

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ВЕТЕРИНАРНОЇ  
МЕДИЦИНИ ТА БІОТЕХНОЛОГІЙ ІМ. С. З. ГЖИЦЬКОГО  
ВІДДІЛ ЗАОЧНОГО НАВЧАННЯ  
ЦЕНТРУ ПЕРЕПІДГОТОВКИ ТА ПІДВИЩЕННЯ КВАЛІФІКАЦІЇ  
КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

## **КВАЛІФІКАЦІЙНА РОБОТА**

першого (бакалаврського) рівня вищої освіти

на тему:

**« Проектування системи « Розумний будинок » з використанням сервісів  
Z-WAY »**

Виконала: студентка групи Кн-42зСП  
спеціальності 122 Комп'ютерні науки  
Павлик А.А.

(прізвище та ініціали)

Керівник: Пташник В. В.

(прізвище та ініціали)

**ДУБЛЯНИ 2025**

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ВЕТЕРИНАРНОЇ МЕДИЦИНИ ТА  
БІОТЕХНОЛОГІЙ ІМ. С. З. ГЖИЦЬКОГО  
ВІДДІЛ ЗАОЧНОГО НАВЧАННЯ  
ЦЕНТРУ ПЕРЕПІДГОТОВКИ ТА ПІДВИЩЕННЯ КВАЛІФІКАЦІЇ  
КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Рівень вищої освіти перший (бакалаврський)  
Спеціальність 122 Комп'ютерні науки

ЗАТВЕРДЖУЮ  
Завідувач кафедри

(підпис)

д.т.н., професор, Тригуба А. М.

(вч. звання, прізвище, ініціали)

“ ” 202\_\_ року

**ЗАВДАННЯ  
НА КВАЛІФІКАЦІЙНУ РОБОТУ**

Павлик Анни Андріїни

(прізвище, ім'я, по батькові)

1. Тема роботи «Проектування системи «Розумний будинок» з використанням сервісів Z-WAY»

керівник роботи к.т.н., доцент. Пташник В. В.

(наук. ступінь, вч. звання, прізвище, ініціали)

затвержені наказом Львівського НУП № 172/к-с від 08.03.2024 р

2. Строк подання студентом роботи 09.06.2025 р.

3. Вихідні дані: Вихідні дані: опис предметної області – система «Розумний будинок» на Z-Wave; вимоги до веб-додатку – керування пристроями через Z-Way; використані бібліотеки – JavaScript, REST API; конфігурація – локальний сервер Z-Way; джерела – техдокументація Z-Wave, фахова література.

4. Зміст кваліфікаційної роботи (перелік питань, які потрібно розробити)  
Вступ

1. Аналіз предметної області системи “Розумний будинок”

2. Аналіз апаратного та програмного забезпечення

3. Реалізація системи “Розумний будинок” на базі сервісу Z-WAY

4. Охорона праці

Висновки

Бібліографічний список

## 5. Перелік графічного матеріалу

*Графічний матеріал подається у вигляді презентації*

## 6. Консультанти розділів

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата		Відмітка про виконання
		завдання видав	завдання прийняв	
1, 2, 3	<i>Пташник В.В., к.т.н., доцент кафедри інформаційних технологій</i>			
4	<i>Городецький І.М., к.т.н., доцент кафедри інженерної механіки</i>			

7. Дата видачі завдання 11 березня 2024 року

## КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломної роботи	Строк виконання етапів роботи	Відмітка про виконання
1	<i>Отримання завдання. Вивчення рекомендованої літератури по темі роботи. Написання аналітичного огляду предметної області.</i>	<i>08.03.2024 – 30.09.2024</i>	
2	<i>Проектування та опис технічного завдання, визначення функціональних вимог до системи «Розумний будинок» на базі Z-Wave, розробка архітектури, вибір структури керування, опис сценаріїв автоматизації.</i>	<i>01.10.2024 – 20.12.2024</i>	
3	<i>Програмна реалізація системи: встановлення та налаштування Z-Way, створення інтерфейсу керування, реалізація сценаріїв, тестування функцій, аналіз працездатності та ефективності системи.</i>	<i>21.12.2024 – 15.02.2025</i>	
4	<i>Розгляд питань з охорони праці та виробничого середовища</i>	<i>16.02.2025 – 30.04.2025</i>	
5	<i>Завершення оформлення основної частини, написання висновків та підготовка презентаційного матеріалу</i>	<i>01.05.2025 – 20.05.2025</i>	
6	<i>Завершення роботи в цілому. Підготовка до захисту кваліфікаційної роботи</i>	<i>21.05.2025 – 09.06.2025</i>	

Студентка

( підпис )

Павлик А. А.

