Міністерство освіти і науки України
Криворізький національний університет
Кафедра моделювання та програмного забезпечення

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти бакалавра

зі спеціальності 121 – Інженерія програмного забезпечення

На тему: Комп'ютеризація проєктів відновлюваних джерел енергії в умовах нестабільного електропостачання

Засвідчую, що в цій кваліфікаційній роботі немає запозичень із праць інших авторів без відповідних посилань.
Студент гр. ІПЗ–22-2

_____ / Б. О. Клюцковський /

Керівник кваліфікаційної роботи _____/ І. А. Котов /

Завідувач кафедри _____/ А. М. Стрюк /

Кривий Ріг

2026

Криворізький національний університет

Факультет: Інформаційних технологій

Кафедра: Моделювання та програмного забезпечення

Ступінь вищої освіти: бакалавр

Спеціальність: 121 – Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:

Зав. кафедри

_____ А. М. Стрюк

«____» _____ 2026 р.

ЗАВДАННЯ

на кваліфікаційну роботу

студенту групи ІПЗ–22-2 Клюцковському Богдану Олексійовичу

1. Тема: Комп'ютеризація проєктів відновлюваних джерел енергії в умовах нестабільного електропостачання

затверджена наказом по КНУ № 285с від «28» травня 2026 р.

2. Термін подання студентом закінченої роботи: «08» червня 2026р.

3. Вихідні дані по роботі: програмний комплекс для автоматизації проєктування систем відновлюваних джерел енергії в умовах нестабільного електропостачання.

4. Зміст пояснювальної записки (перелік питань, що їх треба розробити):
проаналізувати підходи до автоматизації проєктування систем відновлюваних джерел енергії в умовах нестабільного електропостачання, проаналізувати існуючи програмні системи автоматизації відновлюваних джерел енергії, спроектувати додаток, розробити програмне забезпечення системи автоматизації проєктування систем відновлюваних джерел енергії в умовах нестабільного електропостачання, здійснити тестування розробленого додатку.

5. Перелік ілюстративного матеріалу: функціональна схема, блок–схема алгоритму, зображення екранних форм додатку.

Календарний план:

№	Найменування етапів кваліфікаційної роботи	Термін виконання етапів роботи
1	Пошук науково-практичних джерел та інтернет-ресурсів з тематики роботи	02.02.26 – 09.02.26
2	Аналіз існуючих підходів і методів вирішення проблеми	10.02.26 – 23.02.26
3	Формулювання актуальності і постановка завдань роботи	24.02.26 – 02.03.26
4	Оформлення матеріалів першого розділу роботи	03.03.26 – 09.03.26
5	Розробка структурної і функціональної схем та основних алгоритмів програмного проєкту	10.03.26 – 23.03.26
6	Оформлення матеріалів другого розділу роботи	24.03.26 – 06.04.26
7	Розробка візуального інтерфейсу, баз даних і програмних модулів проєкту	07.04.26 – 20.04.26
8	Оформлення матеріалів третього розділу роботи	21.04.26 – 11.05.26
9	Оформлення додатків	12.05.26 – 18.05.26
10	Тестування розробленої програмного комплексу	19.05.26 – 01.06.26
11	Оформлення пояснювальної записки та супровідних документів	02.06.26 – 08.06.26

Дата видачі завдання: «02» лютого 2026 р.

Студент: _____ / Б. О. Клюцковський /

Керівник роботи: _____ / І. А. Котов /

РЕФЕРАТ

ІНТЕРФЕЙС, КОНТРОЛЕР, ВИПРОМІНЮВАННЯ, ДАТЧИК, КОЛЕКТОР, ОБ’ЄКТ, РЕГУЛЯТОР, СЕРВЕР

Пояснювальна записка: 79 с., 45 рис., 3 табл., 1 дод., 28 джерел.

Кваліфікаційна робота: «Комп'ютеризація проєктів відновлюваних джерел енергії в умовах нестабільного електропостачання».

Мета випускової кваліфікаційної роботи: проєктування і розробка програмних рішень автоматизації процесу керування системами сонячного підігріву при дефіциті електроенергії на базі методів, моделей, структур даних і алгоритмів для вирішення завдань підвищення ефективності систем відновлюваних джерел енергії.

Об’єктом дослідження є: процес керування системами сонячного енергопостачання в умовах нестабільного електропостачання.

Теоретична частина роботи присвячена проблемі автоматизації процесу керування системами сонячного підігріву в умовах нестабільного електропостачання. Проведено аналіз підходів, методів та засобів проєктування та побудови систем автоматичного регулювання систем сонячного підігріву. На основі проведеного аналізу зроблено висновок про необхідність подальшого удосконалення програмних засобів систем сонячного подопідігріву. Отже, тема кваліфікаційної роботи є актуальною.

Практична частина кваліфікаційної роботи визначає розробку структурних і функціональних моделей, структур даних, алгоритмів, бази даних, моделей інтерфейсу, програмних модулів, звітів проєкта відновлюваних джерел енергії в умовах нестабільного електропостачання.

Аналіз розробленого програмного комплексу показує, що отримані програмні результати можуть бути використані в наукових установах, промисловості і цивільних об'єктах для автоматичного керування системами сонячного підігріву, а також у вищих навчальних закладах. Використання розробленого застосунку підвищить ефективність проєкта відновлюваних джерел енергії в умовах нестабільного електропостачання.

ABSTRACT

INTERFACE, CONTROLLER, RADIATION, SENSOR, COLLECTOR, OBJECT, REGULATOR, SERVER

Explanatory note: 79 p., 45 fig., 3 tab., 1 supplement, 28 sources.

Qualification work: «Computerization of renewable energy sources projects in conditions of unstable power supply».

The purpose of the final qualifying work: design and development of hardware and software solutions for automating the process of controlling solar heating systems during electricity shortages based on methods, models, data structures and algorithms to solve the problems of increasing the efficiency of renewable energy systems.

The object of research and design are: the process of automatic control of solar heating systems in conditions of unstable power supply.

The theoretical part of the work is devoted to the problem of automating the process of controlling solar heating systems in conditions of unstable power supply. An analysis of approaches, methods and means of designing and building automatic control systems for solar heating systems was conducted. Based on the analysis, a conclusion was drawn about the need for further improvement of software tools for solar heating systems. Therefore, the topic of the qualification work is relevant.

The practical part of qualification work determines the development of structural and functional models, data structures, algorithms, databases, interface models, software modules, and project reports for renewable energy sources in conditions of unstable power supply.

Analysis of the developed software package shows that the obtained software results can be used in scientific institutions, industry and civil facilities for automatic control of solar heating systems, as well as in higher educational institutions. The use of the developed application will increase the efficiency of the renewable energy project in conditions of unstable power supply.

ЗМІСТ

ВСТУП

Переваги сонячної енергії: велика кількість, низька вартість, не забруднює навколишнє середовище. Сонячні теплові системи дозволяють заощадити в будівлях значні обсяги енергії — до 95 % для готелів і до 60 % для житлових будинків. Сонячні теплові системи окупаються приблизно за 2–7 років [1]. При цьому термін їхньої експлуатації становить понад 25 років. Витрати на технічне обслуговування мінімальні. Їх можна використовувати для гарячого водопостачання, опалення приміщень та басейнів, а також для кондиціонування. Тобто вони можуть покривати 85 % енергетичних потреб будівлі. Їх можна використовувати в будь-якому типі будівлі, навіть у пам’ятках архітектури, кам’яних, дерев’яних тощо. Адже виробники сонячних колекторів вже мають усі необхідні рішення. Їх можна поєднувати з усіма існуючими системами будівлі, наприклад: мазутними та газовими котлами, тепловими насосами, дров’яними та біомасовими котлами, геотермальними системами тощо [1, 2]. Сонячна теплоенергетика — це технологія, яка робить нас незалежними від палива. Навіть якщо в майбутньому вартість усіх джерел теплової енергії зросте, та частина енергії, яку ми отримуємо від Сонця, завжди буде безкоштовною.

Завдяки пасивним системам опалення (правильне архітектурне проектування, орієнтація, теплоізоляція, розташування будівельних об’ємів, відповідні будівельні матеріали) будинок може покривати значну частину (навіть 80–100 %) своїх потреб [2]. Найважливішими елементами структури пасивної сонячної системи є: вікна, орієнтовані на південь, для збору та утримання сонячного випромінювання, компактні об’єми матеріалів із відносно великою теплоємністю для накопичення уловленого тепла. Іншим способом є використання сонячної енергії в теплицях [2].

Сучасні сонячні системи теплопостачання використовуються в різних умовах. За результатами аналізу можна зробити висновок, що сучасні комплекси проектування систем використання сонячного світла мають

підтримку і технічне забезпечення в світі. Тому тема кваліфікаційної роботи є актуальною.

Метою дослідження є розробка програмних автоматизованих застосунків для керування системами сонячного енергопостачання при нестачі електроенергії на базі методів, моделей, структур даних і алгоритмів для вирішення завдань підвищення ефективності комплексів сонячного підігріву приміщень.

Об’єктом дослідження є процес функціонування систем сонячного енергопостачання в умовах нестабільного електропостачання.

Предметом дослідження є математичні, структурні та функціональні моделі, програмні модулі для автоматизації процесу керування системами сонячного енергопостачання в умовах нестабільного електропостачання.

Кваліфікаційна робота має новизну. Новизна полягає в тому, що створений програмний застосунок відрізняється від існуючих подібних застосунків тим, що базується на програмної платформі існуючих інтерованих середовищ розробки.

Кваліфікаційна робота має практичне значення. Результати роботи можуть бути використовувані в програмних застосунках управління обладнанням теплопостачання, у навчальних закладах при викладанні дисциплін, пов’язаних з проблематикою автоматизованих систем управління [2].

1 ЗАДАЧА АВТОМАТИЗАЦІЇ ПРОЄКТУВАННЯ СИСТЕМ ВІДНОВЛЮВАНИХ ДЖЕРЕЛ ЕНЕРГІЇ В УМОВАХ НЕСТАБІЛЬНОГО ЕЛЕКТРОПОСТАЧАННЯ

1.1. Актуальність проблеми проєктування відновлюваних джерел енергії в умовах нестабільного електропостачання

Сьогодні сонячні теплові системи дозволяють заощадити в будівлях значні обсяги енергії — до 95 % для готелів і до 60 % для житлових будинків. Сонячні теплові системи окупаються приблизно за 2–7 років. При цьому термін їхньої експлуатації становить понад 25 років. Витрати на технічне обслуговування мінімальні [1, 2]. Їх можна використовувати для гарячого водопостачання, опалення приміщень та басейнів, а також для кондиціонування. Тобто вони можуть покривати 85 % енергетичних потреб будівлі. Їх можна використовувати в будь-якому типі будівлі, навіть у пам’ятках архітектури, кам’яних, дерев’яних тощо. Адже виробники сонячних колекторів вже мають усі необхідні рішення [2]. Їх можна поєднувати з усіма існуючими системами будівлі, наприклад: мазутними та газовими котлами, тепловими насосами, дров’яними та біомасовими котлами, геотермальними системами тощо. Сонячна теплоенергетика — це технологія, яка робить нас незалежними від палива. Навіть якщо в майбутньому вартість усіх джерел теплової енергії зросте, та частина енергії, яку ми отримуємо від Сонця, завжди буде безкоштовною [2]. Завдяки пасивним системам опалення (правильне архітектурне проектування, орієнтація, теплоізоляція, розташування будівельних об’ємів, відповідні будівельні матеріали) будинок може покривати значну частину (навіть 80–100 %) своїх потреб. Найважливішими елементами структури пасивної сонячної системи є: вікна, орієнтовані на південь, для збору та утримання сонячного випромінювання, компактні об’єми матеріалів із відносно великою теплоємністю для накопичення уловленого тепла. Іншим способом є використання сонячної

енергії в теплицях.

Активні сонячні системи — це механічні конструкції, здатні збирати сонячну енергію, перетворювати її на корисну (теплову, холодову або електричну), накопичувати її частину та розподіляти для використання [2, 3]. Найпоширенішими активними сонячними системами є сонячні колектори для нагріву води та фотоелектричні панелі (різновид сонячного колектора) для виробництва електроенергії невеликої потужності [2]. Також до цієї категорії належать вакуумні сонячні колектори, їх поєднання з абсорбційними холодильними установками (Absorption Chillers) для забезпечення потреб у охолодженні, а також сонячні колектори високої ентальпії для безпосереднього виробництва електроенергії з використанням парових турбін або органічних циклів [3].

Вартість теплопостачання приватного будинку показано на рис. 1.1.

Рисунок 1.1 – Вартість теплопостачання приватного будинку

Найпоширенішим застосуванням активних сонячних систем є опалення приміщень або нагрівання води, причому вони збирають сонячне випромінювання при низькій температурі [3, 4].

Тобто вони перетворюють сонячне випромінювання на тепло.

Енергетичні сонячні системи можна використовувати скрізь, де потрібне тепло низької температури. Таким чином, використання сонячної енергії для охолодження, кондиціонування приміщень та інших застосувань видається одним із найперспективніших напрямків, оскільки сонячне випромінювання посилюється саме в той період, коли виникає потреба в охолодженні [4].

Ще одним напрямком застосування, що набув поширення на європейському ринку, є поєднання виробництва гарячої води та опалення приміщень за допомогою активних сонячних систем. Використання таких систем у грецьких кліматичних умовах для опалення приміщень вважається технічно та економічно ефективним, якщо поєднується з відповідним проектуванням/будівництвом будівлі (хороша ізоляція, використання пасивних сонячних переваг, тощо) та співпраці користувача [2, 4]. Це дозволяє заощаджувати традиційну енергію як у нових, так і в старих будівлях, у яких вжито всіх можливих заходів для мінімізації втрат та максимізації економічності установки. Проте дуже важливим є правильне проектування сонячної системи та ретельний аналіз економічної ефективності установки, щоб уникнути помилкових рішень та оптимізувати її продуктивність [5]. Схема сонячного енергопостачання води показана на рис. 1.2.

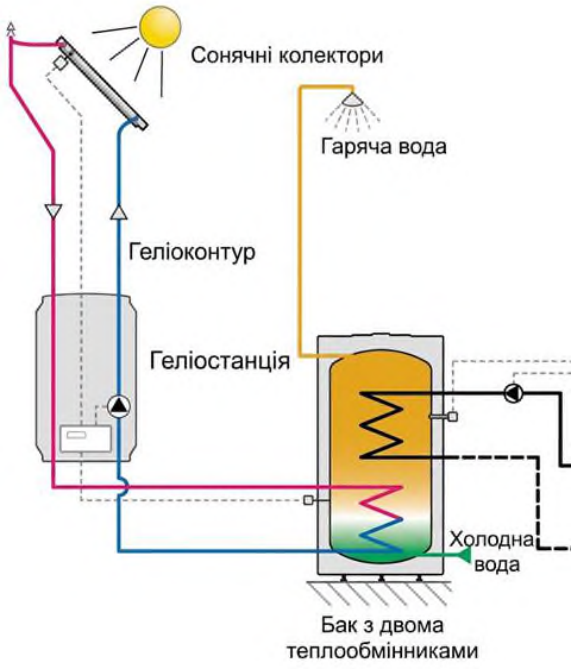


Рисунок 1.2 – Схема сонячного енергопостачання

Переваги сонячного теплопостачання для систем виробництва гарячої води та підтримки опалення забезпечують наступне. Можливість підключення декількох різних джерел тепла, а також можливість задоволення різних потреб за допомогою одного резервуара [4, 5]. Забезпечення для колекторів найнижчої можливої температури, а отже, оптимального ступеня ККД. Великий об’єм, призначений виключно для заправки від колекторів. Пріоритетність джерел тепла. Відсутність рухомих частин усередині накопичувача. Гігієнічне приготування гарячої води. Спрощення монтажу [5].

Переваги накопичувачів гарячої води для побутового призначення наступні. Можливість підключення декількох станцій виробництва гарячої води побутового призначення для забезпечення дуже великих миттєвих витрат [3, 5]. Можливість підключення декількох різних джерел тепла. Можливість нагрівання бака до високої температури для максимального накопичення теплової енергії. Відсутність внутрішнього теплообмінника (змійовика) дозволяє здійснювати пасивний перенос теплової енергії з великою потужністю через резервуар від котла до станцій. Виняткова стійкість резервуара до внутрішньої корозії завдяки відсутності кисню [5]. Гігієнічне приготування гарячої води з точним дотриманням бажаної температури і, як наслідок, економія води завдяки відсутності її змішування під час споживання [5]. Компоновка пасивної системи сонячного теплопостачання показана на рис.1.3.

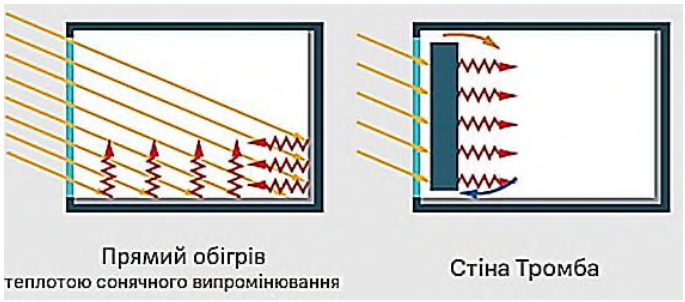


Рисунок 1.3 - Компоновка пасивної системи сонячного теплопостачання

Нище, на рис. 1.4, показано схему удосконаленої сонячної системи взимку та влітку.

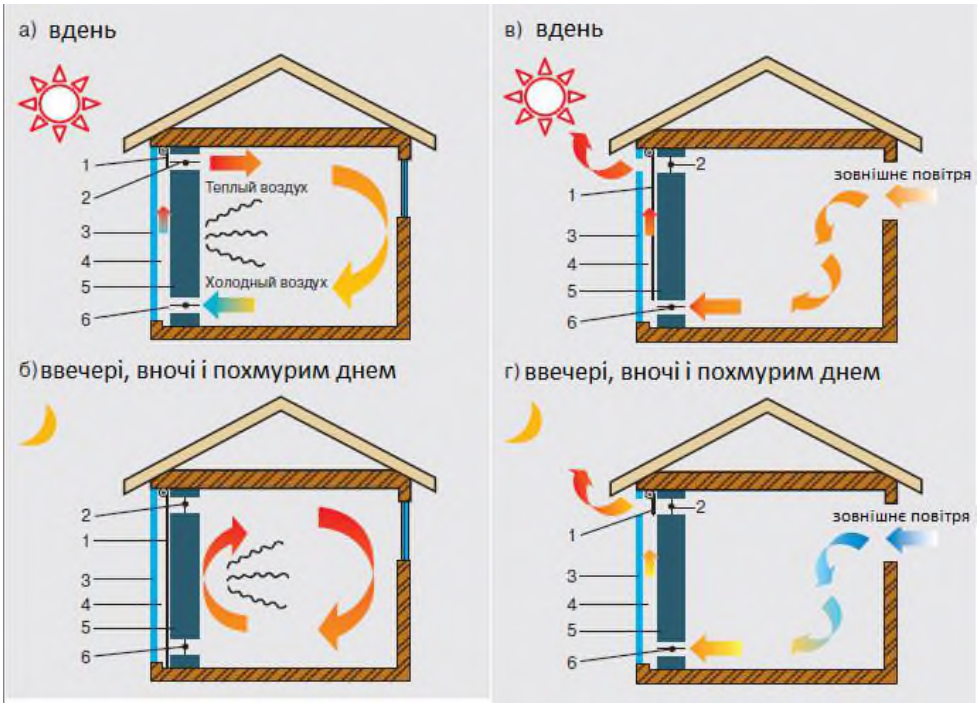


Рисунок 1.4 – Схема сонячної системи

Залежно від сфери застосування, для якої сонячні системи призначені, від використовуваної технології, їхніх розмірів, конфігурації систем, кліматичних умов регіону тощо, можуть використовуватися для різних типів сонячних теплових систем [5, 6]. Різноманітність конфігурацій цих систем зумовлена різними способами їх завдяки яким ці системи захищаються від заморозків, а також від способу забезпечення циркуляції гарячої води. Таким чином, можна виділити два основні типи активних сонячних систем: системи з природною циркуляцією та системи з примусовою циркуляцією [6]. Сонячні водонагрівачі будь-якого типу можуть забезпечити значну частину потреб домогосподарств у гарячій воді споживання води, одночасно знижуючи витрати домогосподарств на енергію [4, 6]. Центральні сонячні системи є найефективнішими, оскільки застосовуються для цілих житлових масивів. Така система забезпечує більш рівномірний розподіл гарячої води протягом цілої доби, що дозволяє зменшити теплові втрати накопиченої води для задоволення потреб усього житлового комплексу.

