МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління

ПРОЦИК Олександр Михайлович

Система оптимізації енергоспоживання розумного будинку / Smart home energy consumption optimization system

Спеціальність 122 – Комп'ютерні науки
Освітньо-професійна програма – Комп'ютерні науки

Дипломний проект

Виконав студент групи КН-42
О.М. Процик

Науковий керівник:
к.т.н., професор В.В. Кочан

Дипломний проект допущено до захисту
«___»_____ 2023 р.

Завідувач кафедри
_____ М.П. Комар

Тернопіль – 2023

Факультет комп’ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління
Освітній ступінь «бакалавр»
Спеціальність 122 – Комп’ютерні науки
Освітньо-професійна програма – Комп’ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ М.П. Комар
«____» _____2022 р.

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЕКТ СТУДЕНТУ

ПРОЦИК Олександр Михайлович
(прізвище, ім’я, по батькові)

1. Тема проекту: Система оптимізації енергоспоживання розумного будинку / Smart home energy consumption optimization system
керівник проекту к.т.н., проф. В.В. Кочан

затверджені наказом по університету від 08 грудня 2022 р. № 491.

2. Строк подання студентом закінченого проекту 01 червня 2023 р.

3. Вихідні дані до проекту: технічне завдання.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

– опис передумови виникнення технологій систем енергоменеджменту для "розумних" будинків;
– опис систем енергоменеджменту розумного будинку;
– опис планування енергоспоживання;
– постановка задачі та опис вимог до системи оптимізації енергоспоживання розумного будинку;
– опис архітектури системи та підмодулів ;
– розробка алгоритмічного забезпечення системи ;
– розробка інформаційного забезпечення системи оптимізації енергоспоживання розумного будинку
– вибір програмного та апаратного забезпечення для реалізації системи;
– розробка інтерфейсу користувача для системи оптимізації

енергоспоживання ;

– проведення тестування системи оптимізації енергоспоживання;

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

– схема структурна взаємозв'язку частин системи;

– блок-схеми алгоритмів розумного сенсора, актуатора та контролю освітлення.

6. Консультанти розділів проекту

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв
Н. контроль	к.т.н., професор В.В. Кочан		

7. Дата видачі завдання 08 грудня 2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту	Строк виконання етапів проекту	Примітка
1	Управління енергоспоживанням розумного будинку	30.12.2022	
2	Структура системи оптимізації енергоспоживання розумного будинку	24.03.2023	
3	Програмно-технологічне забезпечення системи оптимізації енергоспоживання розумного будинку	12.05.2023	
4	Повне завершення та оформлення дипломного проекту	01.06.2023	

Студент_____ О.М. Процик
(підпис)

Керівник проекту_____ В.В. Кочан
(підпис)

РЕФЕРАТ

Пояснювальна записка до дипломного проекту: 64 с., 35 рис., 4 додатки, 39 джерел.

Метою дипломного проекту є розробка системи оптимізації енергоспоживання розумного будинку. Основне призначення даної системи – це ефективний енергоменеджмент будинку за допомогою впровадження концепції IoT..

В роботі використовувалися методи системного аналізу та синтезу, порівняння, логічного узагальнення результатів, методи інтелектуального аналізу даних.

Розроблено архітектуру системи та підмодулів, основні алгоритми функціонування системи: a)розумного сенсора; b) розумного актуатора; c) управляючого модуля (мікроконтролера); d) IoT шлюзу; e) Скриптів роботи з БД. Розроблено структуру бази даних для системи оптимізації енергоспоживання розумного будинку.

Розгорнуто та створено графічний інтерфейс користувача на базі середовища ThingsBoards. Розроблено графічний інтерфейс для мобільного додатку за допомогою сервісу MIT App Inventor

Ключові слова: РОЗУМНИЙ БУДИНОК, ЕНЕРГОМЕНЕДЖМЕНТ, IOT, РОЗУМНИЙ СЕНСОР, РОЗУМНИЙ АКТУАТОР, IOT ШЛЮЗ, THINGSBOARDS.

ABSTRACT

The bachelor’s thesis report: 64 pages, 35 figures, 4 appendices, 39 sources.

The aim of the thesis project is to develop a system for optimizing the energy consumption of a smart home. The main purpose of this system is to ensure efficient energy management of the building by implementing the IoT concept.

The methods of system analysis and synthesis, comparison, logical generalization of results, and data mining methods were used in the work.

The architecture of the system and submodules, the main algorithms of the system functioning were developed: a) smart sensor; b) smart actuator; c) control module (microcontroller); d) IoT gateway; e) database scripts. The database structure for the smart home energy optimization system was developed.

A graphical user interface based on the ThingsBoards environment was deployed and created. A graphical user interface for a mobile application was developed using the MIT App Inventor service.

Keywords: SMART HOME, ENERGY MANAGEMENT, IOT, SMART SENSOR, SMART ACTUATOR, IOT GATEWAY, THINGSBOARDS.

ТЕХНІЧНЕ ЗАВДАННЯ

1. НАЙМЕНУВАННЯ ТА ОБЛАСТЬ ЗАСТОСУВАННЯ

1.1 Система оптимізації енергоспоживання розумного будинку.

1.2 Область застосування – системи енергоменеджменту розумних будинків.

2. ОСНОВА ДЛЯ РОЗРОБЛЕННЯ

Основою для розроблення є завдання на дипломний проект, затверджене кафедрою інформаційно-обчислювальних систем і управління факультету комп’ютерних інформаційних технологій Західноукраїнського національного університету.

3. ПРИЗНАЧЕННЯ РОЗРОБЛЕНОГО КОМПЛЕКСУ

Метою дипломного проекту є є розробка системи оптимізації енергоспоживання розумного будинку.

4. ДЖЕРЕЛА РОЗРОБЛЕННЯ

Джерелами даної розробки є матеріали навчальної і реферативної літератури, технічна документація, науково-дослідні статті, журнали, Інтернет.

5. ТЕХНІЧНІ ВИМОГИ

5.1 Основні функціональні вимоги до програмної системи:

- збір даних;
- попередня обробка даних для перетворення у необхідний формат;
- надання графічного інтерфейсу;
- планування енергоспоживання.

5.2 Вимоги до апаратних засобів:

- Модуль датчика напруги;

- датчик струму SCT-013
- датчик струму з ефектом Холла ACS712
- ESP32 DevKit
- Raspberry PI

5.3 Вимоги до програмних засобів:

- Симулятор Wokwi;
- ThingsBoard;
- Середовище MIT App Inventor.

6. ПОРЯДОК КОНТРОЛЮ

6.1 Представлення дипломного проекту на попередній захист.

6.2 Представлення дипломного проекту на захист.

Завдання прийняв до виконання _____ О.М. Процик

(підпис) (прізвище та ініціали)

Керівник дипломного проекту_____ В.В. Кочан

(підпис) (прізвище та ініціали)

ЗМІСТ

Вступ 9
1 Управління енергоспоживанням розумного будинку 11
1.1 Передумови виникнення технологій систем енергоменеджменту для "розумних" будинків 11
1.2 Системи енергоменеджменту розумного будинку 13
1.3 Планування енергоспоживання 15
1.4 Постановка задачі та опис вимог до системи оптимізації енергоспоживання розумного будинку 19
2 Структура системи оптимізації енергоспоживання розумного будинку 21
2.1 Архітектура системи та підмодулів 21
2.2 Алгоритмічне забезпечення системи 28
2.3 Інформаційне забезпечення системи оптимізації енергоспоживання розумного будинку 34
3 Програмно-технологічне забезпечення системи оптимізації енергоспоживання розумного будинку 38
3.1 Вибір програмного та апаратного забезпечення для реалізації системи 38
3.2 Інтерфейси користувача для системи оптимізації енергоспоживання ... 44
3.3 Тестування системи оптимізації енергоспоживання 49
Список використаних джерел 55
Додаток А текст програми мікроконтролера відправки даних на IoT шлюз .. 59
Додаток Б Блок-програма мобільного додатку 62
Додаток В Схема структурна взаємозв'язку частин системи 63
Додаток Г блок-схеми алгоритмів розумного сенсора, актуатора та контролю освітлення 64

ВСТУП

Різні види системних компонентів, включаючи апаратні елементи, програмні алгоритми, мережеві з'єднання та датчики, повинні співпрацювати один з одним для надання різних послуг у розумному будинку. З розвитком інтелектуальних мереж, що забезпечують двосторонню комунікаційну інфраструктуру, мешканці мають можливість планувати свій графік використання електроенергії, щоб зменшити витрати на електроенергію.

Енергоменеджмент на рівні домогосподарств повинен враховувати, як зменшити навантаження на навколишнє середовище, підтримуючи при цьому спосіб життя людини. Стимули та мотивація до енергоменеджменту на рівні домогосподарств обумовлені комерційними та технічними причинами. Комерційні аспекти пропонують пасивному побутовому споживачеві бути активним на енергетичному ринку. Технічні аспекти включають в себе згладжування піків, зміщення навантаження, гнучку криву навантаження, стратегічне збереження та стратегічне зростання навантаження. Іншим важливим аспектом систем енергоменеджменту є соціально-економічний вплив такої системи.

Метою дипломного проекту є розробка системи оптимізації енергоспоживання розумного будинку. Основне призначення даної системи – це ефективний енергоменеджмент будинку за допомогою впровадження концепції IoT.

Об’єктом дослідження є системи енергоменеджменту в розумних будинках.

Предметом дослідження процес управління та оптимізації енергоспоживання в розумних будинках.

Методи та технології розробки. Технологією розробки було обрано сервіс ThingsBoard для реалізації графічного інтерфейсу користувача, середовище симуляції wokwi для створення програм, які працюють на МК.

Практичне значення одержаних результатів. В результаті проведеної роботи буде реалізовано: систему управлінням енергонавантаженням в залежності від вподобань користувача та відносно об’ємів енергоспоживання приладами в розумному будинку.

1 УПРАВЛІННЯ ЕНЕРГОСПОЖИВАННЯМ РОЗУМНОГО БУДИНКУ

1.1 Передумови виникнення технологій систем енергоменеджменту для "розумних" будинків

Оскільки електроенергія економічно не зберігається, оптові ціни змінюються з дня на день і зазвичай коливаються на порядок від нічних годин з низьким попитом до денних годин з високим попитом. Загалом, майже всі роздрібні споживачі наразі платять деяку середню ціну, яка не відображає фактичну ціну продажу в момент споживання [1]. Для вирішення цієї проблеми були запропоновані різні моделі диференційованого ціноутворення в залежності від часу: ціноутворення в режимі реального часу (RTP), ціноутворення на добу наперед (DAP), ціноутворення в залежності від часу використання (TOUP), ціноутворення в залежності від критичного піку (CPP) тощо.

У всіх цих варіаціях основна ідея є подвійною: по-перше, дозволити роздрібним цінам відображати коливання оптових цін для кінцевих споживачів, щоб вони платили стільки, скільки коштує електроенергія в різний час доби; по-друге, заохочувати користувачів переносити високонавантажені побутові прилади на непікові години, щоб не лише зменшити свої витрати на електроенергію, але й допомогти зменшити пікове співвідношення до середнього (PAR) у попиті на електроенергію [2-5].

Під управлінням попитом (Demand-Side Management, DSM) здебільшого маються на увазі програми, що впроваджуються комунальними компаніями для контролю за споживанням енергії на рівні домогосподарств [6]. Ці програми застосовуються для більш ефективного використання наявної енергії без встановлення нової інфраструктури генерації та передачі. Ці програми включають програми з енергозбереження та енергоефективності, програми заміщення палива, програми реагування на попит, а також програми управління побутовим або комерційним навантаженням [4,5,7]. Програми управління побутовим навантаженням зазвичай спрямовані на одну або обидві

з наступних проектних цілей: скорочення споживання та зміщення споживання [8]. Першої мети можна досягти шляхом заохочення споживачів до енергоощадливого споживання та будівництва більш енергоефективних будівель. Однак існує також потреба в практичних рішеннях для перенесення використання потужних побутових приладів на непікові години, щоб зменшити пікове та середнє співвідношення (PAR) у попиті на електроенергію.

В багатьох країнах світу існує постійна проблема неефективного використання активів з виробництва та передачі електроенергії. Наприклад, у зоні обслуговування Dominion Virginia Power приблизно 20 відсотків генеруючих потужностей використовуються лише 5 відсотків часу [9]. Цю проблему частково вирішило управління попитом.

З впровадженням "Smart Grid" з'явилася можливість здійснювати управління попитом на рівні споживача, щоб отримати більш точний контроль над наявними ресурсами. Реагування на попит визначається як "зміни у використанні електроенергії ресурсами на стороні попиту порівняно з їх звичайним режимом споживання у відповідь на зміни в ціні на електроенергію або на стимулюючі платежі, призначені для зниження споживання електроенергії в періоди високих оптових ринкових цін або коли надійність системи знаходиться під загрозою" [10]. Зазначимо , що майже 80% загального потенціалу зниження пікових навантажень припадає на програми стимулювання [10].

Через цю причину, а також через те, що для побутових споживачів не існує тарифів, що змінюються в залежності від часу, концепція РЗ базується на стимулюючій програмі, яка передбачає, що споживач отримує певні сигнали управління навантаженням від постачальника послуг. Ця концепція керування енергоспоживанням детально розглянута в [11], де представлено алгоритм керування кількома енергоємними навантаженнями в будинку для досягнення певних цілей зі зниження пікового навантаження з урахуванням

встановленого власником будинку пріоритету навантаження та бажаного рівня комфорту.

У цьому випадку власник будинку має свободу вибору, якими навантаженнями керувати і як довго. Це відрізняється від попередньо встановленої мети зниження навантаження (кВт), яку встановлює енергопостачальна компанія в програмах прямого керування навантаженням.

Для того, щоб реалізувати запропоновану функцію енергоуправління, необхідно розгорнути повністю автоматизоване рішення, що може стати можливим завдяки використанню системи домашнього енергоменеджменту (HEM). Сьогодні інтерес до систем HEM значно зріс.

1.2 Системи енергоменеджменту розумного будинку

Системи енергоменеджменту (EMS) існують в енергетичному секторі вже кілька десятиліть. Основними функціями таких систем є моніторинг, контроль та оптимізація потоку та використання енергії. Загалом, системи енергоменеджменту мають широке застосування в системах генерації, передачі та розподілу електроенергії в електричних мережах. Перші системи енергоменеджменту базувалися на аналогових лічильниках, які надавали малу по об’єму, але швидку і зрозумілу інформацію. Однак вони були обмеженими за обсягом та застосуванням [12]. На початку 1970-х років застосування швидко розвивалося. Більшість систем, поставлених до 1975 року, були засновані на Xeros Sigma 5 і Sigma 9 [13]. Технологічний розвиток у 1980-х роках ще більше змінив EMS, особливо з появою персональних комп'ютерів. Ранні розробки EMS таких виробників, як General Electric, Hitachi, Siemens і Toshiba, базувалися на власному апаратному забезпеченні та операційних системах [14].

