимальну потужність СБ. Пропонується визначити її, як трикутне нечітке число:

$$P =< P_{min}, P_{mod}, P_{max} > \quad (3.4)$$

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

Для знаходження залежності $P_{min}, P_{mod}, P_{max}$ від конструктивних та зовнішніх факторів було оброблено певні експериментальні дані для визначення впливу на електричні характеристики сонячних батарей. Було досліджено 16 груп сонячних елементів площиною 0,0403 м2 при освітленості (550-1260) Вт/м2 та

температурі (12-70)°С. До складу груп було включено фотоелементі різних розмірів, щоб врахувати невизначеність і такого роду.

Як результат обробки даних було отримано залежності, що входять до нечіткої моделі (3.4):

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

де S – загальна площина сонячних батарей, що входять до складу гібридної електромережі; E – освітленість; t – температура повітря.

У зв'язку з тим, що потужність СБ представлена у вигляді нечіткого кортежу P, де $P_{min} < P_{mod} < P_{max}$, для якої функція належності $\mu_P(x)$ має вигляд:

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

де для довільного числа $x \in [P_{min}, P_{mod}]$ справедливим є представлення $x = P_{min} + \alpha(P_{mod} - P_{min})$, а для довільного $x \in [P_{mod}, P_{max}]$ – $x = P_{max} - \alpha(P_{max} - P_{mod})$, де $\alpha \in [0,1]$ – заданий рівень належності числа x нечіткій множині P.

Для визначення коефіцієнта α було опрацьовано річні дані інсоляції в різних умовах хмарності (таблиця 3.1). У звязку з тим, що кажне метеорологічне джерело визначає ступінь хмарності (слабка хмарність, хмарно та ін.) у відсотках по різному, вирішено взяти за основу дані сайту Gismeteo.com, де зазначалось, що:

– «Ясно» відповідає 0% хмарності;
– «Слабка хмарність» відповідає 25% хмарності;

– «Помірна хмарність» відповідає 50%;
– «Хмарно» відповідає 100% хмарності.

Таблиця 3.1 Інсоляція в умовах хмарності

Місяць	Інсоляція при безхмарному небі, кВт*год/м2	Інсоляція в умовах повної хмарності, кВт*год/м2
1	63,25	38,75
2	94,37	59,08
3	160,99	95,48
4	197,34	131,4
5	240,51	175,15
6	245,57	175,5
7	244,57	187,24
8	207,51	165,23
9	166,81	117,9
10	119,06	78,12
11	76,31	42,16
12	56,44	32,34

Визначено, що при сильній та помірній хмарності потужність СБ буде належати проміжку $[P_{min}, P_{mod}]$, при умові ясної погоди або малої хмарності потужність СБ належить проміжку $[P_{mod}, P_{max}]$.

Тож відповідно до попередніх даних коефіцієнт α для «Хмарно» та «Помірна хмарність» визначатиметься за наступною формулою:

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

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

У випадку малохмарної та ясної погоди коефіцієнт α визначається за формулою:

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

де $E2(n)$ – значення інсоляції в умовах повної хмарності, кВт*год/м2; $E1(n)$ – значення інсоляції в умовах безхмарного неба, кВт*год/м2; n – номер місяця; φ – рівень безхмарності,% (для «Ясно» - 1, для «Помірна хмарність» - 0,25).

Таблиця 3.2 – Значення коефіцієнта α

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

Тож можна побачити, що за умови безхмарної ясної погоди потужність СБ досягає свого максимуму при заданих погодних та технічних умовах.

Відповідно до умов роботи гібридної електромережі, при недостатньому рівні генерації енергії від СБ та ВЕУ використовуються акумуляторні батареї, які працюють в режимі заряд-розряд. Якщо АБ заряджається, то величина енергії E_B в батареї за час t:

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

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

Кількість енергії E_B в батареї за час t за умови, що батарея розряджається, визначається наступним чином:

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

Якщо кількість електроенергії перевищує кількість потрібної споживачу, то надлишок буде нагромаджуватись в АБ, доки кількість енергії в АБ не досягне свого максимуму $E_{B\,max}$. Якщо за такої ситуації виникне надлишок згенерованої енергії $EPG(t)$ в АБ, то цей надлишок буде направлено до загальної електромережі. Погодинне значення $EPG(t)$ визначається за формулою:

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

де $E_{B\,max}$ – максимальне значення енергії в акумуляторній батареї.

Наступним етапом треба визначити рівень споживання електроенергії користувачем. Електроспоживання являє собою циклічний процес, який залежить

від низки зовнішніх, таких як кліматичні особливості або інше, та внутрішніх факторів, таких як графік святкових та вихідних днів. Якщо проаналізувати добові графіки споживання електроенергії (рис. 3.11), можна побачити, що споживання є нерівномірним протягом доби та поділити його на 3 періоди: ранковий, вечірній та фоновий.

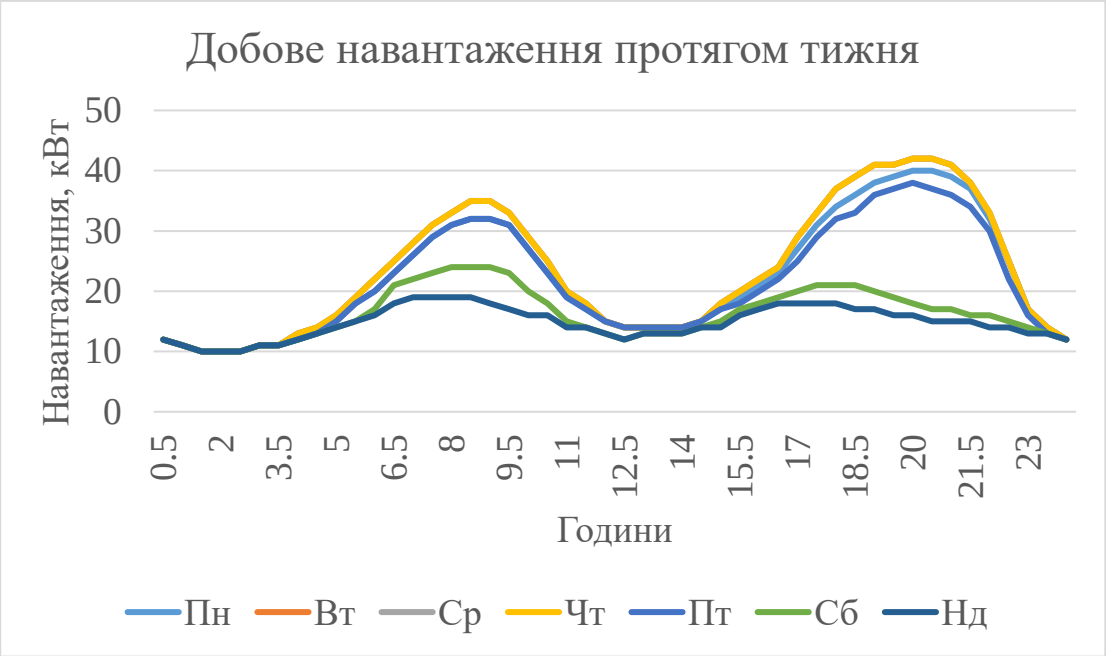

Рисунок 3.11 – Приклад добового графіка електроспоживання

Оскільки в графіку електроспоживання спостерігаються протягом доби чітко виражені піки, то доречно представити прогнозну функцію як суперпозицію Гауссових кривих для піків електроспоживання і як пряму для фонового споживання (рис. 3.12).

Рисунок 3.12 – Приклад прогнозної добової функції

Процес споживання електроенергії несе в собі певну невизначеність, тому миттєве споживання представляється нечітким трикутним числом $W(t) =< W^{mod}(t), W^{min}(t), W^{max}(t) >$. Тож відповідно до добового циклу споживання електроенергії функція набуває наступного вигляду:

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

$$W_{фон}^{mod}(t) = a_1 t + a_2; W_{фон}^{min}(t) = b_1 t + b_2; W_{фон}^{max}(t) = c_1 t + c_2;$$
$$W_{1пік}^{mod}(t) = a_3 \cdot exp(-(t - a_4)^2 / a_5); W_{1пік}^{min}(t) = b_3 \cdot exp(-(t - b_4)^2 / b_5);$$
$$W_{1пік}^{max}(t) = c_3 \cdot exp(-(t - c_4)^2 / c_5); \quad (3.13)$$
$$W_{2пік}^{mod}(t) = a_6 \cdot exp(-(t - a_7)^2 / a_8); W_{2пік}^{min}(t) = b_6 \cdot exp(-(t - b_7)^2 / b_8);$$
$$W_{2пік}^{max}(t) = c_6 \cdot exp(-(t - c_7)^2 / c_8).$$

де t – поточний час протягом доби; a_1-a_8, b_1-b_8, c_1-c_8 – коефіцієнти. Тож функція споживання набуває вигляду, який представлено на рис. 3.13.

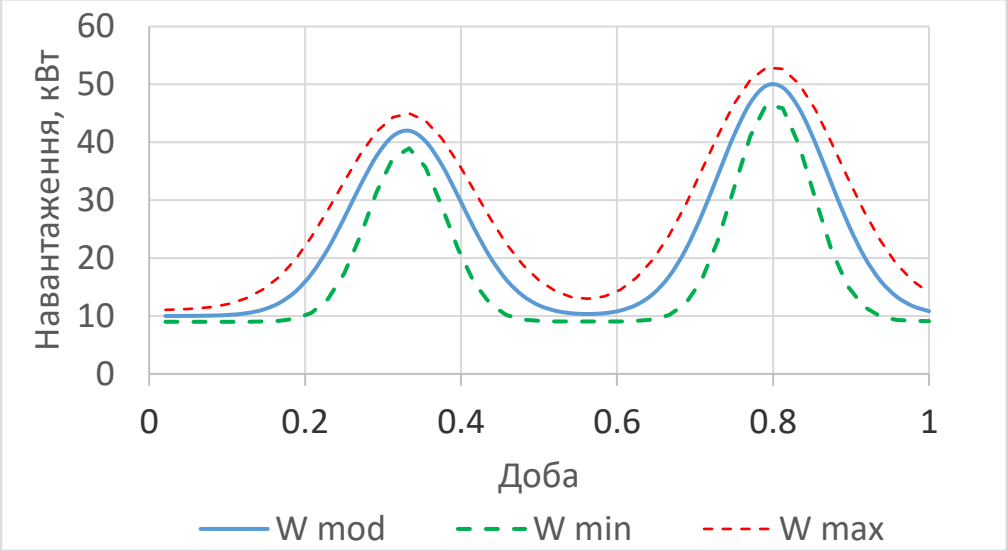

Рисунок 3.13 – Ілюстрація побудови нечіткої прогнозної моделі добового електроспоживання

Коефіцієнти a_1-a_8, b_1-b_8, c_1-c_8 несуть в собі залежність від пори року та тижня в році. Пропонується врахувати тижневі цикли як поліноміальні моделі в наступному вигляді:

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

де d_n – номер дня тижня. У свою чергу отримані коефіцієнти залежать від номера тижня в році і представлені залежністю (3.13), де n – номер тижня в році.

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

3.4.2 Модель нечітких правил

Далі для коректної роботи прогнозної моделі роботи гібридної електромережі на основі складу мережі з урахуванням погодних даних було розроблено перелік нечітких правил. Розглянемо методику побудови функції належності на прикладі критерію «ДжерелоЕлектроенергії». Приклад одного з правил нечіткого логічного виведення рішення щодо визначення джерела електроенергії (Z) наведено в таблиці 3.3.

Чіткі значення змінних «Хмарність» (OY), «Швидкість вітру» (WY), «ЄмністьАБ» (AY), «ПотужністьВЕУ» (BY) та «ПотужністьСБ» (SY) є складовими передумов правил у вигляді предикатів.

$$Z = \{Z1, Z2, Z3, Z4\} \quad (3.16)$$

$$Z = F(OY, WY, AY, BY, SY) \quad (3.17)$$

де OY – змінна, що характеризує «Хмарність», WY – змінна, що характеризує «ШвидкістьВітру», AY – змінна, що характеризує «ЄмністьАБ», BY – змінна, що характеризує «ПотужністьВЕУ», SY – змінна, що характеризує «ПотужністьСБ».

$$OY = OY1, OY2, OY3, OY4 \quad (3.18)$$

$$WY = WY1, WY2, WY3 \quad (3.19)$$

$$AY = AY1, AY2 \quad (3.20)$$

$$BY = BY1, BY2 \quad (3.21)$$

$$SY = SY1, SY2 \quad (3.22)$$

Таблиця 3.3 – Приклад нечіткого правила для критерію «Джерело електроенергії»

Передумова	OY is OY1, WY is WY1, AY is AY1
Формулювання правила	If OY is OY1 and WY is WY1 then If AY is AY1 then Z=Z1, else Z=Z4.
Інтерпретація	Якщо «Хмарність»=OY1і «ШвидкістьВітру»=WY1, то Якщо «ЄмністьАБ»=AY1, То Z=Z1, Інакше Z=Z4.

Лінгвістична змінна OY – «Хмарність» визначається в межах універсальної множини U(OY)=[0,1]. Терм-множина лінгвістичної змінної T(OY)=<сильна хмарність, хмарність, слабка хмарність, ясно> містить терми лінгвістичних змінних OY1, OY2, OY3, OY4.

Лінгвістична змінна WY – «ШвидкістьВітру» визначається в межах універсальної множини U(WY)=[0,30] (м/с). Множина термів лінгвістичної змінної T(WY)=<мала, середня, велика> містить відповідно терми лінгвістичних змінних WY1, WY2, WY3.

Лінгвістична змінна AY – «ЄмністьАБ» визначається в межах універсальної множини U(AY)=[0,10] (кВт). Множина термів лінгвістичної змінної T(AY)=<достатня, недостатня> містить терми лінгвістичних змінних AY1, AY2.

Лінгвістична змінна BY – «ПотужністьВЕУ» визначається в межах універсальної множини U(BY)=[0,10] (кВт). Множина термів лінгвістичної змінної T(BY)=<достатня, недостатня> містить терми лінгвістичних змінних BY1, BY2.

Лінгвістична змінна SY – «ПотужністьСБ» визначається в межах універсальної множини U(SY)=[0,10] (кВт). Множина термів лінгвістичної змінної T(SY)=<достатня, недостатня> містить терми лінгвістичних змінних SY1, SY2.

Лінгвістична змінна Z – «ДжерелоЕлектроенергії» визначається в межах універсальної множини U(Z)=[0,1]. Множина термів лінгвістичної змінної T(Z)=<АБ, ВЕУ, СБ, Загальна мережа> містить терми лінгвістичних змінних Z1, Z2, Z3, Z4.

Тож далі сформовано повний перелік нечітких правил, що були використані в роботі прогнозної моделі роботи гібридної електромережі:

1. Якщо Хмарність є Сильна Хмарність і Швидкість вітру є Мала, то Якщо АБ є достатній, то АБ підключено, Інакше Загальна мережа підключена;
2. Якщо Хмарність є Сильна Хмарність і Швидкість вітру є Середня, то Якщо ВЕУ є достатній, то ВЕУ підключено, Інакше Якщо АБ є достатній, то АБ підключено, Інакше Загальна мережа підключена;
3. Якщо Хмарність є Сильна Хмарність і Швидкість вітру є Велика, то

Якщо ВЕУ є достатній, то ВЕУ підключено, Інакше Якщо АБ є достатній, то АБ підключено, Інакше Загальна мережа підключена;

4. Якщо Хмарність є Хмарно і Швидкість вітру є Мала, то

Якщо АБ є достатній, то АБ підключено, Інакше Загальна мережа підключена;

5. Якщо Хмарність є Хмарно і Швидкість вітру є Середня, то

Якщо ВЕУ є достатній і СБ є достатній, то ВЕУ підключено і СБ підключено, Інакше Якщо АБ є достатній, то АБ підключено, Інакше Загальна мережа підключена;

6. Якщо Хмарність є Хмарно і Швидкість вітру є Велика, то

Якщо ВЕУ є достатній і СБ є достатній, то ВЕУ підключено і СБ підключено, Інакше Якщо АБ є достатній, то АБ підключено, Інакше Загальна мережа підключена;

7. Якщо Хмарність є Слабка хмарність і Швидкість вітру є Мала, то

Якщо ВЕУ є достатній і СБ є достатній, то ВЕУ підключено і СБ підключено, Інакше Якщо АБ є достатній, то АБ підключено, Інакше Загальна мережа підключена;

8. Якщо Хмарність є Слабка хмарність і Швидкість вітру є Середня, то

Якщо ВЕУ є достатній і СБ є достатній, то ВЕУ підключено і СБ підключено, Інакше Якщо АБ є достатній, то АБ підключено, Інакше Загальна мережа підключена;

9. Якщо Хмарність є Слабка хмарність і Швидкість вітру є Велика, то

Якщо ВЕУ є достатній і СБ є достатній, то ВЕУ підключено і СБ підключено, Інакше Якщо АБ є достатній, то АБ підключено, Інакше Загальна мережа підключена;

10. Якщо Хмарність є Ясно і Швидкість вітру є Мала, то

Якщо ВЕУ є достатній і СБ є достатній, то ВЕУ підключено і СБ підключено, Інакше Якщо АБ є достатній, то АБ підключено, Інакше Загальна мережа підключена;

11.Якщо Хмарність є Ясно і Швидкість вітру є Середня, то
Якщо ВЕУ є достатній і СБ є достатній, то ВЕУ підключено і СБ підключено, Інакше Якщо АБ є достатній, то АБ підключено, Інакше Загальна мережа підключена;

12.Якщо Хмарність є Ясно і Швидкість вітру є Велика, то
Якщо ВЕУ є достатній і СБ є достатній, то ВЕУ підключено і СБ підключено, Інакше Якщо АБ є достатній, то АБ підключено.

Провівши інформаційне моделювання було створено всі необхідні моделі для інформаційної технології. Спираючись на зібрані дані освітленості та хмарності було розроблено математичну модель з нечіткими правилами та прогнозною моделлю роботи гібридної електромережі.