1.2. Аналіз загальних підходів проєктування систем відновлюваних джерел енергії

Загальні підходи до проєктування систем відновлюваних джерел енергії потребують відповіді на наступні питання щодо проектування сонячної теплової системи [6]. Чи вистачить місця для колекторів та бака? Чи можна правильно зорієнтувати колектори? Чи існує ймовірність затінення колекторів? Чи можна підключити додаткові споживачі (наприклад, басейн, пральну машину тощо)? Які саме вимоги до опалення та яка площа приміщення? Які температури потрібні системі опалення? Скільки мешканців у будинку і коли їм потрібна гаряча вода? Вранці, вдень чи протягом усього дня? Чого очікує клієнт від своєї сонячної системи? Високого рівня покриття потреб чи високої ефективності? Для вирішення цих задач можуть бути використані теплові насоси [5, 7]. Теплові насоси, ефективно використовуючи електроенергію, можуть використовувати тепло навколишнього середовища з природних джерел (повітря, вода, ґрунт) або відпрацьоване тепло (наприклад, рекуперація тепла з повітря, що виводиться з приміщень), щоб зменшити енергоспоживання на опалення та охолодження й досягти значного скорочення викидів парникових газів [6, 7]. Можливість зменшення викидів за допомогою системи теплового насоса безпосередньо пов’язана з трьома факторами: її річним коефіцієнтом корисної дії; типом системи, що використовується для виробництва тепла, та ефективністю виробництва й розподілу тепла; типом використовуваного холодоагенту та відсотком витоку з теплового насоса [7]. Чим вища ефективність системи теплового насоса, тим менше електроенергії витрачається на її роботу та на виробництво необхідного тепла. Звісно, тепло навколишнього середовища, яке використовує тепловий насос, походить із відновлюваних джерел і не пов’язане з викидами парникових газів. Різні види методів встановлення колекторів показані нижче, на рис. 1.5 [6, 7].

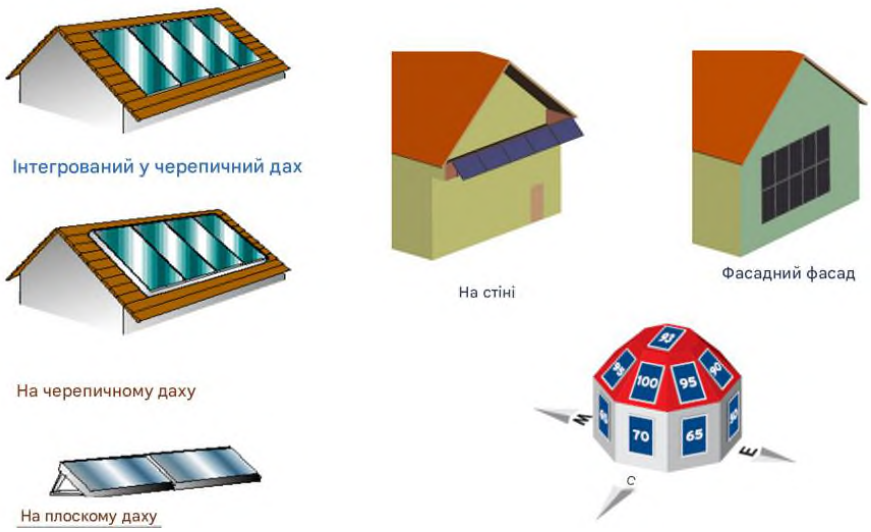


Рисунок 1.5 – Види методів встановлення колекторів

Метод розподілу тепла, який найбільше підходить для приміщень, — це система підлогового опалення, в якій використовуються пластикові, еластичні або мідні труби, вбудовані в бетонну підлогу [6, 7]. Коли гаряча вода протікає по трубах, підлога нагрівається і віддає тепло в приміщення. Системи підлогового опалення використовують помірні температури води. Оскільки різниця температур між підлогою, що випромінює тепло, та колектором є більшою, ніж у системі «колектор/накопичувальний бак», колектори є більш ефективними, оскільки віддають менше тепла у холодне зовнішнє повітря. Розглянемо основні технічні вимоги до систем сонячного енергопостачання [7].

Основним фактором для досягнення оптимальної ефективності сонячного водонагрівача є правильний вибір кута нахилу та орієнтації, з урахуванням місця його встановлення та періоду, протягом якого ми хочемо отримати максимальну віддачу. Сонячний водонагрівач повинен бути орієнтований таким чином, щоб колекторна поверхня була спрямована в бік географічного Півдня для північної півкулі (і географічного Півночі для південної півкулі), тобто завжди була спрямована в бік екватора. Відхилення від правильної орієнтації означає зниження ефективності системи [6, 7].

Якщо відхилення від правильної орієнтації неможливо уникнути, то ефективність системи слід компенсувати за рахунок збільшення площі колектора після вивчення та оцінки конкретних умов. Оскільки кут падіння

сонячного випромінювання змінюється з часом, а також залежно від місця встановлення системи, кут нахилу колекторів повинен приблизно дорівнювати географічній широті місця встановлення. Саме при такому нахилі досягається максимальний річний вихід енергії [7]. Сонячна система енергопостачання показана на рис. 1.6.

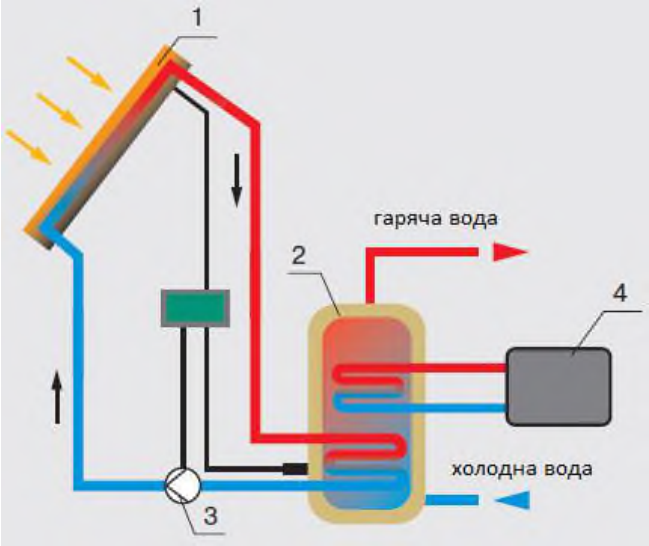


Рисунок 1.6 – Конструкція сонячної системи енергопостачання

Маршрут прокладення трубопроводів і кабелів повинен бути узгоджений між монтажником і замовником, щоб забезпечити правильний монтаж сонячної системи відповідно до чинних місцевих норм, що стосуються гідравлічних та електричних установок [7]. Переконайтеся, що труби, які з’єднують тепловий накопичувач із колектором, а також трубопроводи, що ведуть до та від сонячного водонагрівача, мають ізоляцію, яка витримує температури в діапазоні від -30 °C до 120 °C [5–7]. Необхідно використовувати спеціальний захист від ультрафіолетового випромінювання (anti-UV) Антифриз: Спеціальний теплоносій, що використовується в замкнутому контурі, захищає систему від замерзання та від накопичення солей усередині труб колектора [6]. Оболонка, в якій циркулює теплоносій, не з'єднана з резервуаром для води. Теплоносій має бути добре змішаний з водою у такій пропорції, щоб забезпечити належний захист. Після завершення монтажу місце, де проводилися роботи, має бути чистим та прибраним. Резервуар рідини показано на рис. 1.7.

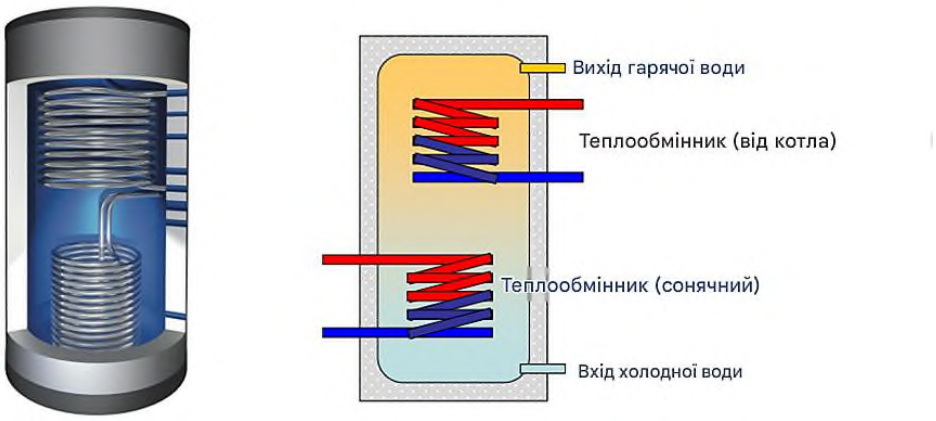


Рисунок 1.7 – Колектор рідини

Монтаж дозволяється лише на похилих та рівних поверхнях з достатньою несучою здатністю. Перш ніж приступити до монтажу, потрібно переконатися, що дах та/або конструкція мають достатню несучу здатність з точки зору статики, відповідно до максимальних очікуваних навантажень у місці встановлення. Якщо місце встановлення знаходиться в районі з надзвичайно великим вітровим та сніговим навантаженням, вся система повинна пройти статичну перевірку, виконану кваліфікованим фахівцем [5, 7]. У окремих випадках можуть знадобитися підсилення або більш міцні конструкції [7].

1.3. Аналіз методів проєктування структури систем відновлюваних джерел енергії

Основним компонентом систем автоматизованого проектування є програмне забезпечення. Наведемо кілька прикладів такого спеціалізованого програмного засобу для проектування солярних водонагрівачів [8].

Щоб спроектувати економічно вигідну сонячну систему, спочатку потрібно розрахувати її теплову ефективність. Найвідомішим наближеним методом розрахунку є метод кривих f, розроблений американцями С. Кляйном, В. Бекманом та Дж. Даффі з Університету Вісконсіна [7, 8]. Цей метод підходить для розрахунку систем опалення, а також може використовуватися

для розрахунку систем гарячого водопостачання або для поєднання обох. Метод кривих f для розрахунків систем гарячого водопостачання застосовується лише у наступних випадках. Розподіл споживання є середнім для житлових будинків. Інші будівлі можуть мати інший добовий розподіл споживання [7]. Ще одна умова застосування цього методу полягає в тому, що сонячна енергія, яка використовується для нагрівання води в резервуарі вище температури Tw, вважається втраченою. Насправді, це не є цілком правильним, оскільки певна кількість гарячої води з температурою, вищою за Tw, змішуючись із холодною водою, дає більшу кількість води з температурою Tw [8]. Однак, незважаючи на ці обмеження, метод кривих f залишається дуже корисним для розрахунку ККД систем гарячого водопостачання [8].

На рис. 1.8 приведений один з проектів сонячних колекторів, розташованих на місцевості.

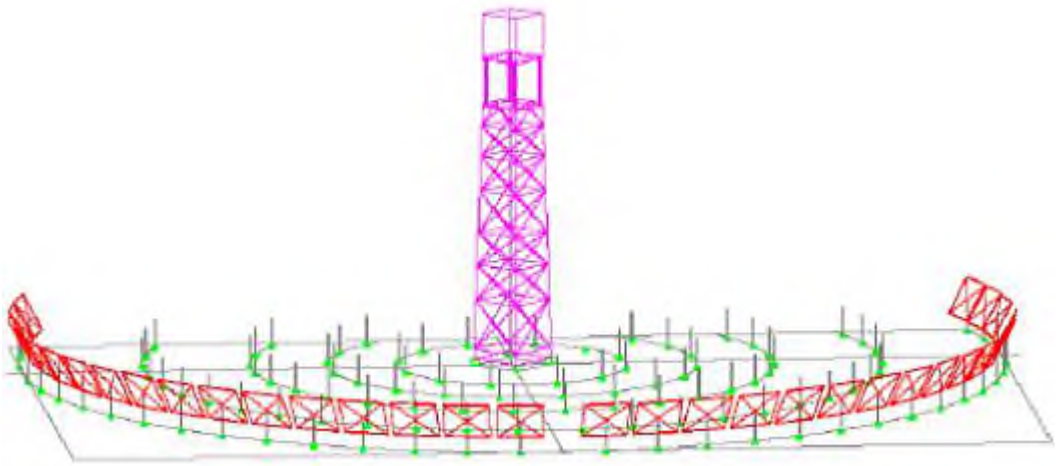


Рисунок 1.8 – Проект сонячних колекторів, розташованих на місцевості

Тепер проаналізуємо проектування моделей сонячних систем енергопостачання [8].

Центральні вежі (Power Towers) використовують низку геліостатів (плоских дзеркал, що слідують за рухом Сонця) для фокусування сонячного випромінювання на центральний приймач, розміщений на вежі (рис. 1.8). Робоча рідина: синтетична олія, розплавлена сіль (нагрівається до 550 °C).

Нагріта рідина виробляє електроенергію безпосередньо або опосередковано. Прикладом є система центральної вежі Solar One у Барстоу, Каліфорнія, потужністю 10 МВт [8]. 1818 плоских дзеркал розміром 7×7 м відбивають сонячну енергію на котел вежі для нагрівання води. Вона успішно працювала з 1983 по 1988 рік, але скорочення державної субсидії призвело до закриття станції. Станція була реконструйована і успішно працювала (під назвою «Solar Two») з 1996 по 1999 рік, використовуючи синтетичну олію як теплоносій. Інші станції працюють в Іспанії (переважно), США та Японії. На сьогодні загальна встановлена потужність концентрованих сонячних систем у світі становить 4400 МВт [5-8]. Ще один приклад використання концентрації сонячного світла показано на рис. 1.9.

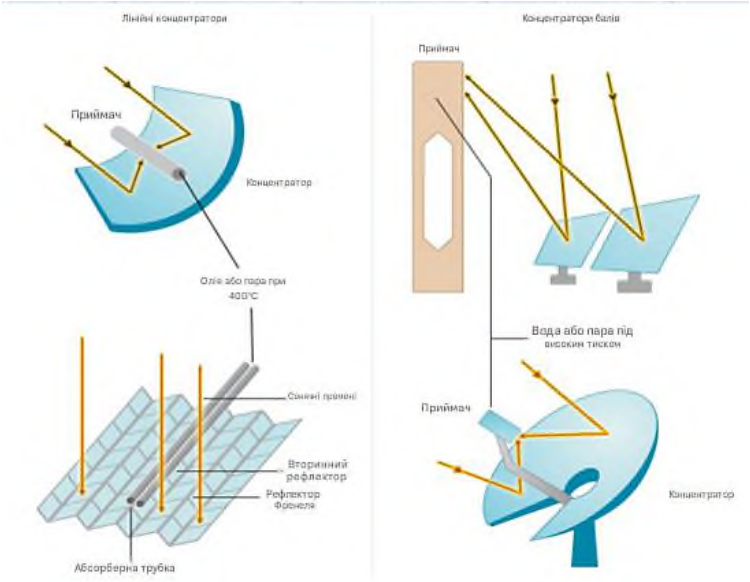


Рисунок 1.9 – Використання моделы концентрації сонячного світла

Окрім проблеми вартості, ще однією проблемою, що виникає (як і в разі інших альтернативних джерел енергії), є необхідність накопичення енергії та її використання в години, коли сонячне випромінювання відсутнє. У будь-якому разі, першочерговим завданням є продовження наукових досліджень [8].

Водночас необхідно приділяти увагу питанням проектування систем комплексної підготовки виробництва комплексів сонячного енергопостачання [7, 8].

Загальна схема етапів підготовки виробництва показана на рис. 1.10.

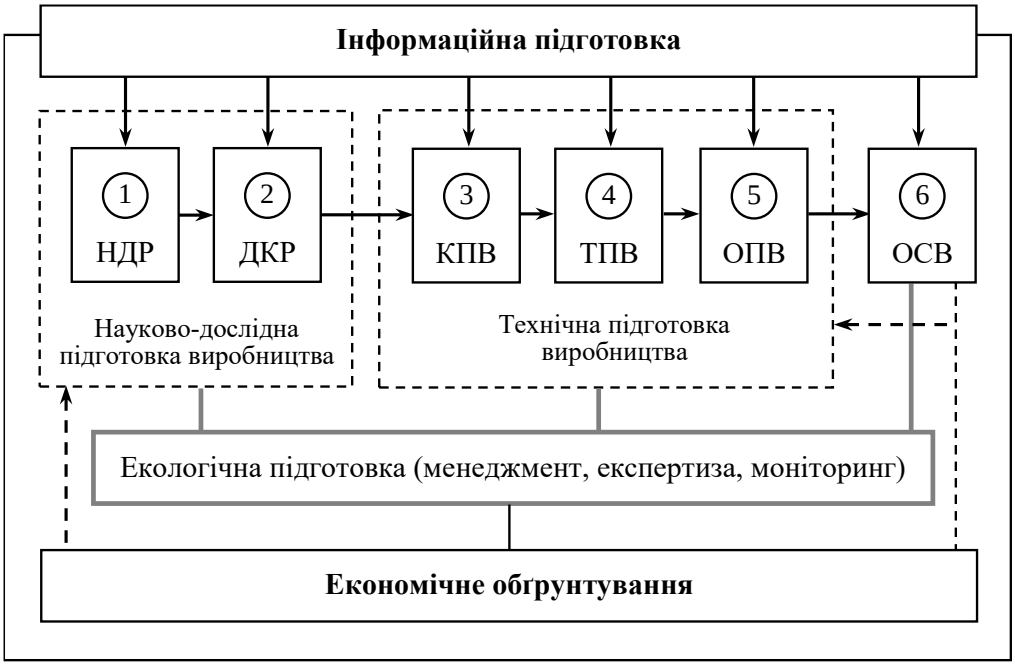


Рисунок 1.10 – Загальна схема етапів підготовки виробництва систем сонячного енергопостачання

Розглянемо детальніше системи комп'ютерної підтримки і забезпечення процесу розробки систем сонячного енергопостачання. До таких систем, перш за все необхідно віднести програмні комплекси для моделювання і автоматизації підготовки виробництва – System Advisor Model (SAM) [7, 8]. Системи виконуються як комплекси автоматизації проектування (САПР).

Ці комплекси спрямовані на розробку систем сонячного енергопостачання, які забезпечують максимальну ефективність у зборі сонячної енергії завдяки інтеграції передових технологій, таких як датчики сонячного випромінювання та автоматизовані системи керування [8]. Ці системи призначені для впровадження в технічному середовищі з метою ефективного використання відновлюваних джерел енергії та забезпечення практичного рішення економії енергоресурсів. Метою комплексів SAM є оцінка позитивного впливу сонячної енергії на зниження експлуатаційних витрат підприємств, сприяючи більш сталому управлінню енергетичними ресурсами [8].

На рис. 1.11 приведено структуру системи SAM, яка передбачає всі модулі проєктування структури систем відновлюваних джерел енергії [8, 9].

Рисунок 1.11 – Структура системи SAM

За результатами аналізу можна сформулювати загальні вимоги до проєкту. Проєкт об’єднує сонячні панелі, сервомотори, датчики та мікроконтролер для оптимізації збору сонячної енергії [8]. Система, що автоматично налаштовується, забезпечує надійне джерело енергії. Проєкт сприяє впровадженню сталого розвитку та зниженню експлуатаційних витрат. Впровадження сонячного енергопостачання створює модель, яку можна повторити в майбутніх проєктах у сфері технологій та сталого розвитку [9]. Крім того, проект застосовує науковий метод для нових ідей, що сприяють поєднанню наукових знань та технологічного прогресу, щоб таким чином впливати на промислову та соціальну сферу.

1.4. Дослідження існуючих комплексів автоматизації проєктування систем відновлюваних джерел енергії

Важливим аспектом, який слід враховувати при проектуванні сонячних водонагрівачів, є режим використання гарячої води, оскільки він може змінюватися під впливом різних факторів [8]. Наприклад, у сім'ї, де всі члени працюють, гаряча вода використовується на початку або наприкінці дня; натомість у спільній ванній кімнаті гаряча вода використовується протягом усього дня, тобто вона споживається одразу після її нагрівання. Ці два способи використання гарячої води називаються відповідно «пунктовою системою» та «безперервною системою» [9]. Безперервна система є ефективнішою та менш витратною, ніж пунктова; це зумовлено, в основному, двома умовами: у безперервній системі використовується накопичувальний бак меншої місткості, а температура води, що зберігається, є нижчою, завдяки чому втрати тепла значно зменшуються [9].

Цієї стратегії дотримується більшість розробників систем сонячного енергопостачання. На рис. 1.12 показаний відомий виробник сонячних систем – Venman [9].