На початку 2000-х років операційні системи, такі як UNIX, LINUX та Windows, додали багато можливостей до рішень EMS, а технологічні розробки вбудованих систем ще більше розширили функціональні можливості EMS.

Багато громіздких, зайвих у просторі твердотільних технологій поступилися місцем більш компактним, невеликим і ефективним вбудованим системам або системам на основі мікросхем [14].

EMS також має давнє застосування в житловому секторі. Використання термостату, як форми автоматизованого управління енергоспоживанням датується початком 1900-х років. Однак управління енергоспоживанням стало справжньою проблемою, особливо у зв'язку з численними енергетичними кризами, зростанням вартості та ідеєю енергозбереження у 1970-х роках [15]. Розробка функціональної та зручної для споживача EMS на рівні житлових будинків вимагає відносно іншого підходу, ніж існуючі EMS у розподільчих та передавальних мережах. Компанія Honeywell розробила унікальну систему управління сонячною енергією на основі мікропроцесорних систем для вирішення енергетичної кризи [16]. Як і будь-яка інша EMS, системи енергоменеджменту будинку мають кінцеву мету - зберегти енергію, зменшити витрати та підвищити комфорт. В основному, HEMS пропонують п'ять ключових послуг, визначених у [17], а саме: моніторинг, реєстрацію, контроль, управління та сигналізацію.

Системи домашнього енергоменеджменту (HEMS) для побутових споживачів останнім часом набули значного розвитку. Реагування на попит (DR), управління попитом (DSM), відключення пікових навантажень та перерозподіл навантаження, які розглядаються, як рішення для оператора мережі, ще більше стимулювали перехід до більш надійних та інтелектуальних HEMS.

На даний час при будівництві будівель, все частіше використовують концепцію "розумного будинку", що включає в себе обладнання домашньої системи управління енергоспоживанням (EMS), яка інтелектуально контролює навантаження домогосподарств за допомогою інтелектуальних лічильників, інтелектуальних приладів, гібридних електромобілів (PHEV), а також домашнього обладнання для виробництва та зберігання електроенергії, як показано на рисунку 1. Під час дослідження домашнього

енергоменеджменту було помічено, що його можна класифікувати або як планування споживання енергії, або як домашню мережу.

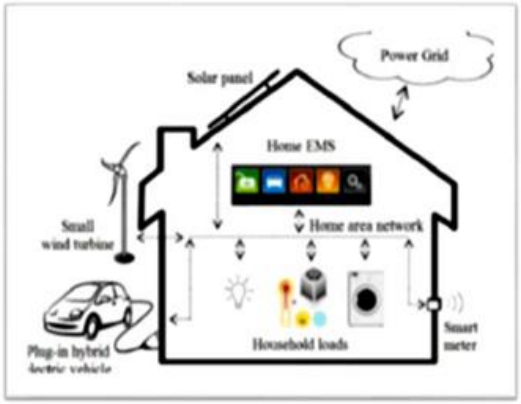


Рисунок 1.1 – Розумний будинок

1.3 Планування енергоспоживання

Також при розробці систем управління енергоспоживанням є проблема розумного планування з точки зору або з боку споживача. В роботі [18], пропонується затверджувати графік споживання електроенергії на основі RTP у поєднанні з цінами IBR, що виявилося ефективним для зменшення витрат на електроенергію в житлових будинках та зменшення загальних витрат у комунальному підприємстві.

Кілька алгоритмів оптимізації було порівняно в роботі [19], де розглядалося планування енергоресурсів у розумному будинку, і результати порівняння показали, що алгоритм лінійного програмування (ЛП) отримав найкраще рішення з меншою складністю, але не підходить для розширень, а евристичне динамічне програмування, залежне від дії (ADHDP), показало більш високу продуктивність у цьому випадку.

У [20] автор порівнює деякі методи оптимізації енергоресурсів. Планування в розумних будинках з акцентом на незалежність досяжних показників від характеристик батареї, а також ADHDP дає більше переваг в моделі батареї за рахунок зменшення розміру батареї при збереженні інших

характеристик і загальних витрат на енергію незмінними. В [21] запропоновано алгоритм оптимізації, який мав загальну цільову функцію витрат, сумісну зі стимулюванням побутових користувачів, і алгоритм був ефективним, по-перше, в економії витрат, по-друге, в енергозбереженні, по-третє, в можливості змінювати графік роботи в залежності від цін на день наперед.

У роботі [22] було вирішено кілька цілей, таких як мінімізація споживання електроенергії шляхом планування роботи приладів та перенесення використання приладів на позапікові періоди без порушення комфорту користувача, керування мережевим накопичувачем енергії та домашніми відновлюваними джерелами енергії.

Автори в [23] представили схему планування для побутових приладів, що працюють через мережу зв'язку "споживач-споживач", поєднану з інтелектуальним ціноутворенням або DLC на основі інших схем DR, що реагують на зміну навантаження. Модель прямого керування навантаженням була оцінена як 2-вимірний ланцюговий протокол Маркова.

В роботі [24] запропоновано метод часових рядів для електроспоживання індивідуальних домогосподарств з використанням протоколу визначення місцезнаходження послуг (SLP). Цей метод генерує реалістичні часові ряди стандартних профілів навантаження з високою роздільною здатністю до 1 секунди.

Автори в [25] представляють метод стохастичного планування, який базується на утиліті, що порівнює цінову інформацію, яка змінюється в часі. Ця схема дозволяє досягти точного моніторингу на 41 відсоток у порівнянні з традиційною схемою і здатна генерувати ефективний розклад за 10 секунд. Автор роботи [26] запропонував систему управління домашнім енергоспоживанням з підтримкою конфіденційності (PeHEMS), засновану на простому алгоритмі для забезпечення більш рівномірного профілю навантаження і зниження витрат на споживання. Алгоритм є менш економічним у порівнянні з TOU.

Адаптивна нейронна система нечіткого інтерфейсу (ANFIS) на основі головного контролера була запропонована в роботі [27] для врахування попиту на електроенергію користувачів в залежності від їх способу життя та соціальних факторів, що впливають на споживання електроенергії. Розроблений алгоритм планування для керування електроприладами базувався на алгоритмі з перериванням та обмеженням. Мікроконтролер (МК), що використовується для оптимізації планування роботи електроприладів, взаємодіє з клієнтом на основі ціни TOU.

В роботі [28] йдеться про енергетичний портал споживача та HEMS, які дозволяють споживачеві розуміти своє споживання електроенергії та сплачувати рахунки, контролюючи їх. За допомогою моделювання було показано, що техніка перерозподілу навантаження та ціноутворення на час користування електроенергією (TOU) може забезпечити до 12% щомісячної економії та до 50% пікової економії. Апаратний дизайн системи енергоменеджменту розумного будинку (SHRMS) представлений в [29] із застосуванням комунікаційних, сенсорних технологій та алгоритму машинного навчання. Цей проект допоміг споживачам реалізувати стратегію управління навантаженням у житлових будинках, що реагує на РТП, і взаємодіяти з постачальниками або організаціями, що обслуговують навантаження (ОНО), щоб полегшити управління з боку постачальника. В роботі [30] в якості EMS для планування електроенергії використовується домашня мережа, яка поєднує в собі RTP та Inclining Block Rate (IBR) з вподобаннями мешканців, що дозволило знизити вартість електроенергії та зменшити час затримки роботи побутових приладів, а також зменшити PAR з боку енергопостачальника, що підвищує стабільність всієї електроенергетичної системи і дає кращі результати порівняно з підходом RTP.

У роботі [31] на основі вимірювань споживання та динаміки цін було складено графік для побутової техніки. У роботі [32] описано основну мету проекту Reactive-Home, яка полягає у створенні інтелектуальної системи

оптимізації виробництва, зберігання та використання енергії в будинках. У роботі [33] описано технічне рішення пілотного проекту розумного обліку, який отримав назву "3e-hours project", що мав на меті зменшити використання енергії лише за допомогою ІКТ на 20 відсотків. Близько 100 годин було використано як практичний досвід, а технічний підхід побудований на передачі даних між "розумним" лічильником та віддаленим сервером колектора.

Система управління розумним будинком на основі протоколу Power Line Communicator представлена в [34]. Для інтеграції розетки змінного струму було розроблено модуль PLC Power Controlled Outlet Module (PPCOM). Концептуальні параметри енергоспоживання та управління пристроями вимірюються за допомогою ПЛК та мікропроцесора. В роботі [35] PSO використовується для планування завдань з експлуатаційними обмеженнями, визначеними побутовим користувачем, і алгоритм здатний зменшити PAR в попиті на енергію. Алгоритм називається плануванням пріоритету реального навантаження домогосподарства, який базується на прогнозуванні RES.

У роботі [36] для вирішення проблем з перерозподілом навантаження було створено модель прогнозованого керування (MPC), керовану подіями, для вирішення проблем з перерозподілом навантаження. Споживачеві було дозволено за допомогою додаткового управління підтримувати мінімальну добову вартість енергії в сценаріях, що характеризуються моделлю ціноутворення TOU та DSM.

Автори в [37] представили та реалізували програмований менеджер навантаження для домогосподарств, який дозволяє користувачеві або запрограмувати графік, або скласти графік навантаження для декількох навантажень, що було протестовано і дало близько 14 відсотків ефективності домогосподарства та надало можливість власнику вибирати щоденні години роботи для не проміжних навантажень в домі. Практична реалізація алгоритму під назвою "Спостерігай, вчись і переймай" (Observe, Learn and Adopt, OLA)

дозволила досягти потенційної економії близько 79 кВт-год за 30 днів у порівнянні з будинками, що не використовували OLA.

Автори в [38] запропонували схему зміщення навантаження та обмеження навантаження, яка може працювати з декількома операторами та різними типами приладів і є підходящим рішенням для управління коливанням цін в умовах ціноутворення в реальному часі та зниження вартості енергії шляхом перенесення деяких навантажень на періоди з низькими цінами та обмеження споживання в періоди з високими цінами.

1.4 Постановка задачі та опис вимог до системи оптимізації енергоспоживання розумного будинку

Опираючись на аналіз предметної області та систем енергоменеджменту які представлені в наукових роботах та компаніями виробниками було складено вимоги, яким повинна відповідати система оптимізації енергоспоживання розумного будинку та опис основного функціоналу для неї який описано нижче.

- моніторинг та контроль споживання енергії та мінімізація витоків енергії в розумному будинку;
- використання домашньої мережі (HAN);
- використання датчиків присутності;
- можливість управління через мобільний додаток;
- можливість використання безпровідного зв’язку між модулями.

Сама система повинна обробляти інформацію для енергозбереження та повідомляти про випадки втрати енергії мешканців, наприклад, виключення світла в кімнаті, якщо люди вийшли з кімнати та перейшли в іншу.

Крім того користувачі можуть відстежувати енергоспоживання побутових приладів за допомогою своїх мобільних пристроїв та керувати приладами через систему HEMS Коли сигнал зміни стану буде надходити від

електроприладів, сервер обиратиме один з двох режимів: ручний або автоматичний.

У ручному режимі призначений користувач обирає пристрої, якими він буде керувати у відповідь на сигнал, коли сигнал надходить.

В автоматичному режимі пристрої визначатимуться користувачем заздалегідь та буде вказуватись поведінка в разі отримання нового сигналу

Так як метою дипломного проекту є створення системи оптимізації енергоспоживання розумного будинку, то для досягнення мети необхідно вирішити такі завдання:

- дослідити предметну область і сформувати вимоги до системи;
- розробити архітектуру системи та алгоритми її функціонування;
- розробити інтерфейс користувача та програмне забезпечення для керуючих мікроконтролерів;
- провести тестування системи.

2 СТРУКТУРА СИСТЕМИ ОПТИМІЗАЦІЇ ЕНЕРГОСПОЖИВАННЯ РОЗУМНОГО БУДИНКУ

2.1 Архітектура системи та підмодулів

Враховуючи поставленні завдання в розділі 1.4 планується застосувати клієнт серверну архітектуру для даної системи.

На рисунку 2.1 представлено клієнт-серверну архітектуру для використання в різних галузях (універсальну).

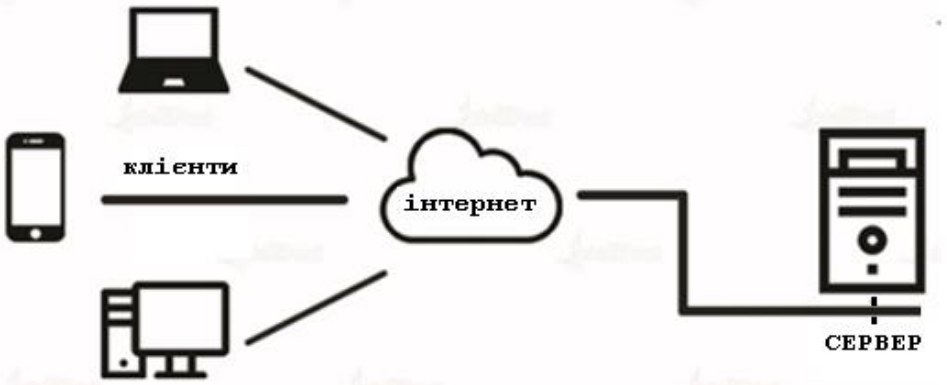


Рисунок 2.1 – Спрощена клієнт – серверна архітектура

Клієнт-серверна архітектура працює за аналогією із замовленням товару з доставкою. Клієнт телефонує в магазин, щоб замовити товар, і хтось відповідає на дзвінок, приймає замовлення, а потім доставляє його. Ця аналогія майже повністю відповідає фундаментальному принципу архітектури клієнт-сервер.

Простіше кажучи, тут задіяні два фактори:

- Сервер - це той, хто надає запитувані послуги.

- Клієнти - це ті, хто запитує послуги.

Сама ж архітектура клієнт-сервер - це обчислювальна модель, в якій сервер розміщує, надає і керує більшістю ресурсів і послуг, запитуваних клієнтом. Вона також відома як мережева обчислювальна модель або мережа

клієнт-сервер, оскільки всі запити та послуги надаються через мережу. Архітектура або модель клієнт-сервер має інші системи, підключені через мережу, де ресурси розподіляються між різними комп'ютерами.