4 РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ

4.1 Підготовчий процес перед створенням головного модулю продукту

4.1.1 Створення прототипу

Для розуміння роботи програмного продукту з додатковими елементами застосовується архітектура програмного продукту(рис.4.1). При відсутності продуманої структури програмного продукту можуть відчуватися труднощі при пошуку інформації або навігації по додатку.

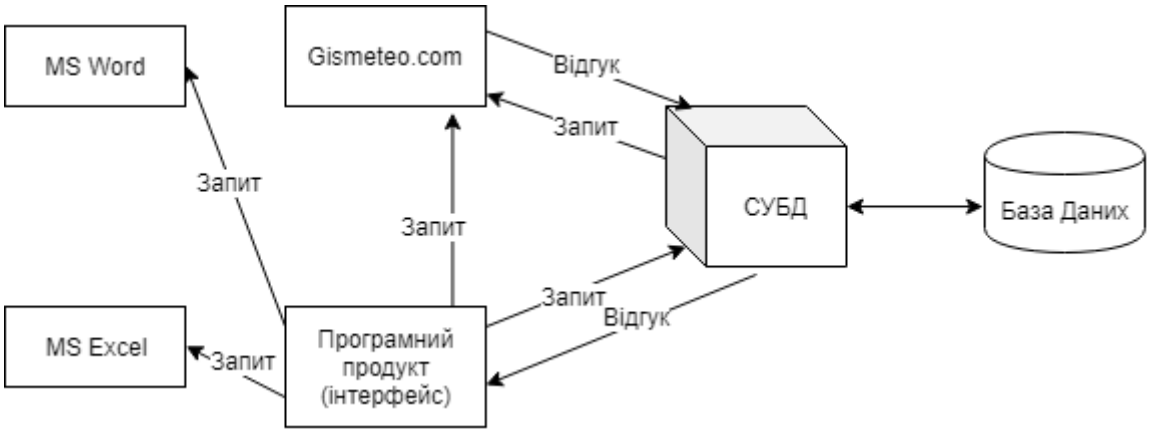

Рисунок 4.1 – Схема взаємодії компонентів додатку «HYBRID»

Пред розробленням додатку за допомогою мови програмування Python треба розробити прототип майбутнього діалогового вікна. Прототип має містити всі необхідні блоки, що відповідають потрібному функціоналу додатку. Прототип зображено на рисунку 4.2, який дає загальне уявлення про розміщенні основних блоків.

Рисунок 4.2 – Прототип додатку

Функціональна панель повинна надавати можливість користувачу зберігати отриманий результат у вигляді звіту за бажанням користувача, за що відповідає кнопка «Звіт». Також одною з задач було надання користувачу доступу до метеорологічного сайту, в нашому випадку – це Gismeteo.com. Ця функція виконується натисканням кнопки «Прогноз погоди».

У зв’язку з тим, що інформаційна технологія планується використовуватися і поза межами України, то існує 2 кнопки, що відповідають за зміну мови додатку. Загальний вигляд першого функціонального блоку зображено на рисунку 4.3.

Рисунок 4.3 – Перший функціональний блок

Далі за умовою роботи додатку, треба визначити місце знаходження користувача для подальшого визначення погодних умов. За виконання цих функцій відповідає блок «Пошук міста», який складається з рядка введення та кнопки. Загальний вигляд цього блоку представлено на рисунку 4.4.

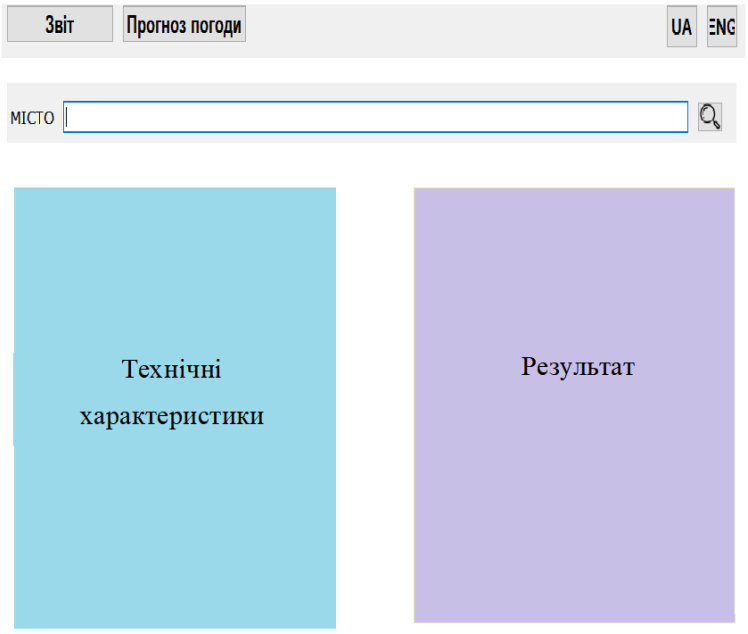

Рисунок 4.4 – Другий функціональний блок

Далі переходимо до наступного блоку – «Технічні характеристики». Дані цього блоку відіграють важливу роль в обрахуванні потужності складових гібридної мережі та прогнозної моделі її роботи. На рисунку 4.5 зображено прототип з блоком «Технічні характеристики» з усіма потрібними даними, які передаються з натисканням відповідної кнопки.

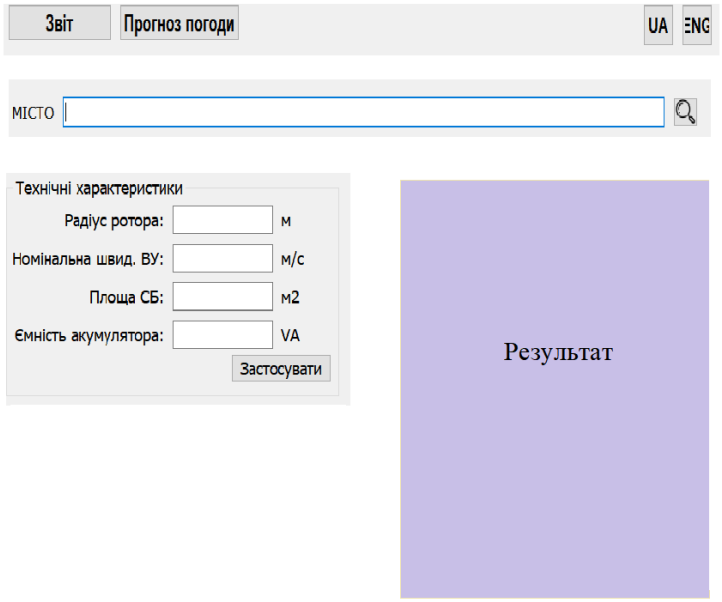

Рисунок 4.5 – Вигляд блоку «Технічні характеристики»

Останній розроблений блок «Результат», який повинен відображати поточний стан гібридної мережі з джерелом живлення та отриманими даними обрахунку потужностей ВЕУ та СБ.

Загальний вигляд прототипу з блоком «Результат» представлено на рисунку 4.6.

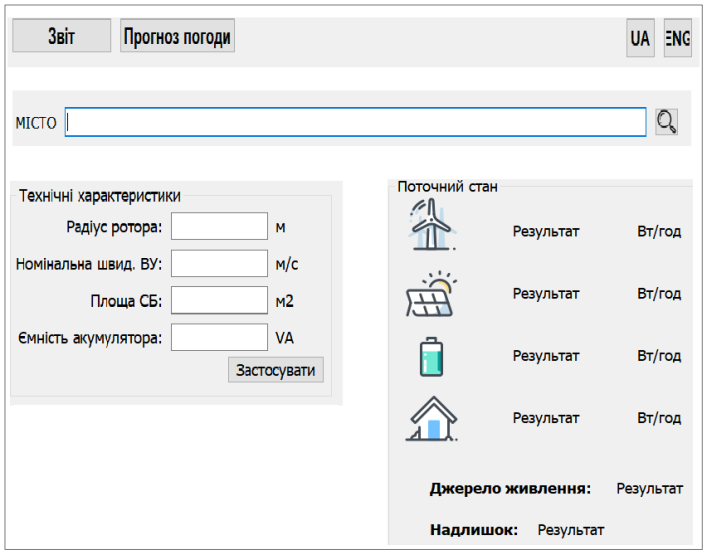

Рисунок 4.6 – Загальний прототип додатку

4.1.2 Розробка бази даних

Для коректної роботи інформаційної технології перш за все створюється база даних «hybrid», що зберігатиме в собі данні місця знаходження гібридної мережі, погодні дані та дані про потужності, що виробляються ВУЕ та СБ, та результат роботи інформаційної технології – визначене джерело живлення. Структуру бази даних представлено на рисунку 4.7.

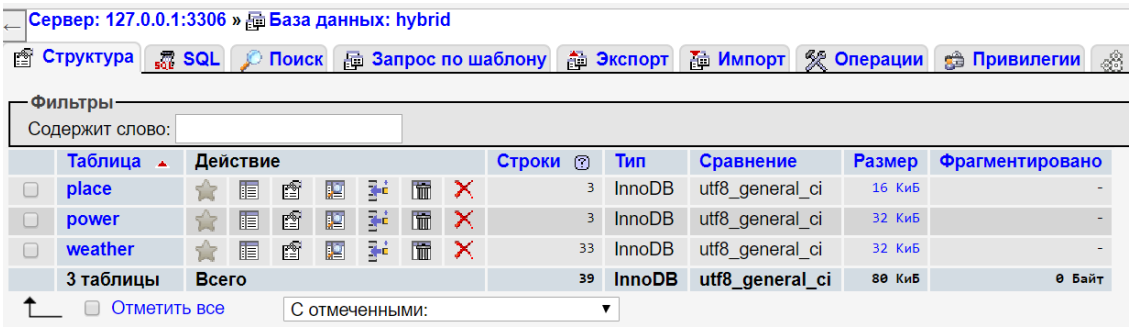

Рисунок 4.5 – Структура база даних «hybrid»

Таблиця «place» (рис. 4.8) зберігає назву міста, області та країни, а також ідентифікаційний номер даного міста на сайті Gismeteo.com, ці дані потрібни саме для парсінгу погодних даних.

+ Параметры

place_id	city_name	country	number	region
1	Гадяч	Україна	12190	Полтавська область
2	Суми	Україна	4936	Сумська область
3	Вишенки	Украина	94297	Сумская область

Отметить все С отмеченными:

Рисунок 4.8 – Структура таблиці «place»

Погодні дані з сайту зберігаються в таблиці «weather» (рис. 4.9). Для обрахування потужностей потрібен рівень хмарності, показники вітру та температура повітря. Далі вже отримані результати роботи інформаційної технології записуються до таблиці «power» (рис 4.10).

Параметры

weather_id	w_speed	w_direction	temperature	overcast	dat_e	tim_e	place_id
3	0	штиль	17	Ясно	2019-06-03	00:00:00	1
4	3	СВ	19	Ясно	2019-06-03	21:00:00	2
5	3	СВ	19	Ясно	2019-06-03	21:00:00	2

Рисунок 4.9 – Структура таблиці «weather»

Параметры

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

Отметить все С отмеченными:

Рисунок 4.10 – Структура таблиці «power»

Для роботи з базою даних програмно було створено додатковий модуль myconnutils.py. Фрагмент коду get_connection(), що відповідає за підключення до бази даних представлено на рис. 4.11.

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

Рисунок 4.11 – Функція, що повертає з’єднання з базою даних

Модуль також має функції підключення до бази даних, функції запитів додавання інформації до бази (insert_city(), insert_power() та insert_weather()), приклад якого зображено на рисунку 4.12, та функції виконання запитів зі змінними execute_query() та без execute_select() (рис. 4.13).

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

Рисунок 4.12 – Функція виконання запиту додавання даних місця знаходження

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

Рисунок 4.13 – Функція виконання запиту зі змінними

4.1.3 Створення парсера

Однією з найважливіших етапів роботи додатку – парсинг відповідних погодних даних з сайту Gismeteo.com. За логікою роботи дані збираються відповідно до введеного міста. Для обрахунку потужності ВЕУ потрібно знати швидкість та напрямок вітру, а для потужності СБ – температуру повітря та рівень хмарності. Тож було вирішено створити додатковий модуль gparser.py, куди було винесено всі потрібні функції.

Функція extract_datetimes(row), яка парсить дату та час, за який було отримано погодні дані,. Ці дані будуть записані до бази даних для правильного подальшого їх використання. Програмний код даної функції зображено на рисунку 4.14.

```
# extract full date
# return list contains 8 el (every 3h of day)  [['YYYY-mm-dd', 'hh:mm:ss(UTC)'],
def extract_datetimes(row):
    dates = []
    for el in row.xpath('./div/div/@title'):
        dates.append(split_todatetime(el))
    return dates
```

Рисунок 4.14 – Функція extract_datetimes

Далі функція, яка визначає температуру повітря за кожні 3 години доби – є extract_temperature(row) (рис. 4.15). Так як температура на сайті зберігається у цельсіях та фаренгейтах, то далі для визначення потрібної використовується функція clean_data(data).

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

Рисунок 4.15 – Функція extract_temperature

Для роботи ВЕУ потрібно визначити характеристики вітру(швидкість та напрямок). Ці дані збирає функція extract_wind(row), програмний код якої зображено на рисунку 4.16.

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

Рисунок 4.16 – Функція extract_wind

Головна функція, що підключається до сайту та збирає дані, – req_city_info(city, lang), до якої передається введене в формі місто та мова, якою представлений сайт, зображена на рисунку 4.17. Далі також існує допоміжна функція to_list_of_dict(keys, val_list), яка повертає отримані дані у вигляді списку словників, що в подальшому облегшує роботу.

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

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

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

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

    return res
```

Рисунок 4.17 – Функція req_city_info

4.2 Створення головного модулю додатку та його реалізація

Для розробки програмного інтерфейсу мовою програмування Python використовуються зазвичай пакети Tkinter або PyQt. Останній є більш зручним саме для розробки віконних додатків, тому у якості одного з інструментів було обрано саме його. За допомогою програми QT Designer було створено макет діалогового вікна з відповідним функціоналом, що представлено на рисунку 4.18, який містить в собі, як поля введення даних, функціональні кнопки, так і зображення, для підгруження яких використовувалась бібліотека QPixmap, та календар, як додатковий елемент вікна.

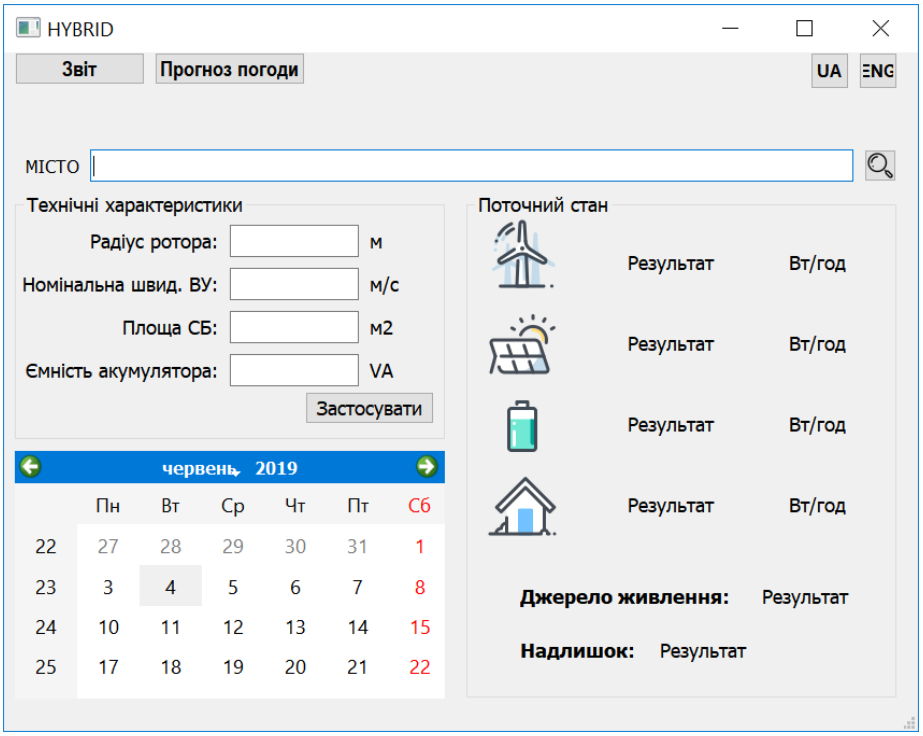

Рисунок 4.18 – Головне вікно додатку

Отриманий макет діалогового вікна потрібно за допомогою командного рядка записати у вигляді коду формату «.py», для чтого використовується відповідний запит(рис. 4.19).

```
python -m PyQt5.uic.pyuic -x [FILENAME].ui -o [FILENAME].py
```

Рисунок 4.19 – Команда для створення програмного файлу макета вікна

Після наступних операцій було створено файл mydesign.py, де відображені всі елементи вікна та їх характеристики. До основного модулю програми myprogram.py вона імпортується, як і всі стандартні бібліотеки за допомогою команди import.

Як було вказано раніше, то додаток відображає дані, як українською, так і англійською мовами, для іноземного населення. Для функціональної кнопки «ENG» було функцію, яка викликається натисканням кнопки. Програмний код функції зображено на рисунку 4.20, а його реалізація – рисунок 4.21.