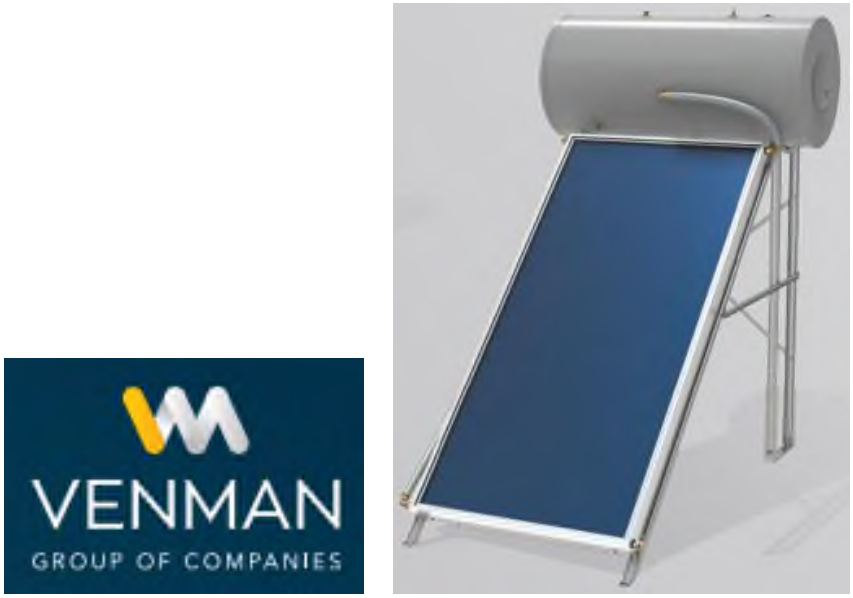


Рисунок 1.12 – Фірма Venman – розробник систем сонячного енергопостачання

При проєктуванні обов’язково береться до уваги наступне. За статистичними даними, середнє споживання в сім’ї становить від 35 до 50 літрів на добу на одну особу. Якщо додати споживання води пральною машиною та посудомийною машиною, якщо вони підключені до сонячної системи, то на кожну з них (на один цикл прання) потрібно приблизно 20 літрів на добу [9]. Щоб повною мірою використовувати можливості сонячної системи, гарячу воду слід використовувати переважно протягом дня, щоб система мала можливість безперервно готувати гарячу воду, поки світить сонце, підтримуючи свою ефективність на максимальному рівні [9, 10]. Таким чином можна сформулювати вимоги до систем сонячного енергопостачання. В таблиці 1.1 наведені технічні вимоги до систем сонячного енергопостачання.

Таблиця 1.1 – Технічні вимоги до систем сонячного енергопостачання

Атрибут	Значення
Матеріал внутрішнього контейнера	Листовий метал постійного струму в робочій ємності (EN 10130/2006) Постійний струм в оболонці (теплообміннику) (EN 10130/2006)
Внутрішній антикорозійний захист	a) рідка емаль (EN4753-3), безпечна для здоров'я населення (51032 та EN 1388-2) та b) магнієвий анод (EN 12438)
Зварка	MIG - дугове зварювання розплавним дротом у середовищі інертного газу
Ізоляція	Твердий поліуретан 48 кг/м^3 (53420), самозатухаючий (EN4102)
Максимальний робочий тиск внутрішньої ємності	10 бар
Випробувальний тиск внутрішньої судини	15 бар (EN 12976-1, 4.1.6)
Максимальний робочий тиск кожуха (теплообмінника)	3,2 бар
Випробувальний тиск мантії (теплообмінник)	5 бар (EN 12976-1/2006, 4.1.6)
Максимальна робоча температура внутрішнього бака	95^{o}C
Зовнішнє покриття	Попередньо пофарбований оцинкований листовий метал, 0,5 мм (EN 10204/2.2)
Іноземні інвестиції	За бажанням

Готова система розроблена, випробувана та реалізується виробником, а завдання проектувальника або технічного спеціаліста полягає у виборі найбільш підходящої системи та її адаптації до конкретного житлового будинку чи будівлі, однак вони не беруть участі у розробці її внутрішньої конструкції [9, 10]. У разі індивідуальної сонячної системи необхідна участь досвідченого фахівця, який розробить повну систему, підбере всі компоненти та розрахує розміри внутрішнього контуру. Далі покажемо інформацію, необхідну для повного проєкту сонячної установки — як готової, так і індивідуально розробленої — яка має бути визначена протягом усього процесу проектування [9, 10]. У таблиці 1.2 наведено основні технічні характеристики одного з класів водонагрівачів.

Таблиця 1.2 – Паспортні дані одного класу водонагрівачів [10]

Параметр	Тип1	Тип2	Тип3	Тип4	Тип5	Тип6
Тип ECO - FLS	130^0C	160^0C	190C	260^0C	320^0C	550^0C
Повна місткість (л)	120	145	210	255	325	525
Довжина контейнера (мм)	1058	1330	1330	1565	1805	1737
Діаметр (мм)	500	500	580	580	580	750
Діаметр резервуара (мм)	400	400	480	480	480	640
Ємність (л)	6	8	9	12	19	24
Діаметр фланця (мм)	140	140	140	140	140	140
Вага (кг)	49	60	70	84	102	132

На рис. 1.13 проілюстровано варіант накопичувального бака водонагрівача в розрізі [8].

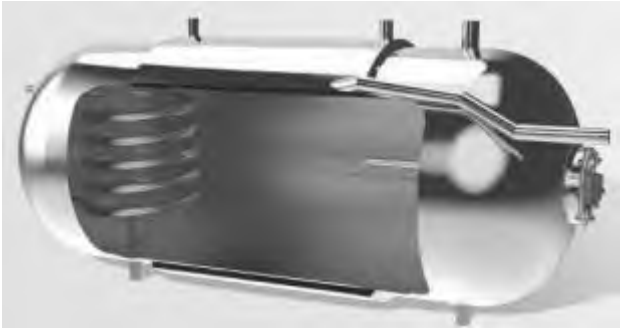


Рисунок 1.13 –Бак для водонагрівання

Колектори слід перевіряти на наявність витоків у місцях з'єднання труб, тріщини в скляних панелях, пошкодження автоматичних запобіжних клапанів, старіння пластикових матеріалів та доливання теплоносія [9]. Також слід перевірити гідравлічний контур.

Усі зазначені положення реалізуються в програмних комплексах для проектування сонячних водонагрівачів. На рис. 1.14 показано приклади застосунків для розробки водонагрівачів.

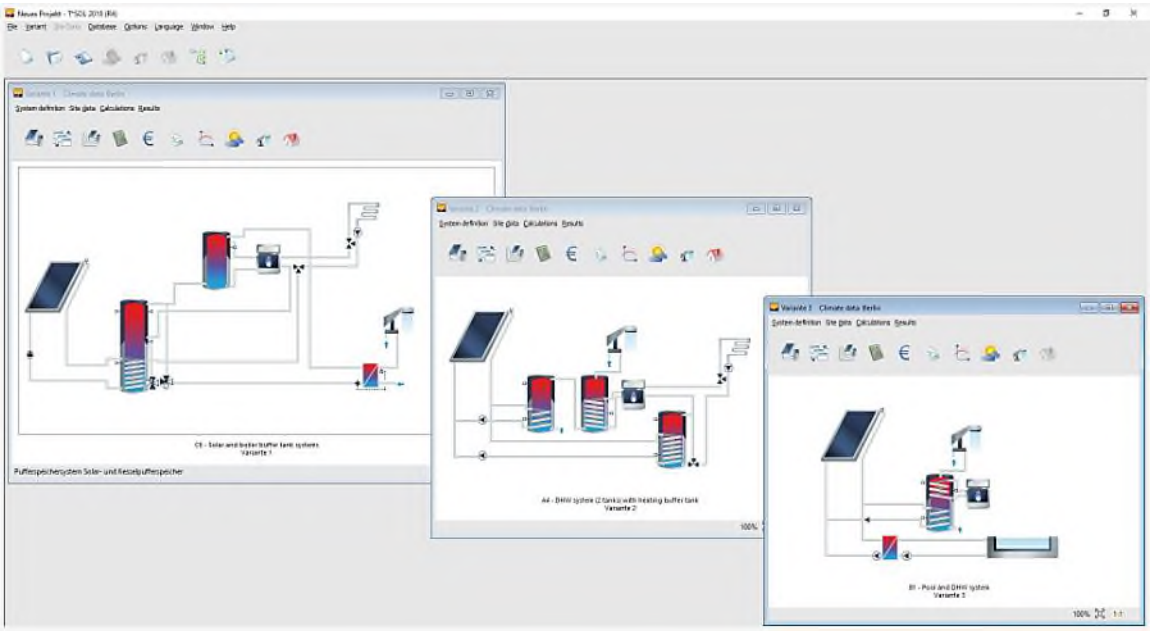


Рисунок 1.14 – Застосунок для розробки водонагрівачів

На рис. 1.15 приведений програмний застосунок Polysun, призначений для розробки солярних панелей для нагрівання води на будівлях і спорудах [10]. Тут враховуються площі споруд, також орієнтація їх до Сонця. Графічний інтерфейс забезпечує об’ємне моделювання [9, 10].

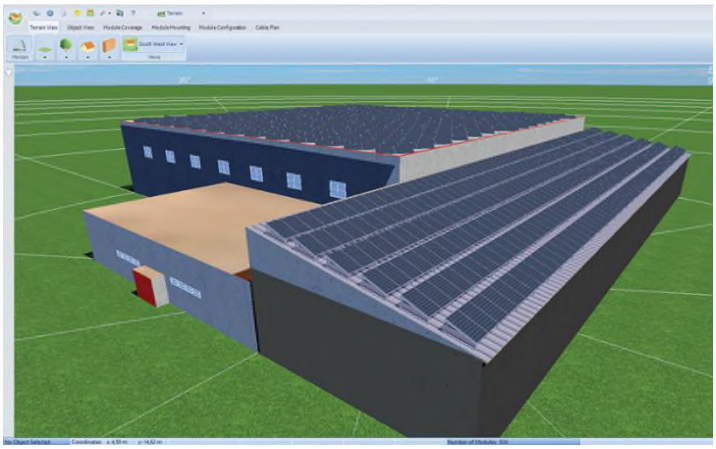


Рисунок 1.15 – Програма для розробки Polysun

Далі приведено систему, яка призначена для проектування коплексів сонячного енергопостачання і проведення математичних розрахунків та інтеграції статистичних даних [10]. Зовнішній вигляд цього застосунку показано на рис. 1.16.

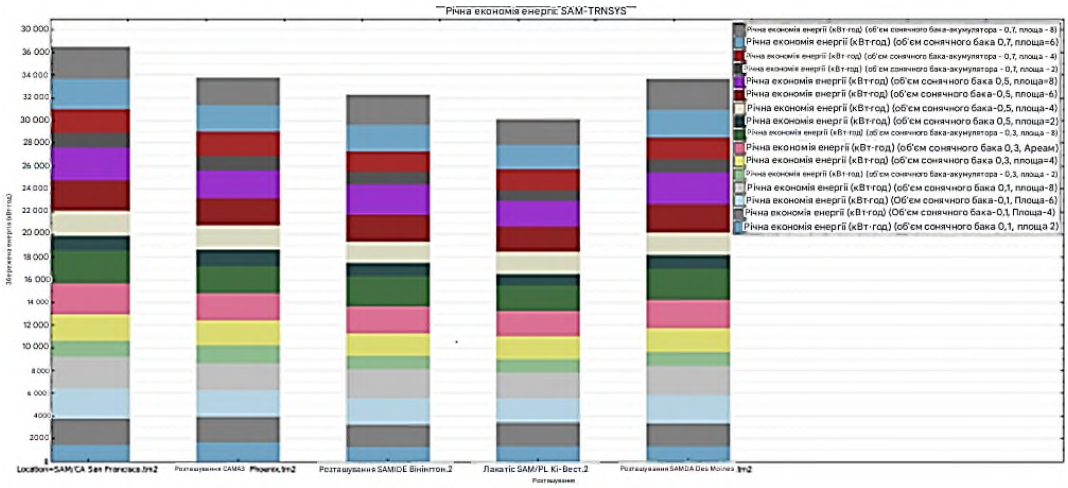


Рисунок 1.16 – Застосунок для аналізу методів збереження енергії

Як показав аналіз, існує два основних напрямки використання систем сонячного підігріву. Це – виробництво гарячої водопровідної води і опалення приміщень [10]. Останніми роками набирає популярності також використання сонячної теплової енергії для підігріву води у басейнах, а в найближчому майбутньому передбачається також використання сонячної теплової енергії для сонячного кондиціонування [10]. З цією метою розробляються різні програмні комплекси.

Приклад такого комплексу приведений на рис. 1.17.

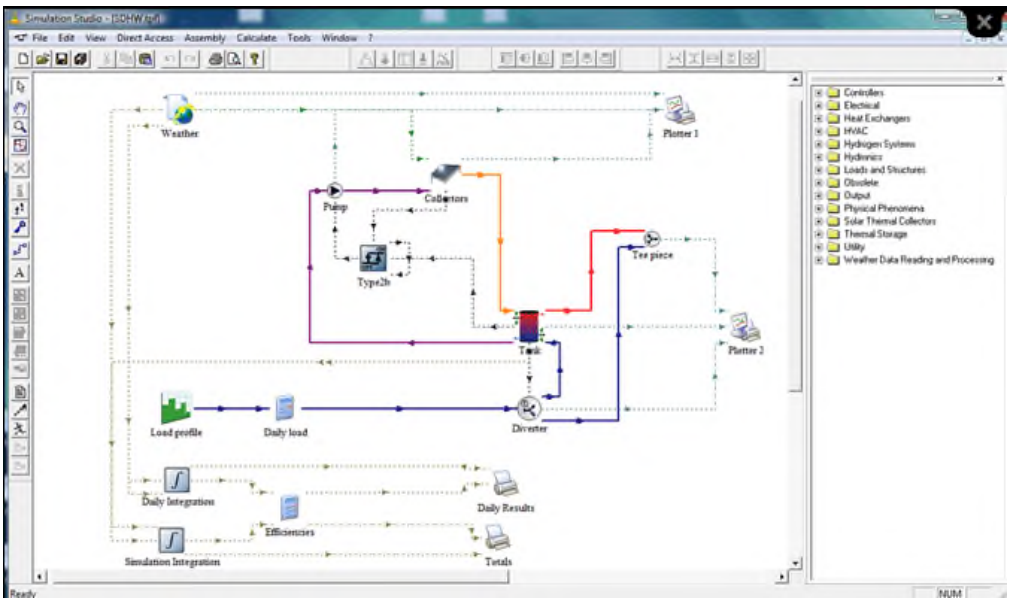


Рисунок 1.17 – Застосунок для моделювання роботи систем сонячного енергопостачання

В цих моделях особливої уваги потребують елементи безпеки експлуатації – підсистеми безпеки, регулювання та контролю. Вони включають розширювальний бак, запобіжний клапан, вентиляційний клапан та/або газовідвідник, сонячний контролер. У сучасних установках частини вищезазначених підсистем інтегровані в один «багатофункціональний» пристрій — сонячну станцію або контролер [11].

1.5. Завдання розробки автоматизованого програмного комплексу проєктування систем відновлюваних джерел енергії в умовах нестабільного електропостачання

Сонячні теплові системи дозволяють заощадити в будівлях значні обсяги енергії — до 95 % для готелів і до 60 % для житлових будинків. Сонячні теплові системи окупаються приблизно за 2–7 років [1, 10]. При цьому термін їхньої експлуатації становить понад 25 років. Витрати на технічне обслуговування мінімальні. Їх можна використовувати для гарячого водопостачання, опалення приміщень та басейнів, а також для кондиціонування. Тобто вони можуть покривати 85 % енергетичних потреб

будівлі. Їх можна використовувати в будь-якому типі будівлі, навіть у пам’ятках архітектури, кам’яних, дерев’яних тощо. Адже виробники сонячних колекторів вже мають усі необхідні рішення. Їх можна поєднувати з усіма існуючими системами будівлі, наприклад: мазутними та газовими котлами, тепловими насосами, дров’яними та біомасовими котлами, геотермальними системами тощо [10, 11]. Сонячна теплоенергетика — це технологія, яка робить нас незалежними від палива. Навіть якщо в майбутньому вартість усіх джерел теплової енергії зросте, та частина енергії, яку ми отримуємо від Сонця, завжди буде безкоштовною.

Активні сонячні системи — це механічні конструкції, здатні збирати сонячну енергію, перетворювати її на корисну (теплову, холодову або електричну), накопичувати її частину та розподіляти для використання [10, 11]. Найпоширенішими активними сонячними системами є сонячні колектори для нагріву води та фотоелектричні панелі (різновид сонячного колектора) для виробництва електроенергії невеликої потужності [11]. Також до цієї категорії належать вакуумні сонячні колектори, їх поєднання з абсорбційними холодильними установками (Absorption Chillers) для забезпечення потреб у охолодженні, а також сонячні колектори високої ентальпії для безпосереднього виробництва електроенергії з використанням парових турбін або органічних циклів [10, 11].

На підґрунті аналізу проблеми поставлено нанступні завдання кваліфікаційної роботи:

– розробка математичних, структурних моделей і алгоритмів програмного комплексу проєктування відновлюваних джерел енергії;

– математичні моделі параметрів автоматизованого комплексу проєктування систем відновлюваних джерел енергії;

– розробка структурної моделі автоматизованої системи проєктування систем відновлюваних джерел енергії;

– функціональна модель застосунку автоматизації проєктування систем відновлюваних джерел енергії;

– синтез структури бази даних автоматизованого застосунку створення систем відновлюваних джерел енергії;

– проєктування основних алгоритмів функціонування програмних модулів автоматизованої системи;

– проєктування елементів інтерфейсу користувача програмного комплексу системи автоматизації;

– розробка програмних блоків автоматизованої системи проєктування відновлюваних джерел енергії в умовах нестабільного електропостачання;

– розробка методів класів програмних модулів автоматизованої системи проєктування систем відновлюваних джерел енергії;

– розробка програмних модулів автоматизованої системи автоматизації проєктування систем відновлюваних джерел енергії;

– реалізація бази даних застосунку проєктування систем відновлюваних джерел енергії;

– розробка користувацького інтерфейсу системи автоматизації проєктування систем відновлюваних джерел енергії;

– розробка модулів візуалізації результатів роботи програмного комплексу системи автоматизації.

2 РОЗРОБКА МАТЕМАТИЧНИХ, СТРУКТУРНИХ МОДЕЛЕЙ І АЛГОРИТМІВ ПРОГРАМНОГО КОМЛЕКСУ ПРОЄКТУВАННЯ ВІДНОВЛЮВАНИХ ДЖЕРЕЛ ЕНЕРГІЇ

2.1. Математичні моделі параметрів автоматизованого комплексу проєктування систем відновлюваних джерел енергії.

На ефективність сонячних систем нагріву води впливають два фактори. Перший фактор — це рівень сонячного випромінювання у місці встановлення водонагрівача, а другий — конструкція сонячного колектора. Поєднання цих двох факторів визначає вартість системи. Ці витрати значно знижуються в районах з високим рівнем сонячної активності [11].

У кваліфікаційної роботі наведено деякі математичні критерії проектування систем нагрівання води. Ці критерії ґрунтуються на фізичних процесах та досвіду, накопиченому розробниками під час оцінки прототипів сонячних водонагрівачів, на теоретичних дослідженнях та на розробці програм для розрахунку сонячних водонагрівачів [11, 12].

Основною є формула (2.1).

$$Q_u = AGC_p(T_0 - T_i), \tag{2.1}$$

де AG — витрата води на 1 м^2 площини колектору; C_p — теплоємність води; T_0 — температура води в колекторі.

Підсумки розрахунків можна подати у вигляді графіку функції ефективності колектору від компонента $\frac{(T_i - T_a)}{I_T}$ [12].

В якості основи представлення формули лежить залежність (2.1). Тоді можна отримати таку формулу ефективності колектору:

$$\eta = \frac{Q_u}{I_T A} = F_R(\tau\alpha)_n - F_R U_L \frac{(T_i - T_\alpha)}{I_T}. \tag{2.2}$$

На основі виразу (2.2), можна відобразити ефективність нагрівача, як приведено на рис. 2.1 [12].

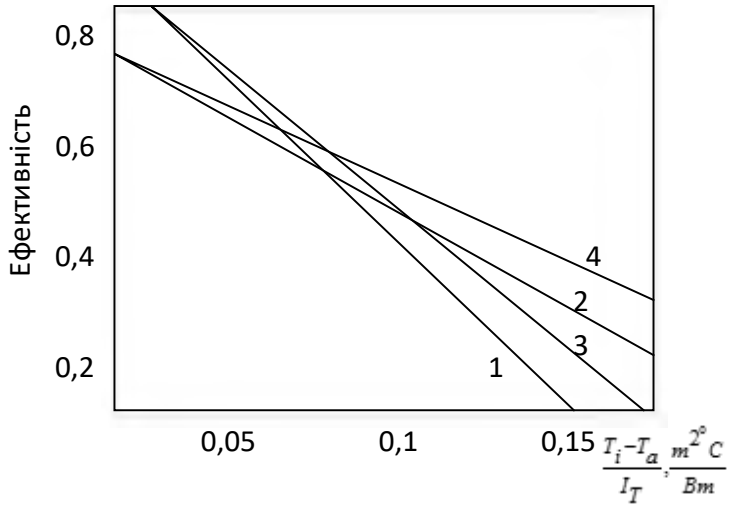


Рисунок 2.1 – Ефективність нагрівачів різних видів з рідинним теплоносієм

Дані заносяться на графік в залежності від $(T_m — T_a)I_T$ або $(T_0 — T_a)I_T$, де T_m — температура води в обладнанні [11, 12]. Для цього значення коефіцієнта і ординати перетину з вертикальною віссю рівняння необхідно помножити на *K:*

$$K = \left\{ \begin{array}{l} \dfrac{GC_p}{\left(GC_p - \dfrac{k}{2}\right)}, \eta = f\left(\dfrac{T_m - T_a}{I_T}\right); \\ \dfrac{GC_p}{\left(GC_p - k\right)}, \eta = f\left(\dfrac{T_0 - T_a}{I_T}\right). \end{array} \right\} \qquad (2.3)$$

Маємо [11, 12]

$$\begin{array}{l} F_R U_L = -Kk; \\ F_R(\tau\alpha)_n = Kb. \end{array} \qquad (2.4)$$

Відношення $\frac{F_R'}{F_R}$ залежить від $\frac{F_R U_L}{Gc_p}$ і $\frac{AGc_p}{\varepsilon_c C_{min}}$. Ця залежність приведена на рис. 2.2.