Зазвичай архітектура клієнт-сервер влаштована таким чином, що клієнти в тому числі і «розумні» пристрої часто знаходяться працюють, як робочі станції, а сервери розташовані в іншому місці мережі, зазвичай на більш потужних обчислювальних вузлах. Така модель вигідна, коли клієнти і сервер виконують рутинні завдання. Наприклад, якщо взяти IoT пристрої то для них рутинною роботою буде опитування сенсорів, конвертація даних, надсилання даних у хмару чи до IoT - роутера, а на серверному комп'ютері може бути запущена програма для отримання та управління базою даних, аналізу надійшовших показників та формування відповідей до пристрою згідно логіки управління. Крім того сервер обслуговує і «звичайних» клієнтів які працюють/комунікують/керують пристроями IoT через різні додатки, що теж рахуються в загальному клієнтами.

Компоненти архітектури клієнт-сервер:

Для того, щоб архітектура клієнт-сервер працювала, потрібні три компоненти. Ці три компоненти - робочі станції, IoT пристрої, розумні пристрої, сервери та мережеві пристрої.

Робочі станції або IoT пристрої називають клієнтами, вони працюють у підпорядкуванні серверів і надсилають їм запити на доступ до спільних файлів ,баз даних або інформацію про виміряні фізичні величини. Сервер запитує інформацію від клієнта і виконує кілька функцій, як центральне сховище файлів, програм, баз даних та аналізу даних. Клієнти керуються політиками, визначеними сервером.

Сервери (хмара або кластер) - це швидкодіючі обчислювальні вузли, які діють як централізовані сховища мережевих файлів, даних, програм, баз даних і політик. Сервери достатньо великий простір для зберігання даних і надійну пам'ять, щоб обробляти запити, які надходять одночасно від різних клієнтів (клієнтських програм). Сервери можуть виконувати багато ролей, таких як

поштовий сервер, сервер баз даних, файловий сервер, аналітична платформа і контролер домену, в архітектурі клієнт-сервер одночасно.

Найпростішою архітектурою є однорангова (рисунок 2.2). В ній ключовими елементами є тільки два - це клієнт і сервер.

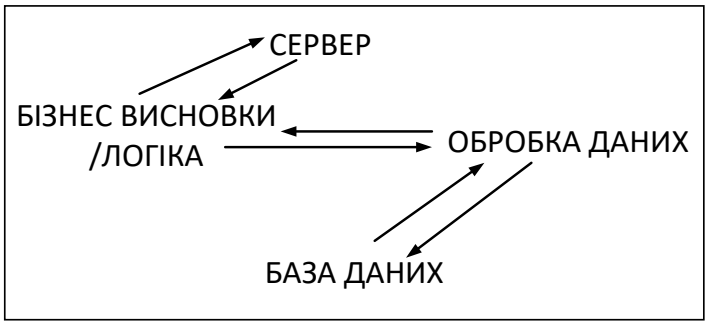


Рисунок 2.2 – Однорангова клієнт – серверна архітектура

У цій категорії архітектури клієнт-сервер містить всі види налаштувань, такі як налаштування конфігурації та логіку, на одному пристрої. Хоча різноманітність послуг, які пропонує однорівнева архітектура, робить її одним з надійних джерел, працювати з такою архітектурою складно. В першу чергу це пов'язано з різноманітністю даних. Це часто призводить до дублювання роботи. Однорівнева архітектура складається з декількох рівнів, таких як презентаційний рівень, бізнес-рівень і рівень даних, які об'єднуються за допомогою унікального програмного забезпечення. Така архітектура підходить наприклад до IoT пристроїв які працюють автономно і взаємодіють з користувачем безпосередньо тільки через мережу

До більш складнішої відноситься 2-рівнева архітектура (рисунок 2.3). Ця архітектура має найкраще середовище. У ній інтерфейс «користувача» зберігається на стороні клієнта, а база даних - на сервері, тоді як логіка бази даних і бізнес-логіка підтримується або на стороні клієнта, або на стороні сервера.

Дворівнева архітектура є швидшою порівняно з однорівневою архітектурою; це тому, що дворівнева не має жодного посередника між клієнтом і сервером. Її часто використовують, щоб уникнути плутанини між

клієнтами. Наприклад, коли використовується розумний лічильник електроенергії, то доцільно, щоб взаємодія з користувачем була не безпосередньо з пристроєм а якимось проміжним вузлом – домашнім сервером на базі мікрокомп'ютера.

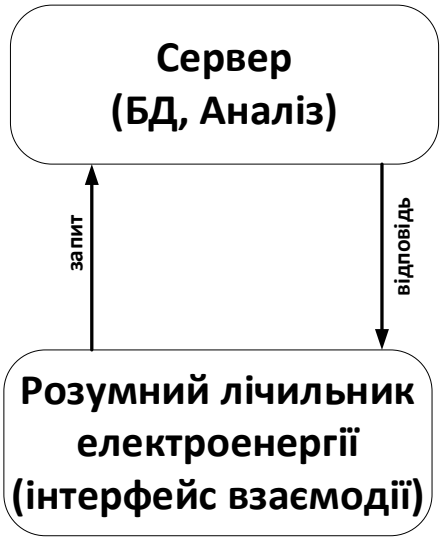


Рисунок 2.3 - Дворівнева архітектура

На відміну від дворівневої архітектури, яка не має посередника, у трирівневій архітектурі (рисунок 2.4) клієнт-сервер між клієнтом і сервером знаходиться проміжне програмне забезпечення.

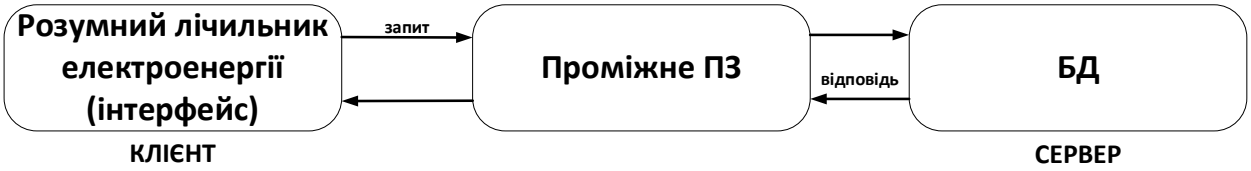


Рисунок 2.4 – Трьох рівнева архітектура

Якщо клієнт надсилає запит на отримання певної інформації з сервера, запит спочатку отримує проміжне програмне забезпечення. Потім він буде відправлений на сервер для подальших дій. Такий самий шаблон буде дотримуватися, коли сервер надсилає відповідь клієнту. Структура 3-рівневої

архітектури поділяється на три основні рівні: рівень представлення, рівень додатків та рівень бази даних.

Всі три рівні контролюються з різних кінців. У той час, як рівень представлення контролюється на клієнтському пристрої, проміжне програмне забезпечення і сервер обробляють рівень додатків і рівень бази даних відповідно. Завдяки наявності третього рівня, який забезпечує контроль даних, 3-рівнева архітектура є більш безпечною, має невидиму структуру бази даних і забезпечує цілісність даних.

Багаторівнева архітектура (рисунок 2.5) є масштабованою формою трьох інших типів архітектури. Ця архітектура передбачає розміщення кожної функції як ізольованого рівня, що включає в себе функції представлення, обробки додатків та управління даними.

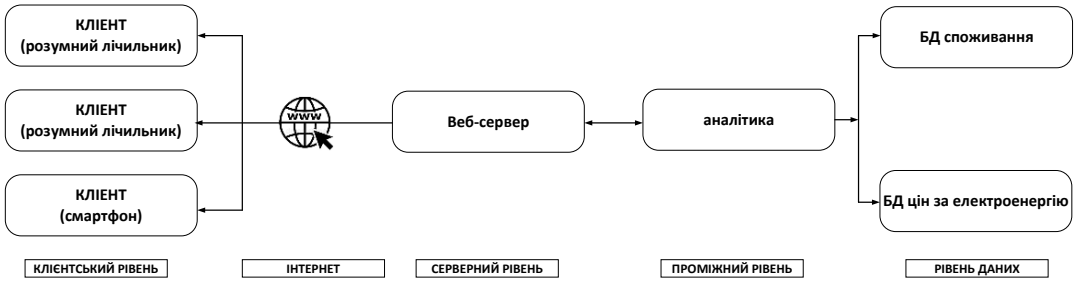


Рисунок 2.5 Багаторівнева архітектура

Загальна архітектура системи представлена на рисунку 2.6.

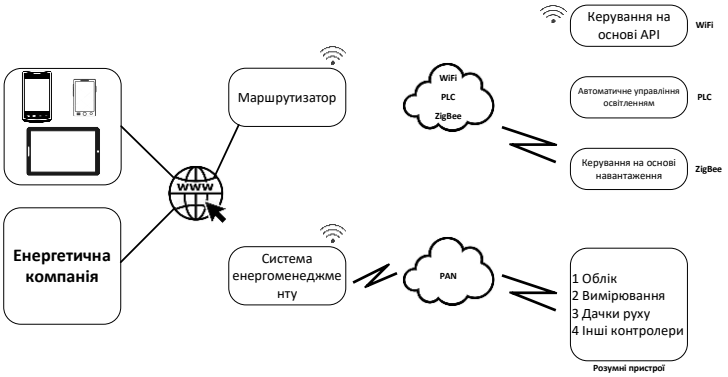


Рисунок 2.6 – Загальна архітектура системи

Як видно з загальної архітектури одним з елементів який запускається самим першим є сенсори струму, виявляє сусідню станцію(модуль) та надсилає запит на асоціацію до HEMS. Потім HEMS повідомляє новий SP призначеному резиденту. В даному представлені резидент є адміністратором HEMS. Він за допомогою свого мобільного пристрою вводить інформацію для розумних лічильників та сенсорів в HEMS, де кожному РЛ/сенсору присвоюється відповідний прилад. Після того, як інформація встановлена, РЛ надсилає дані обліку для приладу в HEMS, періодично або на вимогу, як це встановлено мешканцем. Після цього всі уповноважені мешканці можуть контролювати та управляти приладами.

Датчики присутності визначають і надсилають стан присутності певної зони, і HEMS може керувати світлом відповідно до заданого правила. Наприклад, якщо нікого немає, HEMS автоматично вимикає світло або надсилає повідомлення про це призначеному мешканцю.

Сенсори (SP) - це новий елемент розумного будинку, який збирає дані про енергоспоживання та керує приладами. SP надсилають миттєві та накопичені дані про енергоспоживання для кожного приладу до HEMS, а вона зберігає та керує даними для кожного приладу та за кожен період часу. Потім користувачі можуть відстежувати та шукати інформацію про енергоспоживання приладів у HEMS. Користувачі також контролюють прилади, щоб запобігти витоку енергії, а витрати на електроенергію можна заощадити, зберігаючи при цьому комфорт користувачів.

Опираючись на архітектуру системи та опису функціоналу було вибрано багаторівневу архітектуру яка представлення на рисунку 2.7.

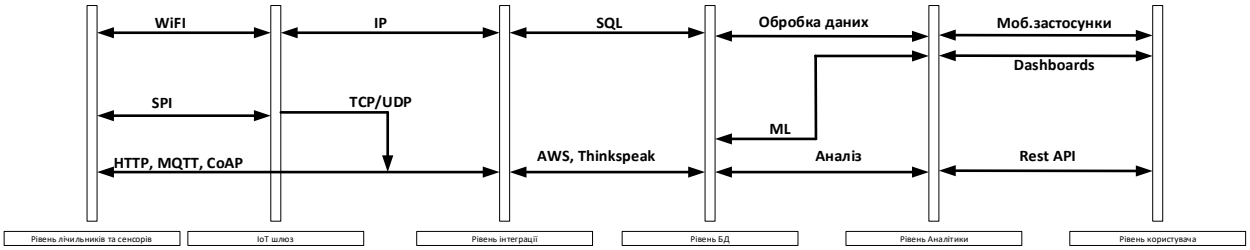


Рисунок 2.7 – Багаторівнева архітектура системи оптимізації енергоспоживання розумного будинку

Згідно рисунку система буде поділятись на 6 рівнів і буде працювати за наступним принципом:

Рівень лічильників і сенсорів – на даному рівні забезпечується комунікація між сенсорами, лічильниками з IoT шлюзом. Зв’язок може організовуватись доступними засобами, такими як бездротовий та дротовий зв'язок на рівні спеціальних протоколів, так і на рівні інтернет протоколу. Це залежить від того який тип сенсорів та лічильників імпульсів буде використовуватись в системі.

Рівень IoT шлюзу. Даний рівень може бути, але якщо сенсори мають прямий зв'язок з сервером та підтримують ряд спеціальних протоколів верхнього рівня, то цим рівнем можна знехтувати. В більшості випадків виробники обладнання і системні адміністратори або адміністратори мереж рекомендують використовувати шлюз, тому що він забезпечує захист сенсорів і розумних пристроїв від несанкціонованого доступу, може робити попередню обробку даних (фільтрація) та формувати запити до БД вкладаючи в них дані з сенсорів і лічильників.

Рівень інтеграції тісно працює з IoT шлюзом та хмарною або туманно інфраструктурою. На даному рівні забезпечується узгодження даних від шлюзу, доставка пакетів по мережі і виявлення помилок, створення сеансів між хмарою та розумними пристроями для передачі даних.

Рівень БД орієнтований на комунікацію з різними типами СУБД в силу того, що різні хмарні та туманні сервіси використовують різноманітне ПЗ. На даному рівні також відбувається зберігання, сортування, пошук даних.

На рівні аналітики проводиться аналіз масивів даних, які збережені в БД, аналізується поведінка об’єктів, встановлюються сценарії поведінки системи відповідно до умов, які склались. Також на цьому рівні відбувається пошук аномалій за допомогою машинного навчання для вироблення відповідних керуючих команд.

Останнім рівнем в даній архітектурі є рівень користувача, що забезпечує взаємозв’язок з клієнтською стороною, а саме мобільними застосунками на смартфонах та інших засобах комунікації. Цей рівень також забезпечує надання інтерфейсів користувача для роботи з системо. Також є можливість додавання API системи для інтеграції в більш глобальніші системи, наприклад систем розумних вулиць чи міст.

2.2 Алгоритмічне забезпечення системи

Так як система розділяється на 7 рівнів то і потрібно розробити алгоритми функціонування ПЗ на даних окремих рівнях. На першому рівні знаходяться сенсори, лічильники та актуатори – це рахується найнижчим рівнем де працює апаратне забезпечення, яке керується спеціальним ПЗ. До основних функцій розумного сенсора в кімнаті буде відноситись.