(прізвище та ініціали)

Керівник роботи

( підпис )

Пташник В. В.

(прізвище та ініціали)

УДК 004.912:004.738.5:681.5

Кваліфікаційна робота : 80сторінок текстової частини, 20 рисунків, 3 таблиці, 42 літературних джерела, 2 додатки.

Павлик А.А. «Проектування системи« Розумний будинок» з використанням сервісів Z-WAY». Кваліфікаційна робота за спеціальністю 122. Комп'ютерні науки. Львів : ЛНУВМБ, 2025, с.

Кваліфікаційна робота присвячена проектуванню системи «Розумний будинок» з використанням сервісів Z-WAY. Розглянуто актуальність автоматизації житлових приміщень, архітектуру системи, принципи роботи протоколу Z-Wave та можливості керування побутовими пристроями. Проаналізовано особливості побудови сучасних інтелектуальних систем, їх інтеграцію з мережею Інтернет та способи взаємодії між контролерами, сенсорами й виконавчими елементами.

У роботі здійснено розробку вебінтерфейсу для керування освітленням, мікрокліматом і розетками на базі платформи Z-Way. Запропоновану систему протестовано в умовах, наближених до реальних. Результати можуть бути використані для створення адаптивних і масштабованих рішень у сфері автоматизації будинку, зокрема для приватного житла, малих офісів і навчальних проєктів.

## ЗМІСТ

<b>ВСТУП</b> .....	6
<b>РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБСЛАТИ СИСТЕМИ «РОЗУМНИЙ БУДИНОК»</b> .....	8
1.1. Поняття та еволюція технологій «розумного будинку» .....	8
1.2 Сучасні підходи до автоматизації житлових приміщень .....	9
1.3. Протоколи передачі даних у системах «Розумний будинок».....	10
1.4. Платформи управління у системах “Розумний будинок” .....	12
<b>РОЗДІЛ 2. АНАЛІЗ АПАРАТНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ</b> .....	15
2.1. Дослідження особливостей протоколу Z-Wave.....	15
2.2. Архітектура Z-Way: компоненти, API, функціональні можливості.....	17
2.3. Аналіз апаратних пристроїв, сумісних із Z-Wave.....	19
2.4. Інтеграція Z-Way з іншими платформами та системами .....	21
2.5. Порівняння Z-Way з іншими альтернативними системами управління .....	23
<b>РОЗДІЛ 3. РЕАЛІЗАЦІЯ СИСТЕМИ «РОЗУМНИЙ БУДИНОК» НА БАЗІ СЕРВІСУ Z-WAY</b> .....	26
3.1. Постановка задачі та вимоги до системи .....	26
3.2. Архітектура системи “Розумний будинок” .....	28
3.3. Реалізація модулів управління освітленням, мікрокліматом та побутовими приладами.....	31
3.4. Розробка інтерфейсу користувача .....	44
3.5. Тестування системи та результати.....	52
<b>РОЗДІЛ 4. ОХОРОНА ПРАЦІ</b> .....	54
4.1. Аналіз стану виробничої санітарії і гігієни праці.....	54
4.2. Обґрунтування організаційно-технічних рекомендацій з охорони праці .....	56
4.3. Пожежна безпека.....	57
<b>ВИСНОВКИ</b> .....	59
<b>БІБЛІОГРАФІЧНИЙ СПИСОК</b> .....	61
<b>ДОДАТКИ</b> .....	66

## ВСТУП

Упродовж останніх десятиліть у світі спостерігається стійка тенденція до інтеграції інформаційних технологій у всі сфери людської діяльності. Особливо динамічно розвивається напрямок, пов'язаний з автоматизацією житлового простору – так звані системи «Розумний будинок». Йдеться не лише про модну технологічну новинку, а про реальну відповідь на запити сучасної людини: зручність, безпека, економія енергоресурсів, гнучкість у керуванні побутом.