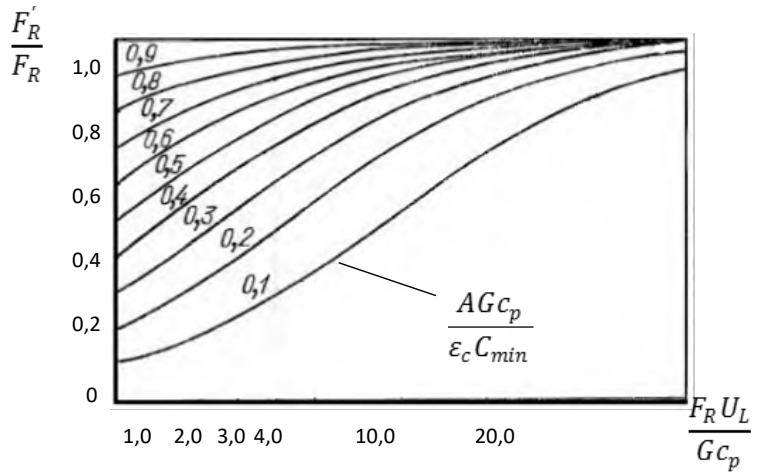


Рисунок 2.2 - Залежності впливу теплообмінника

Тепер врахуємо географічну орієнтацію ємності. Денний прихід тепла за місяць на поверхню H_m дорівнює [12]:

$$H_T = RH, \tag{2.5}$$

де H –денний обсяг випромінювання за місяць на горизонтальну площину; R –денні витрати радіації на похилу і горизонтальну поверхні.

$$R = \left(1 - \frac{H_d}{H}\right) R_b + \frac{H_d}{H} \frac{1 + \cos s}{2} + \rho \frac{1 - \cos s}{2}, \tag{2.6}$$

де H_d - прихід випромінювання на горизонтальну поверхню; $\frac{H_d}{H}$ - відношення витрат прямої радіації на похилу і горизонтальну поверхні; s — кут нахилу колектору до горизонта; ρ — відбивна здатність землі.

Баланс енергії системи сонячного енергопостачання

$$Q_T - L + E = AU, \tag{2.7}$$

де Q_T - теплопродуктивність установки; L – сума місячних навантажень обігріву та гарячого водопостачання [12]; E – загальна кількість енергії отримане протягом місяця від дублюючого джерела; AU – зміна кількості енергії в акумулюючої установці [12].

Математичні моделі в кваліфікаційної роботі були досліджені з технічних джерел з проєктування сонячних систем [12]. Як приклад розглянуті наступні умови. Розглянута точкова система, оскільки гаряча вода

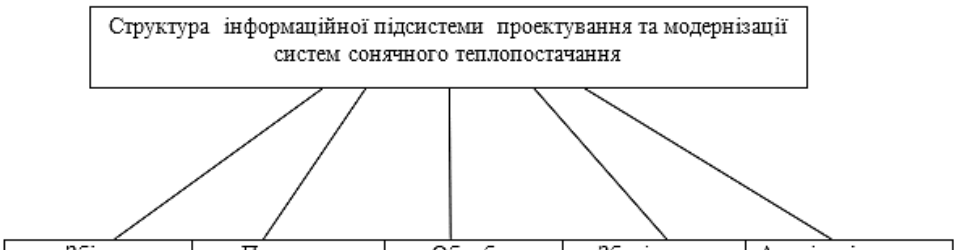
використовуватиметься наприкінці дня; для цього врахований за основу уявний сонячний колектор із загальним ККД 45 %. Продовжуючи розв’язання задачі, розглянуто дві системи: одну для періоду з жовтня по квітень (місяці з високим рівнем сонячної радіації) та іншу — для періоду з травня по вересень (місяці з низьким рівнем сонячної радіації) [13]. Потім, порівнявши площі колекторів кожної системи, обрано систему з більшою площею.

2.2. Розробка структурної моделі автоматизованої системи проєктування систем відновлюваних джерел енергії

Колекторна поверхня, поглинаючи сонячну енергію, нагріває рідину (розчин антифризу), що циркулює в гідроелементі. Ця рідина, нагріваючись, стає легшою і прямує до бойлера, нагріваючи воду, що міститься в ньому. Циркуляція рідини в колекторах відбувається без зусиль і природним чином (термосифонний потік) [13].

Факторів, що впливають на температуру води, яка подається сонячним водонагрівачем, є чимало, і їхні значення коливаються залежно від пори року, часу доби та місця розташування. З огляду на те, що сонячний водонагрівач є системою, яка піддається впливу погодних умов, основними параметрами, що впливають на його ефективність, є температура води у водопровідній мережі, доступна сонячна енергія та температура навколишнього середовища. Температура води у водопровідній мережі не є стабільною протягом року, оскільки взимку вона набагато нижча, ніж влітку [12, 13]. Взявши за орієнтир 45 °C як задовільну температуру води для побутового споживання, щоб задовольнити потреби житлового будинку, на основі статистичних даних можна зробити висновок, що взимку температура водопровідної води має підвищитися приблизно на 35 °C, на відміну від літнього періоду, коли вона має підвищитися на 20 °C [13].

На рис. 2.3 показано структуру системи для розробки систем сонячного енергопостачання [13].



Збір та реєстрація інформації	Передача інформації	Обробка інформації	Зберігання, пошук інформації	Аналіз, підготовка прийняття рішення
Операції				
Збір	Блок каналів системи передачі даних	Введення інформації в систему	Зберігання	Аналіз вихідної інформації
Передача	Перетворення з цифрової форми в наочну	Контроль введення структурованих даних	Запит	Моделювання функцій базових блоків системи
Реєстрація у машинному носії	Передача інформації до користувача	Обробка інформації (розрахунки)	Пошук	Компоновка даних и прогноз проектних рішень
введення в інформаційну систему	Виведення контрольних повідомлень користувачеві	Введення і контроль введення даних	Контроль пошуку	Аналіз і коректування даних, що були розраховані
Контроль введення і реєстрації	Контроль виведення інформації	Відображення результатів у вигляді звітів	Видача та актуалізація даних	Підготовка прийняття проектних рішень

Рисунок 2.3 – Загальна структура системи розробки модулів сонячного енергопостачання

Принципова схема може бути проілюстрована наступним чином. Схема складається з центрального резервуара-акумулятора з різними температурними рівнями для опалення приміщень, виробництва гарячої води та тепла для роботи чилера. Цей резервуар нагрівається як сонячними колекторами, так і резервною системою опалення [13]. Для нагрівання накопиченої води сонячними колекторами є триходовий клапан, через який повернення води до колекторів вибирається з середини або знизу резервуара. Це дозволяє швидше досягти необхідного рівня температури в резервуарі-акумуляторі для роботи чилера [13].

Влітку чилер живиться енергією з резервуара. Для виробництва гарячої води використовується зовнішній пластинчастий теплообмінник. Взимку енергія в резервуарі використовується як для опалення приміщень, так і для виробництва гарячої води.

Схема адаптована до варіанта, де резервний котел не може живити

накопичувальний бак сонячного контуру [11, 13]. З цієї причини резервний котел підключено послідовно з накопичувальним баком. Описані вузли показані на рис. 2.4.

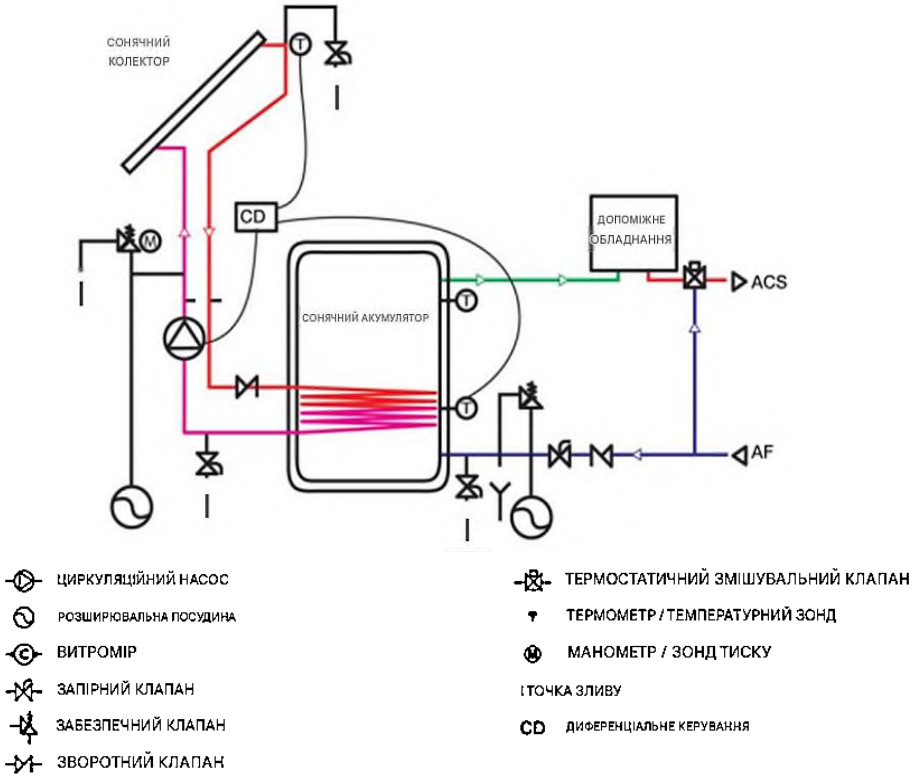


Рисунок 2.4 – Окремі вузли схеми енергопостачання

Використання стельової системи розподілу охолодження [12, 13]. Стельова система перевершує системи фанкойлів з точки зору ефективності чилера завдяки вищому рівню температури в контурі холодної води. Однак вартість цієї системи вища, ніж у інших, тоді як її експлуатація також є критично важливою в офісних будівлях та житлових будинках під час опалювального сезону [13].

2.3. Функціональна модель застосунку автоматизації проєктування систем відновлюваних джерел енергії

Енергія, необхідна для опалення, отримується за допомогою сонячного повітряного нагрівача, встановленого на даху. Найпростіший і найекономічніший варіант — генерувати тепле повітря та подавати його

безпосередньо в приміщення, але його недоліком є те, що вночі приміщення не прогрівається [14]. Ця система призначена переважно для приміщень, які використовуються лише вдень, наприклад, шкіл, дитячих садків та офісів. У цьому випадку важливо зорієнтувати нагрівач на схід, щоб він сприяв нагріванню приміщення рано вранці. Оскільки тепле повітря легше і піднімається вгору, температурний профіль у приміщенні не є дуже сприятливим [14]. Таким чином, функціональна схема з контролем температури в загальному вигляді показана на рис. 2.5.

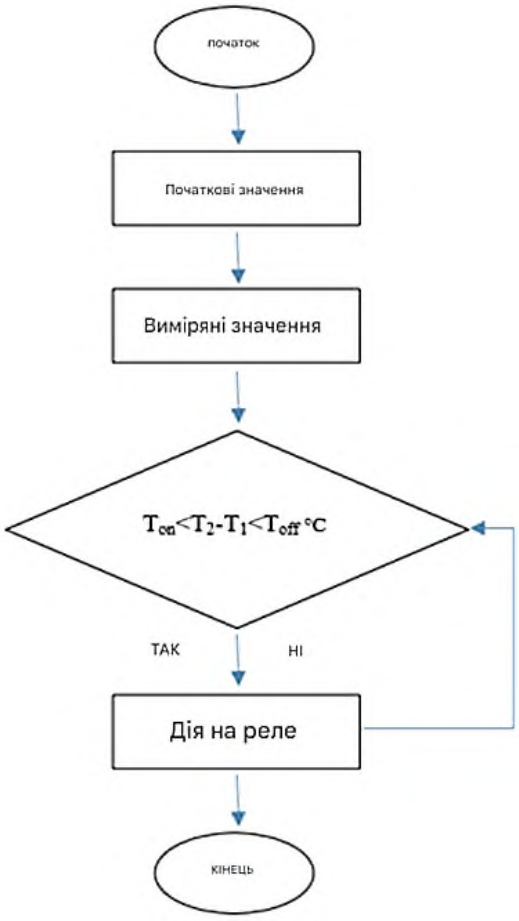


Рисунок 2.5 – Принципи функціонування системи водопідігріву

Опишемо функціональну модель роботи застосунку.

Інтегральна складова системи підсумовує помилку керування за часом. В результаті навіть невелика помилка призведе до повільного зростання загальної складової [14]. Інтегральна реакція буде постійно зростати з часом, доки помилка керування не дорівнюватиме нулю, тому це призводить до зведення помилки в стаціонарному режимі до нуля. Помилка в стаціонарному

режимі — це кінцева різниця між змінною процесу та заданим значенням [14]. Явище, яке називається інтегральним накопиченням, виникає, коли інтегральна дія системи призводить до насичення регулятора, не приводячи при цьому сигнал похибки до нуля. Інша складова призводить до зменшення вихідного сигналу керування, якщо змінна процесу швидко зростає. Диференціальна реакція пропорційна швидкості зміни змінної процесу. Збільшення параметра часу призведе до того, що система керування буде сильніше реагувати на зміни похибки та збільшить швидкість загальної реакції системи керування[13, 14]. У більшості практичних систем керування використовується дуже мале значення коефіцієнту, оскільки диференціальна складова є дуже чутливою до шуму в сигналі змінної процесу. Якщо сигнал зворотного зв’язку від датчика містить шум або швидкість контуру керування є занадто низькою, диференціальна складова може призвести до нестабільності системи керування [14].

Отже, можна сказати, що керування, під час якого виконуються заходи з регулювання системи водопідігріву, необхідні для її оптимальної роботи. Ці заходи з регулювання полягають у зміні кута нахилу сонячних модулів залежно від пори року, в якої вони експлуатуються. Експлуатаційне керування проводитиметься постійно і автоматично, що дозволить уникнути відправлення бригад технічних фахівців для виконання цих робіт [12, 14].

У фізичному масштабі регулювання швидкості обертання двигуна постійного струму за допомогою PID вимагає наявності цифрового енкодера, підключеного до насосу відповідно до похибки, що видається регулятором, або, іншими словами, можна застосувати стратегію диференціального регулювання [14]. Ця стратегія виявляється досить корисною під час керування механічним водяним насосом. Використання PID-регуляторів, безперечно, є більш цікавим завдяки їхній здатності адаптуватися до будь-якої сонячної системи, однак через апаратні обмеження диференціальний регулятор також може бути корисним [14]. Загальна схема функціонування регулятора приведена на рис. 2.6.

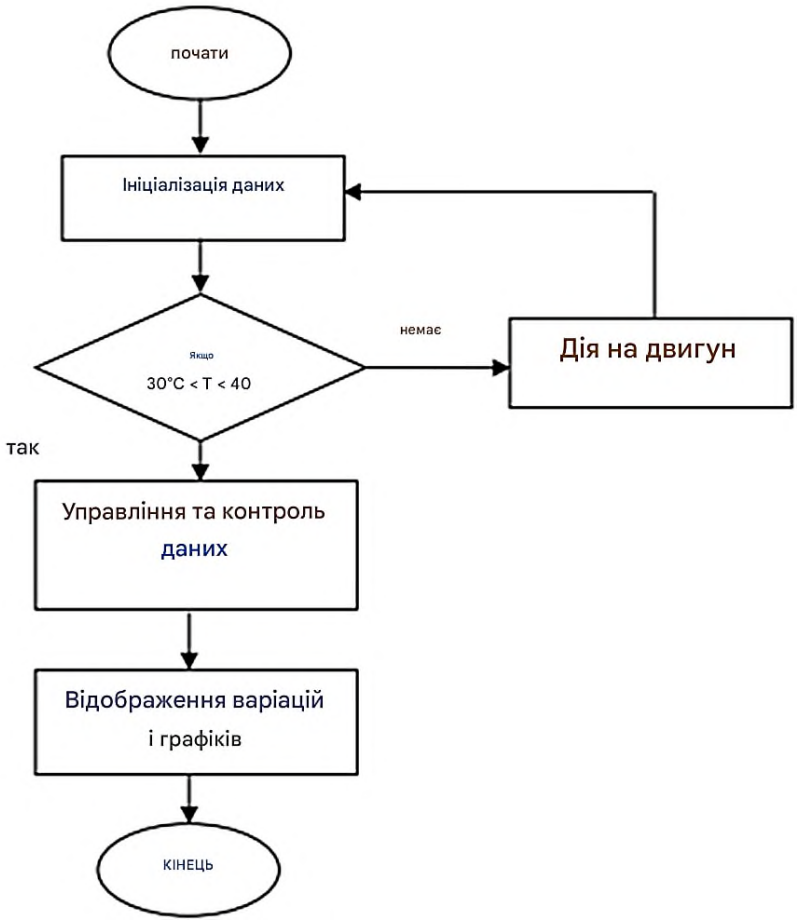


Рисунок 2.6 – Алгоритм функціонування регулятора

Для налаштування ПІД-регулятора можна використовувати різні методи, такі як ручне налаштування або налаштування за допомогою моделювання [14, 15]. Ручне налаштування полягає у зміні значень коефіцієнтів для отримання кращої реакції системи: змініть значення *Kp* таким чином, щоб система якомога швидше досягала заданого значення. Змінюйте *Ti*, щоб отримати точну реакцію за короткий проміжок часу. Змінюйте *Td*, щоб зменшити коливання системи [15]. Цей метод може бути корисним у деяких випадках, однак він є трудомістким і не дуже ефективним у великих системах. Другій метод, визначає граничне або критичне підсилення (*Ku*) та граничний період шляхом регулювання підсилення регулятора *Kp* доти, доки система не почне стійко коливатися, при цьому постійна часу інтегрального члена залишається нескінченною, а постійна часу похідного члена — нульовою [15].

2.4. Синтез структури бази даних автоматизованого застосунку створення систем відновлюваних джерел енергії

При синтезі структури бази даних (БД) автоматизованого застосунку перш за все необхідно розробити схему бази даних. Така схема має назву E/R-діаграма [14, 15]. Розробка схеми E/R, яка відображає семантику заданого проблемного середовища, – це творчий процес, для якого не існує єдиної, заздалегідь визначеної процедури. Однак, можна дотримуватися низки рекомендацій або евристик, які можуть допомогти в процесі проектування. Ці рекомендації не є універсально застосовними правилами; вони підходять у деяких випадках, а в інших – ні. Конструкція, яка розширює семантику, що міститься у зв'язку, є обмеженням кардинальності. Максимальна та мінімальна потужності сутностей, що беруть участь у зв'язку, визначаються як максимальна та мінімальна кількість екземплярів однієї сутності, які можуть бути пов'язані з одним екземпляром іншої або іншими сутностями, що беруть участь у зв'язку [15, 16].

Схему БД показано на рис. 2.7.

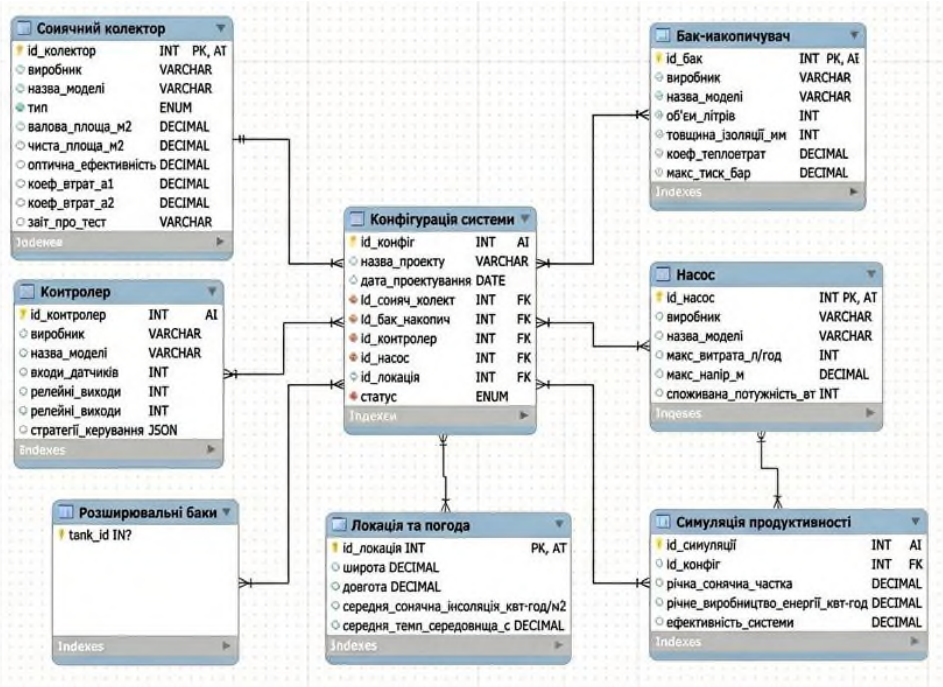


Рисунок 2.7 – Схема бази даних

Система керування базами даних (СКБД) це програмне забезпечення (набір програм) системи керування базами даних для створення та використання бази даних [16]. Схема доступу до даних з клієнтського застосунку приведена на рис. 2.8 [18].