- опитування аналогового сенсора;
- перетворення рівня падіння напруги в фізичну величину;
- формування пакету даних або запиту до IoT шлюзу;
- інкапсуляція даних в пакет;
- встановлення з’єднання
- відправка даних;
- завершення передачі;
- очікування відповіді (у випадку актуатора);

- продовження роботи по збору даних з аналогового сенсора/сенсорів. На рисунку 2.8 представлено псевдокод роботи розумного сенсора в складі системи.

```
старт
налаштування
     налаштування аналогово-цифрового перетворювача
     налаштування інтерфейсу передачі даних
вихід з налаштування
вхід в безкінечний цикл
     читання даних з АЦП
          якщо дані <=0
               виклик функції передачі даних через інтерфейс
               поміщення коду помилки
          якщо інакше дані == Nan
               виклик функції передачі даних через інтерфейс
               поміщення коду про вихід з ладу АЦП
          інакше
               виклик функції передачі даних через інтерфейс
               отримання рівня АЦП
               конвертація рівня в фізичну величину
               відповідно до параметрів сенсора
               формування пакету даних
               вставлення конвертованих значень в пакет
               завершення сеансу передачі
перехід вгору до безкінечного циклу
```

Рисунок 2.8 – Псевдокод функціонування розумного сенсора

Якщо завдання розумного сенсора передавати дані дальше по ланцюжку в структурі, то актуатори працюють за іншим алгоритмом. До основних функцій актуатора відноситься:

- перевірка стану працездатності;
- з’єднання з сервером;
- отримання управляючих команд;
- встановлення механізму відповідно команд;
- відправка нового стану;
- завершення сеансу з’єднання.

На рисунку 2.9 показано алгоритм функціонування управляючого модуля разом з контролером управління для відключення освітлення в кімнаті коли всі особи покинули відповідну територію.

```
старт
налаштування
    налаштування GPIO відповідно до встановленого УМ
    налаштування інтерфейсу передачі даних
вихід з налаштування
вхід в безкінечний цикл
    перевірка стану УМ
    якщо модуль не відповідає
        формування коду помилки
        виклик функції передачі даних через інтерфейс
        поміщення помилки в пакет даних
        відправка даних
    інакше якщо модуль відповідає
        виклик функції передачі даних через інтерфейс
        очікування команди
            якщо команда =<0
                перехід до безкінечного циклу
            якщо інакше команда >0
                отримання команди
                встановлення УМ відповідно команди
                виклик функції передачі даних через
                інтерфейс
                передача нового стану до IoT шлюзу
            інакше
                повернення безкінечний цикл
```

Рисунок 2.9 – Псевдокод функціонування актуатора

До основних функцій IoT шлюзу відносяться:

- отримання даних від розумних сенсорів/контролерів/лічильників;
- перепаковування пакетів даних відповідно до протоколу TCP/IP;
- передача пакетів до хмарних сервісів або серверів БД;
- отримання результатів/команд;

В загальному шлюз працює за досить простою програмою, так як основні комунікаційні протоколи реалізовані апаратно і користувач програмує по суті тільки логіку функціонування. Наприклад функція перевірки мережі повертає 0 коли мережа відсутня, 1 коли все працює, а все що більше 1 є кодом помилки яку включити в обробник помилок.

На рисунку 2.10 представлено псевдо-код для IoT шлюзу який працює в системі енергоменеджменту будинку.

```
старт
while ()
    {
    перевірка цілісності мережі для пристроїв
        Якщо мережа пристроїв працює
          {
             перевірка цілісності мережі
             якщо мережа не працює
               тримання даних від пристроїв
               запис даних у внутрішню
             інакше якщо мережа працює
               створення з'єднання з хмарою
               відправка усіх даних з тимчасового
               файлу до хмари
               отримання даних від пристроїв
               формування запиту до хмарного сервісу
               через RestAPI
               надсилання запиту на додавання нових даних
               надсилання запиту на отримання нової
               команди
               запис команди у файл
             інакше
             засинання на 5 хв
          }
         Якщо мережа пристроїв не працює
          {
          Виставлення статусу newok_err
          }
    }
```

Рисунок 2.10 – Псевдокод IoT шлюзу

Рівень баз даних забезпечує збереження, пошук та сортування даних які надійшли від сенсорів і пристроїв. Так як СУБД є го готовим продуктом з своєю мовою програмування запитів та шаблонами відповідей, то завдання зводиться до написання програми-скрипта, яка б отримувала дані від IoT шлюзу по стеку протоколів TCP/IP а саме http, отримувала результати, втавляла дані результати в запити до БД, отримувала відповіді від БД, вирізала команди/відповіді від загального тексту відповіді та інкапсулювала в відповідь по http для передачі на IoT шлюз.

До основних функцій даного рівня відноситься:

- отримання запитів від IoT шлюзу;
- отримання даних з запиту;
- підключення до БД;
- формування команди з даними до БД;
- отримання відповіді від БД;
- інкапсуляція відповіді у http відповідь до IoT шлюзу.

На рисунку 2.11 наведено псевдокод скрипта запису даних у БД та отримання відповіді.

```
старт
читання GET/POST запиту
 якщо GET/POST == “”
  {
   http відповідь “не має даних”;
  }
 інакше якщо GET/POST != “”
  {
   дані = [GET]
   $посилання = з’єднання з ВД ('адрес ВД', 'користувач',
'пароль');
    якщо (! $посилання)
     {
      вивід “не має з’єднання”
     }
    інакше
     {
      $запит = "вставити в таблицю “дані” (дані) значення
('дані')";
      якщо (! запит)
       {
        вивід 'дані не вставлено в таблицю';
       }
      інакше
       {
        $запит_команда = з’єднання з ВД (ВИВРАТИ ім’я
колонки З таблиці команди)
        вивід “дані вставлено”
        вивід “команда:команда”
       }
     }
  }
 інакше
 {
  вивід “не працює http”
 }
стоп
```

Рисунок 2.11 – Псевдокод скрипта пересилання даних до БД

Рівень аналітики забезпечує аналіз даних, реагування на події та встановлення упрвління в розумному будинку. Так як даний рівень може бути реалізований на тому самому хмарному сервісі, то є декілька варіантів реалізаціїї комунікацій:

- міжпроцесний;
- за допомогою сокетів;
- за допомогою файлів.

Міжпроцесний є найшвидший, тому що використовуються внутрішні механізми ОС і не потрібно піклуватись про надійність з’єднання і передачі, але в той самий час це найважчий спосіб в плані кодування. Комунікація через

файл є самим простим методом, але найбільш ненадійним з боку ОС. Найкращою реалізацією для архітектури яка пропонується є варіант використання сокетів, хоча він і є повільним, але так як і в першому варіанті, багато функцій бере на себе ОС.

На рисунку 2.12 представлено псевдокод програми-скрипта для автоматичного відключення освітлення в кімнаті якщо присутніх немає 1хв.

```
старт
$посилання = з'єднання з БД ('адрес БД', 'користувач',
'пароль');
 якщо (! $посилання)
  {
  вивід "не має з'єднання"
  }
 інакше
  {
   $запит = 'ВИБРАТИ тип_сенсора, рівень_освітлення,
сенсор_присутності, кімната З дані_освітлення';
   функція вибору БД('$запит');
   $отримати = функція запиту ($запит, $посилання);
    якщо ((сенсор_присутності> 10) & (рівень_освітлення>
1000))
     {
      $команда = "ВСТАВИТИ ДО команди (кімната, id, стан)
ЗНАЧЕННЯ ($кімната, $id, $команда')";
       якщо (функція запиту($посилання, $команда)) {
        вивід "команду надіслано";
      }
      інакше {
       вивід "помилка";
      }
     }
  }
стоп
```

Рисунок 2.12 – Псевдокод контролю освітлення

Крім цього в системі передбачено рівень користувача. До основних його функцій відносяться:

- реагування на команди (кнопки) користувача;
- надсилання команд до IoT шлюзу
- отримання даних з БД через IoT шлюз
- візуалізація даних
- попередження про нештатні ситуації (логіка не на аналітиці, а на самому ПЗ додатку)

2.3 Інформаційне забезпечення системи оптимізації енергоспоживання розумного будинку

В даному розділі описується інформаційне забезпечення компонентів системи. Так як в ній передбачено БД то у таблицях 2.1-2,X наведено кілька таблиць для послуг з енергоменеджменту будинку. По-перше, використовуючи таблицю інформації про пристрої, користувач може відстежувати інформацію та стан приладів. По-друге, інформаційна таблиця команд включає в себе команди для управління приладами, щоб керувати приладами з метою енергозбереження. По-третє, інформація про користувача управляється адміністратором для надання дозволу на доступ до приладів. Рівень користувача в таблиці інформації про користувача складається з двох етапів, один з яких - режим адміністратора, а інший - загальний режим.

Адміністратор має повний доступ до пристроїв, керує обліковими записами користувачів та керує пристроями, наприклад, вмикає або вимикає їх.

Таблиця профілю послуг призначена для визначення послуг, таких як DR та RTP. Використовуючи ці таблиці, HEMS може надавати користувачам послуги з енергоменеджменту та енергозбереження.

Таблиця 2.1 – Інформація про прилади

Назва поля	тип	опис
Ідентифікатор пристрою	integer	Унікальний ідентифікатор пристрою
Ім’я пристрою	string	Назва пристрою

Таблиця 2.2 – Інформаційна таблиця приладів

Назва поля	тип	опис
Зона відповідальності пристрою	string	По суті тип кімнати в будинку
Тип пристрою	string	тип пристрою
Споживання енергії пристроєм	integer	Скільки споживає енергії пристрій
Статус пристрою	string	Може бути включений або виключений

Таблиця 2.3 – Інформаційна таблиця команд

Назва поля	тип	опис
Ідентифікатор команди	integer	Унікальний код команди
Назва команди	string	тип пристрою

Таблиця 2.4 – Інформаційна таблиця користувача

Назва поля	тип	опис
Ідентифікатор користувача	integer	Унікальий ідентифікатор користувача
Ім’я користувача	string	Логін користувача
Пароль користувача	string	Пароль користувача
Рівень користувача	string	Наприклад, адміністратор, простий користувач
Мобільний пристрій користувача	string	Назва пристрою

Дата реєстрації користувача	string	Коли зареєструвався користувач
Остання дата	string	Остання дата отримання даних користувачем

Таблиця 2.5 –Таблиця профілю

Назва поля	тип	опис
Ідентифікатор сервісу	integer	Унікальний ідентифікатор сервісу
Назва сервісу	string	Ім’я сервісу
Провайдер сервісу	string	Ім’я провайдера сервісу
Версія сервісу	string	Остання версія сервісу яку обновляв користувач
Тип сервісу	string	Обов’язковий чи додатковий сервіс

На рисунку 2.13 показано, як працює послуга з дистанційного керування енергоспоживанням будинку. На рисунку HEMS отримує сигнал DR від ESP, обирає прилади та керує ними. Процедура надання послуги виглядає наступним чином.

1. ESP генерує сигнал аварійного вимкнення для запобігання відключенню електроенергії та надсилає його до HEMS.

2. HEMS виявляє сигнал і обирає один з двох типів режиму, автоматичний або ручний, який заздалегідь встановлюється адміністратором HEMS. В автоматичному режимі HEMS обирає прилади, які потрібно контролювати для послуги DR. У ручному режимі адміністратор HEMS обирає прилади вручну. 3. HEMS надсилає команду керування до СП.

4. SP керують приладами, наприклад, вимикають їх або контролюють зниження температури обігрівача.

5. SP надсилають результат роботи приладів управління до HEMS.

6. HEMS передає результати роботи приладів управління на ESP та мобільні пристрої мешканців.

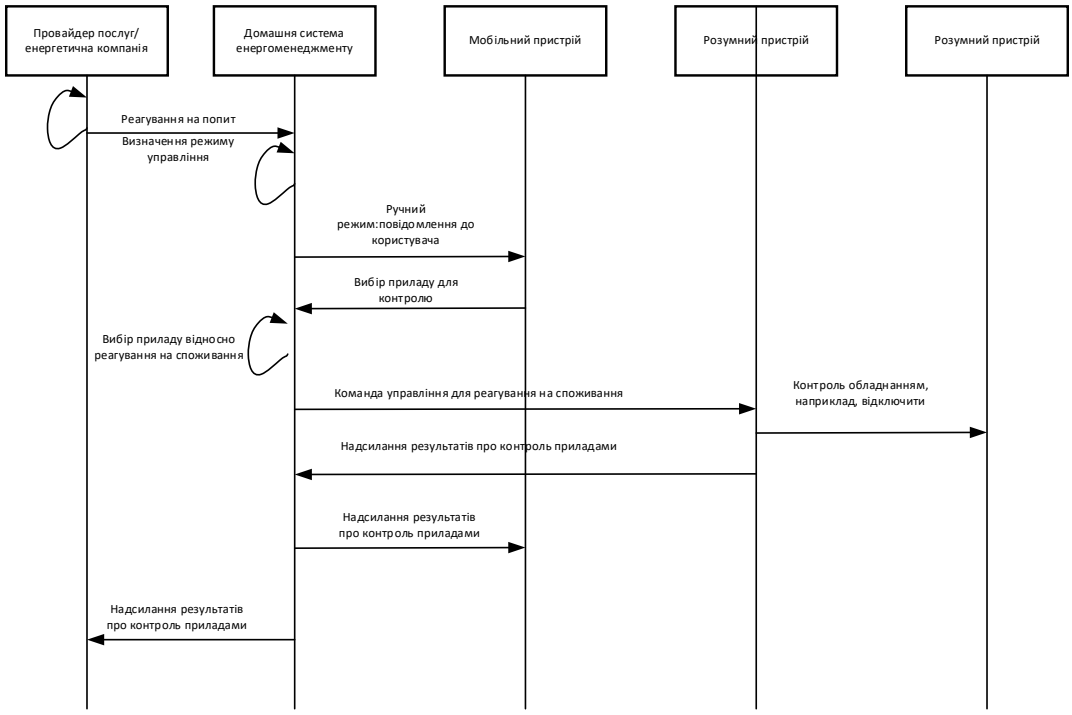


Рисунок 2.13 – Потоки послуг реагування на навантаження

3 ПРОГРАМНО-ТЕХНОЛОГІЧНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ ОПТИМІЗАЦІЇ ЕНЕРГОСПОЖИВАННЯ РОЗУМНОГО БУДИНКУ

3.1 Вибір програмного та апаратного забезпечення для реалізації системи

3.1.1 Апаратне забезпечення

Виходячи із поставленого завдання в 1 розділі, опису архітектури та основних алгоритмів функціонування системи в розділі 2, було вирішено використати наступне програмне і апаратне забезпечення.

На самому нижньому рівні в такій системі повинні працювати сенсори (аналогові, цифрові) в залежності від точності, складності або зручності передавання даних про оточуюче середовище. Так як в системі є два типи даних- це дані про освітлення та дані про споживання то для цієї мети було вибрані наступні сенсори.

Модуль датчика напруги (рисунок 3.1) Цей датчик працює за принципом дільника напруги для вимірювання напруги.