В умовах великої кількості пропозицій на ринку особливу увагу привертають рішення, що базуються на протоколі Z-Wave. Цей бездротовий стандарт створений спеціально для систем домашньої автоматизації та забезпечує надійне з'єднання між пристроями при низькому енергоспоживанні. Його відмінною рисою є орієнтація саме на побутові завдання – від керування освітленням і температурою до організації охоронних і аварійних сповіщень.

Актуальність теми дослідження полягає в необхідності розробки адаптивних, зручних і масштабованих систем керування побутовими процесами, які не тільки відповідають сучасним технологічним вимогам, а й можуть бути впроваджені на практиці без значних фінансових або технічних витрат. Саме рішення на основі Z-Way дозволяють досягти балансу між функціональністю, простотою реалізації та можливістю подальшого розвитку системи.

У контексті зростаючої кількості «розумних» пристроїв і підвищення запитів користувачів на персоналізацію та автоматизацію повсякденного життя, виникає потреба у платформах, які здатні інтегрувати різноманітні елементи в єдину керовану екосистему. Протокол Z-Wave та платформа Z-Way забезпечують стабільну взаємодію між пристроями різних виробників, що особливо важливо для побудови масштабованих систем. До того ж, ураховуючи тенденції до енергоефективності, безпеки та дистанційного керування, дослідження таких рішень має практичне значення як для окремих користувачів, так і для розробників комерційних і житлових інфраструктур. Реалізація подібних систем також сприяє цифровій трансформації побуту, наближаючи концепцію «розумного дому» до широкого

загалу.

**Мета дослідження** полягає у проектуванні та реалізації демонстраційної моделі системи «Розумний будинок», що базується на протоколі Z-Wave і сервісах Z-Way. Проект має продемонструвати базові функції керування освітленням, мікрокліматом і побутовими приладами, забезпечити взаємодію між різними компонентами системи та бути відкритим до подальшого розширення.

Для досягнення поставленої мети в роботі сформульовано такі **завдання**:

- проаналізувати сучасний стан розвитку систем домашньої автоматизації та визначити основні тенденції в цій сфері;
- ознайомитися з технічними характеристиками і перевагами протоколу Z-Wave;
- дослідити функціональні можливості сервісу Z-Way, його архітектуру та інтерфейси взаємодії;
- розробити логічну структуру системи з урахуванням різних типів пристроїв (освітлення, клімат-контроль, побутова техніка);
- реалізувати базові сценарії керування та протестувати їхню працездатність;
- проаналізувати результати експериментального впровадження та сформулювати рекомендації щодо вдосконалення системи.

**Об'єктом дослідження** виступають технології побудови інтелектуальних систем автоматизованого керування у побутовому середовищі.

**Предметом дослідження** є процес розробки, налаштування та практичної реалізації системи «Розумний будинок» з використанням протоколу Z-Wave і платформи Z-Way.

**Практична цінність даної роботи** полягає в тому, що описана система може бути використана як приклад для впровадження подібних рішень у реальних умовах. Крім того, результати роботи можуть стати основою для подальших досліджень у галузі побудови масштабованих Smart Home-систем або ж бути адаптованими під інші протоколи й середовища.

## РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБСЛАТИ СИСТЕМИ «РОЗУМНИЙ БУДИНОК»)

### 1.1. Поняття та еволюція технологій «розумного будинку»

Сучасне розуміння терміну «розумний будинок» охоплює інтегровану інфраструктуру, в якій різноманітні пристрої – освітлення, опалення, охоронні системи, побутова техніка – взаємодіють між собою та користувачем. Це дозволяє забезпечити автоматичне або дистанційне управління за допомогою мобільних додатків, голосових асистентів чи спеціального програмного забезпечення. Головна мета такої системи – підвищення комфорту, енергоефективності та безпеки проживання [1].

Ідея автоматизації домашніх процесів почала реалізовуватись ще в 1970-х роках. Однією з перших технологій, що лягла в основу «розумного будинку», був протокол X10. Цей стандарт дозволяв передавати сигнали керування електроприладами через наявну електромережу. Хоча система була досить примітивною та схильною до завад, саме вона започаткувала нову еру у сфері побутової автоматизації [2].