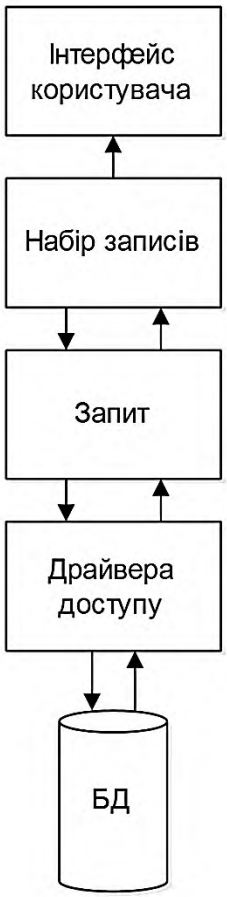


Рисунок 2.8 - Схема модулю керування базою даних

Деякі пропозиції надають перевагу включенню багатозначного атрибута як сутності в схему E/R, тоді як інші вважають за краще представляти його як атрибут [17]. Незалежно від того, чи є атрибут простим чи складеним, якщо відомо, що він матиме обмежену та відносно невелику кількість входжень, то він буде частиною сутності, яку він описує (за умови, що концепція, яку він представляє, не пов'язана з іншими сутностями в схемі E/R) [18].

Центральним елементом реляційної моделі є відношення (Relation). Відношення має ім'я, набір атрибутів, що представляють його властивості, та набір кортежів, що містять значення, які кожен атрибут приймає для кожного

елемента відношення. Іншим важливим елементом у моделі є домен (Domain), який, як і відношення, також має свою власну сутність та описує допустимі значення, які може приймати атрибут, визначений над цим доменом [17, 18]. Відношення можна представити як двовимірну таблицю (стовпці – це атрибути відношення, а рядки – кортежі) з одним значенням у кожній комірці, що перетинається. Цей метод представлення називається екстенсіонним, і він розвивається з часом, оскільки на кортежі відношення можуть впливати операції оновлення [18]. Прикладом такого відношення під назвою students може бути розширення (extension). Це відношення складається з атрибутів registration_code, name, city та group_code (назви стовпців таблиці). Кожен рядок таблиці відображає кортеж, що відповідає даним [18].

Спираючись на зазначені принципи побудови схеми бази даних, можна сформулювати і представити основні режими роботи, які показани на рис. 2.9

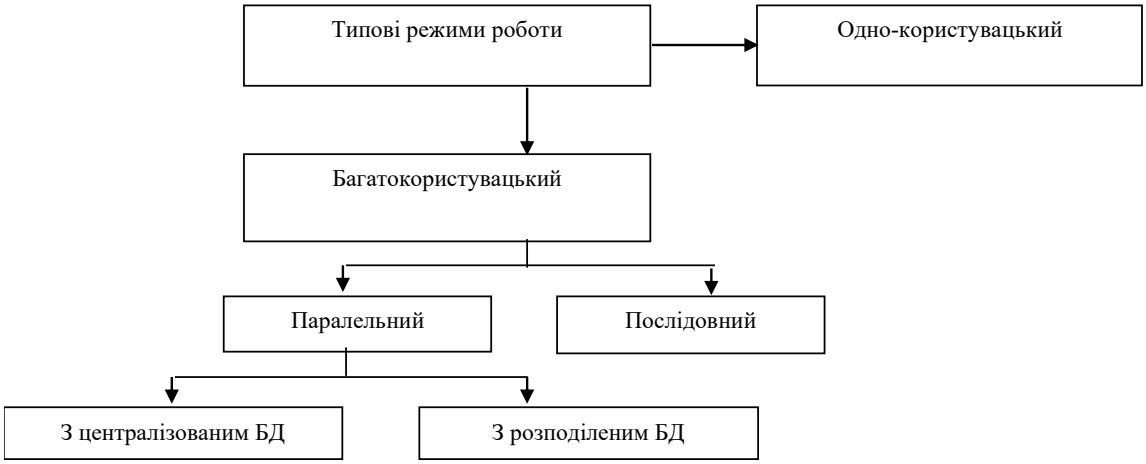


Рисунок 2.9 – Схема режимів роботи с даними

Існує ще один спосіб представлення відношення в реляційній моделі, який називається графом відношення або інтензією, в якому відображаються лише статичні частини відношення. Зокрема, це: назва відношення, назви атрибутів та ім'я домену, для якого вони визначені (хоча останнє зазвичай опускається) [18]. Обмеження зовнішнього ключа, яке також називають посилальною цілісністю, використовується для зв'язування зв'язків у базі даних за допомогою зовнішніх ключів [19].

2.5. Проєктування основних алгоритмів функціонування програмних модулів автоматизованої системи

Алгоритми функціонування програмних модулів автоматизованої системи спираються на механізми керування роботою сонячних теплових установок [19]. Диференціальний терморегулятор забезпечує ефективне використання та керування роботою сонячних теплових або опалювальних систем [19]. Пристрій відрізняється своєю функціональністю та простим, зрозумілим керуванням. Кожна ключова функція пояснюється та пов'язана з відповідною функцією. Окрім пояснювального тексту для вимірювань та налаштувань, меню контролера також містить корисний текст та пояснювальну графіку. Диференціальний терморегулятор можна використовувати для різних конфігурацій сонячної системи [19, 20].

Основні характеристики регулятора наступні. Графіка та текст на дисплеї з підсвічуванням [20]. Легкий доступ до поточних значень. Моніторинг та аналіз системи, наприклад, за допомогою графічної статистики. Великі меню конфігурації з поясненнями. Блокування меню для запобігання випадковим змінам. Повернення до початкових або раніше вибраних значень. Різні додаткові функції доступні та які плануються як опції [20].

Під час першої активації контролера, після встановлення мови та часу, система запитує, чи бажаєте ви скористатися майстром налаштування. Майстер налаштування можна скасувати будь-коли або після повторного доступу до меню спеціальних функцій [20]. Майстер налаштування проведе вас через необхідні налаштування в логічному порядку, пояснюючи кожен параметр на екрані. Натискання клавіші "esc" повертає вас до попереднього параметра для підтвердження або зміни вибору [20]. Повторне натискання клавіші "esc" крок за кроком повертає вас до вибору для скасування налаштування. Нарешті, необхідно перевірити виходи з підключеними навантаженнями та показниками датчика в режимі роботи "Ручний". Потім знову активуйте автоматичний режим роботи [21].

За даними аналізу методів керування системою сонячного підігріву розроблена схема алгоритму роботи всього комплексу застосунку, яка приведена на рис. 2.10.

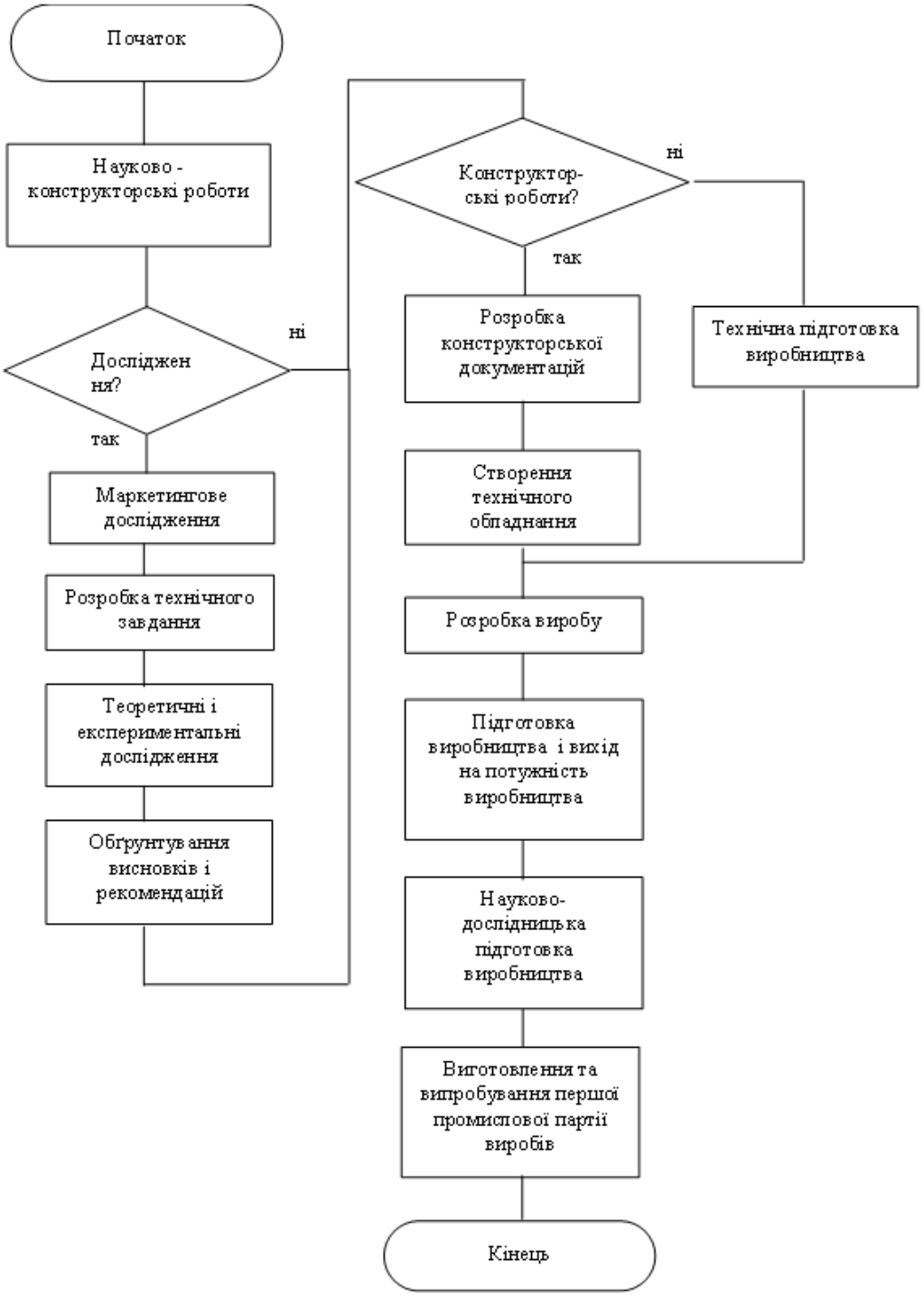


Рисунок 2.10 - Блок-схема роботи комплексу сонячного водопідігріву

На основі наведеного алгоритму покажемо основні етапи та прийоми керування системою сонячного нагрівання для розробки детальних алгоритмів програмних модулів [19]. Коли пристрій керування ввімкнено та в меню

вибрано "автоматичний режим", програмне забезпечення автоматично починає налаштування вхідних та вихідних портів і очікує значення температури керування, яке користувач введе через екран. Потім користувач ініціює вимірювання, натискаючи кнопку на екрані [19, 20]. Якщо користувач не введе значення, пристрій за замовчуванням встановлює 26°C. Потім починається цикл вимірювання та керування; цей цикл відбувається практично миттєво. Датчики температури фіксують температуру на виході з колектора та температуру всередині приміщення, яке потрібно опалювати [20]. Пристрій вмикається та вимикається за допомогою стану "увімк." та стану "вимк.". Ці умови дозволяють пристрою працювати ефективніше, запобігаючи його спонтанне вмикання та вимикання через коливання встановленої температури. Важливо зазначити, що температура всередині приміщення може швидко змінюватися через повітряний потік, що потрапляє в приміщення через отвір вікна або дверей.

Активація вентилятора визначається виконанням двох умов. По-перше, різниця температур між заданим значенням і заданим значенням повинна бути більшою або дорівнювати 10°C [20]. З точки зору логіки керування, ця умова визначає, чи має колектор корисну енергію для подачі в приміщення, яке потрібно обігрівати. Виконання цієї умови вказує на наявність корисної енергії, але вона не враховує поточну температуру приміщення, тобто чи потрібне йому обігрів [20]. Тому додається друга умова, яка порівнює поточну температуру приміщення із заданою температурою. Наприклад, якщо задане значення вважається 26°C, температура опалювального приміщення не повинна перевищувати 26°C. Таким чином, задана температура встановлює комфортну температуру для приміщення. Оцінка теплового стану приміщень проводиться шляхом порівняння його, з одного боку, з температурою на виході з колектора, а з іншого – з контрольною температурою [20]. Тестування датчиків здійснюється миттєво, а дані зберігаються кожну хвилину. Дані зберігаються на SD-карті, щоб оцінити поведінку пристрою. Для остаточної версії пристрою керування зберігання даних не має значення [20]. Незважаючи на те, що вентилятор розташований на виході з колектора, все вищезазначене

вказує на те, що колектор працює за рахунок природної конвекції. Це не є проблемою, оскільки якщо підвищення температури невелике в цьому невеликому опалювальному приміщенні, воно буде непомітним у більших опалювальних площах [20].

Алгоритм встановлення параметрів системи, показаний на рис. 2.11.

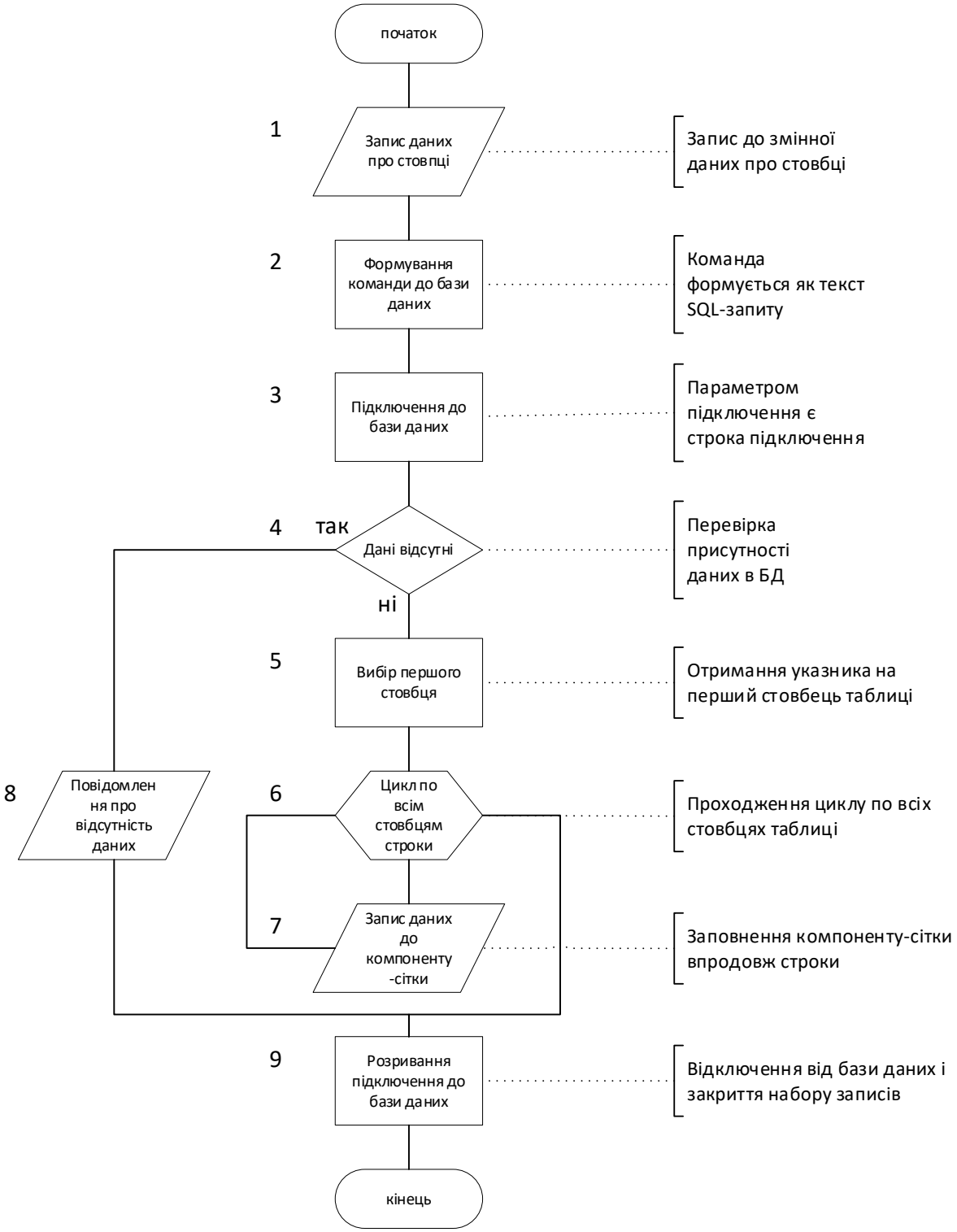


Рисунок 2.11 – Блок-схема алгоритму встановлення параметрів системи

2.6. Проєктування елементів інтерфейсу користувача програмного комплексу системи автоматизації

Проєктування елементів інтерфейсу користувача це галузь програмної інженерії, яка присвячена розробці стандартних та систематичних процесів, які дозволяють розробляти та проектувати графічний інтерфейс програмних продуктів [21]. Ці процеси описують важливі фази розробки продукту та відомі як моделі розробки або життєві цикли програмної системи. Функція цих моделей полягає в забезпеченні основи для розуміння процесу розробки, з конкретними припущеннями та відповідними рекомендаціями щодо дій та співпраці команди дизайнерів інтерфейсу [22]. Проектування розділу UI/UX належить до категорії розробки програмного забезпечення з моделями і, зокрема, у багатьох випадках застосовується модель проектування, орієнтована на людину [21, 22].

Загальна модель меню показана на рис. 2.12.

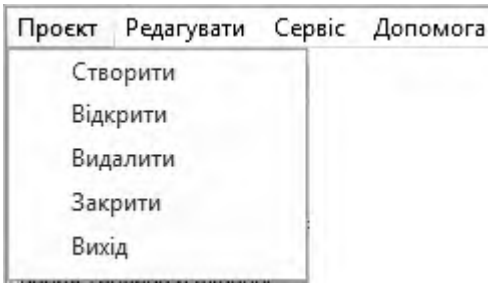


Рисунок 2.12 – Загальна модель меню

Модель людиноцентричного дизайну – це креативний підхід до процесу розробки програмного забезпечення та вирішення проблем, що виникають під час цієї розробки, і на практиці це означає значну участь користувачів у проектуванні та створенні мультимедійних застосунків [22]. Основні причини такого підходу стосуються, головним чином, зручності використання та покращеної ергономіки додатків, які адаптовані до характеристик користувачів, що збираються використовувати застосунок [23]. Більш конкретно, зручність використання додатка є важливою для його прийняття та

успіху, тоді як покращена ергономіка досягається шляхом оптимізації зручності використання пристроїв та інструментів, що є в розпорядженні користувача [22].

Залучаючи користувачів до процесу організації, проектування, розробки та доставки програмного забезпечення, ми можемо отримати кращий користувацький досвід та покращений інтерфейс застосунку [22]. Тож, замість того, щоб проходити всі етапи розробки додатку та зрештою виявляти, що він не відображає аудиторію, до якої він адресований і яка його використовуватиме, ми оцінюємо та адаптуємо застосунок під користувача.

Інформаційна панель з відображенням режимів сонячного підігріву, конструкції нагрівачів показана на рис. 2.13.

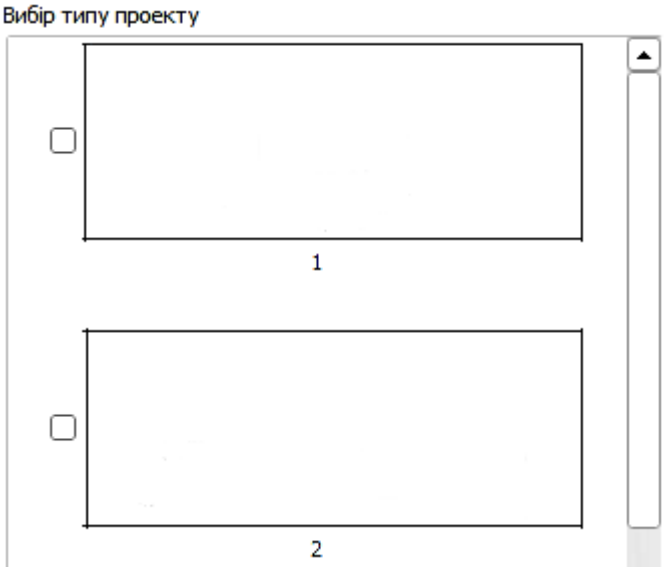


Рисунок 2.13 – Інформаційна панель з відображенням режимів сонячного підігріву

Отже, застосунок пропонує інноваційну ідею, яку користувачі можуть використати для вирішення задачі, або формулює вже реалізовану пропозицію таким чином, щоб вона допомогла користувачеві легше реалізувати бажане завдання та водночас створювала приємні відчуття у користувачів під час його вирішення [23]. Загалом, виходячи з вищезазначеного аналізу, дизайн інтерфейсу застосунку регулюється конкретними окремими цілями та протоколами, які були створені та запропоновані. Спочатку вони стосуються

створення візуальної «мови» та представлення, яке синтезує та об'єднує всі практики правильного та ефективного дизайну системи [23].

На рис. 2.14 приведено форму з можливістю встановлення параметрів.