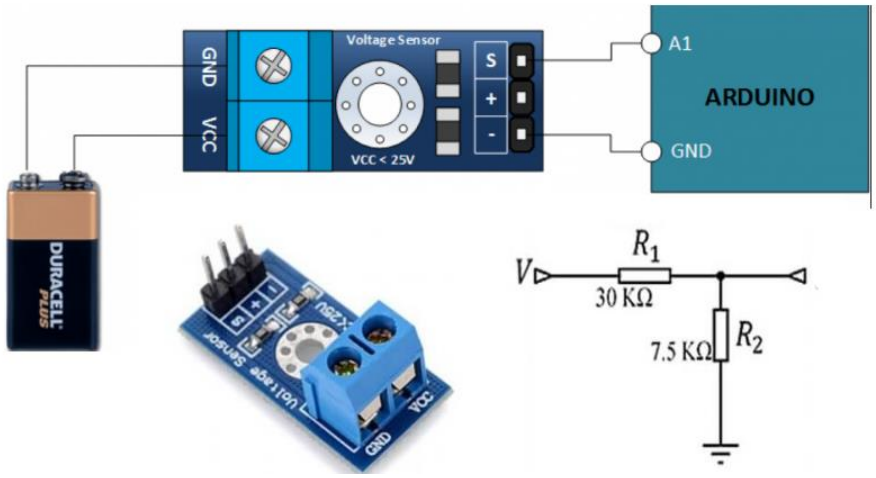


Рисунок 3.1 - Модуль датчика напруги

Він має вбудоване послідовне з'єднання резисторів 7,5 кОм і 30 кОм для досягнення коефіцієнта розподілу напруги 5 до 1 для вимірювання. Діапазон робочої напруги датчика становить 3,3 - 5,0 В, а за допомогою 12-розрядного

АЦП можна вимірювати напругу в діапазоні 0 - 25 В постійного струму. Датчики напруги зазвичай підключаються паралельно до джерела напруги (9В AC/DC адаптер або батарея) для вимірювання його напруги. ESP32 працює з вхідною напругою 3,3В. Датчик підключається безпосередньо для вимірювання напруги: Вивід S датчика напруги підключається до аналогового виводу 32 на ESP32, а вивід - до виводу GND ESP32, при цьому виводи GND і VCC датчика з'єднуються через позитивну і негативну клеми 9В/5В перетворювача, як показано на рисунку 3.1. Даний датчик можна використовувати в приладах з малою напругою, так як стандартні датчики на 220 не реагують на малу зміну струму.

Наступний сенсор, який може працювати з більшими струмами і напругами є датчик струму SCT-013 (рисунок 3.2).

Рисунок 3.2 - Датчик струму SCT-013

Датчик струму з ефектом Холла ACS712 (рисунок 3.3). Цей датчик працює за принципом ефекту Холла - генерує напругу, перпендикулярну, як до струму, так і до магнітного поля. Ця напруга використовується для вимірювання струму.

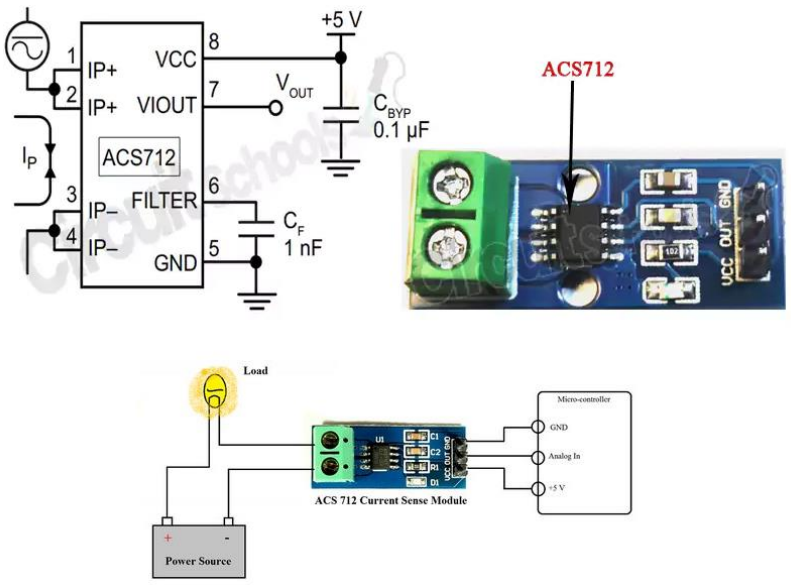


Рисунок 3.3 - Датчик струму з ефектом Холла ACS712

Це повністю інтегрований, економічний і відносно високоточний датчик, який використовує неінверсійний метод вимірювання струму шляхом вимірювання магнітного поля, створеного в провіднику. Датчик на основі ефекту Холла ACS712 може вимірювати як постійний, так і змінний струм, використовуючи провідник з низьким опором. Ланцюг, що вимірюється, і ланцюг датчика електрично ізольовані. Для його живлення потрібно 5 В постійного струму, він поставляється в трьох версіях 5A, 20A і 30A і дає 2,5 В постійного струму на виході, коли струм не виявлено. Його чутливість/масштабний коефіцієнт становить 20A і 185 мВ/A для модуля 30A.

Всі сенсори будуть працювати з платформою ESP32 DevKit (рисунок 3.4).

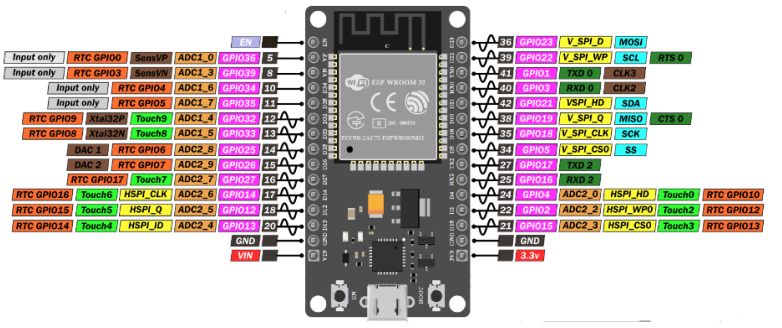


Рисунок 3.4 - ESP32 DevKit

Це потужний мікроконтролер, який підтримує Wi-Fi, Bluetooth і Bluetooth Low Energy (BLE), що використовуються в різних додатках. Радіус дії Wi-Fi ширший, ніж у Bluetooth, який виявляє модуль і з'єднує його зі смартфоном за допомогою низькоенергетичних маячків. Мікропроцесор використовує двоядерний процесор з індивідуальним керуванням, який працює на регульованій частоті від 80 МГц до 240 МГц. Для економії енергії під час виконання завдань з низьким енергоспоживанням можна використовувати співпроцесор.

Для організації аналізу, збору даних надання користувацьких інтерфейсів та зв’язку з мобільними пристроями в якості IoT шлюзу було вибрано мікрокомп’ютер Raspbery PI (рисунок 3.5).

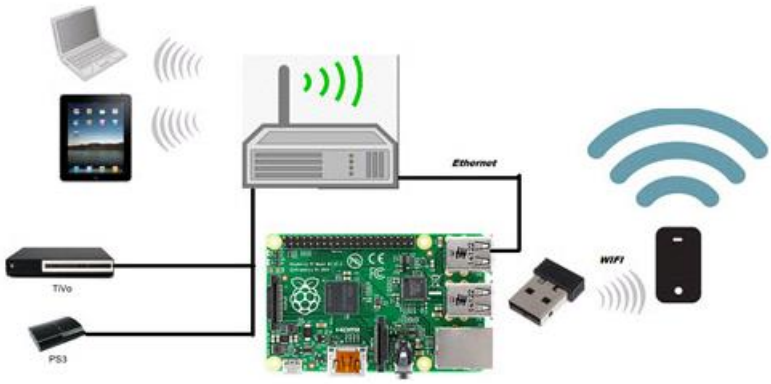


Рисунок 3.5 - Raspberry PI

Raspberry Pi. Це дійсно комп'ютер, за всіма визначеннями. Raspberry Pi - це недорогий комп'ютер розміром з кредитну картку, який підключається до монітора або телевізора за допомогою HDMI і використовує стандартну клавіатуру та мишу. На ньому може працювати безліч операційних систем, таких як Raspbian (Debian Linux), Android, Windows 10, IoT Core тощо.

3.1.2 Програмне забезпечення

Для розробки усіх додатків системи необхідно використати декілька середовищ та емуляторів для тестування. Це пояснюється тим, що система

неоднорідна і має модульну структуру, тобто різні компоненти можуть замінюватись на інші.

Для тестування модуля ESP32 було використано ресурс wokwi.com (рисунок 3.6)

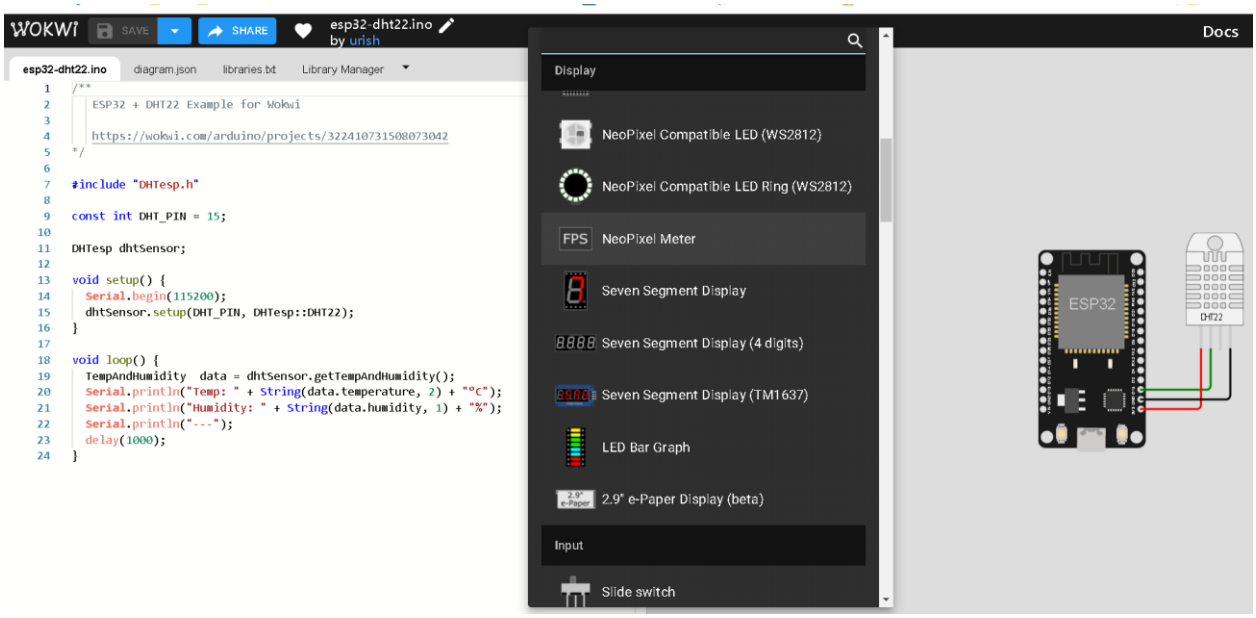


Рисунок 3.6 – Симулятор Wokwi

Wokwi - це симулятор вбудованих систем та IoT, що підтримує ESP32, Arduino та Raspberry Pi Pico. Вся робота відбувається через браузер і не потрібно ставити додаткове ПЗ. Wokwi запускає симуляцію всередині VS Code, використовуючи двійкові файли прошивки.

Наступне що потрібно – це сервіс який забезпечував би зберігання, аналіз, відображення даних. Так як планується зробити все на IoT шлюзі, то в якості такого середовища вибрано ThingsBoard (рисунок 3.7). ThingsBoard - це IoT-платформа з відкритим вихідним кодом для збору, обробки, візуалізації та управління пристроями.

Вона забезпечує підключення пристроїв за допомогою стандартних протоколів Інтернету речей - MQTT, CoAP і HTTP - і підтримує як хмарне, так і локальне розгортання. ThingsBoard поєднує в собі масштабованість, відмовостійкість і продуктивність.

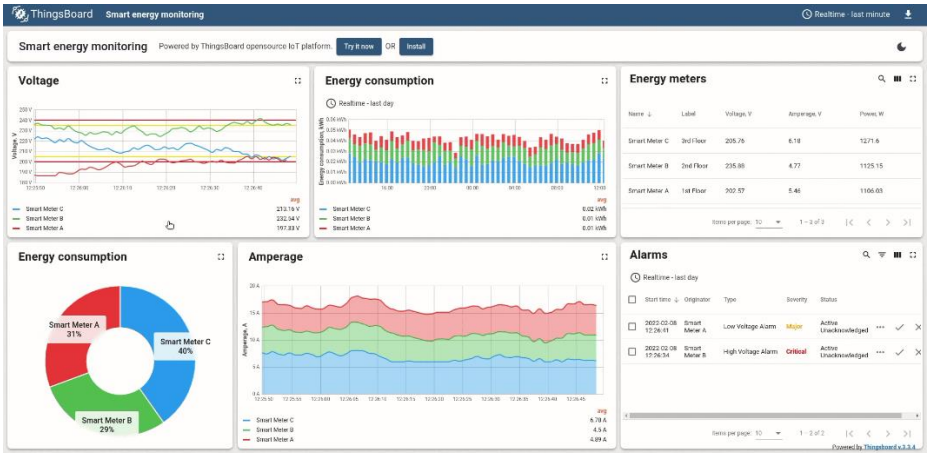


Рисунок 3.7 - ThingsBoard

В проекті також передбачено, що користувач може керувати енергонавантаженням в ручному режимі через мобільний додаток. Для цього було вибрано середовище розробки MIT App Inventor (рисунок 3.8).