У 1980–1990-х роках почали з'являтися комерційні рішення для автоматизації будинку. Вони дозволяли керувати окремими функціями житла – освітленням, безпекою, кліматом. Однак ці системи були доступні переважно для заможних споживачів, оскільки вимагали складного встановлення та налаштування. Широкого розповсюдження вони не набули, але стали базою для подальших технологічних рішень [3].

З початком нового тисячоліття та появою бездротових технологій (Wi-Fi, Zigbee, Z-Wave) почався стрімкий розвиток і масове впровадження систем «розумного будинку». Завдяки цим технологіям зникла потреба у прокладанні додаткових кабелів, а підключення пристроїв стало простішим і дешевшим. З'явилися універсальні платформи, які дозволяли керувати великою кількістю пристроїв з одного інтерфейсу.

Сьогодні «розумні будинки» інтегруються з концепцією Інтернету речей, що



дозволяє пристроям не лише виконувати завдання, а й адаптуватись до поведінки користувача. Наприклад, система може самостійно регулювати температуру в приміщенні залежно від присутності людей. Штучний інтелект поступово впроваджується для аналізу звичок мешканців і створення персоналізованих сценаріїв автоматизації.

Незважаючи на зручність, широке впровадження «розумних» пристроїв породжує серйозні виклики у сфері інформаційної безпеки. Через відсутність регулярних оновлень прошивок або слабкий захист, такі системи можуть стати вразливими до кібератак [4].

## **1.2 Сучасні підходи до автоматизації житлових приміщень**

На сьогодні автоматизація житлового простору виходить далеко за межі простого дистанційного керування приладами. Вона охоплює створення інтелектуального середовища, яке самостійно аналізує поведінку мешканців та адаптується до їхніх звичок. Наприклад, освітлення може вмикатись у відповідь на рух, температура автоматично регулюється відповідно до часу доби, а системи безпеки змінюють рівень охорони залежно від присутності в будинку [5].

Сучасні системи проєктуються за модульним принципом, що дозволяє поступово розширювати функціональність будинку без повного переоснащення. Основу зазвичай становить контролер або шлюз, до якого підключаються різні сенсори, виконавчі пристрої та інтерфейси управління. Це дає змогу користувачу самостійно визначати ступінь автоматизації та інтегрувати пристрої від різних виробників [6].

Однією з ключових особливостей сучасного підходу є використання бездротових технологій передачі даних: Zigbee, Z-Wave, Bluetooth, Wi-Fi. Це значно спрощує встановлення, особливо у вже готових житлових об'єктах, оскільки відсутня потреба в прокладці кабелів. Крім того, завдяки стандартам інтероперабельності, стає можливим об'єднання пристроїв у єдину екосистему[7].

У більшості сучасних реалізацій автоматизованих систем важливу роль відіграє підключення до хмарних платформ, таких як Amazon Alexa, Google Assistant або Apple HomeKit. Це забезпечує доступ до управління з будь-якої точки світу, а також можливість голосового керування. Хмарні технології також дозволяють створювати складні сценарії автоматизації без необхідності глибоких технічних знань [8].

Ще один ключовий напрям – автоматизація для зниження енергоспоживання. Системи розумного будинку дозволяють оптимізувати використання електроенергії, води та опалення, що не лише зменшує витрати мешканців, а й сприяє екологічній сталій практиці. Наприклад, автоматичне вимкнення освітлення у відсутності людей або розклад опалення – стандартні функції в більшості сучасних рішень.

Останніми роками в системи розумного дому почали впроваджувати алгоритми машинного навчання. Це дозволяє пристроям не лише реагувати на події, а й передбачати потреби користувача на основі аналізу його щоденної поведінки. Такі технології відкривають шлях до повної автономізації житлового середовища [9].

### **1.3. Протоколи передачі даних у системах «Розумний будинок»**

Комунікаційні протоколи є основою для взаємодії між пристроями в інтелектуальному будинку. Вони визначають, як дані передаються між сенсорами, актуаторами, контролерами та хмарними платформами. Правильно обраний протокол забезпечує надійність, масштабованість та ефективність роботи всієї системи [10].