```
def ENG_text(self):
    self.ui.groupBox1.setTitle("Specifications")
    self.ui.lbl_veu_r.setText("Rotor radius:")
    self.ui.lbl_veu_n.setText("Rated speed of WG:")
    self.ui.lbl_sb_s.setText("Square SP:")
    self.ui.lbl_bt_e.setText("Battery Capacity:")
    self.ui.lbl_m.setText("m")
    self.ui.lbl_n.setText("m/s")
    self.ui.lbl_m2.setText("m2")
    self.ui.apply_btn.setText("Apply")
    self.ui.lbl_main_city.setText("Sumy, Sumska oblast, Ukraine")
    self.ui.lbl_source.setText("Power supply:")
    self.ui.lbl_overflow.setText("Overflow:")
    self.ui.lbl_city.setText("City")
    self.ui.lbl_source_value.setText("Result")
    self.ui.lbl_overflow_value.setText("Result")
    self.ui.settingBtn.setText("Save")
    self.ui.reportBtn.setText("Report")
    self.ui.weatherBtn.setText("Weather forecast")
    self.ui.label_veu.setText("Result")
    self.ui.label_sb.setText("Result")
    self.ui.label_bt.setText("Result")
    self.ui.label_cons.setText("Result")
    self.ui.groupBox.setTitle("Current condition:")
    self.ui.label.setText("W/h")
    self.ui.label_3.setText("W/h")
    self.ui.label_2.setText("W/h")
    self.ui.label_4.setText("W/h")
```

Рисунок 4.20 – Програмний код функції ENG_text

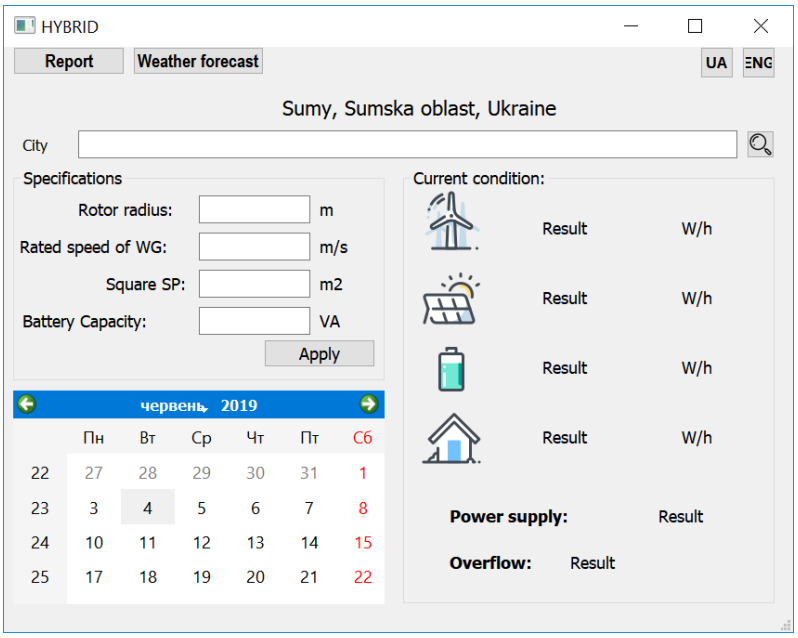

Рисунок 4.21 – Результат натискання кнопки «ENG»

Аналогічно діє кнопка «UA», яка перекладає зміст вікна в зворотньому напрямку – українською мовою, за що відповідає функція UA_text(self), що також активується натисканням кнопки. Програмний код функції зображено на рисунку 4.22, а його реалізація – 4.23.

```
    self.ui.groupBox1.setTitle("Технічні характеристики")
    self.ui.lbl_veu_r.setText("Радіус ротора:")
    self.ui.lbl_veu_n.setText("Номінальна швид. ВУ:")
    self.ui.lbl_sb_s.setText("Площа СБ:")
    self.ui.lbl_bt_e.setText("Ємність акумулятора:")
    self.ui.lbl_m.setText("м")
    self.ui.lbl_n.setText("м/с")
    self.ui.lbl_m2.setText("м2")
    self.ui.apply_btn.setText("Застосувати")
    self.ui.lbl_main_city.setText("Суми, Сумська область, Україна")
    self.ui.lbl_source.setText("Джерело живлення:")
    self.ui.lbl_overflow.setText("Надлишок:")
    self.ui.lbl_city.setText("МІСТО")
    self.ui.lbl_source_value.setText("Результат")
    self.ui.lbl_overflow_value.setText("Результат")
    self.ui.reportBtn.setText("Звіт")
    self.ui.weatherBtn.setText("Прогноз погоди")
    self.ui.label_veu.setText("Результат")
    self.ui.label_sb.setText("Результат")
    self.ui.label_bt.setText("Результат")
    self.ui.label_cons.setText("Результат")
    self.ui.groupBox.setTitle("Поточний стан")
    self.ui.label.setText("Вт/год")
    self.ui.label_3.setText("Вт/год")
    self.ui.label_2.setText("Вт/год")
    self.ui.label_4.setText("Вт/год")
```

Рисунок 4.22 – Програмний код функції UA_text

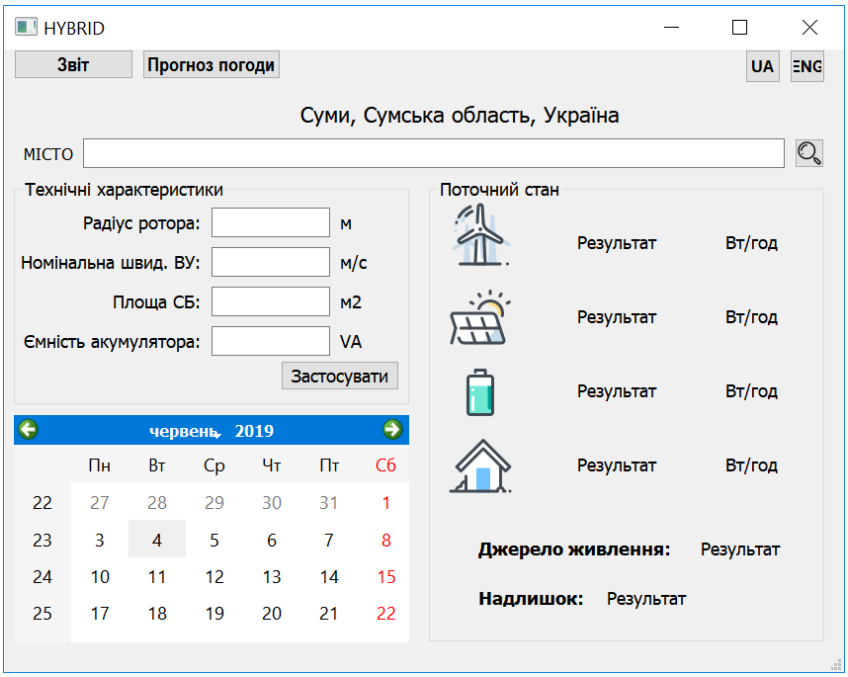

Рисунок 4.23 – Реалізація функції UA_text

Перш за все для подальшого отримання результату користувач повинен внести у поле введення назву бажаного міста та натиснути кнопку пошуку. Результат нашого запиту зображено на рисунку 4.24. Після виконання запиту користувач бачить обране місто, що відображається зверху вікна, а також в блоці «Поточний стан».

При натисканні кнопки пошуку активується функція searchCity(self), програмний код якої представлено на рисунку 4.25. Спочатку за допомогою функції gparser.req_city_info() шукається введене місто на сайті Gismeteo.com, далі за допомогою функції myconnutils.insert_city() записуємо дані міста до бази даних в тому випадку, якщо такі ще не існують. Далі за даними отриманого міста за допомогою функції myconnutils.insert_weather() записуємо отримані погодні дані до бази даних за останню годину (рис. 4.26).

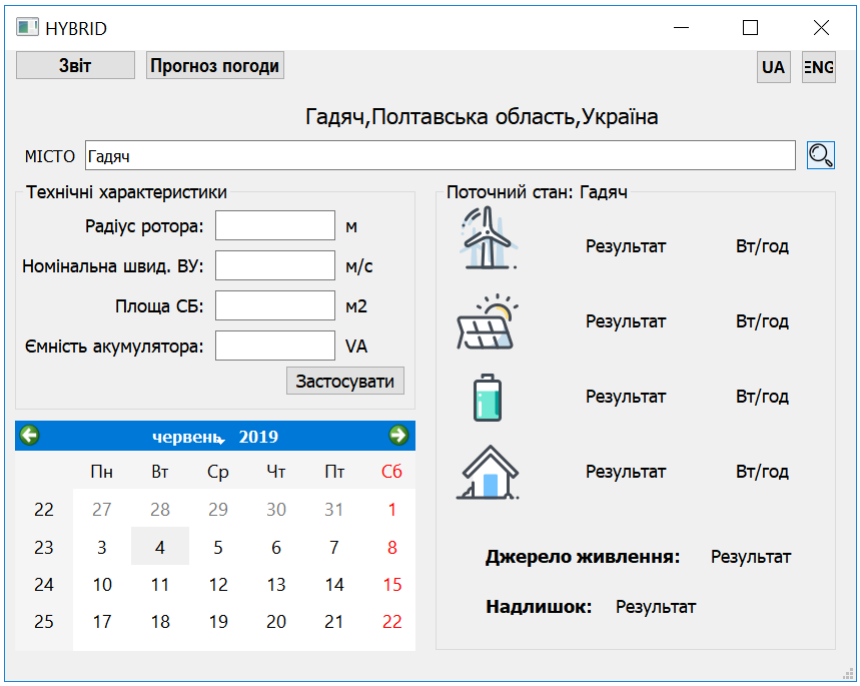

Рисунок 4.24 – Результат натискання кнопки пошуку міста

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

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

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

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

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

Рисунок 4.25 – Програмний код функції searchCity

weather_id	w_speed	w_direction	temperature	overcast	dat_e	tim_e	place_id
36	4	B	22	Малооблачно	2019-06-04	21:00:00	1

Рисунок 4.26 – Результат додавання погодних даних до БД

Далі потрібно внести технічні характеристики вітрогенератора та сонячних панелей. Для прикладу візьмемо вітрогенератор з довжиною лопастей 1,55 та номінальною швидкістю обертання – 14 м/с. Ці показники можна подивитися в технічному паспорті виробу. Далі, на приклад, для невеликого будинку було вирішено обрати сонячні панелі площинною 18 м2 та акумуляторну батарею ємністю 5000 Вт. Після введення всіх необхідних даних користувач натискає кнопку «Застосувати»(рис.4.27), яка викликає функцію applyData(self).

Функція встановлює з’єднання з базою даних, зчитує введенні дані, шукає в базі даних погодні дані за найближчий час, зчитує дані з excel-файлу для обрахування потужності, виконує потрібні обчислення та записує їх до бази даних. Програмний код даної функції зображено на рисунку 4.28.

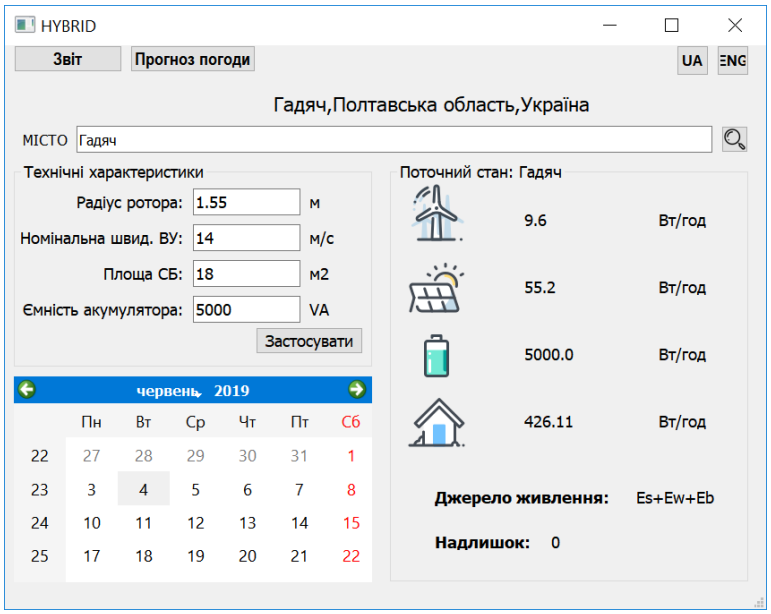

Рисунок 4.27 – Результат роботи кнопки «Застосувати»

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

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

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

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

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

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

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

Рисунок 4.28 – Програмний код функції applyData

Для виведення отримання результату обрахунку потужностей використовуються функції E_veu(), що визначає потужність ВЕУ(рис.4.28), E_sb(), що обраховує потужність СБ(рис.4.30), W_consumption() для визначення рівня споживання електроенергії(рис.4.31) та priority(), що визначає джерело електроенергії та кількість енергії, що передано до загальної мережі(рис.4.32).

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

Рисунок 4.29 – Функція E_veu()

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

Рисунок 4.30 – Функція E_sb()

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

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

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

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

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

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

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

Рисунок 4.31 – Функція W_consumption()

```
def priority(Es, Ew, Eb, W, Eb_max):
    if Es > W:
        Source = 'Es'
        overflow = Es + Ew - W
        if Eb < Eb_max:
            Eb = Eb + overflow
            if Eb >= Eb_max:
                E_send = Eb-Eb_max
    elif Ew > W:
        Source = 'Ew'
        overflow = Ew - W + Es
        if Eb < Eb_max:
            Eb = Eb + overflow
            if Eb >= Eb_max:
                E_send = Eb-Eb_max
    elif Es + Ew > W:
        Source = 'Es+Ew'
        overflow = Ew - W + Es
        if Eb < Eb_max:
            Eb = Eb + overflow
            if Eb >= Eb_max:
                E_send = Eb-Eb_max
    elif Es + Ew + Eb > W:
        Source = 'Es+Ew+Eb'
        Eb = Eb - (W - Es - Ew)
        E_send = 0
    else: Source = 'Power Grid'