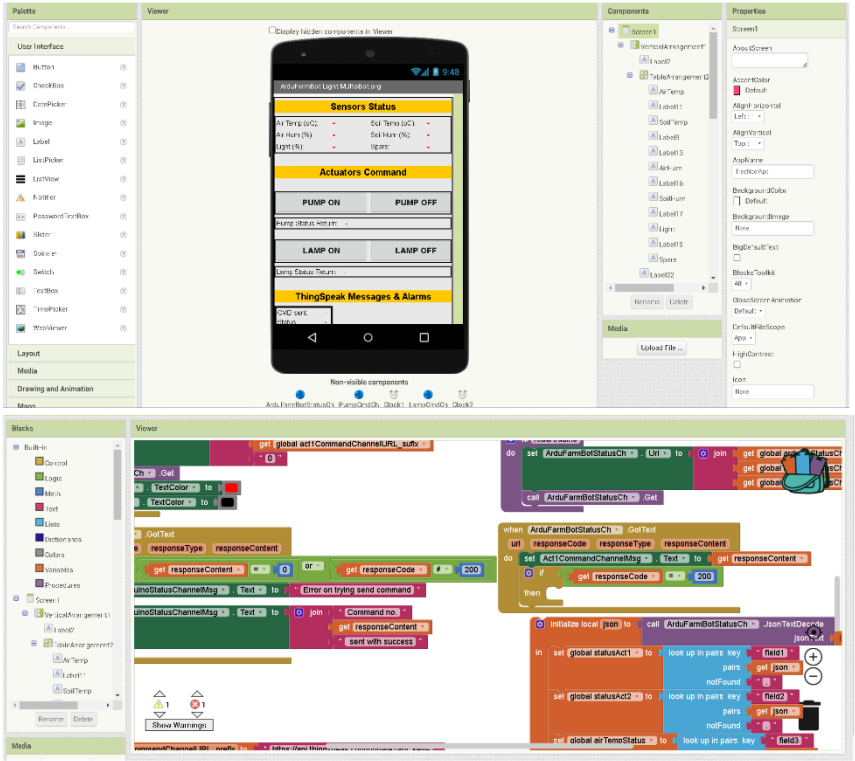


Рисунок 3.8 – Середовище MIT App Inventor

Як і з попереднім середовищем, з ним можна працювати через браузер не встановлюючи спеціалізованого ПЗ. Також - це інтуїтивно зрозуміле візуальне середовище програмування, яке дозволяє кожному створювати повнофункціональні додатки для смартфонів та планшетів. Програмування відбувається на основі блоків, що полегшує створення складних, високоефективних додатків за значно менший час, ніж у традиційних середовищах програмування.

3.2 Інтерфейси користувача для системи оптимізації енергоспоживання

3.2.1 Інтерфейс веб орієнтованого інтерфейсу на IoT шлюзі

Інтерфейс користувача складається з декількох робочих областей. Для навігації зручно використовувати меню де відображено основні елементи системи (рисунок 3.9).

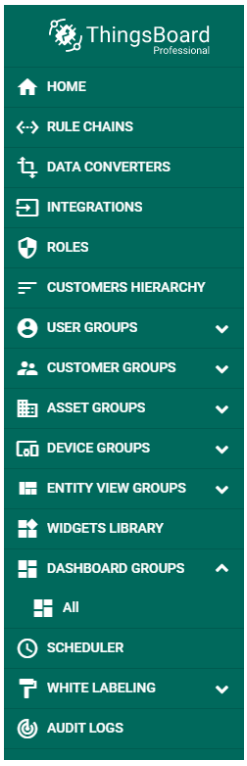


Рисунок 3.9 – Меню навігації

Тут користувач може налаштовувати весь функціонал системи – від загального вигляду до правил автоматизації. Нижче на рисунку 3.10 представлено вікно моніторингу споживання енергії певними приладами.

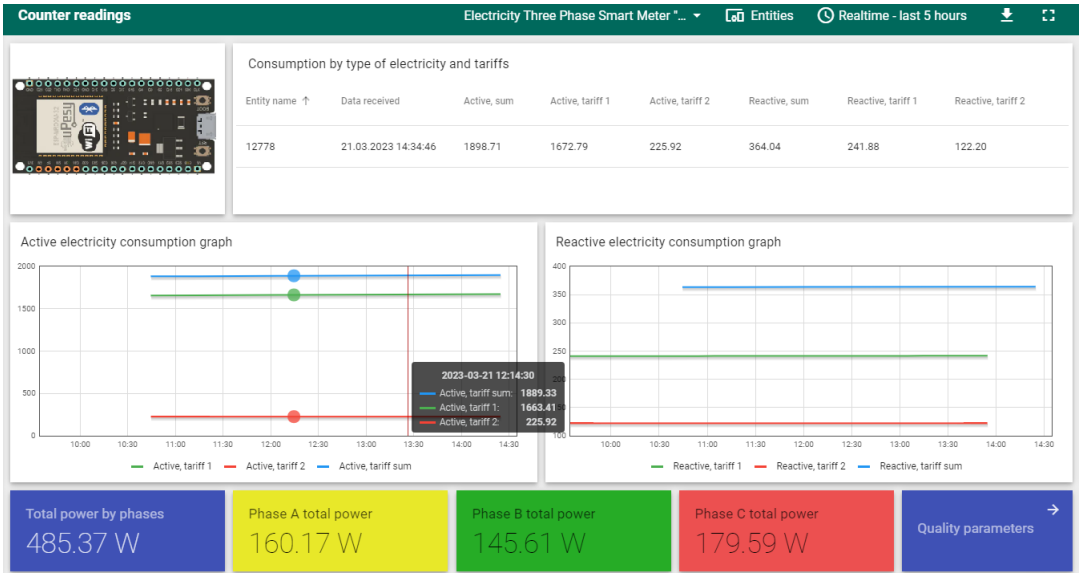


Рисунок 3.10 – Вигляд окремого модуля збору даних про електроспоживання

Наприклад користувач може переглядати споживану енергію по кожній кімнаті та контролювати її (рисунок 3.11).

Рисунок 3.11 – Вигляд контрольної панелі

Як видно з рисунку користувачу системи через цей інтерфейс доступно переглядати споживання окремих пристроїв в кімнаті та загальне використання енергії.

Користувач може в ручному режимі зупиняти та відновлювати роботу приладу (рисунок 3.12). Наприклад, як видно на рисунку користувач може спостерігати споживання певного приладу (обігрівача) і коли споживання буде максимальним, натиснувши кнопку Stop може відключити споживач від мережі 220.

Рисунок 3.12 – Управління обігрівачем

Також в системі реалізовано інтерфейс контролю різних сенсорів, наприклад присутності, температури і струму (рисунок 3.13).

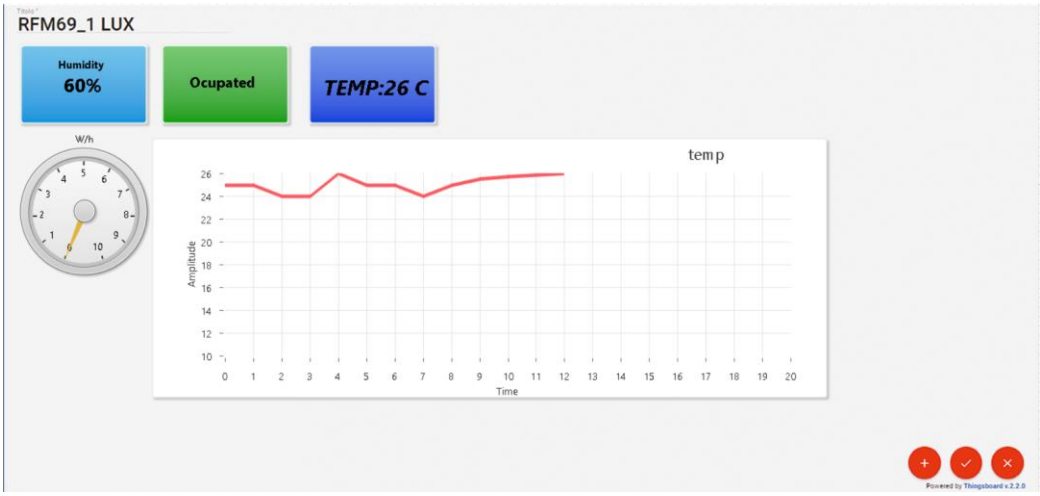


Рисунок 3.13 – Моніторинг мікроклімату в кімнатах

Вхідні дані з лічильників електроенергії мають такий вигляд

```
{
    "applicationID": "2",
    "applicationName": "Energo monitor",
    "deviceName": "0012778",
    "devEUI": "02aaaa02000031ea",
    "rxInfo": [{
        "gatewayID": "647fdafffe00d228",
        "uplinkID": "9d29d67f-8db2-4c7e-9fa8-b7f9bd5be9e6",
        "name": "tectelic_micro_lite_TECH",
        "rssi": -80,
        "loRaSNR": 5.2,
        "location": {
            "latitude": 48.44229794818326,
            "longitude": 35.014479160308845,
            "altitude": 144
        }
    }],
    "txInfo": {
        "frequency": 868500000,
        "dr": 0
    },
    "adr": true,
    "fCnt": 2202,
    "fPort": 15,
    "data": "QF8gJWwgSK4giMogyIYhQHEhgcshwTwiw1BBwABNNw=="
}
```

Рисунок 3.14 – JSON формат інтеграції віджету в графічний інтерфейс

Таким чином сервіс ThingsBoard дозволяє легко налаштовувати як вигляд інтерфейсу так і функціонал за допомогою використання JSON.

Крім того для мобільного додатку було розроблено окреме ПЗ для моніторингу та контролю в разі якщо інтерфейс IoT шлюзу буде не доступний. Загальний вигляд інтерфейсу показано на рисунку 3.15. Він не є дуже складний чи прогресивний в силу того, що середовище орієнтовано на початківців і тому робота має бути зрозумілою і не складною, тому в MIT appinventor не передбачено складних елементів, але можна організувати досить пристойний інтерфейс, наприклад можна змінювати напис на кнопці відповідно до стану приладу в кімнаті, також можна за допомогою вертикальних і горизонтальних ліній створити щось на кшталт таблиці. З одного боку даний інтрефейс не є дружнім до користувача або прогресивним, але він не обтяжує сам мобільний

пристрій, що теж є деякою перевагою, коли потрібний додаток малого обєму з високою швидкодією

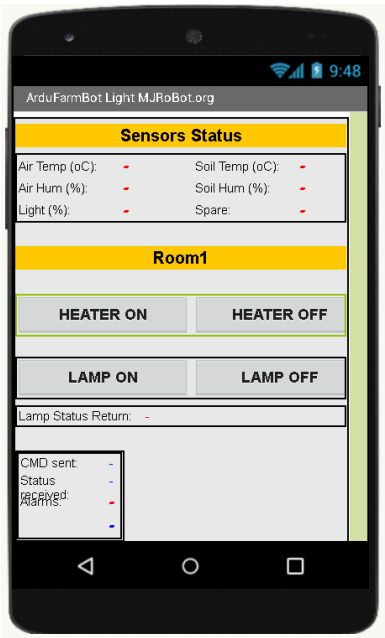


Рисунок 3.15 – Інтерфейс користувача для мобільного пристрою.

На відміну від ThingsBoard де інтерфейс користувача інтегровано в IoT шлюз, мобільний додаток спочатку приєднується до сервера сервісу, отримує поточні дані та історію та відсилає контрольні команди за допомогою POST/GET запитів (рисунок 3.16).

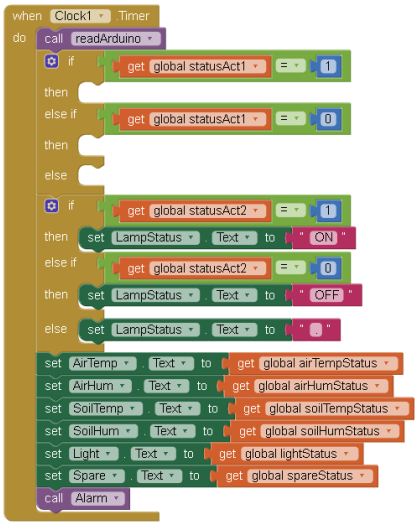


Рисунок 3.16 – Програма (блоки) отримання поточних даних від IoT шлюзу

3.3 Тестування системи оптимізації енергоспоживання

Після створення програмного забезпечення для мікроконтролерів, розгортання сервісу на IoT шлюзі та розробки мобільного додатку потрібно було протестувати основні компоненти системи на працездатність та можливість функціонування як одної системи в цілому.

На першому етапі, потрібно було провірити чи контролери працюють нормально і чи передають дані в систему. Для цього передбачено два етапи, на першому провіряється цілісність мережі від контролера до шлюзу (рисунок 3.17).

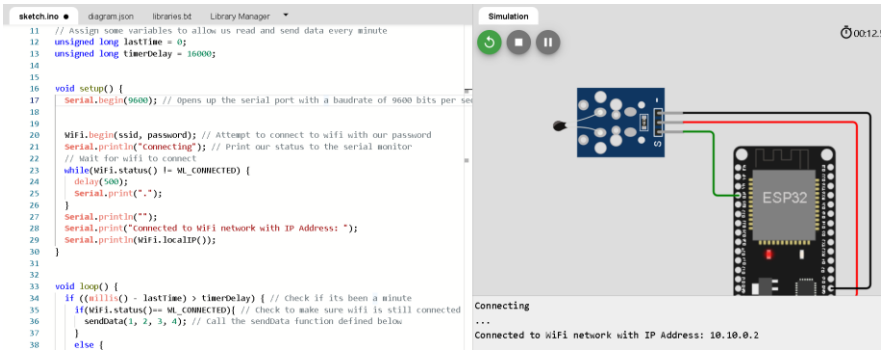


Рисунок 3.17 – Тестування цілісності мережі

На другому етапі тестується можливість прийому даних на IoT шлюзі (рисунок 3.18).

Рисунок 3.18 – Тестування отримання даних

Тестування проводилось двічі, спершу на симуляторі а потім через послідовний порт з реальним ПК та контролером, це було потрібно тому, що модуль який буде аналізувати мікроклімат в кімнаті чи споживання енергіє встановлюється один раз і демонтаж такого вузла не бажаний, тому мають бути виключені усі можливі помилки.

Після того як дані отримуються на стороні IoT шлюзу тестується графічний інтерфейс користувача який був розгорнутий за допомогою ThingsBoard. На рисунку 3.19 представлено головну сторінку де представлено усі вузли системи в розумному будинку.

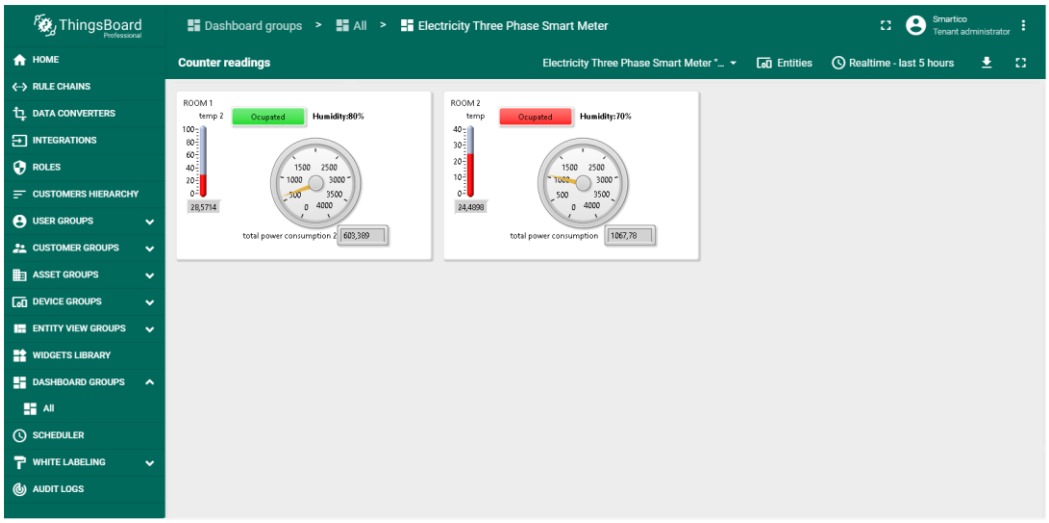


Рисунок 3.19 – Головна сторінка веб інтерфейсу користувача

Клацнувши на малому вікні де представлено основні параметри кімнати можна побачити детальну інформацію про всі прилади які споживають енергію, яка температура і вологість в кімнаті, та чи є хтось присутній зараз в ній (рисунок 3.20).