Zigbee є одним з найпопулярніших протоколів бездротової передачі даних у «розумному будинку». Його основні переваги – енергоефективність та підтримка великої кількості пристроїв у мережі. Zigbee працює у діапазоні 2.4 ГГц, створюючи мережу типу «mesh», де кожен пристрій може ретранслювати сигнал.

Завдяки цьому забезпечується стабільне з'єднання навіть у великих будівлях [11.]

Z-Wave, як і Zigbee, використовує топологію mesh, однак працює у нижчому частотному діапазоні (приблизно 868 МГц в Європі), що дозволяє сигналу легше проходити крізь стіни та інші перешкоди. Протокол підтримує до 232 пристроїв у мережі та має суворі вимоги до сумісності, що забезпечує високу стабільність роботи. Однією з ключових особливостей Z-Wave є сертифікація всіх сумісних пристроїв, що сприяє стандартизації ринку [12].

Wi-Fi – один з найбільш розповсюджених протоколів у побуті, і завдяки цьому багато виробників «розумних» пристроїв використовують його для спрощення підключення до мережі. Його головною перевагою є висока швидкість передачі даних, що ідеально підходить для відеоспостереження або потокової передачі даних. Водночас, Wi-Fi споживає більше енергії, що робить його менш доцільним для батарейних пристроїв типу сенсорів чи датчиків [13].

Хоча всі три протоколи – Zigbee, Z-Wave та Wi-Fi – можуть бути застосовані у системах автоматизації, їх вибір залежить від конкретних вимог. Наприклад, для великої кількості малопотужних пристроїв краще підходить Zigbee або Z-Wave, тоді як Wi-Fi обирають для пристроїв з високим обсягом даних. Важливим чинником є й інтероперабельність – Zigbee та Z-Wave активніше стандартизуються, тоді як Wi-Fi – універсальний, але менш структурований.

У майбутньому очікується посилення позицій універсальних протоколів, таких як Matter – ініціатива, яка об'єднує можливості Zigbee, Z-Wave, Wi-Fi та Bluetooth для досягнення максимальної сумісності між пристроями від різних виробників. Це має суттєво спростити процес налаштування розумного дому для пересічного користувача [14].

Для наочного розуміння технічних характеристик основних протоколів, що застосовуються у системах «розумного будинку», нижче наведено порівняльну таблицю. У ній узагальнено ключові параметри Zigbee, Z-Wave та Wi-Fi, які впливають на вибір тієї чи іншої технології в конкретному проєкті. Порівняльні дані представлено в таблиці 1.1.

Таблиця 1.1 – Порівняльна характеристика протоколів Zigbee, Z-Wave і Wi-Fi для систем «Розумний будинок»

Характеристика	Zigbee	Z-Wave	Wi-Fi
Частота роботи	2.4 ГГц	~868 МГц (Європа)	2.4/5 ГГц
Топологія мережі	Mesh	Mesh	Star
Радіус дії (в приміщ.)	~10–20 м	~30–40 м	~50–100 м
Кількість пристроїв	до 65 000	до 232	Обмежена мережею роутера
Енергоспоживання	Низьке	Дуже низьке	Високе
Сумісність	Часткова	Сертифікована	Висока, але без гарантії
Призначення	Сенсори, керування	Сенсори, автоматика	Камери, стрімінг, IoT

Як видно з таблиці, протоколи Zigbee та Z-Wave демонструють кращі характеристики для енергоефективної та масштабованої автоматизації порівняно з Wi-Fi, який більше підходить для ресурсомістких задач (камери, мультимедіа). Вибір залежить від пріоритетів: якщо необхідно створити гнучку і стабільну мережу датчиків – доцільніше використовувати Z-Wave або Zigbee, тоді як для спрощених сценаріїв або готових пристроїв – Wi-Fi залишається популярним завдяки універсальності.

#### 1.4. Платформи управління у системах “Розумний будинок”

Ринок платформ для автоматизації житлових приміщень демонструє стабільне зростання, що зумовлено підвищеним інтересом до інтелектуальних технологій серед кінцевих користувачів і підприємств. Ці платформи створюють єдиний інтерфейс для керування великою кількістю різномірних пристроїв – від освітлення та клімат-контролю до систем безпеки та побутової техніки. Важливою характеристикою сучасних платформ є їх відкритість та здатність підтримувати численні протоколи й стандарти, що забезпечує гнучкість і розширюваність системи. Активні спільноти розробників і користувачів створюють умови для швидкого оновлення та впровадження нових функцій [15].