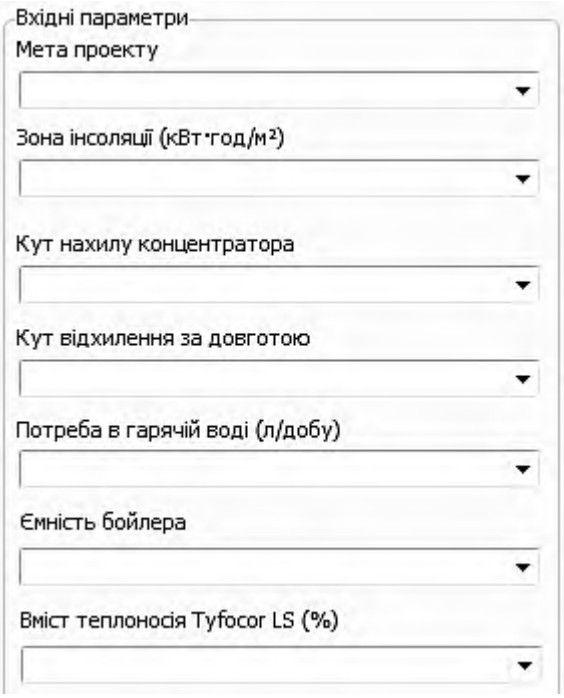


Рисунок 2.14 – Форма з можливістю встановлення параметрів

Таким чином, реалізуються принципи забезпечення процесів безперервної інтеграції та безперервної доставки програмного забезпечення, які називають конвеєром CI/CD [22, 23]. Це набір методів і операційних принципів та набір практик, що дозволяють розробникам застосунків частіше та надійніше вносити зміни до коду. Саме тому процес розробки застосунків використовує ці процеси, щоб створити гнучку основу для легкого проектування та еволюції продукту.

3 РОЗРОБКА ПРОГРАМНИХ БЛОКІВ АВТОМАТИЗОВАНОЇ СИСТЕМИ ПРОЄКТУВАННЯ ВІДНОВЛЮВАНИХ ДЖЕРЕЛ ЕНЕРГІЇ В УМОВАХ НЕСТАБІЛЬНОГО ЕЛЕКТРОПОСТАЧАННЯ

3.1. Інструментальні засоби розробки програмного забезпечення комплексу проєктування систем відновлюваних джерел енергії

Концепція Microsoft .NET включає набір нових програм, які Microsoft та треті сторони розробили (або розробляють) для використання на платформі .NET. До них належать програми, розроблені Microsoft, такі як Windows .NET, Hailstorm, Visual Studio .NET, MSN.NET, Office .NET та нові корпоративні сервери Microsoft (SQL Server .NET, Exchange .NET тощо) [23, 24].

У табл. 3.1 приведена інформація по кожному з компонентів платформи .NET [18, 19].

Таблиця 3.1 – Компоненти платформи .NET

Елемент	Визначення	Приклади
Інструменти розробки	Інтерфейси та засоби програмування для розробки, створення, виконання та розгортання рішень на платформі .NET.	.NET Framework Visual Studio .NET
Сервери	Середовище для розробки, впровадження рішень та супроводження серверів для платформи .NET	Microsoft Windows Server Сервери Microsoft .NET Enterprise Servers
Веб-служби XML	Централізована множина служб, що виконують основні завдання, а також інструмент для створення власних служб	Налаштовані служби Microsoft .NET
Клієнти	Комп'ютери, що працюють під управлінням операційних систем, які взаємодіють з іншими елементами .NET.	Microsoft Windows для портативних пристроїв Microsoft Windows для ПК
Користувацький інтерфейс	Клієнтське програмне забезпечення, яке інтегроване з XML-веб-службами, для користувачів, у зрозумілій для них формі	Майбутні версії Microsoft Central™ Microsoft MSN®

Середовище виконання Common Language Runtime (CLR) є ядром платформи .NET. Це механізм, що відповідає за керування виконанням розроблених для неї програм, і пропонує численні сервіси, що спрощують їхню розробку та підвищують їхню надійність і безпеку [19, 25].

На рис. 3.1 розміщене вікно IDE Visual Studio [18, 19].

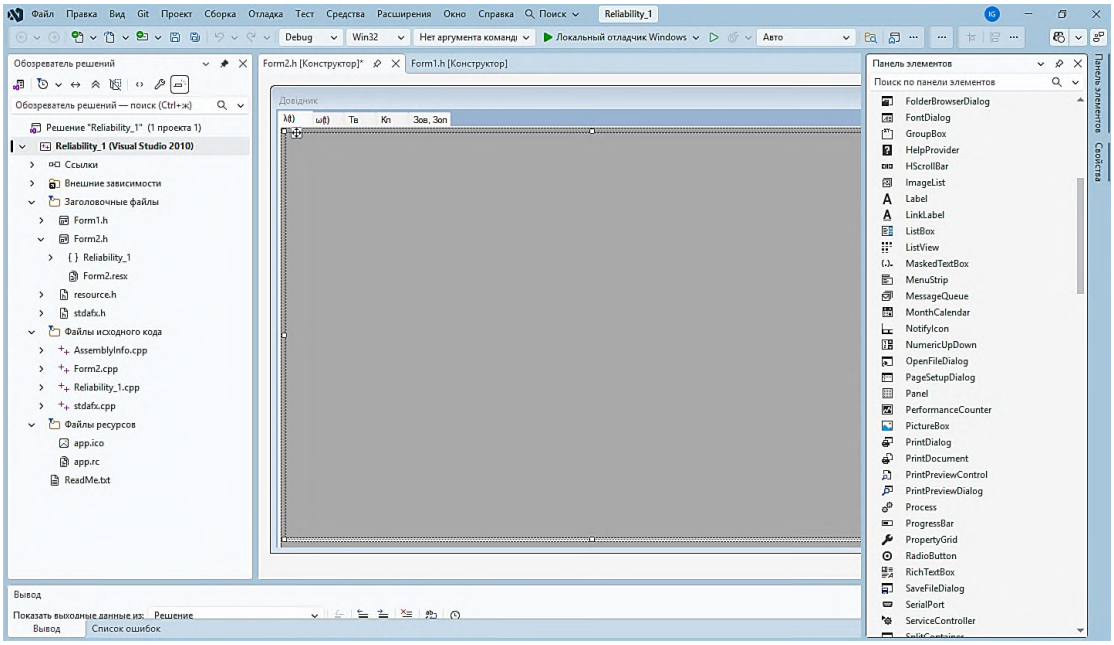


Рисунок 3.1 –Вікно IDE Visual Studio

Microsoft .NET — це набір нових технологій, які Microsoft розробляє протягом останніх кількох років з метою створення простої, але потужної платформи для розповсюдження програмного забезпечення як послуг, які можуть надаватися віддалено, взаємодіяти та поєднуватися один з одним повністю незалежно від платформи, мови програмування та моделі компонентів, що використовуються для їх розробки [19 - 22]. Це платформа .NET, а згадані вище служби називаються веб-сервісами.

Для створення застосунків для платформи .NET, включаючи як веб-сервіси, так і традиційні застосунки (консольні застосунки, настільні застосунки, служби Windows NT тощо), Microsoft випустила .NET Framework SDK, комплект розробки програмного забезпечення, що включає необхідні інструменти для розробки, розповсюдження та виконання, а також Visual

Studio .NET, який дозволяє робити все вищезазначене з візуального інтерфейсу на основі вікна [23, 24].

CLR діє як віртуальна машина, що запускає програми, розроблені для платформи .NET. Це означає, що будь-яка платформа, для якої існує версія CLR, може запускати будь-яку програму .NET. Microsoft розробила версії CLR для більшості версій Windows та Windows CE (які можна використовувати на процесорах, відмінних від x86) [24]. Microsoft також співпрацює з Corel для портування CLR на Linux, і є треті сторони, які незалежно розробляють версії CLR з відкритим кодом для Linux [23, 24]. Крім того, враховуючи, що архітектура CLR є повністю відкритою, можливо, що в майбутньому будуть розроблені версії для інших операційних систем.

Форма з вибором типу проекту показана на рис. 3.2.

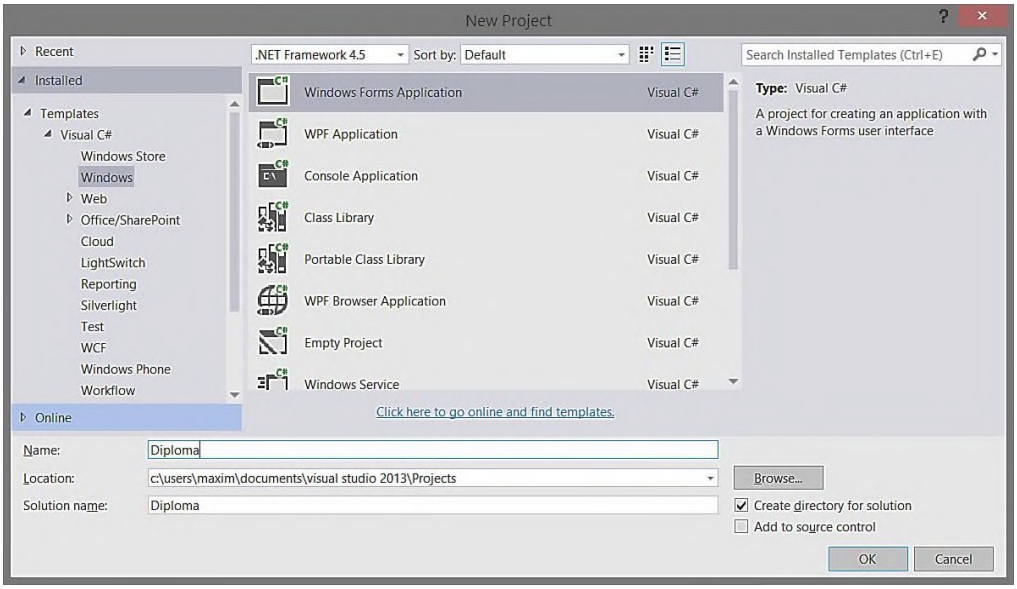


Рисунок 3.2 – Вибір Windows Forms Application

Щоб надати більш досвідченим програмістам загальне уявлення про мову, нижче наведено основні особливості C#. Деякі з згаданих тут особливостей не є специфічними для самої мови, а радше для платформи .NET загалом. Однак вони також мають безпосередній вплив на мову [24, 25].

Сучасність C# включає в саму мову елементи, які протягом багатьох років виявилися дуже корисними для розробки додатків і які інші мови, такі як Java або C++, повинні імітувати. До них належать базовий десятковий тип,

який дозволяє виконувати високоточні операції зі 128-бітними дійсними числами (дуже корисно у фінансовому світі), включення інструкції foreach, яка дозволяє легко ітерувати колекції та розширюється до типів, визначених користувачем, включення базового рядкового типу для представлення рядків та виділення певного типу bool для представлення логічних значень [25, 26].

C# є об'єктно-орієнтований. Як і всі сучасні мови програмування загального призначення, C# є об'єктно-орієнтованою мовою, хоча це скоріше характеристика CTS, ніж самого C#. Одна з відмінностей між цим об'єктно-орієнтованим підходом та підходом інших мов, таких як C++, полягає в тому, що C# є чистішою, оскільки не дозволяє глобальних функцій або змінних. Натомість весь код і дані повинні бути визначені в межах визначень типів даних, що зменшує проблеми, спричинені конфліктами імен, і покращує читабельність коду [26].

3.2. Розробка програмних модулів автоматизації систем відновлюваних джерел енергії

C# реалізує автоматичне керування пам'яттю. Всі мови .NET мають доступ до збирача сміття CLR [26, 27]. Це означає, що інструкції щодо знищення об'єктів не потрібні. Однак, оскільки знищення об'єктів через збирач сміття є недетермінованим і відбувається лише тоді, коли воно спрацьовує — чи то через недостатню кількість пам'яті, завершення програми, чи явний запит у вихідному коді — C# також надає детермінований механізм вивільнення ресурсів через оператор `using` [27].

Особливої уваги вимагає розгляд типів і структур даних на мові C#. Безпека типів C# містить механізми, що гарантують, що доступ до типів даних завжди виконується правильно. Це запобігає важковиявним помилкам, спричиненим доступом до пам'яті, яка не належить жодному об'єкту, що особливо важливо в середовищі зі збирачем сміття [27].

Загальна схема можливих структур даних приведена на рис. 3.3.

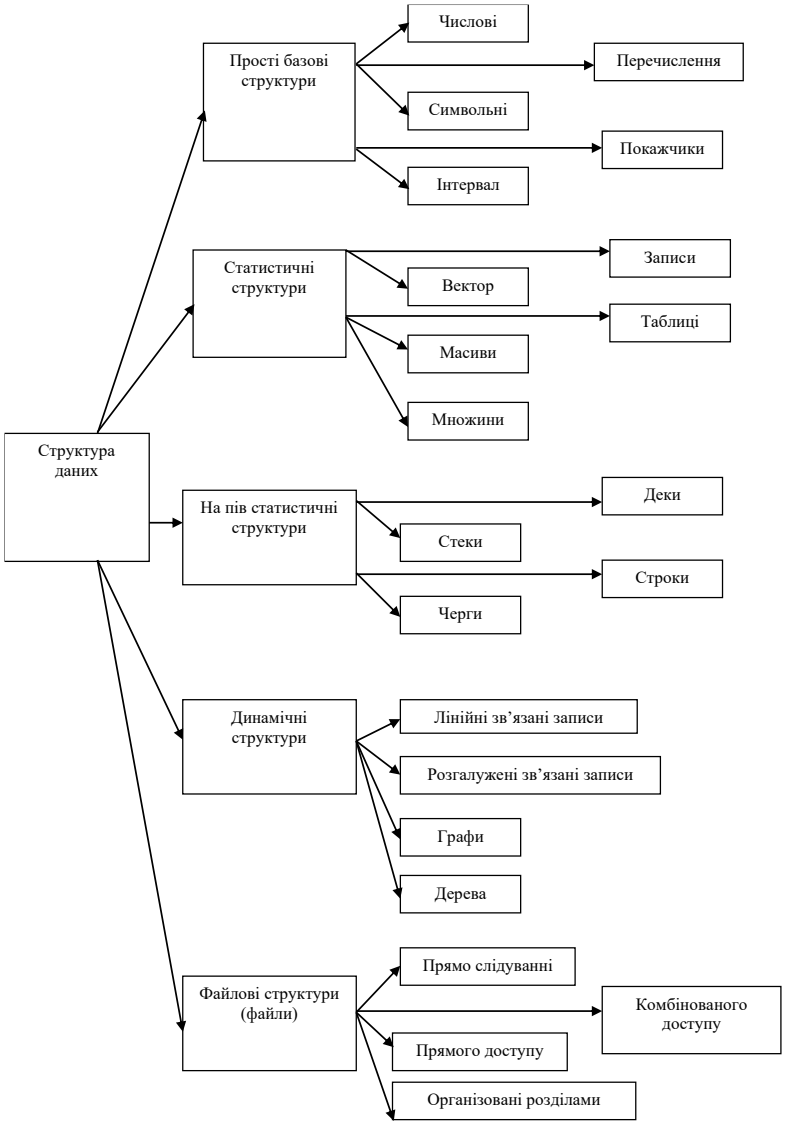


Рисунок 3.3 – Загальна схема можливих структур даних

Хоча жодне посилання на створений об'єкт типу не зберігається, і тому він недоступний для програміста, збирач сміття не знаходиться в однаковій ситуації та завжди має доступ до об'єктів, навіть якщо вони марні для програміста [26]. Важливо наголосити, що включати будь-які модифікатори у визначення деструктора некоректно, навіть модифікатори доступу, оскільки його ніколи не можна викликати явно, і тому він не має рівня доступу для програміста. Однак це не означає, що під час їх виклику справжній тип об'єктів, що знищуються, не враховується [27]. Були розроблені програмні модулі проєкта. Нижче показаний програмний лістинг класу модуля роботи з базою даних, рис. 3.4 [26].

```
using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;
using System.Data;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Dip
{
    public static class SqlConnections
    {
        public static void Run(string commandStr)
        {
            Logger.WriteLine(commandStr);
            MySqlConnection connection = new MySqlConnection
              (Global.ConnectionString);
            MySqlCommand command = new MySqlCommand(commandStr,
              connection);
            connection.Open();
            command.ExecuteNonQuery();
            connection.Close();
        }
        public static DataTable GetTable(String commandStr)
        {
            Logger.WriteLine(commandStr);
            MySqlConnection connection = new MySqlConnection
              (Global.ConnectionString);
            MySqlCommand command = new MySqlCommand(commandStr,
              connection);
            connection.Open();
            var sqlDataReader = command.ExecuteReader();
            if (sqlDataReader.HasRows)
            {

                var result = new DataTable();
                result.Load(sqlDataReader);
                sqlDataReader.Close();
                connection.Close();
                return result;
            }
            connection.Close();
            return null;
        }

        public static String GetValue(String commandStr)
        {
            Logger.WriteLine(commandStr);

            MySqlConnection connection = new MySqlConnection
              (Global.ConnectionString);
            MySqlCommand command = new MySqlCommand(commandStr,
              connection);
           connection.Open();
           var result = command.ExecuteScalar();
           if (result != null)
           {
               connection.Close();
               return result.ToString();
           }
           connection.Close();
           return null;
        }
    }
}
```

Рисунок 3.4 – Лістинг класу модуля роботи з базою даних

Лістинг класу початку взаємодії з даними [27].

```
class CommonDB
{
    public static void Init()
    {
        Tables.Add(new Table("SkillCategory") { Columns =
          { "CategoryID", "CategoryName" } });
        Tables.Add(new Table("Skill") { Columns = { "SkillID",
          "CategoryID", "SkillName" } });
        Tables.Add(new Table("Programmist") { Columns =
          { "ProgrammistID", "HourlyRate", "Name" } });
        Tables.Add(new Table("ProgrammistSkill") { Columns = { "ID",
          "ProgrammistID", "Level", "Experiense" } });
    }
    public static String ConnectionString = "server=localhost;user
      id=root;database=diploma;password=root";
    public static List<Table> Tables = new List<Table>();

    public static Table GetTableByName(string name)
    {
        foreach (Table t in Tables)
        {
            if (t.Name == name) return (t);
        }
        return (null);
    }
}
}
```

Рисунок 3.5 – Лістинг класу початку взаємодії з даними

Програмний код на рис. 3.6 показує дії запитів до бази даних

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace Dip
{
    public class Table
    {
        public String Name { get; set; }
        public List<String> Columns;

        public void Insert(string param)
        {
            StringBuilder colums = new StringBuilder("(");
            for (int i = 0; i < this.Columns.Count; i++)
            {
                colums.Append("`" + this.Columns[i] + "`");
                if (i < this.Columns.Count - 1) colums.Append(",");
            } //-----------------------------------------------
```