Також користувач може відключити пристрої які багато споживають або наприклад коли людина вийшла з кімнати вимкнути обігрівач для того щоб не споживати зайвої електроенергії.

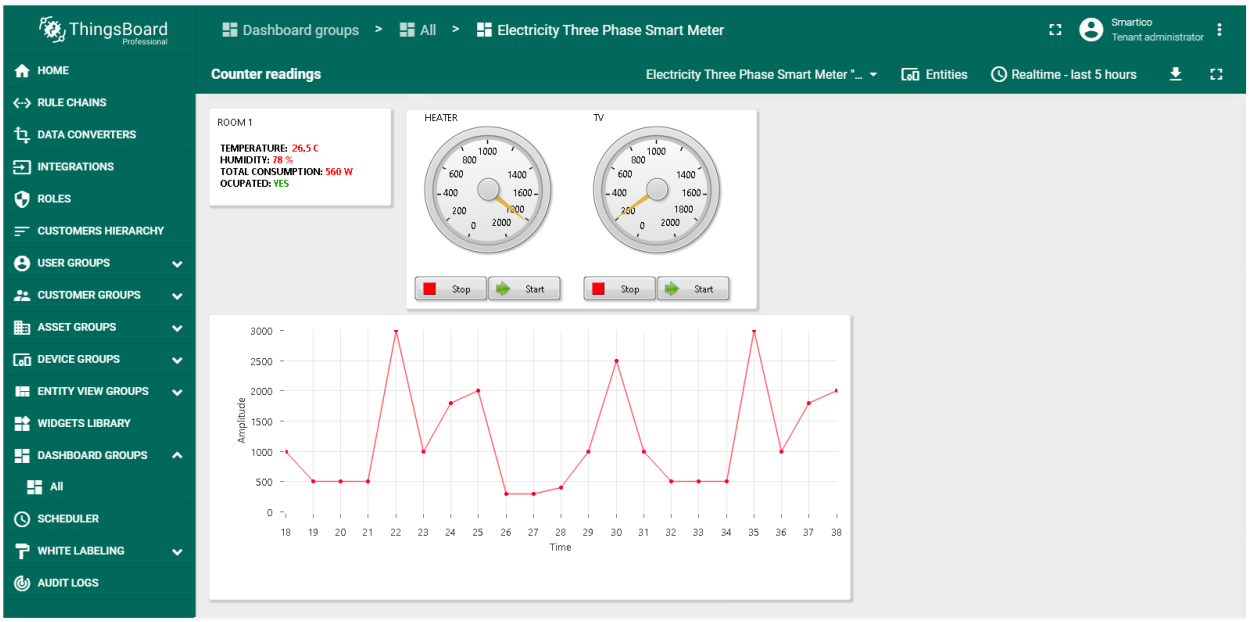


Рисунок 3.20 – Сторінка управління та аналізу окремої кімнати

Також користувач може переглядати дані про споживання та активності в будинку за допомогою мобільного застосунку, в нього менший функціонал і візуалізація (рисунок 3.21).

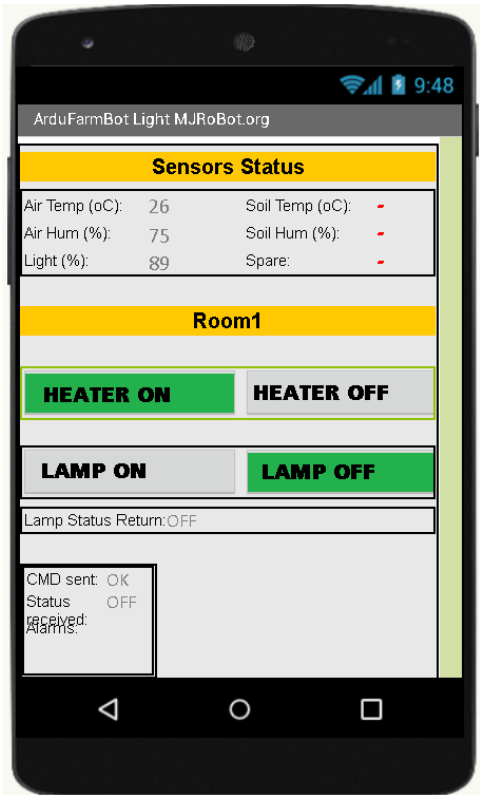


Рисунок 3.21 – Інтерфейс мобільного додатку

За допомогою додатку користувач може переглядати поточний стан в певній кімнаті, температуру, вологість і освітлення, також вмикати або вимикати прилади в ручному режимі.

ВИНОВКИ

В результаті виконання дипломного проекту було :

1. Проаналізовано передумови виникнення технологій систем енергоменеджменту для "розумних" будинків
2. Проведено аналіз систем енергоменеджменту розумних будинків
3. Проведено аналіз методів планування енергоспоживання
4. Сформульовано вимоги та набір функціоналу до системи оптимізації енергоспоживання розумного будинку
5. Розроблено архітектуру системи та підмодулів
6. Розроблено основні алгоритми функціонування системи. Так як система складається з семи рівнів то було окремо розроблено алгоритми для:
   a. розумного сенсора;
   b. розумного актуатора;
   c. управляючого модуля (мікроконтролера);
   d. IoT шлюзу;
   e. Скриптів роботи з БД;
7. Розроблено структуру бази даних для системи оптимізації енергоспоживання розумного будинку
8. Вибрано апаратні та програмні засоби для реалізації проекту
9. Реалізовано апаратну частину системи
10. Реалізовано та протестовано ПЗ для мікроконтролерів за допомогою спеціального симулятора
11. Розгорнуто та створено графічний інтерфейс користувача на базі середовища ThingsBoards
12. Розроблено графічний інтерфейс для мобільного додатку за допомогою сервісу MIT App Inventor
13. Протестовано усі модулі системи
14. Подальшим розвитком даної системи є інтеграція ще одного рівня, який би забезпечував доступ до обладнання з будь якої точки світу через

мережу Інтернет та розробка більш гнучкого мобільного інтерфейсу за допомогою Android Studio та додавання елементів машинного навчання та ШІ до рівня аналітики.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Погодинні ціни купівлі-продажу електроенергії URL: https://www.oree.com.ua/index.php/pricectr (дата звернення: 12.04.2023)
2. КИРИЛЕНКО, О. В.; БЛІНОВ, І. В.; ТАНКЕВИЧ, С. Є. Smart Grid та організація інформаційного обміну в електроенергетичних системах. Технічна електродинаміка, 2012.
3. Vasilchenko, V. I., Gryb, O. G., Leleka, O. V., Gapon, D. A., & Ierusalimova, T. S. (2014). Цифрова підстанція складова системи" Smart Grid". Electrical Engineering & Electromechanics, (6), 72-76.
4. Каплун, В. В., and В. В. Козирський. "Smart Grid як інноваційна платформа розвитку електроенергетичних систем." Енергетика та електрифікація (2011).
5. Lobodzinskiy, V., et al. "ВПЛИВ СИСТЕМИ SMART GRID НА НАЦІОНАЛЬНУ ЕНЕРГЕТИЧНУ МЕРЕЖУ." POWER ENGINEERING: economics, technique, ecology 1 (2022).
6. Sawle, Yashwant, S. C. Gupta, and Aashish Kumar Bohre. "Socio-techno-economic design of hybrid renewable energy system using optimization techniques." Renewable energy 119 (2018): 459-472.
7. Sekhar, D. Chandra, PVV Rama Rao, and R. Kiranmayi. "Conceptual Review on Demand Side Management, Optimization Techniques for the Improvement of Power Quality in Smart Grids." 2022 Second International Conference on Artificial Intelligence and Smart Energy (ICAIS). IEEE, 2022.
8. Li, Chaosu, Yan Song, and Nikhil Kaza. "Urban form and household electricity consumption: A multilevel study." Energy and Buildings 158 (2018): 181-193.
9. Hourly Energy Consumption URL: https://www.kaggle.com/datasets/robikscube/hourly-energy-consumption (дата звернення: 12.04.2023)

10. 2021 Assessment of Demand Response and Advanced Metering URL:https://ferc.gov/media/2021-assessment-demand-response-and-advanced-metering (дата звернення: 12.04.2023)

11. Shareef, Hussain, et al. "Review on home energy management system considering demand responses, smart technologies, and intelligent controllers." Ieee Access 6 (2018): 24498-24509.

12. Hannan, Mahammad A., et al. "State-of-the-art and energy management system of lithium-ion batteries in electric vehicle applications: Issues and recommendations." Ieee Access 6 (2018): 19362-19378.

13. Elmouatamid, Abdellatif, et al. "Review of control and energy management approaches in micro-grid systems." Energies 14.1 (2020): 168.

14. Energy management system URL:https://en.wikipedia.org/wiki/Energy_management_system (дата звернення: 15.04.2023)

15. Bhattacharjee, Somudeep, and Champa Nandi. "Design of a voting based smart energy management system of the renewable energy based hybrid energy system for a small community." Energy 214 (2021): 118977.

16. Chakraborty, Sudip, Jianxin Li, and Prosun Bhattacharya. "The solution to the global energy crisis with new materials, and sustainability." Journal of Phase Change Materials 1.2 (2021).

17. Hu, Mian, et al. "Distributed real-time demand response for energy management scheduling in smart grid." International Journal of Electrical Power & Energy Systems 99 (2018): 233-245.

18. Garifi, Kaitlyn, et al. "Stochastic model predictive control for demand response in a home energy management system." 2018 IEEE power & energy society general meeting (PESGM). IEEE, 2018.

19. Gonçalves, Ivo, Álvaro Gomes, and Carlos Henggeler Antunes. "Optimizing the management of smart home energy resources under different power cost scenarios." Applied Energy 242 (2019): 351-363.

20. Qayyum, Nabeeha, et al. "Optimization techniques for home energy management: A review." 2019 2nd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET). IEEE, 2019.

21. Than, Hoang Phuong Lien, Fengji Luo, and Zhao Yang Dong. "Acoustic Comfort-Aware Home Energy Management System." (2020).

22. Stolojescu-Crisan, Cristina, Calin Crisan, and Bogdan-Petru Butunoi. "An IoT-based smart home automation system." Sensors 21.11 (2021): 3784.

23. Bukhsh, Rasool, et al. "Appliances scheduling using hybrid scheme of genetic algorithm and elephant herd optimization for residential demand response." 2018 32nd International Conference on Advanced Information Networking and Applications Workshops (WAINA). IEEE, 2018.

24. Sattarpour, Tohid, Daryoush Nazarpour, and Sajjad Golshannavaz. "A multi-objective HEM strategy for smart home energy scheduling: A collaborative approach to support microgrid operation." Sustainable cities and society 37 (2018): 26-33.

25. Barelli, L., et al. "Residential micro-grid load management through artificial neural networks." Journal of Energy Storage 17 (2018): 287-298.

26. Zahid, Maheen, et al. "Hill climbing load balancing algorithm on fog computing." Advances on P2P, Parallel, Grid, Cloud and Internet Computing: Proceedings of the 13th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC-2018). Springer International Publishing, 2019.

27. Saeed, Faisal, et al. "IoT-based intelligent modeling of smart home environment for fire prevention and safety." Journal of Sensor and Actuator Networks 7.1 (2018): 11.

28. Ahmad, Sadiq, et al. "A compendium of performance metrics, pricing schemes, optimization objectives, and solution methodologies of demand side management for the smart grid." Energies 11.10 (2018): 2801.

29. Choi, C., Esposito, C., Wang, H., Liu, Z., & Choi, J. (2018). Intelligent power equipment management based on distributed context-aware inference in smart cities. IEEE Communications Magazine, 56(7), 212-217.

30. Grover-Silva, E., Heleno, M., Mashayekh, S., Cardoso, G., Girard, R., & Kariniotakis, G. (2018). A stochastic optimal power flow for scheduling flexible resources in microgrids operation. Applied energy, 229, 201-208.

31. Lu, R., Hong, S. H., & Zhang, X. (2018). A dynamic pricing demand response algorithm for smart grid: Reinforcement learning approach. Applied Energy, 220, 220-230.

32. Shareef, H., Ahmed, M. S., Mohamed, A., & Al Hassan, E. (2018). Review on home energy management system considering demand responses, smart technologies, and intelligent controllers. Ieee Access, 6, 24498-24509.

33. Luo, F., Kong, W., Ranzi, G., & Dong, Z. Y. (2019). Optimal home energy management system with demand charge tariff and appliance operational dependencies. IEEE Transactions on Smart Grid, 11(1), 4-14.

34. Aponte-Luis, J., Gómez-Galán, J. A., Gómez-Bravo, F., Sánchez-Raya, M., Alcina-Espigado, J., & Teixido-Rovira, P. M. (2018). An efficient wireless sensor network for industrial monitoring and control. Sensors, 18(1), 182.

35. Rahman, A., Srikumar, V., & Smith, A. D. (2018). Predicting electricity consumption for commercial and residential buildings using deep recurrent neural networks. Applied energy, 212, 372-385.

36. Tang, R., & Wang, S. (2019). Model predictive control for thermal energy storage and thermal comfort optimization of building demand response in smart grids. Applied Energy, 242, 873-882.

37. Semong, T., Maupong, T., Anokye, S., Kehulakae, K., Dimakatso, S., Boipelo, G., & Sarefo, S. (2020). Intelligent load balancing techniques in software defined networks: A survey. Electronics, 9(7), 1091.

38. Terlouw, T., AlSkaif, T., Bauer, C., & van Sark, W. (2019). Optimal energy management in all-electric residential energy systems with heat and electricity storage. Applied Energy, 254, 113580.

39. Комар М.П., Саченко А.О., Васильків Н.М., Гладій Г.М., Коваль В.С. Методичні рекомендації до виконання кваліфікаційної роботи з освітньопрофесійної програми «Комп'ютерні науки» спеціальності 122 «Комп'ютерні науки» за першим (бакалаврським) рівнем вищої освіти. – Тернопіль: ЗУНУ, 2021. – 56 с.