    return Source, Eb, E_send
```

Рисунок 4.32 – Функція priority()

У разі, якщо користувачу потрібно сформувати звіт з отриманих даних, то рекомендовано натиснути кнопку «Звіт». Файл збережеться в папці з проектом під назвою «Report.docx».Приклад роботи програми представлено на рисунку 4.33 з відповідним повідомленням про результат запису та сам файл звіту – рисунок 4.34. програмний код виконання даної команди представлено на рисунку 4.35.

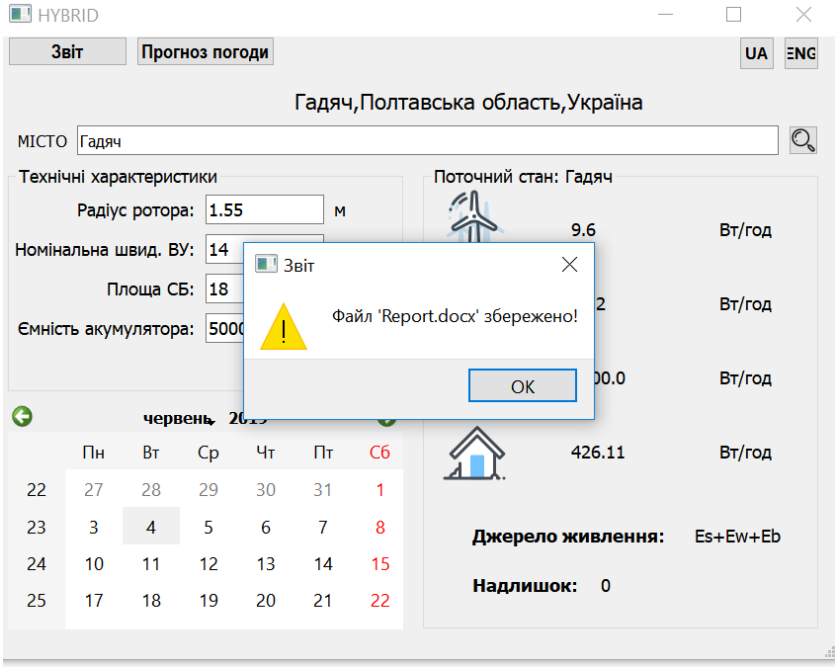

Рисунок 4.33- Результат збереження звіту

РЕЗУЛЬТАТИ·ЕЛЕКТРОСПОЖИВАННЯ¶

¶

Потужність·ВЕУ¤	55.2¤
Потужність·СБ¤	9.6¤
Ємність·акумулятора¤	5000.0¤
Рівень·споживання·електроенергії¤	426.11¤
Джерело·енергії¤	Es+Ew+Eb¤
Передано·до·загальної·мережі¤	0¤

¶

Дата:2019-06-04¶

Рисунок 4.34 – Сформований звіт

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

Рисунок 4.35 – Програмний код, що запускається при натисканні кнопки «Звіт»

Однією з задач проекту є доступ до метеорологічного сайту. При натисканні кнопки «Прогноз погоди» в браузері відкривається сайт Gismeteo.com, який показує одразу погоду місця знаходження користувача(рис.4.36).

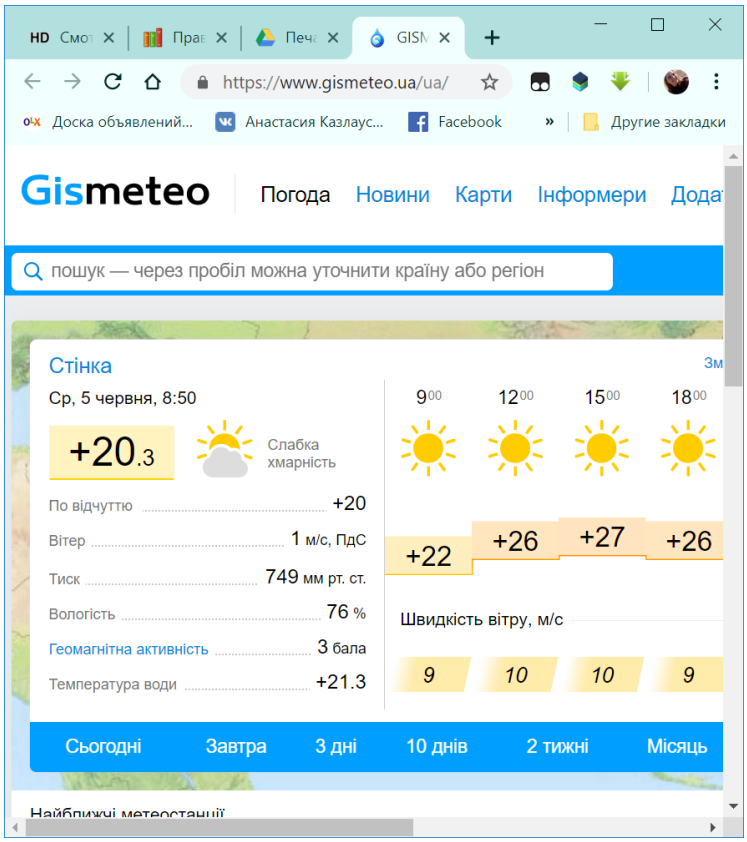

Рисунок 4.36 – Сайт прогнозу погоди

У зв’язку з тим, що погодні дані на сайті можуть змінюватись протягом доби, тому було вирішено парсить дані відповідно до поточного часу, тож програма буде опрацьовувати вся функції кожні 3 години, так як на сайті представлено прогноз погоди для кожного третього часу. Для запуску програми кожні 3 години було створено додатковий клас UpdateResults(threading.Thread), яка містить в собі функцію run(self) (Додаток С)

Після проведення попередніх маніпуляцій була розроблена інформаційна технологія з відповідним функціоналом, що зазначено в Додатку А. Файли коду

myprogram.py, mydesign.py, gparcer.py та myconnutils.py представлено у Додатку С. Інструкцію користувача для коректної роботи представлено у Додатку Д.

ВИСНОВКИ

У дипломному проекту було розроблено інформаційну технологію визначення впливу погодних умов на продуктивність альтернативних джерел енергії з використанням нечіткої логіки. Інформаційна технологія відповідає всім послевленим замовником вимогам:

- вибір місця для збору необхідних даних;
- збереження та пошук даних;
- доступ до метеорологічного сайту;
- визначення впливу погодних умов на ВДЕ;
- автоматичне забезпечення пріоритетності джерела енергії з найменшими затратами.

Виконавши аналіз предметної області, а саме дослідження актуальності у необхідності впровадження та використання гібридних електромереж, а також аналіз існуючих аналогів, було прийнято рішення про створення інформаційної технології, що базується саме на моделі нечіткого вибору режиму гібридної енергомережі, до складу якої входять як сонячні батареї, так і вітрогенератори, оскільки більшість аналогів не використовує одночасно можливості енергії сонця та вітру.

Провівши аналіз існуючих засобів реалізації програмного продукту було обрано мову програмування загального призначення Python з додатковими модулями PyQt5 та GUI, а також СУБД MySQL. Було обрано саме ці засоби, спираючись на відповідні їх характеристики, а саме легкість у розумінні та використанні, наявність великого спектру можливостей, та їх популярність.

Як результат цього етапу було розроблено необхідні діаграми , завдяки яким є можливість виконувати контроль виконання строків робіт, що допомагає знизити ризик невчасно розробленого проекту.

Одним з основних етапів розробки інформаційної технології було розроблення математичної моделі з використанням нечіткої логіки. Визначено залежності потужностей ВЕУ та СБ від зовнішніх та конструктивних факторів, розроблено залежність потужності СБ від рівня хмарності на основі даних попередніх спостережень та створено перелік нечітких правил для прогнозної моделі роботи ГЕСВДЕ.

Після попереднього створення прототипу робочого діалогового вікна було створено інтерфейс за допомогою QT Designer. За допомогою функцій, методів та надбудов мови програмування Python були створені допоміжні модулі та основний функціонал додатку, які відповідають на коректну реакцію користувача. При введенні неприпустимих даних користувач отримає відповідне повідомлення.

Після того, як інформаційна технологія була остаточно розроблена, було проведено її тестування. За результатом тесту було визначено, що з технологічних умов та даних сайту Gismeteo.com бувають випадки, коли назва міста введеного користувачем не співпадає з даними сайту, буде висвітлено назву найближчого міста, для якого збираються метеорологічні показники, наприклад, Глухів – Вишенки. Попри все цей факт не впливає на роботу інформаційної технології.

Отже, після виконання всіх етапів проекту було розроблено інформаційту технологію, що матиме попит серед домашніх господарств та представників малого бізнесу, бо не тільки допоможе обрати краще джерело електроенергії в конкретних умовах, але й дозволить економити кошти за рахунок продажу надлишку електроенергії до загальної мережі за «Зеленим тарифом».

Результати дослідження доповідались на конференції ІМА 2019 (Додаток Е). А також даний проект може бути впроваджено в учбовий процес секції «Інформаційні технології проектування» кафедри комп’ютерні науки.

СПИСОК ЛІТЕРАТУРИ

1. Konechenkov, A. Renewable Energy. Focusing: Ukraine Vision 2050 [Електронний ресурс] / A. Konechenkov. – Available at: URL: https://bitly.su/N8R3tCK6 – 19.01.2005.

2. GE Power ИБП [Електронний ресурс] – режим доступу: https://bitly.su/aXqNEv

3. MAN: Hybrid power plants [Електронний ресурс] – режим доступу: https://bitly.su/iDBI

4. Multin, M. Integration of electric vehicles in smart homes – an ICT-based solution for V2G scenarios [Text] / M. Multin, F. Allerding, H. Schmeck // 2012 IEEE PES Innovative Smart Grid Technologies (ISGT). – IEEE, 2012. – P. 1–8. doi:10.1109/isgt.2012.6175624

5. MySQL [Електронний ресурс] – режим доступу: https://bitly.su/xbHLDWb

6. Python (programming language) [Електронний ресурс] – режим доступу: https://bitly.su/paHJe

7. Ringel, M. Fostering the use of renewable energies in the European Union: the race between feedin tariffs and green certificates [Text] / M. Ringel // Renewable Energy. — 2006. — Vol. 31, № 1. — P. 1–17. doi:10.1016/j.renene.2005.03.015

8. Riverbank | Software | PyQt | What is PyQt? [Електронний ресурс] – режим доступу: https://bitly.su/7QdxtQlJ

9. Shulyma, O. The Features of the Smart MicroGrid as the Object of Information Modeling [Text] / O. Shulyma, V. Shendryk, I. Baranova, A. Marchenko // Communications in Computer and Information Science. — 2014. — Vol. 465. — P. 12–23. doi:10.1007/9783319119588_2

10. Альтернативні джерела енергії [Електронний ресурс] – режим доступу: https://bitly.su/MFMQX

11. Гибридные инверторы с функцией ИБП [Електронний ресурс] – режим доступу: https://bitly.su/gdhi

12. Грекул В. І. Методические основы управления ИТ-проектами / В. І. Грекул, Н. Л. Коровкина, Ю. В. Куприянов. – Москва: Інтернет-Університет Інформаційних Технологій: БІНОМ. Лабораторія знань., 2010. – 391 с.

13. Діаграми „сутність-зв”язок”: призначення, місце застосування, правила побудови, ERD-стандарти. Сутності, відношення та зв’язки в нотації Чена [Електронний ресурс] – режим доступу: https://bitly.su/iUZ40

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

15. Кузьмин Е. В. Управление проектами с использованием Microsoft Project 2013: лабораторный практикум / Е. В. Кузьмин. - Самара: ПГУТИ, 2016. –151 с.

16. Марк Лутц Изучаем Python, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 1280 с.

17. Методологія функціонального моделювання SADT [Електронний ресурс] – режим доступу: https://bitly.su/kQFQ542

18. На какие расстояния эффективно передавать электроэнергию? [Електронний ресурс] – режим доступу: https://bitly.su/iJk7sk

19. Проблемы передачи электроэнергии на дальние расстояния энергии [Електронний ресурс] – режим доступу: https://bitly.su/rerniFAS

20. Просницкий А. В. Самоучитель «Microsoft Project 2013. Управление проектами» [Электронный ресурс] / А. В. Просницкий. - Электрон. текстовые дан. – Киев: 2013. – режим доступу: https://bitly.su/gfN6

21. Розвиток енергетичної кризи та її вплив на енергетичну безпеку країни [Електронний ресурс] – режим доступу: https://bitly.su/32qB

22. Розподілена генерація як альтернатива ресурсозатратної енергетики [Електронний ресурс] – режим доступу: https://bitly.su/mV9ijsSQ

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

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

25. Солнечная энергетика в Украине: кто и где строит новые станции [Електронний ресурс] – режим доступу: https://bitly.su/rFgV

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

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

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

29. Электрик Инфо: Альтернативные источники энергии [Електронний ресурс] – режим доступу: https://bitly.su/Kdzi

30. Энергетический кризис [Електронний ресурс] – режим доступу: https://bitly.su/g9BJJds0

Додаток А

Технічне завдання

Назва програмного продукту «HYBRID».
Область застосування програмного продукту – програмний продукт призначений для оптимізації використання виробленої та споживаної електроенергії.
Об'єкт, у якому використовують програму – секція Інформаційних технологій проектування.

1 Основи для розробки

Розробка виконується на основі завдання, виданого викладачем секції Інформаційних технологій проектування.

2 Призначення розробки

Розробка повинна надати можливість оброблювати, переглядати та зберігати в базах даних інформацію, отриману з метеорологічних сайтів, підтримувати оптимальний рівень споживання електроенергії та бути універсальною для будь-якої місцевості.

Назва організації: Казлаускайте Анастасія, ІТ-51, кафедра Комп'ютерних Наук.
Тема проекту: «Інформаційна технологія визначення впливу погодних умов на продуктивність альтернативних джерел енергії»

3 Вимоги до програмного продукту

Програмний продукт повинен бути реалізований у вигляді віконного додатку з підключенням до баз даних, що забезпечує виконання функціональних можливостей з пункту 3.3.

3.1 Вимоги до програмного продукту

Програмний продукт розробляється мовою Python 3.7 з використанням технології PyQt5 та підключенням до баз даних, створення таблиць баз даних та запитів до них реалізується за допомогою мови запитів SQL. Математична модель створюється на основі нечіткої логіки для використання даних у вигляді якісних значень.

3.2 Вимоги до програмного забезпечення

Для коректної роботи програмного продукту необхідно встановити веб-сервер (наприклад OpenServer) для використання баз даних та швидкого виконання запитів. Також для перегляду даних на метеорологічних сайтах є обов'язковим наявність веб-браузера: Google Chrome – версії 9.0 або новіша, Intertnet Explorer – від версії 11.0 або будь-який інший веб-браузер.

3.3 Вимоги до функціональних характеристик

Програмний продукт повинен забезпечувати виконання наступних функцій:

- вибір місця для збору необхідних даних;
- збереження та пошук даних;
- доступ до метеорологічного сайту;
- визначення впливу погодних умов на ВДЕ;
- визначення кращого джерела електроенергії або їх поєднання в даних погодних та експлуатаційних умовах;

4 Перелік програмної документації

- Опис проекту продукту із використанням UML-діаграм;
- Макет інтерфейсу програмного продукту;
- Програмний код розробки програмного продукту;
- Технічне завдання;
- Інструкція користувача;

5 Порядок виконання робіт і етапи розробки

Стадії та етапи розробки повинні складатися з наступних пунктів:

- Оформлення завдання для дипломної роботи;
- Планування роботи. Розроблення ТЗ, побудова мережевого графіку та діаграми Ганта;
- Розроблення математичної моделі програмного продукту;
- Розроблення структури програмного продукту;
- Розроблення функціоналу програмного продукту, базових маніпуляцій та інтерфейсу;
- Провести тестування програмного продукту;
- Розроблення інструкції користувача;
- Оформлення пояснювальної записки про виконання дипломної роботи;
- Здача пояснювальної записки до дипломної роботи та розробленого програмного продукту;
- Презентація роботи та її захист.

6 Порядок контролю та приймання

Контроль коректності функціонування та придатності програмного продукту здійснюється замовником (секцією Інформаційні технології проектування) на основі наданої пояснювальної записки до дипломної роботи та програмних файлів. Контроль ходу виконання проекту здійснюється на основі календарного плану виконання дипломної роботи:

- Перевірка завдання дипломної роботи.
- Перевірка ТЗ, мережевого графіка та діаграми Ганта.

- Перевірка математичної моделі програмного продукту.
- Перевірка структури програмного продукту
- Перевірка наявності функціоналу, базових маніпуляцій та інтерфейсу.
- Перевірка інструкції користувача.
- Здача ПЗ.