Home Assistant виділяється серед інших рішень завдяки широкій підтримці

пристроїв та можливості гнучкого налаштування під потреби конкретного користувача. Система орієнтована на локальне розгортання, що гарантує більшу безпеку та приватність, адже дані не обов'язково передаються на сторонні сервери. Користувачі можуть інтегрувати пристрої різних виробників, створювати складні сценарії автоматизації, що базуються на численних тригерах і умовах, а також використовувати мобільні додатки та веб-інтерфейс для керування системою з будь-якої точки світу [16].

OpenHAB – це універсальне рішення, що підтримує запуск на багатьох операційних системах і апаратних платформах, завдяки використанню Java. Його відмінність полягає в широкому наборі плагінів, які дозволяють підключати до системи різноманітні пристрої незалежно від виробника чи протоколу зв'язку. Хоча OpenHAB вимагає певного рівня технічної підготовки для налаштування, він забезпечує стабільність роботи, гнучкість та можливість масштабування системи.[17].

Z-Way – це пропріетарна платформа, розроблена спеціально для роботи з протоколом Z-Wave, що має на меті максимально ефективно використовувати всі можливості цієї технології. Платформа пропонує повний спектр інструментів для управління мережею пристроїв, діагностики та автоматизації. Вона орієнтована на користувачів, які хочуть глибоко працювати з Z-Wave без зайвих складнощів налаштування. Такий підхід робить Z-Way привабливим вибором для тих, хто використовує багато пристроїв з підтримкою Z-Wave і бажає мати централізоване управління [18].

Кожна з розглянутих платформ має свої сильні сторони і може задовольнити різні вимоги. Home Assistant підходить для тих, хто хоче максимальної гнучкості та має бажання і час для налаштування системи, тоді як OpenHAB – відмінне рішення для користувачів, які цінують стабільність і багатоплатформеність. Z-Way, у свою чергу, є кращим вибором для систем, орієнтованих виключно на Z-Wave пристрої, забезпечуючи найкращу сумісність та простоту використання. Вибір залежить від рівня технічної підготовки користувача, необхідного функціоналу та типу обладнання, що планується інтегрувати [19].

На рисунку 1.1 наведено орієнтовну кількість активних інсталяцій платформ Home Assistant, OpenHAB та Z-Way станом на 2025 рік.

Home Assistant займає лідерську позицію з приблизно 2 мільйонами активних інсталяцій, що підтверджується офіційною аналітикою [analytics.home-assistant.io](https://analytics.home-assistant.io). OpenHAB, за оцінками спільноти та публічної статистики, має близько 150–200 тисяч інсталяцій. Платформа Z-Way, як комерційне рішення з вузькою спеціалізацією, демонструє нижчі показники – орієнтовно 50–80 тисяч користувачів у всьому світі.