Додаток А

Текст програми мікроконтролера відправки даних на IoT шлюз

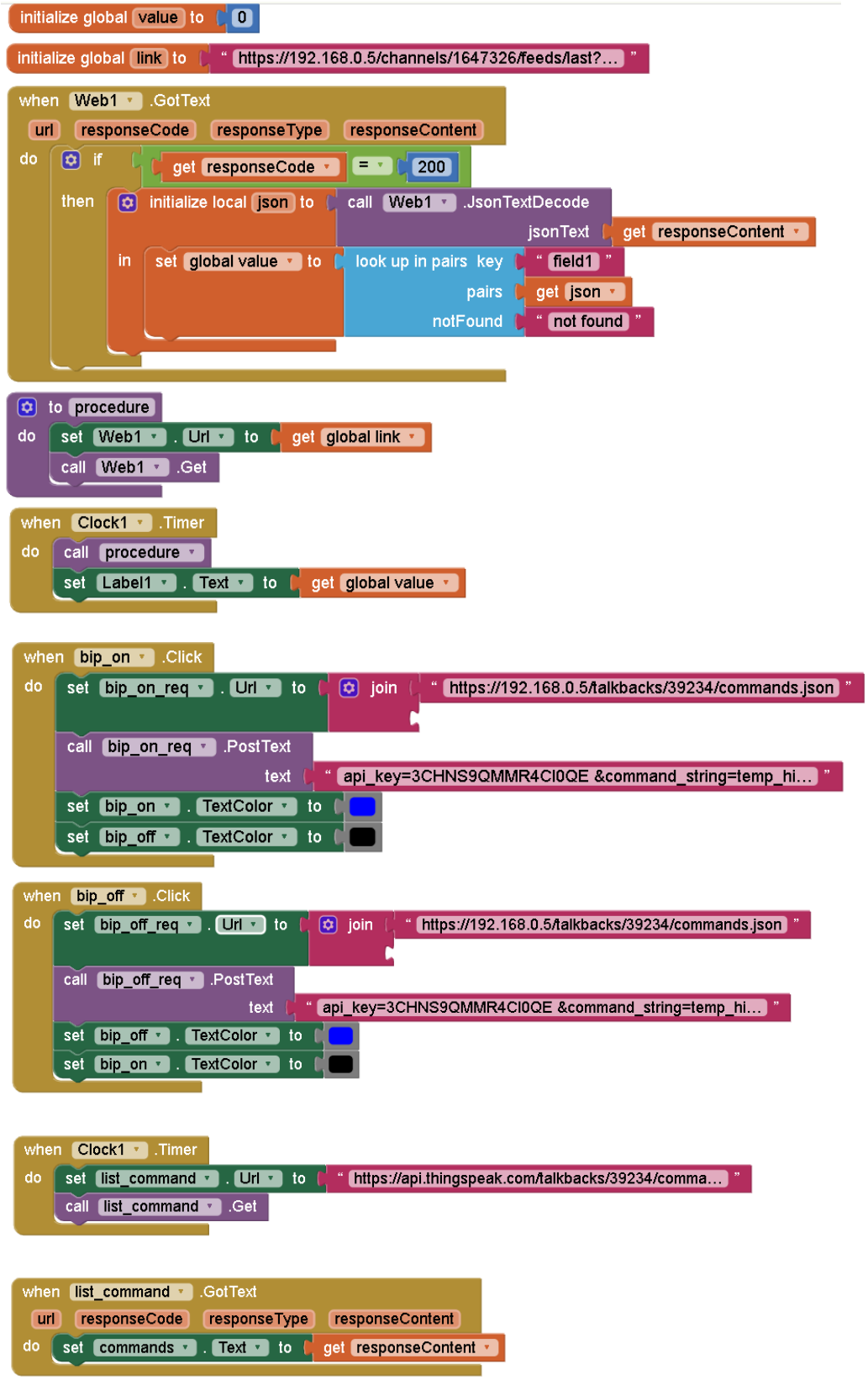
```
#include "DHTesp.h"
#include <WiFi.h>
#define TIMEOUT  5000
#define WIFI_NAME "Wokwi-GUEST"
#define PASSWORD ""
#define THING_SPEAK_ADDRESS "192.168.0.5"
String writeAPIKey="73WH71L2MMK9EW0U";
const char *ssid =  "Wokwi-GUEST";
const char *pass =  "";
const char* ca_cert = \
"-----BEGIN CERTIFICATE-----\n" \
"MIIFMzCCBNigAwIBAgIQCvZfxEx4IkD2bpmHf3g4HjAKBggqhkjOPQQDAj
BKMQsw\n" \
"CQYDVQQGEwJVUzEZMBcGA1UEChMQQ2xvdWRmbGFyZSwgSW5jLjEgMB4GA1
UEAxMX\n" \
"Q2xvdWRmbGFyZSBJbmMgRUNDIENBLTMwHhcNMjEwNjI5MDAwMDAwWhcNMj
IwNjI4\n" \
"MjM1OTU5WjB1MQswCQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZvcm5pYT
EWMBQG\n" \
"A1UEBxMNU2FuIEZyYW5jaXNjbzEZMBcGA1UEChMQQ2xvdWRmbGFyZSwgSW
5jLjEe\n" \
"MBwGA1UEAxMVc25pLmNsb3VkZmxhcmVzc2wuY29tMFkwEwYHKoZIzj0CAQ
YIKoZI\n" \
"zj0DAQcDQgAEN39+Pjk02H+YRfHjd7oNS/E+SHHd0MMw+UaLgnVpS3vzOu
/octS7\n" \
"N3zaAjeDhbkqn9stU6cPx3q9L7ZVZnxLnKOCA3MwggNvMB8GA1UdIwQYMB
aAFKXO\n" \
"N+rrsHUOlGeItEX62SQQh5YfMB0GA1UdDgQWBBTMfRL98Aej0j0ihxvu+8
LCraY2\n" \
"2DA4BgNVHREEMTAvggl3b2t3aS5jb22CFXNuaS5jbG91ZGZsYXJlc3NsLm
NvbYIL\n" \
"Ki53b2t3aS5jb20wDgYDVR0PAQH/BAQDAgeAMB0GA1UdJQQWMBQGCCsGAQ
UFBwMB\n" \
"BggrBgEFBQcDAjB7BgNVHR8EdDByMDegNaAzhjFodHRwOi8vY3JsMy5kaW
dpY2Vy\n" \
"dC5jb20vQ2xvdWRmbGFyZUluY0VDQ0NBLTMuY3JsMDegNaAzhjFodHRwOi
8vY3Js\n" \
"NC5kaWdpY2VydC5jb20vQ2xvdWRmbGFyZUluY0VDQ0NBLTMuY3JsMD4GA1
UdIAQ3\n" \
"MDUwMwYGZ4EMAQICMCkwJwYIKwYBBQUHAgEWG2h0dHA6Ly93d3cuZGlnaW
NlcnQu\n" \
"Y29tL0NQUzB2BggrBgEFBQcBAQRqMGgwJAYIKwYBBQUHMAGGGGh0dHA6Ly
9vY3Nw\n" \
"LmRpZ2ljZXJ0LmNvbTBABggrBgEFBQcwAoY0aHR0cDovL2NhY2VydHMuZG
lnaWNl\n" \
"cnQuY29tL0Nsb3VkZmxhcmVJbmNFQ0NDQS0zLmNydDAMBgNVHRMBAf8EAj
AAMIIB\n" \
"fwYKKwYBBAHWeQIEAgSCAW8EggFrAWkAdgBGpVXrdfqRIDC1oolp9PN9ES
xBdL79\n" \
```

```
     "SbiFq/L8cP5tRwAAAXpYYr+HAAAEAwBHMEUCIQCVmobzd+L83fSF0WBPia
CRP7af\n" \
     "qP5Yy7qTPEGGhCPVIQIgLYYArZeD3HDVIRMA4f/gE25DCp2yKaLKCGgBH+
K39XoA\n" \
     "dwAiRUUHWVUkVpY/oS/x922G4CMmY63AS39dxoNcbuIPAgAAAXpYYr+iAA
AEAwBI\n" \
     "MEYCIQDaOd3UZfXDbGfO+Ik8TA9IzRzwTDnMNBP1ecN6huVp6QIhAMweY0
TpBHKJ\n" \
     "LTn0cRReOw0nzms+oklUG43W3yiZSfq/AHYAQcjKsd8iRkoQxqE6CUKHXk
4xixsD\n" \
     "6+tLx2jwkGKWBvYAAAF6WGK/bQAABAMARzBFAiBMfeXhVBX+jeTM+uhpEK
9j2E96\n" \
     "CZGHTSuxUGtC4smacwIhAJ1EobOS8irZ/KDkunN5TLfrtJbP4tZi/81aur
AtgNVq\n" \
     "MAoGCCqGSM49BAMCA0kAMEYCIQC2rquvlc6Omt0/R7ZWpQg1Id4M9H+C+2
iVw50n\n" \
     "JwyTaAIhALVsYs/oGD2ZoNdXX9zLQPTLg78AhiTGw9vjkk+BJdLm\n" \
     "-----END CERTIFICATE-----\n";


     DHTesp dhtSensor;
     WiFiClient client;

     void setup() {
       Serial.begin(115200);
       dhtSensor.setup(15, DHTesp::DHT22);
       //+---------------------------------------------//
       delay(10);
         Serial.println("Connecting to ");
         Serial.println(ssid);
         WiFi.begin(ssid, pass);
         while (WiFi.status() != WL_CONNECTED)
         {
             delay(500);
             Serial.print(".");
         }
         Serial.println("");
         Serial.println("WiFi connected");
     }

     void loop() {
       TempAndHumidity  data = dhtSensor.getTempAndHumidity();
       Serial.println("Temp: " + String(data.temperature, 2) +
"°C");
       Serial.println("Humidity: " + String(data.humidity, 1) +
"%");
       delay(1000);
       if (client.connect( THING_SPEAK_ADDRESS , 80 ), ca_cert){
            String postData= "api_key=" + writeAPIKey + "&field1="
+ String(data.temperature, 2);
            Serial.println(  "Connecting  to  ThingSpeak  for
update..." );
            Serial.println();
```

```
            client.println( "POST /update HTTP/1.1" );
            client.println( "Host: 192.168.0.5" );
            client.println( "Connection: close" );
            client.println(  "Content-Type:  application/x-www-
form-urlencoded" );
            client.println(  "Content-Length:  "  +  String(
postData.length() ) );
            client.println();
            client.println( postData );

            Serial.println( postData );

          String response;
          long startTime = millis();
          delay( 200 );
          while  (  client.available()  <  1  &&  ((  millis()  -
startTime ) < TIMEOUT ) ){
          delay( 5 );
       }

       if( client.available() > 0 ){
          do {
              charIn = client.read();
              response += charIn;
             } while ( client.available() > 0 );
         }
       client.stop();
             Serial.println( response);
         }
         else
         {
           Serial.println ( "Connection Failed" );
         }
         delay(17000);
     }
```

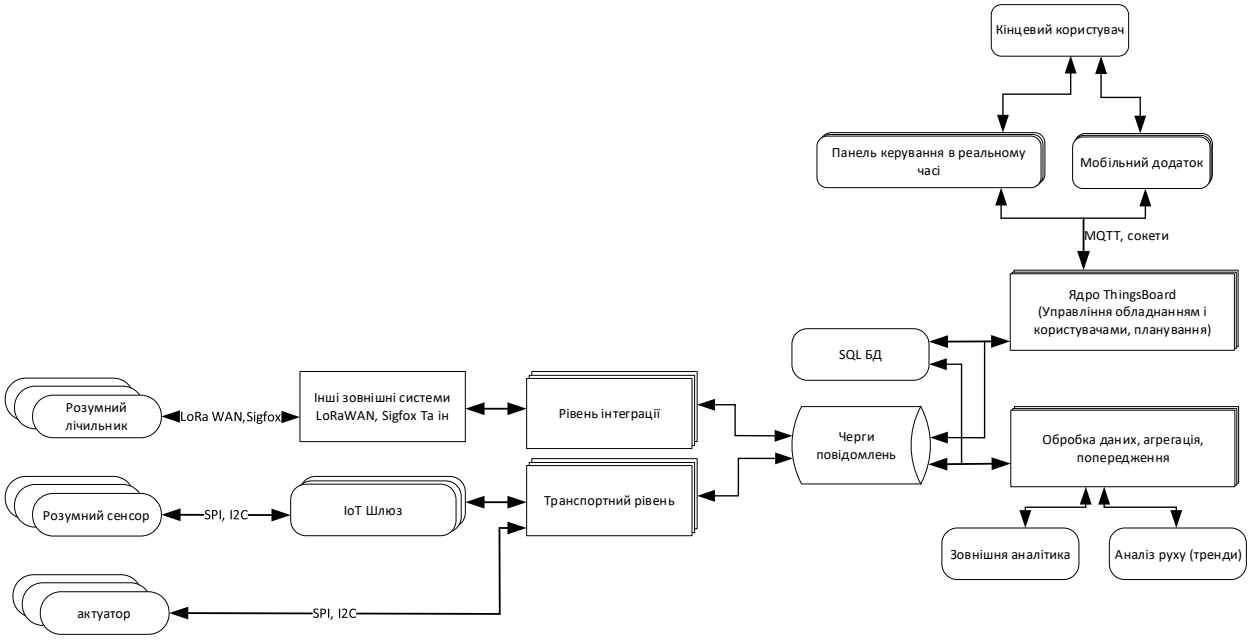
Додаток Б

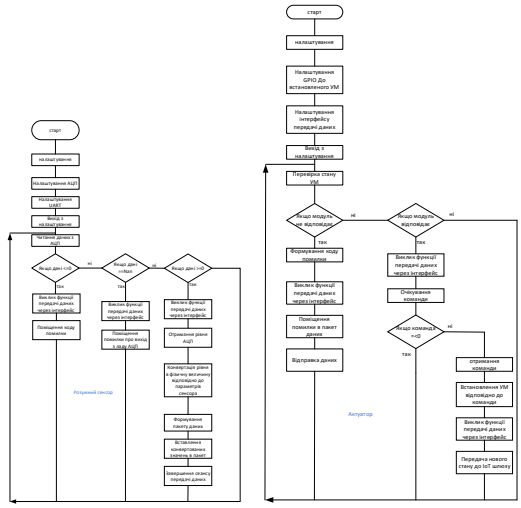
Блок-програма мобільного додатку



Додаток В

Схема структурна взаємозв’язку частин системи



Додаток Г

Блок схеми алгоритмів розумного сенсора, актуатора та контролю освітлення