- Презентація.

Додаток Б

Планування робіт

Метою проекту є розробка моделі та супутнього програмного забезпечення для вибору оптимального режиму роботи гібридної електромережі, що забезпечує регулювання параметрів вироблюваної і вживаної електроенергії, автоматично визначає пріоритетність джерела енергії, з використанням мови програмування Python, де користувач може самостійно внести власні налаштування. Покомпонентне налаштування розподіленої електросистеми використовує дані, запозичені з відповідних сайтів прогнозу погоди. Результати деталізації мети методом SMART розміщенні у таблиці Б.1.

Таблиця Б.1 – Деталізація мети методом SMART

Specific (конкретна)	Розробити програмний продукт для оптимального режиму роботи електромережі з розподіленою концепцією генерації електроенергії.
Measurable (вимірювання)	Результатом роботи проекту є оцінка самого замовника, оскільки проект – не комерційний.
Achievable (досяжна, узгоджена)	Мета даного проекту вважається досяжною, оскільки розробник володіє необхідними навичками у створенні програмних продуктів засобами Python, MySQL та ознайомлений з необхідною кількістю методів, функцій та бібліотек даних мов. Мета була узгоджена з вимогами та потребами замовника.

Продовження таблиці Б.1

Relevant (реалістична)	Для реалізації продукту проекту є всі необхідні технічні та програмні засоби (веб-сервер Open Server, універсальний інтерпретатор Notepad++, open source дистрибутив для мови програмування Python Anaconda) та доступ до мережі Інтернет. Розробник є досить кваліфікованим для виконання поставлених задач.
Time-framed (обмежена в часі)	Програмний продукт розробляється з обмеженням у часі, ґрунтуючись на сформованому календарному плані та матриці відповідальності.

Планування змісту структури робіт. WBS (Work Breakdown Structure) є одним з головних етапів плану проекту. Він є необхідним для визначення вартості проекту і його календарного плану. WBS – це орієнтоване на результати групування компонентів проекту, що визначає які роботи повинні бути виконані в проекті. Зазвичай представляється у вигляді ієрархії робіт, де на першому рівні зафіксовано сам продукт проекту, далі слід розбити проект на кілька підпроектів доки не буде досягнутий необхідний рівень деталізації. На рисунку Б.1. представлена повна та деталізована WBS-структура проекту.

Планування структури організації, для впровадження готового проекту (OBS). Після побудови WBS розробляють організаційну структуру виконавців. OBS-структура проекту (Organizational breakdown structure) – організаційна структура виконавців проекту. Визначається за переліком пакетів робіт нижнього рівня кожної гілки WBS-структури. Організаційна структура являє собою

графічне відображення учасників проекту та їх відповідальних осіб, задіяних в реалізації проекту. На верхньому рівні OBS розташована команда проекту. На наступному рівні фіксуються виконавці: організації, відділи тощо. Список виконавців, що функціонують в проекті представлений в таблиці Б.2. OBS-структура виконуваного проекту представлена на рисунку Б.2

Таблиця Б.2 – Виконавці проекту

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

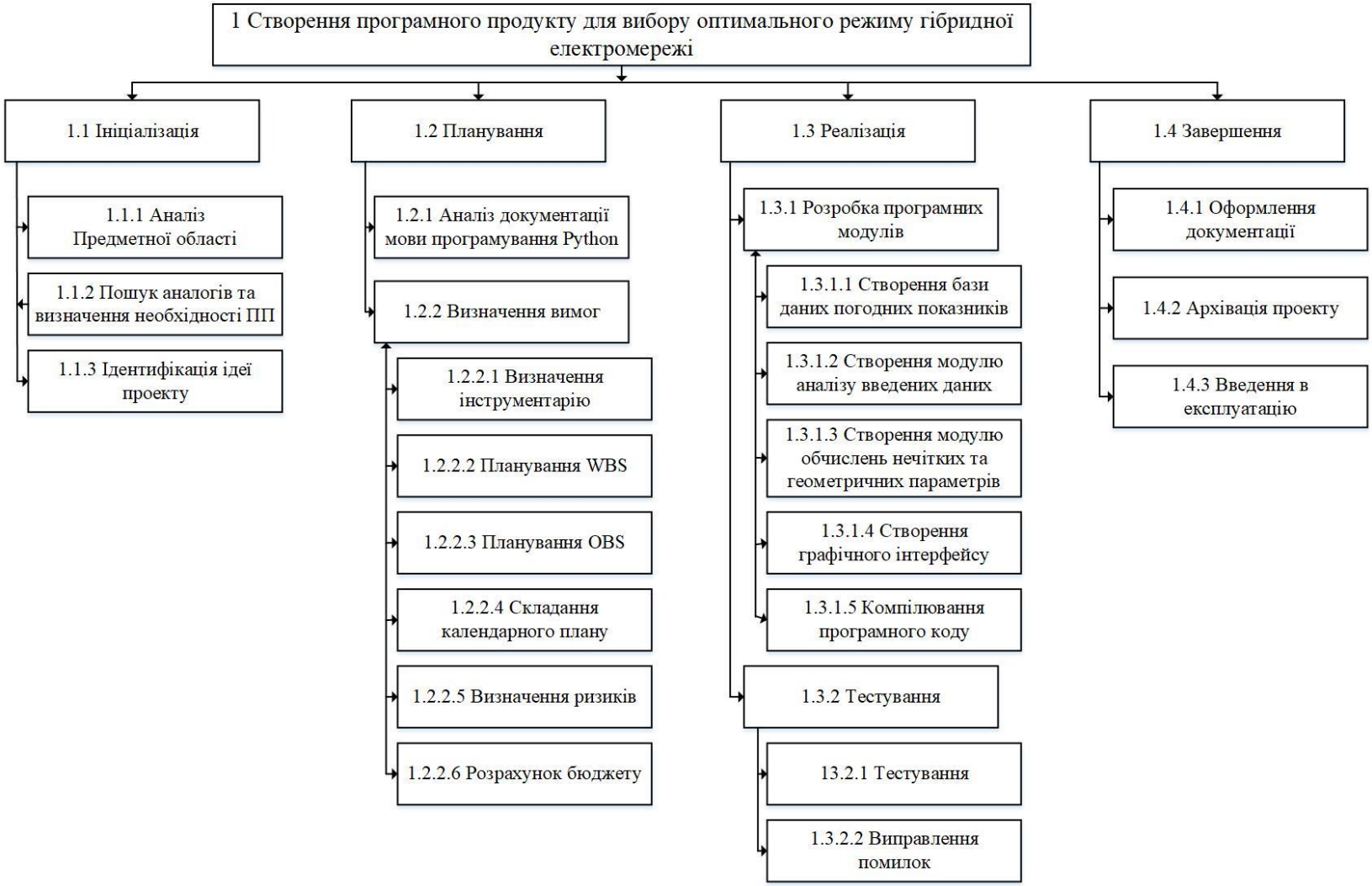

Рисунок Б.1 − WBS. Структура робіт проекту

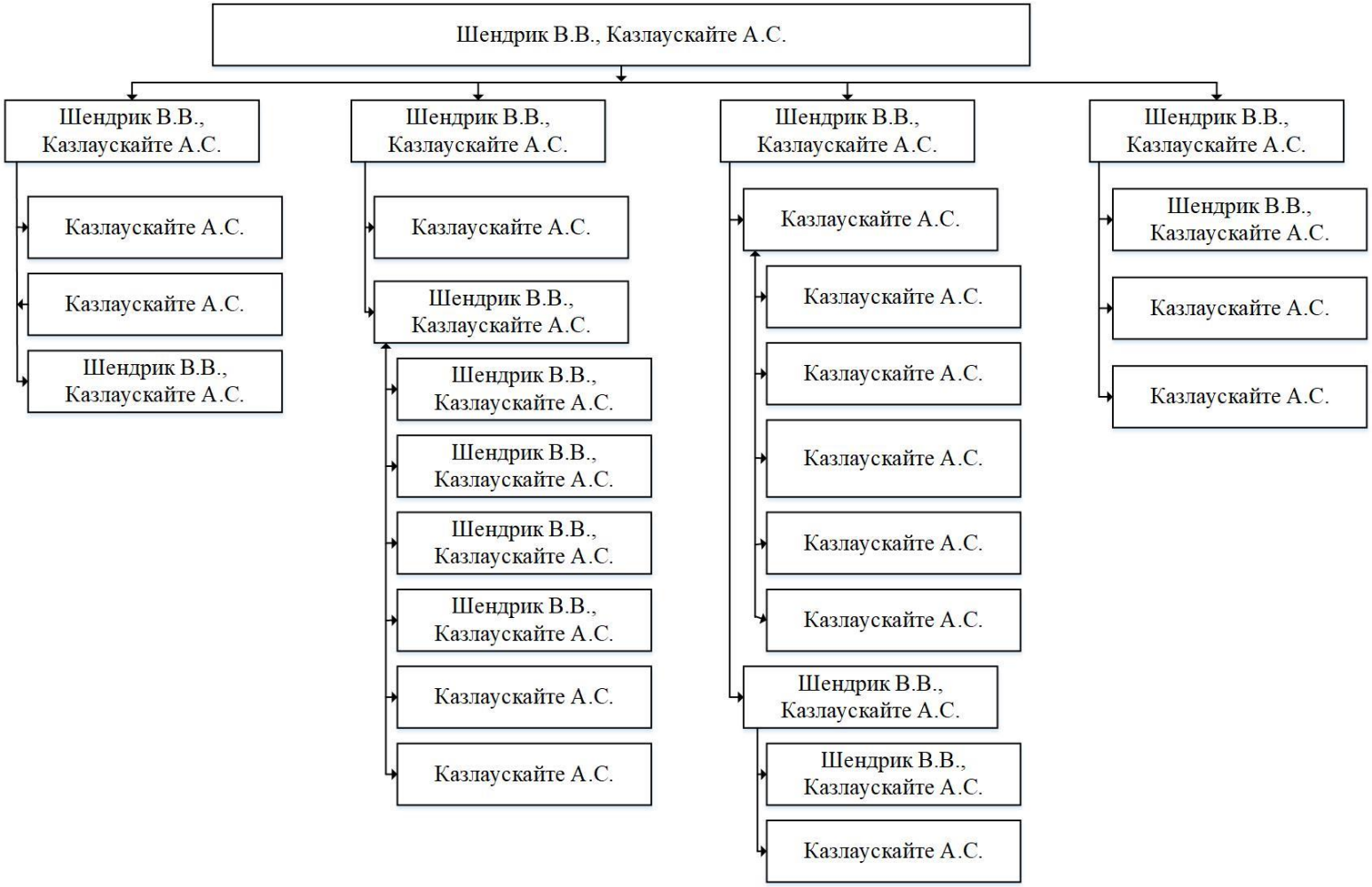

Рисунок Б.2 − OBS. Організаційна структура проекту

Матриця відповідальності. На підставі OBS і WBS структур побудуємо матрицю відповідальності проекту. Матриця відповідальності закріплює за кожною елементарною роботою виконавця. Тобто матриця відповідальності надає інформацію про особу, яка є відповідальною за виконання кожної тої чи іншої елементарної роботи. Потрібно пам’ятати, що за одну елементарну роботу може бути тільки один відповідальний. Часто в лінійному графіку відповідальності, крім виконавців, додають інших учасників проекту. При цьому по кожній елементарної роботі їм доручається або функція контролю, або консультування.

Потім для кожної елементарної роботи з урахуванням технічних і технологічних критеріїв до продукту проекту і його складових частин розробляються робочі завдання. Їх призначення – забезпечення максимальної ймовірності того, що виконавці представлятимуть саме той продукт (або послугу), який був закладений при складанні WBS. Тобто, робоче завдання використовується як основа взаєморозуміння між відповідальним і командою проекту.

На основі розроблених WBS та OBS структур проекту була побудована матриця відповідальності проекту, яка представлена в таблиці Б.3.

Таблиця Б.3 – Матриця відповідальності

WBS\OBS	Казлаускайте	Шендрик
Аналіз предметної області		
Пошук аналогів та визначення необхідності ПП		
Ідентифікація ідеї проекту		
Аналіз документації мови програмування Python		
Визначення інструментарію		
Планування WBS		

Продовження таблиці Б.3

Планування OBS		
Складання календарного плану		
Визначення ризиків		
Розрахунок бюджету		
Створення бази даних погодних показників		
Створення модулю аналізу введених даних		
Створення модулю обчислень нечітких та геометричних параметрів		
Створення графічного інтерфейсу		
Компілювання програмного коду		
Тестування		
Виправлення помилок		
Оформлення документації		
Архівація проекту		
Введення в експлуатацію		

PDM мережа. Для кожного пакету робіт розробляють приватні мережеві моделі, в яких встановлюють логічні взаємозв'язки між усіма роботами, які необхідно виконувати для отримання запланованого продукту з пакета робіт. Мережеві моделі дозволяють визначити тривалість виконання пакету робіт, які рекомендується будувати з використанням програмного інструментарію (MS Project, Spider Project, Project Expert, Open Plan, Primavera або ін.). PDM-мережі складаються з двох типів елементів: робіт, які знаходяться в вузлах, і стрілок, які вказують логічні взаємозв'язки між роботами проекту. Мережевий графік розробляється проекту було побудовано, використовуючи програму MS Project. Мережа представлена на рис. Б.3-Б.6.

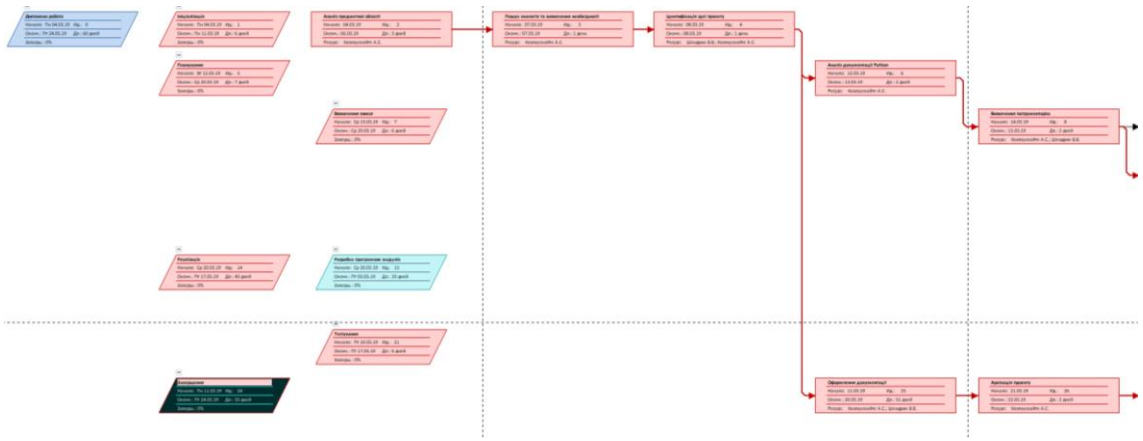

Рисунок Б.3 - PDM – мережа проекту

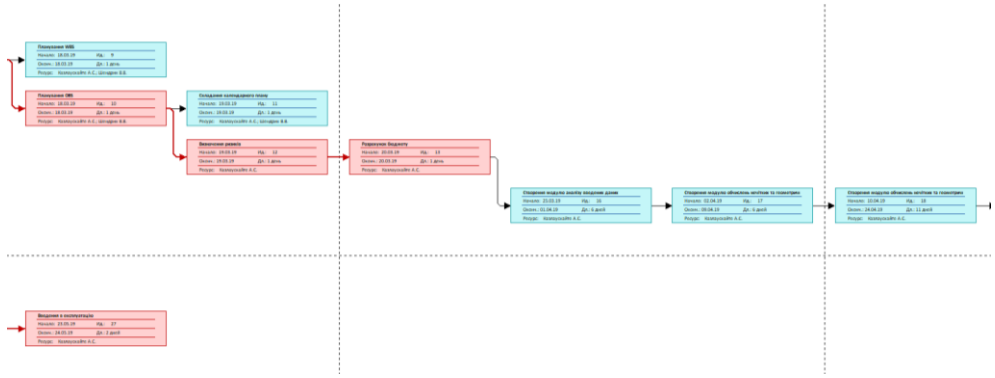

Рисунок Б.4 - Продовження PDM – мережі проекту

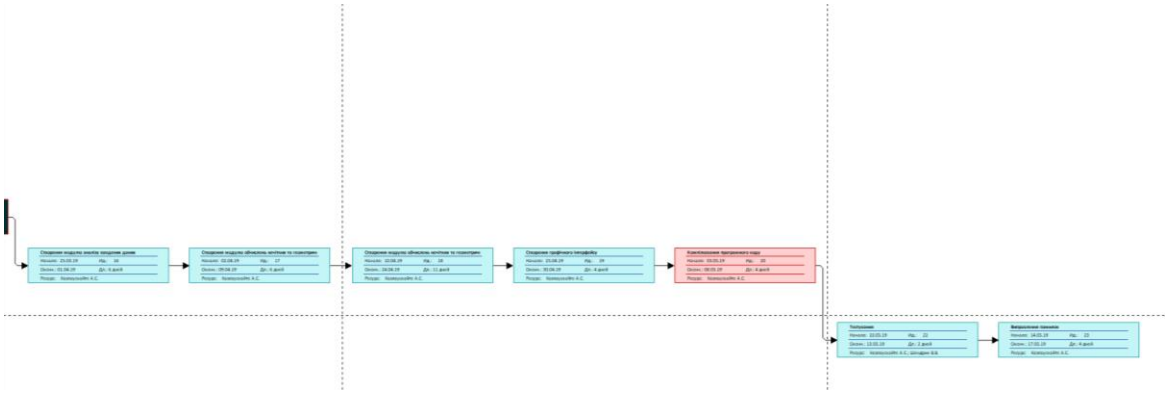

Рисунок Б.5 - Продовження PDM – мережі проекту

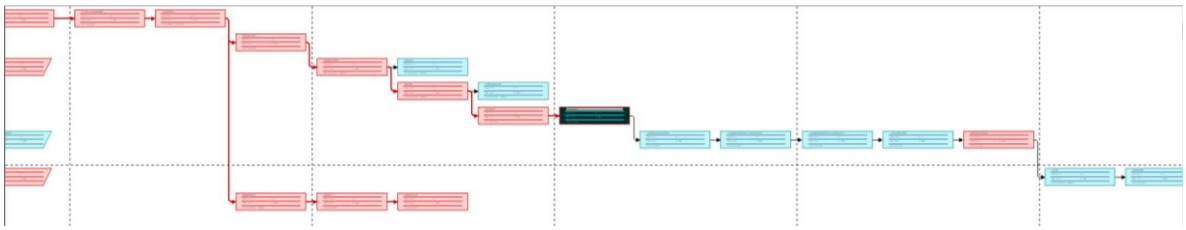

Рисунок Б.6 – Загальний вигляд PDM – мережі проекту

Діаграма Ганта. Для того, щоб мати реальне уявлення про тривалість виконання робіт з урахуванням обмеженості у використанні ресурсів, на підставі часткової мережевої моделі, а також, проекту в цілому з урахуванням вихідних і святкових днів, будують календарний графік робіт. Він є реальним розподілом робіт по календарним датам, тобто своєрідним розкладом виконання робіт. Діаграма Ганта є досить зручним для користування. Його будують таким чином:

- на горизонталі фіксують календар в тих одиницях часу, які обрані для проекту (години, дні);
- зліва на вертикалі розташовують найменування всіх робіт;
- на поле, що утворилося, поставляють у вигляді прямокутників роботи, довжина яких по горизонталі відповідає їх тривалості. Між роботами лініями вказують логічні зв’язки[15].

Діаграму Ганта для розроблюваного проекту було побудовано, використовуючи програму MS Project. Її вигляд представлений на рисунку Б.7, список робіт зазначено на рисунку Б.8.

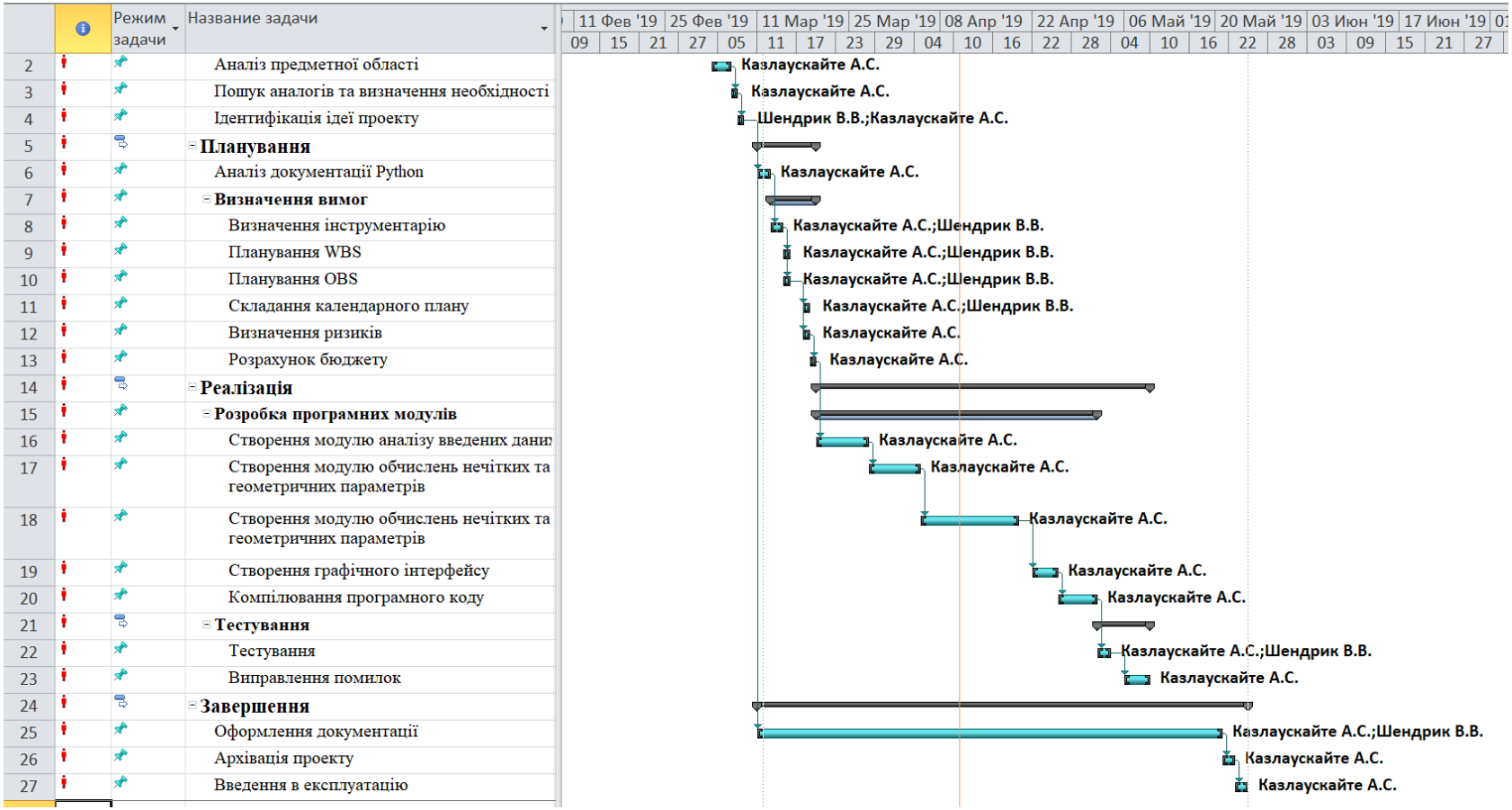

Рисунок Б.7 – Діаграма Ганта

	Режим задачи	Название задачи	Длительно	Начало	Окончание	Предшественники	Названия ресурсов
1		**Ініціалізація**	**6 дней**	**Пн 04.03.19**	**Пн 11.03.19**		**Казлаускайте А.С.;Ш**
2		Аналіз предметної області	3 дней	Пн 04.03.19	Ср 06.03.19		Казлаускайте А.С.
3		Пошук аналогів та визначення необхідності	1 день	Чт 07.03.19	Чт 07.03.19	2	Казлаускайте А.С.
4		Ідентифікація ідеї проекту	1 день	Пт 08.03.19	Пт 08.03.19	3	Шендрик В.В.;Казлау
5		**Планування**	**7 дней**	**Пн 11.03.19**	**Вт 19.03.19**		**Казлаускайте А.С.;Ш**
6		Аналіз документації Python	2 дней	Пн 11.03.19	Вт 12.03.19	4	Казлаускайте А.С.
7		**Визначення вимог**	**5 дней**	**Ср 13.03.19**	**Вт 19.03.19**		**Казлаускайте А.С.;Ш**
8		Визначення інструментарію	2 дней	Ср 13.03.19	Чт 14.03.19	6	Казлаускайте А.С.;Ше
9		Планування WBS	1 день	Пт 15.03.19	Пт 15.03.19	8	Казлаускайте А.С.;Ше
10		Планування OBS	1 день	Пт 15.03.19	Пт 15.03.19	8	Казлаускайте А.С.;Ше
11		Складання календарного плану	1 день	Пн 18.03.19	Пн 18.03.19	10	Казлаускайте А.С.;Ше
12		Визначення ризиків	1 день	Пн 18.03.19	Пн 18.03.19	10	Казлаускайте А.С.
13		Розрахунок бюджету	1 день	Вт 19.03.19	Вт 19.03.19	12	Казлаускайте А.С.
14		**Реалізація**	**37 дней**	**Ср 20.03.19**	**Чт 09.05.19**		**Казлаускайте А.С.**
15		**Розробка програмних модулів**	**31 дней**	**Ср 20.03.19**	**Ср 01.05.19**		**Казлаускайте А.С.**
16		Створення модулю аналізу введених даних	6 дней	Ср 20.03.19	Ср 27.03.19	13	Казлаускайте А.С.
17		Створення модулю обчислень нечітких та геометричних параметрів	6 дней	Чт 28.03.19	Чт 04.04.19	16	Казлаускайте А.С.
18		Створення модулю обчислень нечітких та геометричних параметрів	11 дней	Пт 05.04.19	Пт 19.04.19	17	Казлаускайте А.С.
19		Створення графічного інтерфейсу	4 дней	Пн 22.04.19	Чт 25.04.19	18	Казлаускайте А.С.
20		Компілювання програмного коду	4 дней	Пт 26.04.19	Ср 01.05.19	19	Казлаускайте А.С.
21		**Тестування**	**6 дней**	**Чт 02.05.19**	**Чт 09.05.19**		**Казлаускайте А.С.;Ш**
22		Тестування	2 дней	Чт 02.05.19	Пт 03.05.19	20	Казлаускайте А.С.;Ше
23		Виправлення помилок	4 дней	Пн 06.05.19	Чт 09.05.19	22	Казлаускайте А.С.
24		**Завершення**	**55 дней**	**Пн 11.03.19**	**Пт 24.05.19**		**Казлаускайте А.С.;Ш**
25		Оформлення документації	51 дней	Пн 11.03.19	Пн 20.05.19	4	Казлаускайте А.С.;Ше
26		Архівація проекту	2 дней	Вт 21.05.19	Ср 22.05.19	25	Казлаускайте А.С.

Рисунок Б.8 – Список робіт для діаграми Ганта

Управління ризиками. Під ризиком в проектній діяльності розуміється ймовірна подія, в результаті якого суб’єкт, який прийняв рішення, втрачає можливість досягти запланованих результатів проекту або його окремих параметрів, що мають тимчасову, кількісну і вартісну оцінку. Ризик характеризується певними джерелами або причинами і має наслідки, тобто впливає на результати проекту.

Невизначеності, що існують в кожному проекті, можуть стати причиною виникнення ризиків. Ризики можуть бути «відомі» – ті, які визначені, оцінені, для яких можливо планування. Ризики «невідомі» – ті, які не ідентифіковано і не можуть бути спрогнозовані. Хоча специфічні ризики і умови їх виникнення не визначені, але й вони можуть бути спрогнозовані[20].

Управління ризиками – це процеси, пов’язані з ідентифікацією, аналізом ризиків та прийняттям рішень, які включають максимізацію позитивних і мінімізацію негативних наслідків настання ризикових подій.

Ґрунтуючись на даних було створено класифікацію ризиків для даного проекту, що наведена в таблиці Б.4.

Таблиця Б.4 –Класифікація рисків

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

Використовуючи дану класифікацію, було побудовано матрицю ризиків, що представлена на рис. Б.9.

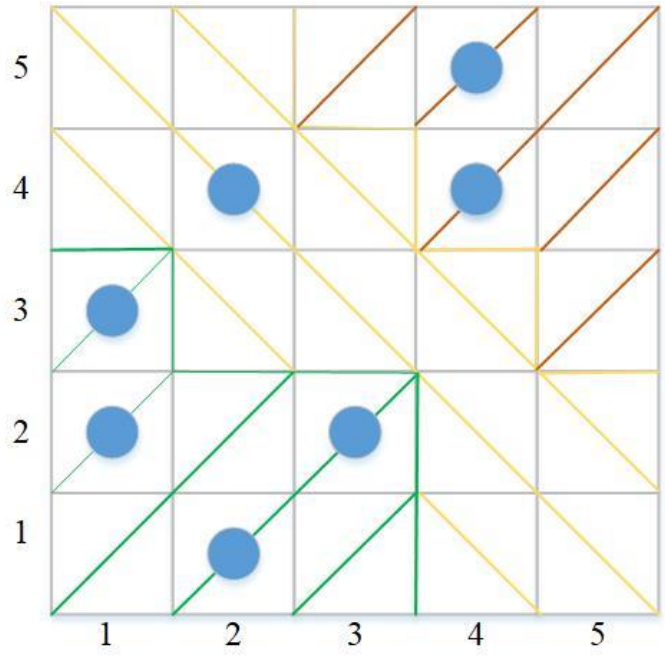

Рисунок Б.9 – Матриця ризиків

Далі було визначено рівні ризиків та ступінь їх дії для розрахунку рангу ризику за відповідною формулою: R = P * L, де R – ранг ризику; P – ймовірність виникнення; L – величина втрат.

Рівні можуть бути:

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

Ступінь впливу ризиків:

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

Ґрунтуючись на описаних вище даних, було виконано оцінку ступенів та рівнів для кожного ризику в проекті. Результати роботи представлені в таблиці Б.5.

Таблиця Б.5 – Визначення ступенів та рівнів ризиків

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

Формування бюджету. Кошторис продукту проекту – це загальні майбутні витрати, які необхідні безпосередньо для створення продукту

проекту. Тобто це витрати на фінансування всіх робіт, передбачених WBS - структурою проекту.

Бюджет продукту проекту – це кошторис продукту проекту, розподілений в часі на основі календарного плану реалізації робіт або за окремими WBS елементами. План фінансування – це кошторис продукту проекту в розрізі основних джерел фінансування робіт з проекту. Розрахуємо бюджет проекту по окремим WBS-елементам.

На початку реалізації планування робіт було визначено учасників проекту, які заплановано мають брати участь в етапах реалізації поставленої задачі. Спираючись на діаграму OBS було виділено таких працівників:

- розробник\тестувальник\дизайнер;
- менеджер проекту.

В табл. Б.6 приведена заробітна плата всіх учасників проекту з урахуванням погодинної ставки та кількості відпрацьованих годин.

Таблиця Б.6 – Заробітна плата учасників проекту

Посада	За 1 год (позаурочні)	Робочих годин	Заробітна плата
розробник\тестувальник\дизайнер	120(150)	296	35520
Менеджер проекту	150(200)	46	6900

Бюджет заробітної плати складає 42420 грн.

Додаток С

Файли коду реалізації

Файл коду gparcer.py