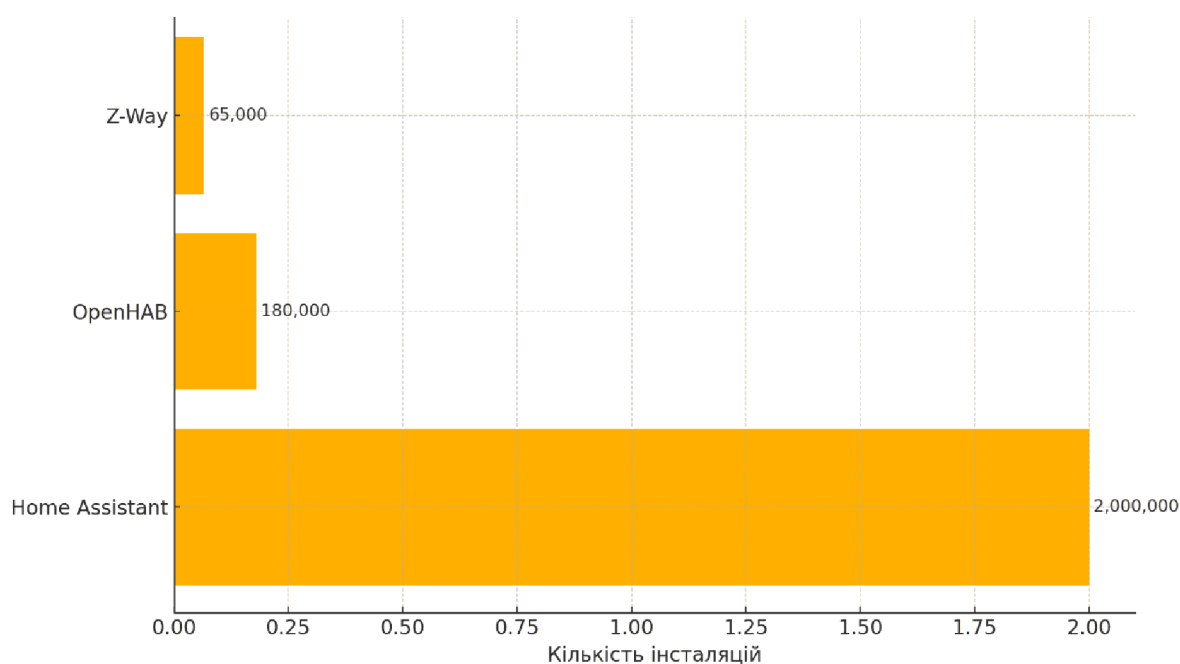


Рисунок 1.1 – Орієнтовна кількість інсталяцій Smart Home-платформ станом на 2025 рік.

З діаграми видно, що Home Assistant домінує на ринку open-source рішень для «розумного будинку» завдяки великій кількості інтеграцій, активній спільноті та універсальності.

## РОЗДІЛ 2. АНАЛІЗ АПАРАТНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

### 2.1. Дослідження особливостей протоколу Z-Wave.

Протокол Z-Wave є одним із найпоширеніших стандартів бездротового зв'язку, що використовується в системах автоматизації житлових і комерційних приміщень. Він був створений компанією Zensys у 2001 році й спеціалізується виключно на забезпеченні стабільної та надійної взаємодії між розумними пристроями в межах одного об'єкта. На відміну від універсальних стандартів, таких як Wi-Fi чи Bluetooth, Z-Wave розроблявся саме для потреб "розумного будинку", тому він вирізняється низьким енергоспоживанням, високою сумісністю пристроїв та стійкістю до перешкод [20].

Ключовою перевагою Z-Wave є мережа типу mesh (сіткова топологія), де кожен пристрій не лише приймає сигнали, а й передає їх далі, тим самим збільшуючи загальну дальність покриття. Завдяки цьому навіть у великих приміщеннях сигнал здатен легко обходити перешкоди, а збої в одному вузлі мережі не зупиняють роботу всієї системи. Більшість Z-Wave пристроїв мають радіус дії близько 30–40 метрів у приміщенні, але через реле-сусідів сигнал може подорожувати до 200 метрів і більше [21].

Ще однією важливою характеристикою Z-Wave є уніфікований стандарт взаємодії, що дозволяє пристроям різних виробників працювати разом без конфліктів. Це стало можливим завдяки суворій сертифікації, яку проходять всі пристрої перед виходом на ринок. На відміну від більш відкритих, але фрагментованих стандартів, Z-Wave гарантує, що будь-який сумісний пристрій буде коректно функціонувати в будь-якій Z-Wave мережі [22.]

Z-Wave працює на ліцензованих частотах у діапазоні близько 800–900 МГц (в залежності від регіону), що дозволяє уникати завад, типових для перевантажених діапазонів Wi-Fi (2.4 ГГц). Завдяки цьому, в умовах щільної забудови чи наявності великої кількості інших бездротових мереж, Z-Wave зберігає стабільність сигналу. Ця частотна особливість також знижує навантаження на акумулятори в мобільних

або автономних пристроях, таких як датчики руху чи температури [23].

Що стосується безпеки, Z-Wave використовує 128-бітове шифрування AES (Advanced Encryption Standard), яке відповідає сучасним вимогам до захисту даних. Крім того, з впровадженням оновленого стандарту Z-Wave S2 (Security 2), процес автентифікації став ще більш надійним завдяки обов'язковій ручній перевірці пристроїв при додаванні до мережі (наприклад, через QR-коди чи унікальні ключі) [24].