```
            columns.Append(")");
            SqlConnections.Run("INSERT INTO `diploma`.`" + this.Name +
              "` " + colums + " VALUES (" + param + ")");
        }
        public void RemoveBy(int paramIndex, string value)
        {
            SqlConnections.Run("DELETE FROM `diploma`.`" + this.Name +
              "` WHERE (" + this.Columns[paramIndex] + "='" + value +
              "')");
        }
        public DataTable GetTable(String columns, String Where = null)
        {
            if (Where == null)
                return SqlConnections.GetTable("SELECT " + columns + "
                  FROM `diploma`.`" + this.Name + "`");
            else
                return SqlConnections.GetTable("SELECT " + columns + "
                  FROM `diploma`.`" + this.Name + "` WHERE " + Where);
        }
        public String GetValue(String columns, String Where = null)
        {
            return SqlConnections.GetValue("SELECT " + columns + " FROM
              `diploma`.`" + this.Name + "` WHERE " + Where);
        }
        public Table(String name)
        {
            this.Name = name;
            Columns = new List<String>();
        }
    }
```

Рисунок 3.6 – Клас програмного коду дій запитів до бази даних

3.3. Реалізація бази даних застосунку проєктування систем відновлюваних джерел енергії

Зберігання та керування даними в комп'ютерній системі є одним з найважливіших застосувань комп'ютерів. Цей організований набір даних називається базою даних. Зокрема, в інформатиці термін "бази даних" стосується організованих, дискретних колекцій пов'язаних збережених даних. Термін "Система керування базами даних" стосується програмного забезпечення, яке обробляє такі колекції [15]. Спосіб проектування та організації даних має бути спрямований на швидкий пошук інформації, а також легке оновлення даних [18]. Сьогодні дані записуються та використовуються в будь-якій діяльності: науковій, комерційній, виробничій, урядовій, військовій тощо. Великі обсяги даних вимагають пошуку ефективних методів зберігання. Дані повинні бути організовані таким чином,

щоб полегшити їх пошук та оновлення. Загальна структура програмного застосунку, якій використовує базу даних, приведена на рис. 3.7.

Рисунок 3.7 – Загальна структура програмного застосунку, якій використовує базу даних

СУБД має своєю основною метою загальну роботу з базою даних, з точки зору створення, підтримки, обробки даних, перевірки безпеки тощо, а також обслуговування користувачів на всіх рівнях [17, 18]. По суті, можна стверджувати, що СУБД є посередником між користувачем і базою даних, і тільки через нього користувач може запитувати інформацію з бази даних. СУБД може бути встановлена на ПК і використовуватися одним користувачем (однокористувацька система) або може бути встановлена на наборі комп’ютерів, які спілкуються один з одним через певну (локальну або віддалену) мережу, і використовуватися багатьма користувачами (багатокористувацька система) [18]. Основні функції, які виконує СУБД наступні. Організовує базу даних на носії інформації (жорсткі диски, оптичні диски, хмара тощо). Забезпечує механізми керування даними. Надає

програмам дані у форматі, який вони запитують.

На даних принципах була розроблена база даних. Для розробки використана база даних MySQL, яка приведена на рис. 3.8.

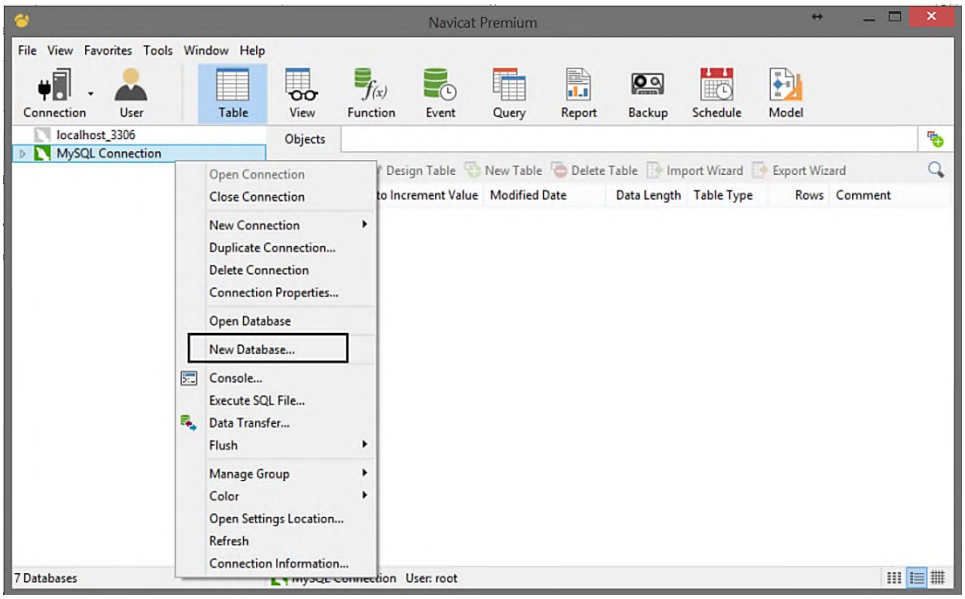


Рисунок 3.8 – Розробка бази даних в середовище MySQL

Необхідність оцінки надійності в усьому, що пов'язано з транзакціями з базами даних, гарантується прийняттям вимог ACID (атомарність, узгодженість, ізоляція, довговічність). Ми розглядатимемо будь-яку логічну дію, пов'язану з даними, як транзакцію [18, 27].

Проєктування таблиць реалізоване інструкціями мови SQL, однак, таблиці можна розробляти безпосередньо в середовище розробки, як показано на рис. 3.9 [19].

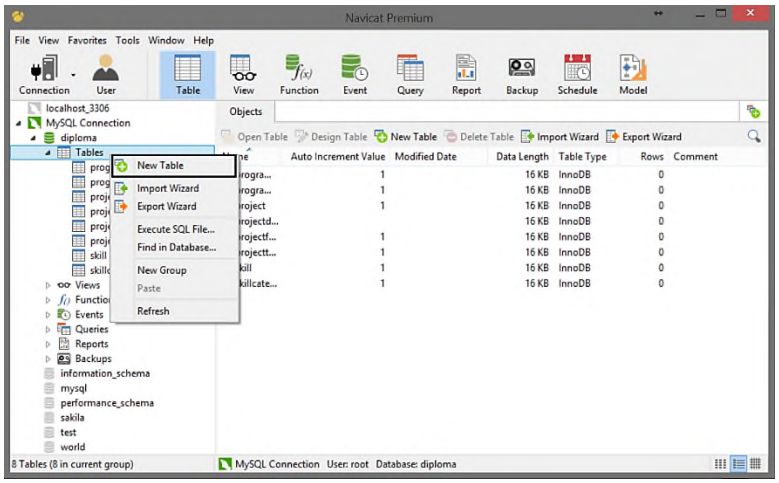


Рисунок 3.9 –Створення таблиць у середовище MySQL

Реляційна модель складається з таблиць, де кожна таблиця має унікальне ім’я та ідентифікується набором рядків і набором стовпців [16]. Кожна таблиця містить записи певного типу, і кожен тип запису визначає фіксовану кількість полів або властивостей [18]. Масив можна використовувати як для записів, що описують сутності, так і для записів, що описують зв’язки. Зазвичай ми думаємо про стовпці як про впорядковані, і ми прагнемо спочатку перерахувати стовпці, які відповідають атрибутам, які є ключами [18]. Дані значень теплоносія показано на рис. 3.10.

id	brand	base_substance	freezing_point	density
1	Antifrogen L	Пропіленгліколь	-30	1.04
2	Solaris Fluid	Поліпропіленгліколь	-28	1.03
3	Еко-Тепло	Гліцерин-вода	-25	1.12
4	Тепло-Мікс	Етиленгліколь	-35	1.08
5	Дистильована вода	Вода	0	1

Рисунок 3.10 – Дані значень теплоносія

Дані типів теплообмінних панелей приведені на рис. 3.11.

id	model	type	area_sqm	manufacturer
1	Sunsys S4	Плоский	2.51	Atmosfera
2	VacuumTube X1	Вакуумний	1.84	Meibes
3	Heat-X Pro	Термосифонний	2.1	Hewalex
4	SolarStar 200	Плоский	2.05	Viessmann
5	BlueClean	Вакуумний	3.2	Vaillant
6	EcoSun S3	Плоский	2.35	Atmosfera

Рисунок 3.11 – Дані типів теплообмінних панелей

3.4. Розробка користувацького інтерфейсу системи автоматизації проєктування систем відновлюваних джерел енергії

CI/CD – одна з найкращих практик, яку можуть впроваджувати розробники програмного забезпечення [19, 28]. Це також найкраща методологія, яка є частиною гнучких методів програмування, оскільки вона дозволяє розробникам програмного забезпечення зосередитися на виконанні

бізнес-вимог, якості коду та безпеці, за умови автоматизації етапів розробки. Безперервна інтеграція (CI) – це філософія кодування та набір практик, які спонукають команди розробників впроваджувати невеликі зміни та часто перевіряти свій код розробки у релізах у репозиторіях [27, 28]. Оскільки більшість сучасних програм вимагають розробки коду на різних платформах та інструментах, розробникові потрібен механізм для завершення та перевірки своїх змін. Форма застосунку показана на рис. 3.12.

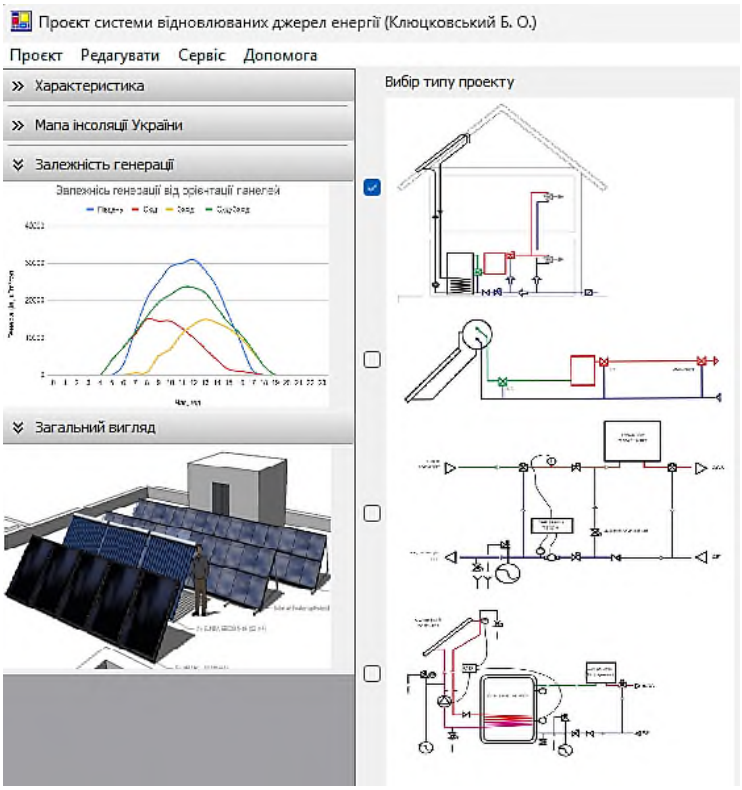


Рисунок 3.12 – Головний інтерфейс застосунку

Визначення вимог до інтерфейсу визначає вивчення та аналіз проблеми інтерфейсу, визначення призначення програмного забезпечення та цілі, які будуть поставлені, а також визначено середовище, в якому воно працюватиме, та користувачів, яких воно обслуговуватиме [25, 28]. Проектування, де з числа запропонованих обирається найпоширеніший формат інтерфейсу користувача. При виборі враховуються вартість, час навчання, доступні матеріальні ресурси. Чітко створюються об'єкти, які його реалізовуватимуть, дії, які кожен з користувачів виконуватиме, та події, на які він реагуватиме [27].

Щодо інформаційної архітектури, зазначимо, що інформаційна архітектура – це план, тобто візуальне представлення інфраструктури, функцій та ієрархії застосунку. Застосунок використовує інформаційний метод, який мовою називається структурою потоку [26]. Таким чином, структура потоку – це структура, що базується на завданнях, що вимагають доступу до екранів у послідовному порядку. Навігація між екранами здійснюється у послідовному порядку, що вказує на ієрархію доступу до інформації. Для переходу між ними здійснюються відповідні рухи та вибір подій [27]. Форми застосунку приведені на рис.3.13.

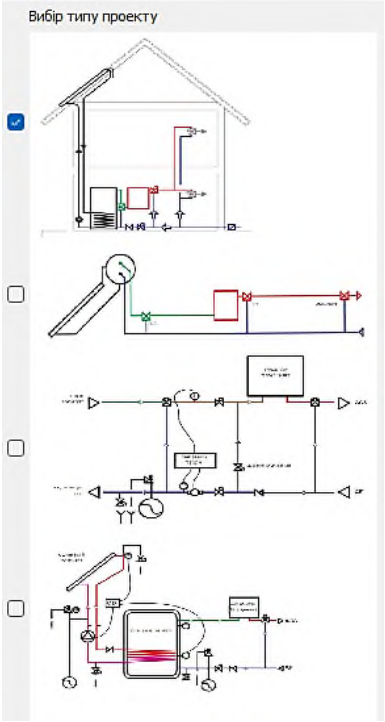


Рисунок 3.13 – Форми застосунку

Ергономічність інтерфейсу суттєво впливає на сприйняття його користувачем івпливає на його сприйняття зручності використання, надійності та продуктивності системи [28]. Вищезазначені візуальні елементи призводять до високої зручності використання та мотивують користувачів використовувати інтерфейс програми, що робить її отримувачем позитивних відгуків. Таким чином, ми робимо висновок, що її якість можна оцінити на основі зручності використання [28].

3.5. Розробка модулів візуалізації результатів роботи програмного комплексу системи автоматизації

При розробці модулів візуалізації результатів роботи програмного комплексу перш за все необхідно визначити основні засоби інтерфейсу для формування даних для звітів. Вони такі [27]. Запитання та відповіді –це діалог, що встановлюється з комп'ютерною програмою для виконання низки завдань; користувач ставить запитання, а комп'ютер відповідає. Інтерактивні форми – у них користувач вводить дані або команди у спеціально розроблені поля. Природна мова – взаємодія людини з машиною, подібна до діалогу між людьми, називається природною мовою [27]. Наразі цей тип діалогу переважно ведеться за допомогою клавіатури. Основна проблема полягає поки ще в нездатності комп'ютера розуміти природну мову [28].

Звітування включає макет, вибір даних, дії звіту (наприклад, фільтри, посилання та пов’язані вибори) і візуалізацію (у веб-форматах і рідних форматах мобільних пристроїв). Візуалізація звіту показана на рис. 3.14.

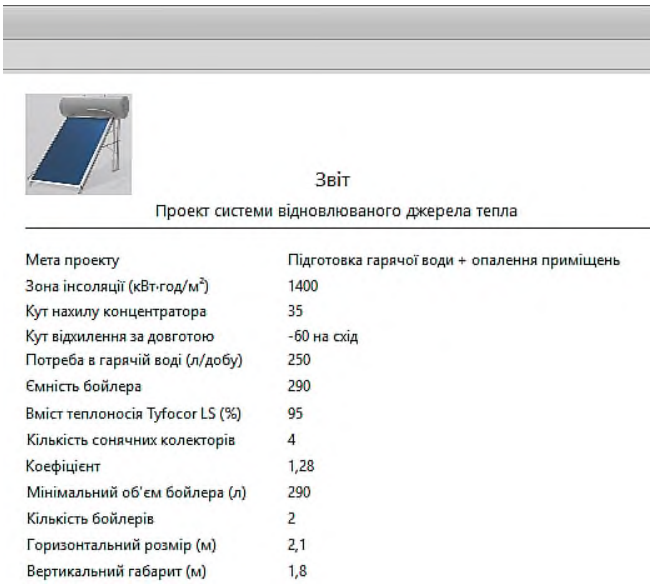


Рисунок 3.14 – Форма візуалізації звіту

Засоби візуалізації інтуїтивно зрозумілі і швидко реагують навіть на надзвичайно великі джерела даних. Вони надають усе необхідне для створення та розповсюдження чітких і відповідних звітів [27].

Visual Analytics містить такі інтерфейси. SAS Visual Analytics дозволяє вибирати дані, створювати звіти, ділитися звітами та об’єктами з іншими користувачами, а також друкувати звіти та об’єкти у форматі PDF [28]. Той самий інтерфейс дозволяє переглядати та взаємодіяти зі звітами, створеними іншими користувачами. SAS Graph Builder дозволяє користувачам створювати спеціальні об’єкти графіків, які можуть бути доступні в SAS Visual Analytics для використання у звітах і на інформаційних панелях [27]. Програми SAS Visual Analytics дозволяють мобільним користувачам переглядати звіти та взаємодіяти з ними.

Усі джерела даних, доступні в SAS Visual Analytics, містять один або кілька елементів даних, які можна використовувати у звітах [28]. Ви можете вибрати всі елементи даних у джерелі даних або їх підмножину. Якщо об’єкт має неповні призначення даних, відображається кнопка «Призначити дані». Натисніть кнопку та скористайтеся спливаючим меню, щоб призначити елементи даних необхідним ролям даних. Потім ви можете вручну призначити додаткові елементи даних на панелі «Дані» або панелі «Ролі даних».

Одним із варіантів формування звіту є таблиця-список. Таблиця-список – це двовимірне представлення даних. Таблиця-список не може використовувати ієрархію. Таблиці-списки містять агреговані дані (якщо не вибрано опцію «Детальні дані») [28]. Можна додавати міні-діаграми до стовпця (якщо джерело даних містить елемент даних дати), коли агреговані дані відображаються в таблиці-списку. Ви можете додати клітинкову діаграму (гістограму або теплову карту) до стовпця в таблиці-списку, якщо джерело даних містить принаймні один елемент даних міри [26, 28].

Спільнота SAS Visual Analytics призначена для користувачів, які зосереджені на дослідницькій візуалізації та аналітичних методах, підготовці даних, звітності на інформаційній панелі та мобільних пристроях. Використовуючи спільноту, можна ділитися своїм досвідом, обговорювати теми та ідеї, просити своїх колег про допомогу та ділитися інформацією про майбутні події [28].

ВИСНОВКИ

Виконана дипломна кваліфікаційна робота: «Комп'ютеризація проєктів відновлюваних джерел енергії в умовах нестабільного електропостачання».

У цій роботі досліджено, як проєктувати системи сонячного водопідігріву. Сонячна енергія має безліч плюсів: її багато, вона дешева та екологічна. Але є й складності — вона має низьку щільність (мало енергії на одиницю площі), і її важко зберігати у великих обсягах. В роботі реалізоване наступні компоненти. Математичні моделі – розглянути моделі управління сонячним підігрівом та описані основні чинники, які впливають на роботу таких систем. Схема регулювання системою – розроблена принципова схема з елементами, що спрощують налаштування, користування, контроль та обслуговування обладнання. Розглянуті програмні регулятори – окремо зупинилися на регуляторах, адже вони прості у використанні та чудово працюють у найрізноманітніших умовах.

Розглянуті та обрані найкращі програмні технології для створення блоків програмного застосунку управління системою сонячного підігріву. Була обрана платформа Microsoft .NET. Основні етапи проєктування застосунка. Проєктування програмного забезпечення – опис структури програми, моделі й структури даних, алгоритми, а також те, як компоненти системи взаємодіють між собою. Практична реалізація системи – створені та налаштувані основні програмні класи. Розроблена схема системи – запропонувана зручна схема компонування для блоків програмного комплексу.

Спираючись на отримані результати, можна зробити такі висновки:

– сформульовано загальний опис проблеми, чітко сформульовані та проаналізовані труднощі, з якими стикаються системи сонячного підігріву в умовах низького рівня чи дефіциту електроенергії сьогодні;

– проведені базові дослідження, вивчені фізичні принципи роботи сонячного підігріву та проаналізовані аналогічні рішення, які вже є на ринку;

– поставлені цілі та завдання кваліфікаційної роботи, визначені, які саме задачі має вирішувати новий апаратно-програмний комплекс;

– проведено проєктування та розроблені схеми, розроблені структурні компоненти комплексу та структурні схеми компонування його програмних модулів;

– розроблені алгоритми для програмного комплексу з ціллю управління автоматизованою системою;

– розроблена база даних програмного застосунку, спроєктовані зв'язки між даними та створена готова база даних для автоматизованого застосунку;

– розроблений графічний інтерфейс застосунку, розроблені зручні функціональні елементи графічного інтерфейсу для користувача.

,

ПЕРЕЛІК ПОСИЛАНЬ

1. Дафі ДжА., Бекман В.А. «Теплові процеси з використанням сонячної енергії», 2019. 324 с.

2. ТкаченкоС.І., Степанов Д.В. *«Проектування геліосистем теплопостачання».* 2017. 232 с.

3. Kalogiru S.A. «Solar Energy Engineering. Processes and Systems». (Academic Press). 2021. 168 p.

4. Желіх В.М., Юркевич Ю.С. «Сонячне теплопостачання». Видавництво Львівської політехніки – 2020. – 173 с.

5. Dufie J.A., Bekman V.A. «Solar Engineering of Thermal Processes, Photovoltaics and Wind» (Wiley). 2021. 187 p.

6. Форкун Я. Б. Сонячна теплоенергетика. Конспект лекцій для спеціальності 141 – Електроенергетика, електротехніка та електромеханіка / Я.Б. Форкун, О.О. Шкурпела. Харків нац. ун-т госп-ва ім. О. М. Бекетова. – Харків : ХНУМГ ім. О. М. Бекетова, 2020. – 88 с.

7. EuroSun2026 conference in Freiburg, 2026. URL: https://solarthermalworld.org/news/eurosun2026-opens-registration-secure-your-early-bird-discount-before-1-july/

8. Carbonell, D., et al. (2018), “Simulations of combined solar pump systems for domestic hot water and space heating”, Proceedings of the 2nd International Conference on Solar Heating and Cooling for Buildings and Industry (SHC 2013), Energy Procedia, Vol. 48, pp. 524-534, https://www.sciencedirect.com/science/article/pii/S1876610214003245.

9. Кузик М. П., Заяць М. Ф. Пасивна система сонячного теплопостачання // Кузик М. П., Заяць М. Ф. – Науковий вісник НЛТУ України, 2019, т. 29, №5. С.111-114

10. Frantzis, T. (2015), “Solar Thermal Systems in the Tourist Industry”, IRENA-Cyprus Event on Renewable Energy Applications for Island Tourism, 29-30 May, www. irena.org/DocumentDownloads/events/2014/June/9_Frantzis.pdf.

11. Solar Heat Europe (ESTIF): Decarbonising Heat with Solar Thermal - Market Outlook 2023/2024, December 2024

12. D. Yogi Goswami.2014. Principles of Solar Engineering. Taylor & Francis Group Link: https://www.advan-kt.com/principlesofsolarengi.pdf

13. Natural Resources Canada 2024: Report about the Canadian Solar Thermal Market Survey

14. Wattana Ratismith, ―A Novel Non-Tracking Solar Collector for High Temperature Application.‖, proceedings of ecos 2012 - the 25th international conference on efficiency, cost, optimization, simulation, environmental impact of energy systems June 26- 29, 2012, Perugia, Italy.

15. Зощенко С.А. Сонячне теплопостачання: різновиди систем перетворення, ефективність – Відновлювана енергетика. №4/2022. С. 43-48

16. Handbook. (2019). Solar Electricity Handbook. Edition. Solar Irradiance. 2019. Retrieved from: http://solarelectricityhandbook.com/solarirradiance.html

17. K.K. Chong, K.G. Chay, K.H. Chin, ―Study of a solar water heater using stationary collector‖, Renewable Energy 39 (2012) 207-215.

18. Коноваленко І.В. Платформа .NET та мова програмування C# 7.0: навчальний посібник / Коноваленко І.В., Марущак П.О. – Тернопіль: ФОП Паляниця В. А., 2020 – 320 с.

19. Daniel Solis. Illustrated C#7. The C# Language Presented Clearly, Concisely, and Visually. – APress, 2018. – 799 p.

20. Об’єктно-орієнтоване програмування. Частина 1. Основи об’єктно-орієнтованого програмування на мові C#: Навчальний посібник. / Д.В. Настенко, А.В. Нестерко. – К.: НТУ «КПІ», 2016. - 76с.

21. Коноваленко І.В. Платформа .NET і мова програмування C#8.0: Навчальний посібник/ І.В. Коноваленко, П.О. Марущак. – Тернопіль: ФОП Паляниця В. А., 2022. – 320 с.

22. Олефіренко Н.В., Курганський А.Р., Остапенко Л.П. Об’єктно-орієнтоване програмування мовою C#. Навчальний посібник – Харків, 2024. – 254 с.

23. Jeffrey Richter CLR via C#. Programming on Microsoft.NET Framework 4.5 in C#: Print2print. – 2020. – 896 p.

24. T. Dmytrenko, T. Derkach, A. Dmytrenko Using JAVA and C# programming languages for server platforms and workstations. Control, Navigation and Communication Systems. Vol. 3 No. 73 (2023). P. 93-94

25. Joseph Albahari, C#10 in a Nutshel. The Definitive Reference: O'Reilly. – 2022. – 1058 p.

26. Івохін, Е.В.; Махно, М.Ф.; Піскунов, О.Г. Розробка додатків засобами мови програмування С#: Навч.-метод. посібник для проведення лабораторних робіт для студентів вищих навчальних закладів спеціальності «системний аналіз» /Е.В.Івохін, М.Ф.Махно, О.Г.Піскунов. – К.: Видавничо-поліграфічний центр "Київський університет", 2021. – 100 с.

27. Vystavel, V. C# Programming for Absolute Beginners: Learn to Think Like a Programmer and Start Writing [Text]/ V. Vystavel. - 2nd Edition. - Ondreyov, Czech Republic: APress, 2021 - 365 p.

28. Nagel, C. Professional C# and .NET [Текст] / C. Nagel. - Hoboken, New Jersey: John Willey & Sons, Inc., 2022 — 970 p

ДОДАТОК А

Основні лістинги програмного коду проекту