```
import logging
import datetime

import requests
from lxml import html
import json

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

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


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

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

```
            return dates


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


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


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


        # parse html page to get info about weather
        def extract_winfo(page):
            widget_info_rows     =     page.xpath("//*[@class='forecast_frame
hw_wrap']//*[@class='widget__container']/*")[:3]
            dates = extract_datetimes(widget_info_rows[0])
            weathers = extract_weather(widget_info_rows[1])
```

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

            return info


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

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

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

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

            return res


        # make reqest to url
        # return string with html
```

```
    def req_html(url, headers):
        response = requests.get(url, headers=headers)

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

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

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

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

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


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

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

        weather_info = extract_winfo(today_page) # extract all info to
list, each element is info about some period 00:00, 00:03, ... 21:00
```

```
        weather_info.pop(0)  #  remove  measurement  for  previous  day  from
dataset
        weather_info.append(extract_winfo(tomorrow_page)[0])  #  add  last
measurement of day to dataset

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

Файл коду myconnutils.py

```
import logging

import pymysql.cursors
from PyQt5.QtWidgets import QMessageBox

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

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

def insert_city(connection, place):
    # Prepare SQL query to INSERT a record into the database.
    id = str(place['id'])
    sql_check =("SELECT number FROM place")
    cursor = execute_select(connection, sql_check)
    result = cursor.fetchall()
    n=0
```

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


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


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


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

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


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

Файл коду mydesign.py

```
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'MainW.ui'
#
# Created by: PyQt5 UI code generator 5.11.3
#
# WARNING! All changes made in this file will be lost!

from PyQt5 import QtCore, QtGui, QtWidgets
from PyQt5.QtGui import QPixmap, QIcon
from PyQt5.QtCore import QSize, Qt
from settings import Ui_Dialog
import webbrowser
import pymysql.cursors
import myconnutils

class Ui_MainWindow(object):
```

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

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


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

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


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


            self.lbl_city = QtWidgets.QLabel(MainWindow)
            self.lbl_city.setGeometry(QtCore.QRect(20, 90, 50, 30))
            self.lbl_city.setObjectName("lbl_city")
            self.input_city = QtWidgets.QLineEdit(MainWindow)
            self.input_city.setGeometry(QtCore.QRect(80, 90, 710, 30))
            self.input_city.setObjectName("input_city")
            self.search_btn = QtWidgets.QPushButton(MainWindow)
            self.search_btn.setGeometry(QtCore.QRect(800, 90, 30, 30))
            self.search_btn.setObjectName("search_btn")
            self.search_btn.setIcon(QIcon('E:\\Python-
projects\\Diplom\\search.png'))
            self.search_btn.setIconSize(QSize(30, 30))
```

```
self.groupBox1 = QtWidgets.QGroupBox(self.centralwidget)
self.groupBox1.setGeometry(QtCore.QRect(10, 130, 400, 230))
font = QtGui.QFont()
font.setPointSize(9)
self.groupBox1.setFont(font)
self.groupBox1.setObjectName("groupBox1")

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


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


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

```
            self.lbl_n.setObjectName("lbl_n")
            self.lbl_m2 = QtWidgets.QLabel(self.groupBox1)
            self.lbl_m2.setGeometry(QtCore.QRect(330, 110, 35, 30))
            self.lbl_m2.setObjectName("lbl_m2")
            self.lbl_wt = QtWidgets.QLabel(self.groupBox1)
            self.lbl_wt.setGeometry(QtCore.QRect(330, 150, 70, 30))
            self.lbl_wt.setObjectName("lbl_wt")

            self.apply_btn = QtWidgets.QPushButton(self.groupBox1)
            self.apply_btn.setGeometry(QtCore.QRect(270, 185, 120, 30))
            self.apply_btn.setObjectName("apply_btn")

            self.calendarWidget                                          =
QtWidgets.QCalendarWidget(self.centralwidget)
            self.calendarWidget.setGeometry(QtCore.QRect(10,   370,   400,
250))
            self.calendarWidget.setBaseSize(QtCore.QSize(400, 300))
            self.calendarWidget.setObjectName("calendarWidget")

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

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

      #results labels
```

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

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

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



self.img_vey = QtWidgets.QLabel(self.groupBox)
self.img_vey.setGeometry(QtCore.QRect(20, 20, 75, 75))
self.img_vey.setObjectName("img_vey")
pixmap = QPixmap('E:\\Python-projects\\Diplom\\veu.png')
self.img_vey.setPixmap(pixmap)
self.img_vey.resize(75, 75)

self.img_sb = QtWidgets.QLabel(self.groupBox)
self.img_sb.setGeometry(QtCore.QRect(20, 100, 75, 75))
self.img_sb.setObjectName("img_sb")
```

```
            pixmap2 = QPixmap('E:\\Python-projects\\Diplom\\sb.png')
            self.img_sb.setPixmap(pixmap2)
            self.img_sb.resize(75, 75)

            self.img_bt = QtWidgets.QLabel(self.groupBox)
            self.img_bt.setGeometry(QtCore.QRect(20, 180, 75, 75))
            self.img_bt.setObjectName("img_bt")
            pixmap3 = QPixmap('E:\\Python-projects\\Diplom\\battery.png')
            self.img_bt.setPixmap(pixmap3)
            self.img_bt.resize(75, 75)

            self.img_cons = QtWidgets.QLabel(self.groupBox)
            self.img_cons.setGeometry(QtCore.QRect(20, 250, 75, 75))
            self.img_cons.setObjectName("img_cons")
            pixmap4 = QPixmap('E:\\Python-projects\\Diplom\\consump.png')
            self.img_cons.setPixmap(pixmap4)
            self.img_cons.resize(75, 75)


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

            self.lbl_overflow_value = QtWidgets.QLabel(self.groupBox)
            self.lbl_overflow_value.setGeometry(QtCore.QRect(180,    410,
100,30))
            self.lbl_overflow_value.setObjectName("lbl_overflow_value")


            self.horizontalLayoutWidget                                  =
QtWidgets.QWidget(self.centralwidget)
            self.horizontalLayoutWidget.setGeometry(QtCore.QRect(10,   0,
270, 30))

self.horizontalLayoutWidget.setObjectName("horizontalLayoutWidget")
            self.horizontalLayout                                        =
QtWidgets.QHBoxLayout(self.horizontalLayoutWidget)
            self.horizontalLayout.setContentsMargins(0, 0, 0, 0)
            self.horizontalLayout.setObjectName("horizontalLayout")
```

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

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

            self.horizontalLayout.addWidget(self.weatherBtn)

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

self.horizontalLayoutWidget_2.setObjectName("horizontalLayoutWidget_2")
            self.horizontalLayout_2                                 =
QtWidgets.QHBoxLayout(self.horizontalLayoutWidget_2)
            self.horizontalLayout_2.setContentsMargins(0, 0, 0, 0)
            self.horizontalLayout_2.setObjectName("horizontalLayout_2")
            self.UA_btn                                             =
QtWidgets.QPushButton(self.horizontalLayoutWidget_2)
            self.UA_btn.setBaseSize(QtCore.QSize(35, 35))
            font = QtGui.QFont()
```

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

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

        def retranslateUi(self, MainWindow):
            _translate = QtCore.QCoreApplication.translate
            MainWindow.setWindowTitle(_translate("MainWindow", "HYBRID"))
            self.groupBox.setTitle(_translate("MainWindow",      "Поточний
стан"))
            self.label.setText(_translate("MainWindow", "Вт/год"))
            self.label_3.setText(_translate("MainWindow", "Вт/год"))
            self.label_2.setText(_translate("MainWindow", "Вт/год"))
            self.label_4.setText(_translate("MainWindow", "Вт/год"))

            self.label_veu.setText(_translate("MainWindow", "Результат"))
            self.label_sb.setText(_translate("MainWindow", "Результат"))
            self.label_bt.setText(_translate("MainWindow", "Результат"))
```

```
            self.label_cons.setText(_translate("MainWindow",
"Результат"))

            self.lbl_source.setText(_translate("MainWindow",      "Джерело
живлення:"))
            self.lbl_overflow.setText(_translate("MainWindow",
"Надлишок:"))
            self.lbl_city.setText(_translate("MainWindow", "МІСТО"))
            self.lbl_source_value.setText(_translate("MainWindow",
"Результат"))
            self.lbl_overflow_value.setText(_translate("MainWindow",
"Результат"))
            #self.settingBtn.setText(_translate("MainWindow",
"Зберегти"))
            self.reportBtn.setText(_translate("MainWindow", "Звіт"))
            self.weatherBtn.setText(_translate("MainWindow",      "Прогноз
погоди"))
            self.UA_btn.setText(_translate("MainWindow", "UA"))
            self.ENG_btn.setText(_translate("MainWindow", "ENG"))

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

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

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

Файл коду myprogram.py

```
import gparser
import myconnutils

import datetime
from time import sleep
import threading

import xlrd
from openpyxl import load_workbook

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

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

import numpy as np
import pandas as pd
import math

def report_create(Es, Ew, Eb, W, source, overflow):
    date = datetime.datetime.now().date()
```

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


def priority(Es, Ew, Eb, W, Eb_max):
    if Es > W:
        Source = 'Es'
        overflow = Es + Ew - W
        if Eb < Eb_max:
            Eb = Eb + overflow
            if Eb >= Eb_max:
                E_send = Eb-Eb_max
    elif Ew > W:
        Source = 'Ew'
        overflow = Ew - W + Es
        if Eb < Eb_max:
            Eb = Eb + overflow
            if Eb >= Eb_max:
                E_send = Eb-Eb_max
    elif Es + Ew > W:
        Source = 'Es+Ew'
        overflow = Ew - W + Es
        if Eb < Eb_max:
            Eb = Eb + overflow
            if Eb >= Eb_max:
                E_send = Eb-Eb_max
    elif Es + Ew + Eb > W:
        Source = 'Es+Ew+Eb'
        Eb = Eb - (W - Es - Ew)
        E_send = 0
    else: Source = 'Power Grid'
    return Source, Eb, E_send
```