```
using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;
using System.Data;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Dip
{
    public static class SqlConnections
    {
        public static void Run(string commandStr)
        {
            Logger.WriteLine(commandStr);
            MySqlConnection connection = new MySqlConnection
              (Global.ConnectionString);
            MySqlCommand command = new MySqlCommand(commandStr,
              connection);
            connection.Open();
            command.ExecuteNonQuery();
            connection.Close();
        }
        public static DataTable GetTable(String commandStr)
        {
            Logger.WriteLine(commandStr);
            MySqlConnection connection = new MySqlConnection
              (Global.ConnectionString);
            MySqlCommand command = new MySqlCommand(commandStr,
              connection);
            connection.Open();
            var sqlDataReader = command.ExecuteReader();
            if (sqlDataReader.HasRows)
            {

                var result = new DataTable();
                result.Load(sqlDataReader);
                sqlDataReader.Close();
                connection.Close();
                return result;
            }
            connection.Close();
            return null;
        }

        public static String GetValue(String commandStr)
        {
            Logger.WriteLine(commandStr);

            MySqlConnection connection = new MySqlConnection
              (Global.ConnectionString);
            MySqlCommand command = new MySqlCommand(commandStr,
              connection);
           connection.Open();
           var result = command.ExecuteScalar();
           if (result != null)
           {
               connection.Close();
               return result.ToString();
           }
           connection.Close();
           return null;
       }
    }
}
```

```
using System;
using System.Drawing;
using System.Windows.Forms;
namespace FaultTreeAnalysis
{
    public partial class Form1 : Form
    {
        private Random random = new Random();
        private Graphics graphics;
        private bool showBrakeSystemTree = true;
        private bool showEngineControlSystemTree = false;
        private bool showCoolingSystemTree = false;
        private bool showElectricalSystemTree = false;
        private bool showTransmissionSystemTree = false;

        public Form1()
        {
            InitializeComponent();
            LoadTreeNames();
        }

        private void LoadTreeNames()
        {
            listBox1.Items.AddRange(new string[] { });
            listBox1.SelectedIndex = 0;
        }

        private void listBox1_SelectedIndexChanged(object sender,
          EventArgs e)
        {
            switch (listBox1.SelectedItem.ToString())
            {
                case "1":
                    showBrakeSystemTree = true;
                    showEngineControlSystemTree = false;
                    showCoolingSystemTree = false;
                    showElectricalSystemTree = false;
                    showTransmissionSystemTree = false;
                    break;
                case "2":
                    showBrakeSystemTree = false;
                    showEngineControlSystemTree = true;
                    showCoolingSystemTree = false;
                    showElectricalSystemTree = false;
                    showTransmissionSystemTree = false;
                    break;
                case "3":
                    showBrakeSystemTree = false;
                    showEngineControlSystemTree = false;
                    showCoolingSystemTree = true;
                    showElectricalSystemTree = false;
                    showTransmissionSystemTree = false;
```

```
            break;
        case "4":
            showBrakeSystemTree = false;
            showEngineControlSystemTree = false;
            showCoolingSystemTree = false;
            showElectricalSystemTree = true;
            showTransmissionSystemTree = false;
            break;
        case "5":
            showBrakeSystemTree = false;
            showEngineControlSystemTree = false;
            showCoolingSystemTree = false;
            showElectricalSystemTree = false;
            showTransmissionSystemTree = true;
            break;
    }
    buttonDrawTree_Click(sender, e);
}
private void buttonFind_Click(object sender, EventArgs e)
{
    graphics = panel1.CreateGraphics();
    if (showBrakeSystemTree)
    {
        DrawBrakeSystemTree(graphics);
    }
    else if (showEngineControlSystemTree)
    {
        DrawEngineControlSystemTree(graphics);
    }
    else if (showCoolingSystemTree)
    {
        DrawCoolingSystemTree(graphics);
    }
    else if (showElectricalSystemTree)
    {
        DrawElectricalSystemTree(graphics);
    }
    else if (showTransmissionSystemTree)
    {
        DrawTransmissionSystemTree(graphics);
    }
}

private void EngineControlSystem(Object g)
{
    g.Clear(Color.White);
    Pen pen = new Pen(Color.Black, 2);
    Font font = new Font("Times New Roman", 10);
    Brush brush = Brushes.Black;

    int centerX = panel1.Width / 2;
```

```
int startY = 20;
int nodeWidth = 170;
int nodeHeight = 50;
int verticalSpacing = 80;
int horizontalSpacing = 200;
Rectangle rootRect = new Rectangle(centerX - nodeWidth /
  2, startY, nodeWidth, nodeHeight);
g.DrawRectangle(pen, rootRect);
Rectangle nodeB = new Rectangle(centerX - nodeWidth / 2 -
  horizontalSpacing / 2,
    startY + nodeHeight + verticalSpacing, nodeWidth,
      nodeHeight);
g.DrawRectangle(pen, nodeB);
Rectangle nodeC = new Rectangle(centerX - nodeWidth / 2 +
  horizontalSpacing / 2,
    startY + nodeHeight + verticalSpacing, nodeWidth,
      nodeHeight);
g.DrawRectangle(pen, nodeC);
Rectangle nodeD = new Rectangle(centerX - nodeWidth / 2,
    startY + (nodeHeight + verticalSpacing) * 2 + 100,
      nodeWidth, nodeHeight);
g.DrawRectangle(pen, nodeD);
Rectangle nodeE = new Rectangle(nodeB.X -
  horizontalSpacing,
    nodeB.Y + nodeHeight + verticalSpacing, nodeWidth,
      nodeHeight);
g.DrawRectangle(pen, nodeE);
Rectangle nodeF = new Rectangle(nodeB.X +
  horizontalSpacing,
    nodeB.Y + nodeHeight + verticalSpacing, nodeWidth,
      nodeHeight);
g.DrawRectangle(pen, nodeF);
Rectangle nodeI = new Rectangle(nodeE.X, nodeE.Y +
  nodeHeight + verticalSpacing,
    nodeWidth, nodeHeight);
g.DrawRectangle(pen, nodeI);
Rectangle nodeG = new Rectangle(nodeC.X -
  horizontalSpacing, nodeC.Y + nodeHeight + verticalSpacing,
    nodeWidth, nodeHeight);
g.DrawRectangle(pen, nodeG);
Rectangle nodeH = new Rectangle(nodeC.X +
  horizontalSpacing, nodeC.Y + nodeHeight + verticalSpacing,
    nodeWidth, nodeHeight);
g.DrawRectangle(pen, nodeH);
Rectangle nodeJ = new Rectangle(nodeH.X, nodeH.Y +
  nodeHeight + verticalSpacing,
    nodeWidth, nodeHeight);
g.DrawRectangle(pen, nodeJ);
int operatorWidth = 50;
int operatorHeight = 20;
Rectangle or1 = new Rectangle(centerX - operatorWidth / 2,
    rootRect.Y + nodeHeight + 10, operatorWidth,
```

```
        operatorHeight);
    Rectangle or2 = new Rectangle(nodeB.X + nodeWidth / 2 -
      operatorWidth / 2,
        nodeB.Y + nodeHeight + 10, operatorWidth,
          operatorHeight);
    Rectangle and1 = new Rectangle(nodeE.X + nodeWidth / 2 -
      operatorWidth / 2,
        nodeE.Y + nodeHeight + 10, operatorWidth,
          operatorHeight);
    Rectangle not1 = new Rectangle(nodeJ.X + nodeWidth / 2 -
      operatorWidth / 2,
        nodeH.Y + nodeHeight + 10, operatorWidth,
          operatorHeight);
    Rectangle or3 = new Rectangle(nodeC.X + nodeWidth / 2 -
      operatorWidth / 2,
        nodeC.Y + nodeHeight + 10, operatorWidth,
          operatorHeight);
    Brush whiteBrush = Brushes.White;
}
private void CoolingSystem(Graphics g)
{
    g.Clear(Color.White);
    Pen pen = new Pen(Color.Black, 2);
    Font font = new Font("Times New Roman", 10);
    Brush brush = Brushes.Black;
    int centerX = panel1.Width / 2;
    int startY = 20;
    int nodeWidth = 150;
    int nodeHeight = 50;
    int verticalSpacing = 60;
    int horizontalSpacing = 160;

    Rectangle rootRect = new Rectangle(centerX - nodeWidth / 2,
        startY, nodeWidth, nodeHeight);
    g.DrawRectangle(pen, rootRect);
    Rectangle nodeB = new Rectangle(centerX - nodeWidth -
      horizontalSpacing / 2);
    g.DrawRectangle(pen, nodeB);


    Rectangle nodeC = new Rectangle(centerX +
      horizontalSpacing / 2,
        startY + nodeHeight + verticalSpacing, nodeWidth,
          nodeHeight);
    g.DrawRectangle(pen, nodeC);
    Rectangle nodeD = new Rectangle(centerX - nodeWidth / 2,
        startY + (nodeHeight + verticalSpacing) * 2,
          nodeWidth, nodeHeight);
    g.DrawRectangle(pen, nodeD);
    Rectangle nodeE = new Rectangle(nodeD.X -
      horizontalSpacing / 2 - nodeWidth / 2,
        nodeD.Y + nodeHeight + verticalSpacing, nodeWidth,
```

```
            nodeHeight);
        g.DrawRectangle(pen, nodeE);
        Rectangle nodeF = new Rectangle(nodeD.X +
          horizontalSpacing / 2 - nodeWidth / 2,
            nodeD.Y + nodeHeight + verticalSpacing, nodeWidth,
              nodeHeight);
        g.DrawRectangle(pen, nodeF);
        int operatorWidth = 50;
        int operatorHeight = 20;
        Rectangle or1 = new Rectangle(centerX - operatorWidth / 2,
            rootRect.Y + nodeHeight + 10, operatorWidth,
              operatorHeight);
        Rectangle or2 = new Rectangle(nodeD.X + nodeWidth / 2 -
          operatorWidth / 2,
            nodeD.Y + nodeHeight + 10, operatorWidth,
              operatorHeight);
        g.DrawLine(pen, new Point(rootRect.X + nodeWidth / 2,
            rootRect.Y + nodeHeight), new Point(centerX, or1.Y +
              operatorHeight / 2));
        g.DrawLine(pen, new Point(centerX, or1.Y +
          operatorHeight / 2), new Point(nodeB.X + nodeWidth / 2,
          nodeB.Y));
        g.DrawLine(pen, new Point(centerX, or1.Y +
          operatorHeight / 2), new Point(nodeC.X + nodeWidth / 2,
          nodeC.Y));
        g.DrawLine(pen, new Point(centerX, or1.Y +
          operatorHeight / 2), new Point(nodeD.X + nodeWidth / 2,
          nodeD.Y));
    }

    private void DrawElectricalSystemTree(Graphics g)
    {
        // Налаштування графіки
        g.Clear(Color.White);
        Pen pen = new Pen(Color.Black, 2);
        Font font = new Font("Times New Roman", 10);
        Brush brush = Brushes.Black;

        // Координати та розміри елементів
        int centerX = panel1.Width / 2;
        int startY = 20;
        int nodeWidth = 150;
        int nodeHeight = 50;
        int verticalSpacing = 60;
        int horizontalSpacing = 160;

        // Малювання вузлів
        Rectangle rootRect = new Rectangle(centerX - nodeWidth / 2,
            startY, nodeWidth, nodeHeight);
        g.DrawRectangle(pen, rootRect);
        g.DrawString("Повна відмова електричної системи", font,
          brush, rootRect);
```

```
    g.FillRectangle(whiteBrush, or2);
    g.DrawRectangle(pen, or2);
    g.DrawString("OR", font, brush, CenterText("OR", font,
      or2));

    g.FillRectangle(whiteBrush, or3);
    g.DrawRectangle(pen, or3);
    g.DrawString("OR", font, brush, CenterText("OR", font,
      or3));
}

private void DrawTransmissionSystemTree(Graphics g)
{
    g.Clear(Color.White);
    Pen pen = new Pen(Color.Black, 2);
    Font font = new Font("Times New Roman", 10);
    Brush brush = Brushes.Black;

    int centerX = panel1.Width / 2;
    int startY = 20;
    int nodeWidth = 150;
    int nodeHeight = 50;
    int verticalSpacing = 60;
    int horizontalSpacing = 160;

    Rectangle rootRect = new Rectangle(centerX - nodeWidth /
      2, startY, nodeWidth, nodeHeight);
    g.DrawRectangle(pen, rootRect);
    Rectangle nodeB = new Rectangle(centerX - nodeWidth -
      horizontalSpacing, startY + nodeHeight + verticalSpacing,
      nodeWidth, nodeHeight);
    g.DrawRectangle(pen, nodeB);
    Rectangle nodeC = new Rectangle(centerX +
      horizontalSpacing, startY + nodeHeight + verticalSpacing,
      nodeWidth, nodeHeight);
    g.DrawRectangle(pen, nodeC);
    Rectangle nodeD = new Rectangle(centerX - nodeWidth -
      horizontalSpacing / 2 + 155,
        startY + (nodeHeight + verticalSpacing) * 2 - 53,
          nodeWidth, nodeHeight);
    g.DrawRectangle(pen, nodeD);
    Rectangle nodeE = new Rectangle(centerX +
      horizontalSpacing / 2 + 80,
        startY + (nodeHeight + verticalSpacing) * 2 - 200,
          nodeWidth, nodeHeight);
    g.DrawRectangle(pen, nodeE);
    Rectangle nodeF = new Rectangle(nodeB.X -
      horizontalSpacing / 2, nodeB.Y + nodeHeight +
      verticalSpacing, nodeWidth, nodeHeight);
    g.DrawRectangle(pen, nodeF);
    Rectangle nodeG = new Rectangle(nodeB.X +
```

```
  horizontalSpacing / 2, nodeB.Y + nodeHeight
  verticalSpacing, nodeWidth, nodeHeight);
g.DrawRectangle(pen, nodeG);
Rectangle nodeH = new Rectangle(nodeC.X -
  horizontalSpacing / 2, nodeC.Y + nodeHeight
  verticalSpacing, nodeWidth, nodeHeight);
g.DrawRectangle(pen, nodeH);
Rectangle nodeI = new Rectangle(nodeC.X +
  horizontalSpacing / 2, nodeC.Y + nodeHeight
  verticalSpacing, nodeWidth, nodeHeight);
g.DrawRectangle(pen, nodeI);
Rectangle nodeJ = new Rectangle(nodeD.X -
  horizontalSpacing / 2, nodeD.Y + nodeHeight
  verticalSpacing, nodeWidth, nodeHeight);
g.DrawRectangle(pen, nodeJ);
Rectangle nodeK = new Rectangle(nodeD.X +
  horizontalSpacing / 2, nodeD.Y + nodeHeight
  verticalSpacing, nodeWidth, nodeHeight);
g.DrawRectangle(pen, nodeK);
int operatorWidth = 50;
int operatorHeight = 20;
Rectangle or1 = new Rectangle(centerX - operatorWidth / 2,
  rootRect.Y + nodeHeight + 10, operatorWidth,
  operatorHeight);
Rectangle or2 = new Rectangle(nodeB.X + nodeWidth / 2 -
  operatorWidth / 2, nodeB.Y + nodeHeight + 10,
  operatorWidth, operatorHeight);
Rectangle or3 = new Rectangle(nodeC.X + nodeWidth / 2 -
  operatorWidth / 2, nodeC.Y + nodeHeight + 10,
  operatorWidth, operatorHeight);
Rectangle or4 = new Rectangle(nodeD.X + nodeWidth / 2 -
  operatorWidth / 2, nodeD.Y + nodeHeight + 10,
  operatorWidth, operatorHeight);

// Лінії з'єднань
g.DrawLine(pen, new Point(rootRect.X + nodeWidth / 2,
  rootRect.Y + nodeHeight), new Point(centerX, or1.Y +
  operatorHeight / 2));
g.DrawLine(pen, new Point(centerX, or1.Y +
  operatorHeight / 2), new Point(nodeB.X + nodeWidth / 2,
  nodeB.Y));
g.DrawLine(pen, new Point(centerX, or1.Y +
  operatorHeight / 2), new Point(nodeC.X + nodeWidth / 2,
  nodeC.Y));
g.DrawLine(pen, new Point(centerX, or1.Y +
  operatorHeight / 2), new Point(nodeD.X + nodeWidth / 2,
  nodeD.Y));
g.DrawLine(pen, new Point(centerX, or1.Y +
  operatorHeight / 2), new Point(nodeE.X + nodeWidth / 2,
  nodeE.Y));
```

```
private bool SimulateEngineControlSystemFailure()
{
    double pE = 0.010;
    double pG = 0.012;
    double pF = 0.003;
    double pH = 0.007;
    double pI = 0.005;
    double pJ = 0.006;
    double pD = 0.002;

    bool failureE = random.NextDouble() < pE;
    bool failureG = random.NextDouble() < pG;
    bool failureF = random.NextDouble() < pF;
    bool failureH = random.NextDouble() < pH;
    bool failureI = random.NextDouble() < pI;
    bool failureJ = random.NextDouble() < pJ;
    bool failureD = random.NextDouble() < pD;

    bool oilPressureFailure = failureE && failureI;
    bool powerReduction = oilPressureFailure || failureF;
    bool fuelSensorFailure = failureH && !failureJ;
    bool sensorFailure = failureG || fuelSensorFailure;
    bool overallFailure = powerReduction || sensorFailure ||
      failureD;

    return overallFailure;
}

private bool SimulateCoolingSystemFailure()
{
    double pB = 0.010;
    double pC = 0.012;
    double pD = 0.003;
    double pE = 0.005;
    double pF = 0.007;
    bool failureB = random.NextDouble() < pB;
    bool failureC = random.NextDouble() < pC;
    bool failureD = random.NextDouble() < pD;
    bool failureE = random.NextDouble() < pE;
    bool failureF = random.NextDouble() < pF;
    bool radiatorClogging = failureE || failureF; //
    bool overallFailure = failureB || failureC || failureD ||
      radiatorClogging;
    return overallFailure;
}

private bool SimulateElectricalSystemFailure()
{
    double pB = 0.010;
    double pC = 0.012;
    double pD = 0.003;
```

```
            double pF = 0.007;
            double pG = 0.006;
            double pH = 0.004;
            bool failureB = random.NextDouble() < pB;
            bool failureC = random.NextDouble() < pC;
            bool failureD = random.NextDouble() < pD;
            bool failureE = random.NextDouble() < pE;
            bool failureF = random.NextDouble() < pF;
            bool failureG = random.NextDouble() < pG;
            bool failureH = random.NextDouble() < pH;
            bool batteryFailure = failureE || failureF;
            bool shortCircuit = failureG || failureH;
            bool overallFailure = failureB || failureC || failureD ||
              batteryFailure || shortCircuit;
            return overallFailure;
        }
    }
}
```