```
A = {}
A[1] = np.array([[0,    0,    0],
                 [-0.058001325,     3.129664419,     -59.33981437],
                 [0.650827154, -37.25557963,     399.3108268]])
A[2] = np.array([[0,    0,    -16.48632757],
                 [0,     0,    84.01107552],
                 [0,     0,    531.0846902]])
A[3] = np.array([[-0.031114566,     2.021197585,     -90.76184864],
                 [0.068892225, -4.369347687,     355.3810179],
                 [0.812020337, -55.20939546,     1831.403134]])
A[4] = np.array([[0,    0,    0],
                 [0,     0,    0],
                 [0,     0,    0.00522]])
A[5] = np.array([[0,    0,    0],
                 [0,     0,    0],
                 [0,     0,    0.3966659]])
A[6] = np.array([[0.008553425,     -0.32813861,     -60.79002498],
                 [-0.235862408,     14.08462039,     125.6567786],
                 [1.054414961, -69.26651236,     1898.321959]])
A[7] = np.array([[0,    0,    0],
                 [0,     0,    0],
                 [0,     0,    0.015005432]])
A[8] = np.array([[0,    0,    0],
                 [0,     0,    0],
                 [0,     0,    0.55680151]])

B = {}
B[1] = np.array([[0,    0,    0],
                 [-0.058001325,     3.129664419,     -59.33981437],
                 [0.650827154, -37.25557963,     399.3108268]])
B[2] = np.array([[0,    0,    -16.48632757],
                 [0,     0,    84.01107552],
                 [0,     0,    276.9132438]])
```

```
B[3] = np.array([[-0.031114566,     2.021197585,      -90.76184864],
                 [0.068892225, -4.369347687,     355.3810179],
                 [0.812020337, -55.20939546,     1439.758948]])
B[4] = np.array([[0,    0,    0],
                 [0,     0,    0],
                 [0,     0,    0.00522]])
B[5] = np.array([[0,    0,    0],
                 [0,     0,    0],
                 [0,     0,    0.3966659]])
B[6] = np.array([[0.008553425,      -0.32813861,      -60.79002498],
                 [-0.235862408,      14.08462039,      125.6567786],
                 [1.054414961, -69.26651236,     1403.405653]])
B[7] = np.array([[0,    0,    0],
                 [0,     0,    0],
                 [0,     0,    0.015005432]])
B[8] = np.array([[0,    0,    0],
                 [0,     0,    0],
                 [0,     0,    0.55680151]])


C = {}
C[1] = np.array([[0,    0,    0],
                 [-0.058001325,      3.129664419,      -59.33981437],
                 [0.650827154, -37.25557963,     399.3108268]])
C[2] = np.array([[0,    0,    -16.48632757],
                 [0,     0,    84.01107552],
                 [0,     0,    770.4549488]])
C[3] = np.array([[-0.031114566,     2.021197585,      -90.76184864],
                 [0.068892225, -4.369347687,     355.3810179],
                 [0.812020337, -55.20939546,     2357.63803]])
C[4] = np.array([[0,    0,    0],
                 [0,     0,    0],
                 [0,     0,    0.00522]])
C[5] = np.array([[0,    0,    0],
                 [0,     0,    0],
```

```
                [0,     0,    0.3966659]])
C[6] = np.array([[0.008553425,      -0.32813861,      -60.79002498],
                [-0.235862408,      14.08462039,      125.6567786],
                [1.054414961, -69.26651236,     2286.862064]])
C[7] = np.array([[0,    0,    0],
                [0,     0,    0],
                [0,     0,    0.015005432]])
C[8] = np.array([[0,    0,    0],
                [0,     0,    0],
                [0,     0,    0.55680151]])


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

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

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

    if V == 'Велика' and overcast == 'Пасмурно':
        if Ew > W:
```

```
        Source = 'Ew'
    elif Eb > W:
        Source = 'Eb'
    else: Source = 'Power Grid'

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

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

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

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

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

```
    elif Eb > W:
        Source = 'Eb'
    else: Source = 'Power Grid'

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

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

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

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

```
def time_define():
    now = datetime.datetime.now().time()
    h = now.hour//3
    if h==0:
        t=0
    elif h==1:
        t=3
    elif h==2:
        t=6
    elif h==3:
        t=9
    elif h==4:
        t=12
    elif h==5:
        t=15
    elif h==6:
        t=18
    elif h==7:
        t=21
    return t

def W_concumption(A, B, C, data):

    curr_date = datetime.datetime.strptime(data, '%Y-%m-%d')
    n = float(curr_date.isocalendar()[1])
    d = float(curr_date.weekday()+1)
    #---------------
    k = time_define()
    t = k/24.0
    #---------------
    a1=A[1]
    a2=A[2]
    a3=A[3]
    a4=A[4]
```

```
    a5=A[5]

    a6=A[6]

    a7=A[7]

    a8=A[8]

    Wmod_f =
(a1[1,0]*d+a1[2,0])*n**2+(a1[1,1]*d+a1[2,1])*n+(a1[1,2]*d+a1[2,2])*t+a2[0,2
]*d**2+a2[1,2]*d+a2[2,2]

    Wmod_1 =
((a3[0,0]*d**2+a3[1,0]*d+a3[2,0])*n**2+(a3[0,1]*d**2+a3[1,1]*d+a3[2,1])*n+(
a3[0,2]*d**2+a3[1,2]*d+a3[2,2]))*math.exp(-((t-a5[2,2])**2)/a4[2,2])

    Wmod_2 =
((a6[0,0]*d**2+a6[1,0]*d+a6[2,0])*n**2+(a6[0,1]*d**2+a6[1,1]*d+a6[2,1])*n+(
a6[0,2]*d**2+a6[1,2]*d+a6[2,2]))*math.exp(-((t-a8[2,2])**2)/a7[2,2])



    b1=B[1]

    b2=B[2]

    b3=B[3]

    b4=B[4]

    b5=B[5]

    b6=B[6]

    b7=B[7]

    b8=B[8]

    Wmin_f =
(b1[1,0]*d+b1[2,0])*n**2+(b1[1,1]*d+b1[2,1])*n+(b1[1,2]*d+b1[2,2])*t+b2[0,2
]*d**2+b2[1,2]*d+b2[2,2]

    Wmin_1 =
((b3[0,0]*d**2+b3[1,0]*d+b3[2,0])*n**2+(b3[0,1]*d**2+b3[1,1]*d+b3[2,1])*n+(
b3[0,2]*d**2+b3[1,2]*d+b3[2,2]))*math.exp(-((t-b5[2,2])**2)/b4[2,2])

    Wmin_2 =
((b6[0,0]*d**2+b6[1,0]*d+b6[2,0])*n**2+(b6[0,1]*d**2+b6[1,1]*d+b6[2,1])*n+(
b6[0,2]*d**2+b6[1,2]*d+b6[2,2]))*math.exp(-((t-b8[2,2])**2)/b7[2,2])



    c1=C[1]

    c2=C[2]

    c3=C[3]

    c4=C[4]

    c5=C[5]

    c6=C[6]

    c7=C[7]
```

```
    c8=C[8]

    Wmax_f =
(c1[1,0]*d+c1[2,0])*n**2+(c1[1,1]*d+c1[2,1])*n+(c1[1,2]*d+c1[2,2])*t+c2[0,2
]*d**2+c2[1,2]*d+c2[2,2]

    Wmax_1 =
((c3[0,0]*d**2+c3[1,0]*d+c3[2,0])*n**2+(c3[0,1]*d**2+c3[1,1]*d+c3[2,1])*n+(
c3[0,2]*d**2+c3[1,2]*d+c3[2,2]))*math.exp(-((t-c5[2,2])**2)/c4[2,2])

    Wmax_2 =
((c6[0,0]*d**2+c6[1,0]*d+c6[2,0])*n**2+(c6[0,1]*d**2+c6[1,1]*d+c6[2,1])*n+(
c6[0,2]*d**2+c6[1,2]*d+c6[2,2]))*math.exp(-((t-c8[2,2])**2)/c7[2,2])


    #--------------

    W_mod=Wmod_f+Wmod_1+Wmod_2

    W_min=Wmin_f+Wmin_1+Wmin_2

    W_max=Wmax_f+Wmax_1+Wmax_2


    W=[W_min, W_mod, W_max]

    return W


def E_veu(tp, t, R, Vw, V):

    Q=0.00001661*tp**2-0.004764*tp+1.2924

    f=0.003869*Vw**2-0.128*Vw+6.650827154

    w=V/R

    Cp=0.351

    E=Q*Cp*Vw*math.pi*R**2

    return E


def E_sb(tp, Square, E, overcast, month):

    N1=0.0901 * E + 0.0873 * tp - 0.00032 * E * tp

    P_mod = Square*N1

    N2 = 0.0876*E + 0.0499*tp - 0.00027*E*tp

    P_min = N2*Square

    N3 = 0.0918*E + 0.1055*tp - 0.00035*E*tp

    P_max = N3*Square

    alfa = np.array([[0.6126, 0.8063,     0.0970,     0],

                     [0.6261,  0.8130,     0.0935,     0],

                     [0.5931,  0.7966,     0.1017,     0],
```

```
                    [0.6658,  0.8329,     0.0835,     0],
                    [0.7282,  0.8641,     0.0679,     0],
                    [0.7146,  0.8573,     0.0713,     0],
                    [0.7656,  0.8828,     0.0586,     0],
                    [0.7963,  0.8982,     0.0509,     0],
                    [0.7068,  0.8534,     0.0733,     0],
                    [0.6561,  0.8281,     0.0860,     0],
                    [0.5525,  0.7763,     0.1119,     0],
                    [0.5713,  0.7857,     0.1072,     0]])
    if overcast == 'Пасмурно':
        x = P_min + alfa[month, 0]*(P_mod - P_min)
    elif overcast == 'Облачно':
        x = P_min + alfa[month, 1]*(P_mod - P_min)
    elif overcast == 'Малооблачно':
        x = P_max - alfa[month, 2]*(P_max - P_mod)
    elif overcast == 'Ясно':
        x = P_max - alfa[month, 3]*(P_max - P_mod)
    return x

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

#windows works
class myMainWindow(QtWidgets.QMainWindow):

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

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

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

def UA_text(self):
    self.ui.groupBox1.setTitle("Технічні характеристики")
    self.ui.lbl_veu_r.setText("Радіус ротора:")
    self.ui.lbl_veu_n.setText("Номінальна швид. ВУ:")
    self.ui.lbl_sb_s.setText("Площа СБ:")
    self.ui.lbl_bt_e.setText("Ємність акумулятора:")
    self.ui.lbl_m.setText("м")
    self.ui.lbl_n.setText("м/с")
    self.ui.lbl_m2.setText("м2")
    self.ui.apply_btn.setText("Застосувати")
    self.ui.lbl_main_city.setText("Суми, Сумська область, Україна")
    self.ui.lbl_source.setText("Джерело живлення:")
    self.ui.lbl_overflow.setText("Надлишок:")
    self.ui.lbl_city.setText("МІСТО")
    if self.ui.lbl_source_value.text() == 'Result':
        self.ui.lbl_source_value.setText("Результат")
    else: pass
    if self.ui.lbl_overflow_value.text() == 'Result':
        self.ui.lbl_overflow_value.setText("Результат")
```

```
        else: pass
        self.ui.reportBtn.setText("Звіт")
        self.ui.weatherBtn.setText("Прогноз погоди")
        if self.ui.label_veu.text() == 'Result':
            self.ui.label_veu.setText("Результат")
        else: pass
        if self.ui.label_sb.text =='Result':
            self.ui.label_sb.setText("Результат")
        else: pass
        if self.ui.label_bt.text() == 'Result':
            self.ui.label_bt.setText("Результат")
        else: pass
        if self.ui.label_cons.text() == 'Result':
            self.ui.label_cons.setText("Результат")
        else: pass
        self.ui.groupBox.setTitle("Поточний стан")
        self.ui.label.setText("Вт/год")
        self.ui.label_3.setText("Вт/год")
        self.ui.label_2.setText("Вт/год")
        self.ui.label_4.setText("Вт/год")

    def ENG_text(self):
        self.ui.groupBox1.setTitle("Specifications")
        self.ui.lbl_veu_r.setText("Rotor radius:")
        self.ui.lbl_veu_n.setText("Rated speed of WG:")
        self.ui.lbl_sb_s.setText("Square SP:")
        self.ui.lbl_bt_e.setText("Battery Capacity:")
        self.ui.lbl_m.setText("m")
        self.ui.lbl_n.setText("m/s")
        self.ui.lbl_m2.setText("m2")
        self.ui.apply_btn.setText("Apply")
        self.ui.lbl_main_city.setText("Sumy, Sumska oblast, Ukraine")
        self.ui.lbl_source.setText("Power supply:")
        self.ui.lbl_overflow.setText("Overflow:")
```

```
        self.ui.lbl_city.setText("City")
        if self.ui.lbl_source_value.text() == 'Результат':
            self.ui.lbl_source_value.setText("Result")
        else: pass
        if self.ui.lbl_overflow_value.text() == 'Результат':
            self.ui.lbl_overflow_value.setText("Result")
        else: pass
        self.ui.reportBtn.setText("Report")
        self.ui.weatherBtn.setText("Weather forecast")
        if self.ui.label_veu.text() == 'Результат':
            self.ui.label_veu.setText("Result")
        else: pass
        if self.ui.label_sb.text =='Результат':
            self.ui.label_sb.setText("Result")
        else: pass
        if self.ui.label_bt.text() == 'Результат':
            self.ui.label_bt.setText("Result")
        else: pass
        if self.ui.label_cons.text() == 'Результат':
            self.ui.label_cons.setText("Result")
        else: pass
        self.ui.groupBox.setTitle("Current condition:")
        self.ui.label.setText("W/h")
        self.ui.label_3.setText("W/h")
        self.ui.label_2.setText("W/h")
        self.ui.label_4.setText("W/h")

    def searchCity(self):

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

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

```
            myconnutils.insert_city(con, res)

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

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

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


    def applyData(self):

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

            cit = self.ui.lbl_main_city.text()
            place = cit.split(',')
            city = place[0]
```

```
            t= time_define()
            if t<10:
                time = '0'+str(t) + ':00:00'
            else: time = str(t) + ':00:00'
            sql ="SELECT w.w_speed, w.temperature, w.overcast, w.dat_e,
w.tim_e, w.weather_id, p.city_name FROM weather AS w, place AS p WHERE
w.place_id = p.place_id AND p.city_name = '"
            a = sql+city+"' AND w.tim_e = '"+time+"' "
            cursor.execute(a)
            weth = cursor.fetchone()

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

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

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

            source = sourse(speed, over, Prior[1], P_w, P_sb, W[1])
            self.ui.label_veu.setText(str(round(P_w, 2)))
            self.ui.label_sb.setText(str(round(P_sb, 2)))
```

```
            self.ui.label_bt.setText(str(round( Prior[1], 2)))
            self.ui.lbl_source_value.setText(source)
            self.ui.label_cons.setText(str(round(W[1], 2)))
            self.ui.lbl_overflow_value.setText(str(round(Prior[2])))

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

        except:
            msgBox = QMessageBox()
            msgBox.warning(self, 'Помилка ',"Не вірно введено дані!")



class UpdateResults(threading.Thread):
    def __init__(self, label_w, label_v, label_s, label_e, label, label1,
label2, label3, label4, label5, label6):
        super(UpdateResults, self).__init__()
        self.daemon = True
        self.labe_w = label_w
        self.label_v = label_v
        self.label_s = label_s
        self.label_e = label_e
        self.label = label
        self.label1 = label1
        self.label2 = label2
        self.label3 = label3
        self.label4 = label4
        self.label5 = label5
        self.label6 = label6
        self.start()

    def run(self):

        connection = myconnutils.get_connection()
```

```
        cursor = connection.cursor()
        cit = self.label.text()
        place = cit.split(',')
        city = place[0]
        #city = 'Суми'
        print(city)
        t= time_define()-3.0
        s = str(t)
        s.split('.')[0]
        if t<10:
            time = '0'+s[0] + ':00:00'
        else: time = s[0] + ':00:00'
        sql ="SELECT p.Pb FROM weather AS w, power AS p, place AS pl WHERE
p.weather_id = w.weather_id  AND w.place_id = pl.place_id AND w.tim_e = '"
        a = sql+time+"' AND pl.city_name = '"+city+"' "
        print(a)
        cursor.execute(a)
        power = cursor.fetchone()
        print (power)
        t1= time_define()
        s1 = str(t1)
        s1.split('.')[0]

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

        ins=pd.read_excel('ins.xlsx', index_col = 0)
        lst = list(ins.index)
```

```
        lst = [str(s) for s in lst]
        ins.index = lst

        n = int(t1/3.0)
        dat = weth['dat_e']
        mounth = dat.split('-')
        m = int(mounth[1])
        hour = weth['tim_e']
        tp = int(weth['temperature'])
        over = weth['overcast']
        speed = float(weth['w_speed'])

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

        P_w = E_veu(tp, t, r, speed, v )
        P_sb = E_sb(tp, s, float(ins.loc[hour,m]), over, m-1)# power of
solar pannels at current time
        W = W_concumption(A, B, C, dat)#electricity consumprion at current
time
        P_b = float(power['Pb'])
        sour = sourse(speed, over, P_b, P_w, P_sb, W[1] )
        if sour == 'Ew' or sour == 'Es' or sour == 'Ew + Es' or sour ==
'Power Grid':
            P_b_new =  P_b + P_sb + P_w - W[1]
        else: P_b_new = P_b - W[1]
        if P_b_new>e:
            P_send = P_b_new-e
            P_b_new = e

        w_id = weth['weather_id']
        power = (P_sb, P_w, P_b_new, sour, W[1],P_send, w_id)
        myconnutils.insert_power(connection, power)
```

```
        while True:


            self.label1.setText(str(P_w))

            self.label2.setText(str(P_sb))

            self.label3.setText(str(P_b))

            self.label4.setText(str(W[1]))

            self.label5.setText(str(sour))

            self.label6.setText(str(P_send))

            sleep(10800)



if __name__ == '__main__':


    #open window

    app = QtWidgets.QApplication(sys.argv)

    Win = myMainWindow()

    up = UpdateResults(Win.ui.input_veu_r, Win.ui.input_veu_n,
Win.ui.input_sb_s, Win.ui.input_bt_e, Win.ui.lbl_main_city,
Win.ui.label_veu, Win.ui.label_sb, Win.ui.label_bt, Win.ui.label_cons,
Win.ui.lbl_source_value, Win.ui.lbl_overflow_value)

    Win.show()



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

Додаток Д

ІНСТРУКЦІЯ КОРИСТУВАЧА

1. Запуск додатку

Для коректного використання інформаційної технології користувачу потрібно мати повний пакет файлів (myprogram.exe, ins.xlsx, Звіт.docx). Також користувачу потрібно впевнитись, що всі необхідні файли знаходяться в одній папці з виконуваним файлом додатку та існує інтернет підключення.

2. Виконання програми

Після запуску додатку зявляється основне діалогове вікно (рис.Д.1).

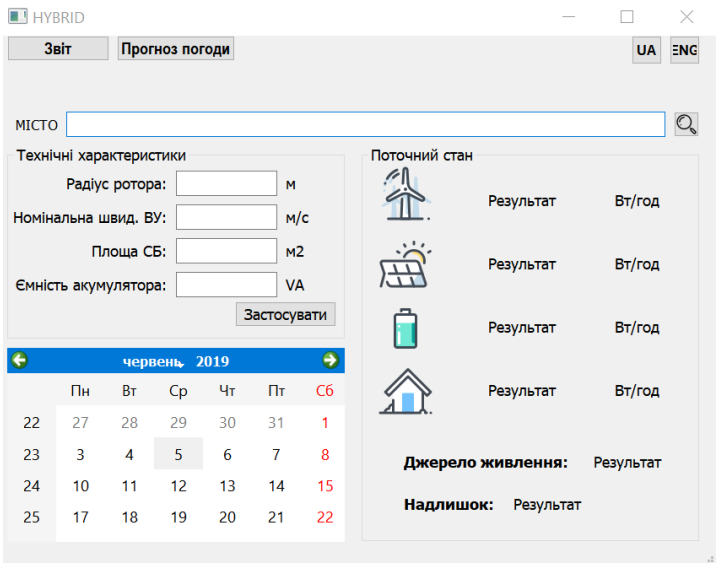

Рисунок Д.1 – Головне вікно додатку

Для пошуку необхідного міста треба внести в поле вводу коротку (лише назва) або повну (назва, область, країна) назву міста та натиснути кнопку пошуку. Як результат зверху відобразиться повна назва шуканого міста (рис.Д.2).

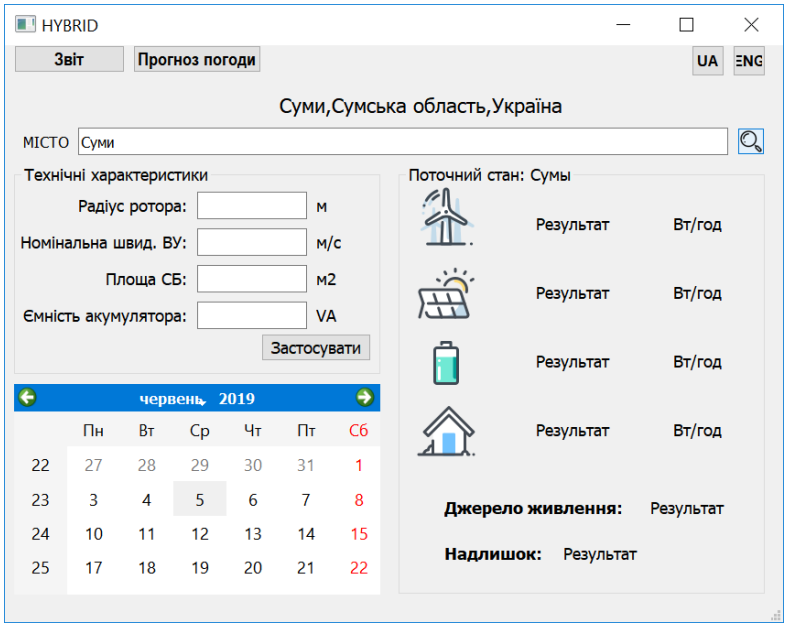

Рисунок Д.2 – Результат пошуку міста

У разі , якщо такого міста не існує, або ж користувач ввів некорректі дані, користувач отримає відповідне повідомлення про помилку (рис.Д.3).

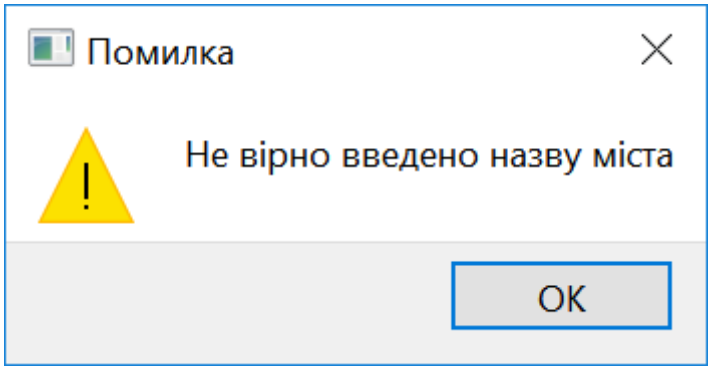

Рисунок Д.3 – Повідомлення про помилку

Далі у випадку якщо користуч не знає української мови, то він має змогу перекласти інтерфейс англійською мовою при натисканні кнопки «ENG» (рис.Д.4), а при натисканні кнопки «UA» інтерфейс знову буде представлено українською мовою.

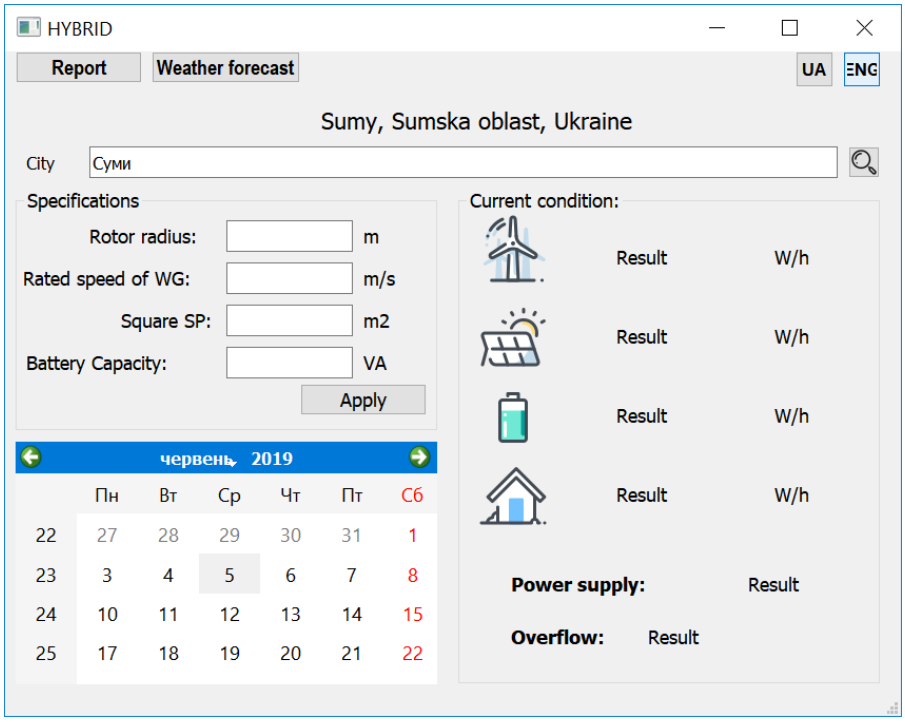

Рисунок Д.4 – Результат перекладу англійською мовою

Далі для потрібного обрахунку та отримання результату потрібно внести до полей вводу технічні характеристики: радіус вітроротора (або довжину лопостей), номінальну швидкість обертання ротора, загальну площину встановлених сонячних батарей та ємність акумуляторної батареї. Всі ці дані можна дізнатися з технічних паспортів виробів.

Після введення даних для отримання результату користувачу потрібно натиснути кнопку «Застосувати», результат дії якої представлено на рисунку Д.5. У випадку, якщо користувач введе некоректні дані, отримає відповідне повідомлення про помилку (рис.Д.6) та повинен буде внести коректні дані.

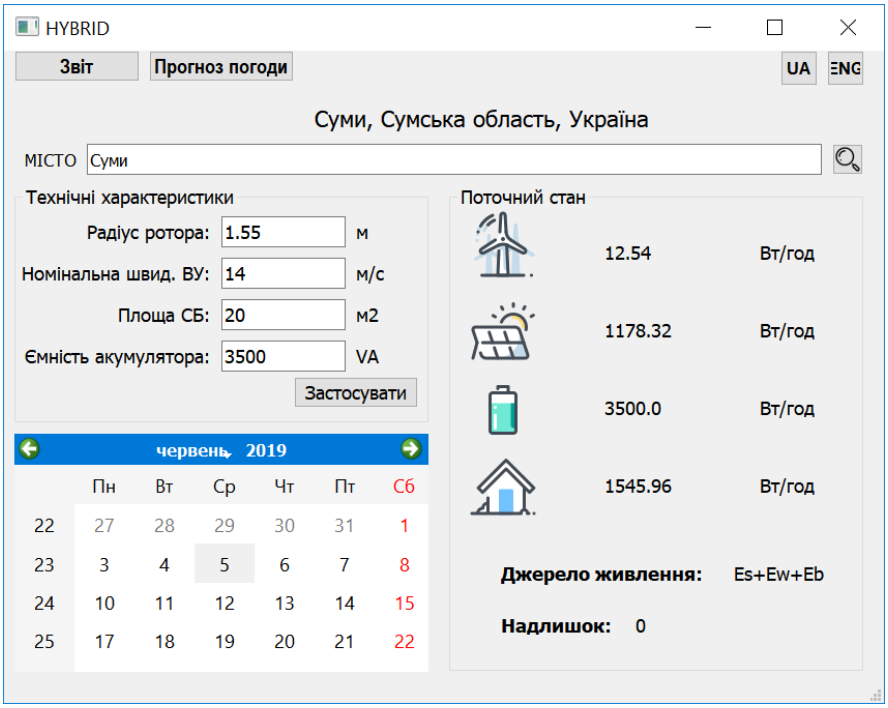

Рисунок Д.5 – Результат натискання кнопки «Застосувати»

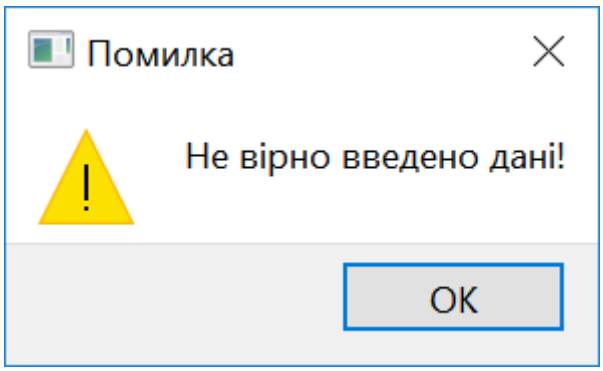

Рисунок Д.6 – Повідомлення про помилку

У результаті користувач отримає поточний результат роботи гібридної електромережі. Зазначається джерело живлення:

– Es – підключено сонячні панелі;
– Ew – підключено вітрогенератор;
– Es+Ew – підключено сонячні панелі та вітрогенератор;

– Es+Ew+Ep – підключено сонячні панеля, вітрогенератор та акумуляторну батарею;
– Power grid – підключено загальну мережу;

та кількість надлишкової електроенергії, що передається до загальної електромережі.

У випадку, якщо користувачу потрібно передивитись погодні дані, то така можливість надається при натисканні кнопки «Прогноз погоди». Як результат зазначеної дії – відкриється сайт «Gismeteo.com» з даними погоди найближчого до користувача населеного пункту (рисД.7).

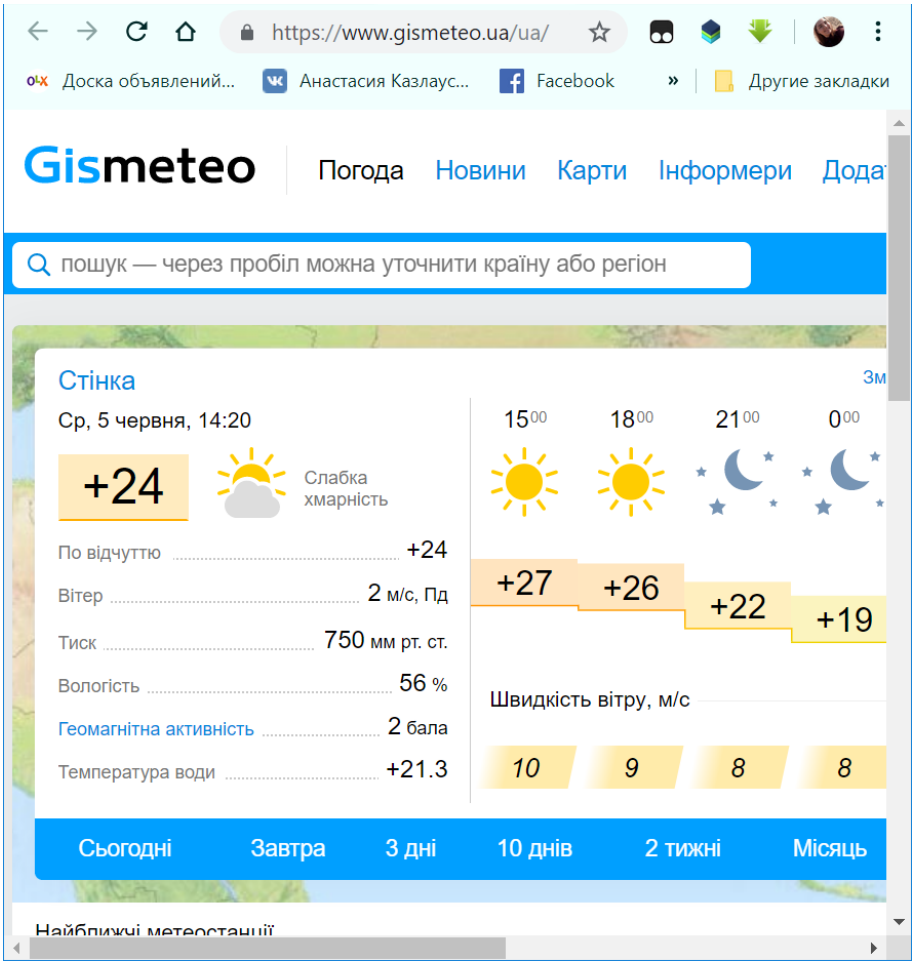

Рисунок Д.7 – Сайт прогнозу погоди

Всі отримані дані зберігаються в базі даних та якщо користувачу потрібно, то він має можливість зберегти результат у вигляді звіту формату .docx, натиснувши кнопку «Звіт» (рис.Д.8) з отриманням відповідного повідомлення про успішне збереження (рис.Д9). Звіт зберігається у папку з додатком.

РЕЗУЛЬТАТИ·ЕЛЕКТРОСПОЖИВАННЯ¶

¶

Потужність·ВЕУ¤	1178.32¤
Потужність·СБ¤	12.54¤
Ємність·акумулятора¤	3500.0¤
Рівень·споживання·електроенергії¤	1545.96¤
Джерело·енергії¤	Es+Ew+Eb¤
Передано·до·загальної·мережі¤	0¤

¶

Дата:2019-06-05¶

Рисунок Д.8 – Збережений звіт

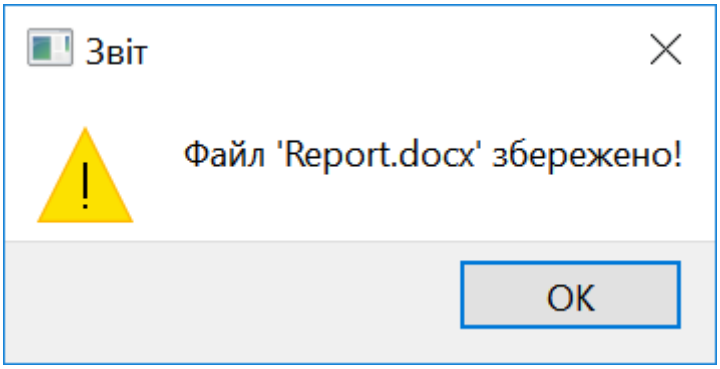

Рисунок Д.9 – Повідомлення про збереження звіту

Додаток Е

Прогнозування рівня електрогенерації сонячних батарей при управлінні гібридною електромережею

Казлаускайте А.С. *студент*; Шендрик С.О., *аспірант*
Сумський державний університет, м. Суми, Україна

Сучасне життя неможливо уявити без енергії. Розподілена генерація з залученням відновлюємих джерел енергії відіграє все більшу роль в електрозабезпеченні багатьох країн світу. У разі використання різнотипних джерел енергії електромережа є гібридною. Рівень потужності генерації в такій мережі характеризується швидкою зміною режимів роботи в залежності від погодних умов та рівня споживання електроенергії. Також важливими завданнями є узгодження потужностей та вибір оптимального режиму експлуатації. Для визначення оптимального режиму в першу чергу необхідно прогнозувати рівень електрогенерації від відновлюємих джерел енергії.

В даному дослідженні розглядається залежність електричних параметрів сонячних батарей від температури та освітленість, які несуть в собі невизначеність.

Метою дослідженя є формування нечіткої прогнозної моделі потужності сонячних батарей для використання у системи підтримки прийняття рішень при керуванні гібридною електромережею.

При створенні прогнозних моделей енергогенерації необхідно враховувати невизначеність вхідних даних. Пропонується визначати максимальну потужність сонячної батареї у вигляді трикутного нечіткого числа, тобто кортежем:

$$P = < P_{\text{mod}}, P_{\text{min}}, P_{\text{max}} >,$$

де P_{mod} – модальне значення, $P_{\text{min}}, P_{\text{max}}$ – ліва та права межа інтервалу невизначеності.

Точність прогнозу в великій мірі залежить від ступеню невизначеності прогнозу інсоляції і температури повітря. Тому в розробленій моделі передбачено можливість використання як результатів прямих вимірювань інсоляції та температури, так і результатів їх оперативного прогнозування.