На сьогоднішній день на ринку представлено понад 4 000 сертифікованих пристроїв Z-Wave від більш ніж 700 виробників. Це свідчить про широке розповсюдження та стабільний розвиток екосистеми. Серед типових пристроїв можна назвати смарт-реле, замки, датчики руху, термостати, контролери штор, а також універсальні шлюзи, які забезпечують об'єднання Z-Wave з іншими протоколами [25].

У порівнянні з іншими бездротовими протоколами, такими як Zigbee чи Thread, Z-Wave демонструє вищу стабільність у малих і середніх за розміром мережах, а також має переваги у простоті налаштування. Проте головним обмеженням є використання ліцензованої технології, що ускладнює створення DIY-рішень без придбання спеціалізованого апаратного забезпечення або контролерів [26].

Z-Wave використовує сіткову (mesh) топологію, де кожен пристрій може не лише отримувати, але й ретранслювати сигнали до інших пристроїв. Це дозволяє будувати надійну мережу з великою зоною покриття навіть при наявності фізичних перешкод (стіни, меблі тощо). У мережі Z-Wave виділяють три типи вузлів:

- Контролер – головний пристрій, який керує всією мережею.
- Роутер – активний пристрій із живленням від мережі, що передає сигнал іншим.
- Кінцевий пристрій – виконує функцію без маршрутизації (наприклад, датчик на батарейці).

На рисунку 2.1 представлено типовий вигляд мережі Z-Wave.



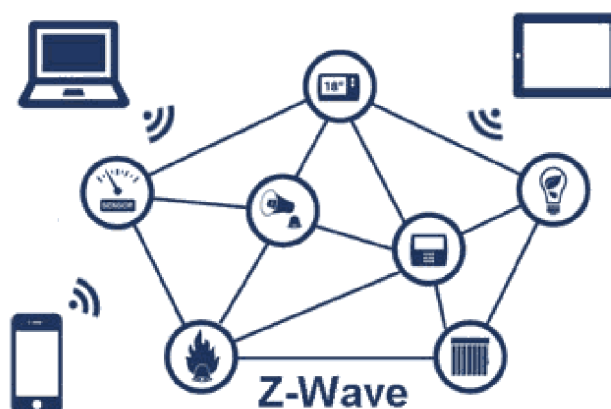


Рисунок 2.1 – Схема топології мережі Z-Wave (mesh-сітка)

Комунікація в мережі Z-Wave побудована на основі стеку протоколів, що включає такі рівні:

1. Фізичний рівень – передача даних на частоті 868,42 МГц (в Європі).
2. Канальний рівень (MAC) – контроль доступу до середовища, адресація, виявлення колізій.
3. Мережевий рівень – маршрутизація даних у сітковій топології.
4. Транспортний рівень – розбиття повідомлень, контроль цілісності.
5. Прикладний рівень – набір команд, що описують функціональність пристрою (вмикання/вимикання, вимірювання температури тощо).

## 2.2. Архітектура Z-Way: компоненти, API, функціональні можливості

Z-Way – це спеціалізована програмна платформа, розроблена компанією Z-Wave.Me, яка реалізує повноцінну підтримку протоколу Z-Wave. На відміну від багатьох систем, що лише частково підтримують специфікацію, Z-Way є першою платформою, офіційно сертифікованою Z-Wave Alliance як повноцінна реалізація Z-Wave Controller Software [18].

Архітектура Z-Way складається з кількох основних шарів: низькорівнева частина взаємодіє з Z-Wave USB-донглом або контролером RaZberry, тоді як верхній рівень забезпечує доступ до керування пристроями через API або вебінтерфейс. Базовим ядром системи є Z-Way Engine, який відповідає за логіку

роботи мережі, маршрутизацію пакетів, створення асоціацій, підтримку сцен та сценаріїв.

Функціональні можливості Z-Way охоплюють автоматизацію на основі подій, створення таймерів, інтеграцію з голосовими асистентами, обробку скриптів, а також побудову дашбордів для користувача. Окремим елементом системи є модулі – розширення, які додають специфічну функціональність, наприклад, інтеграцію з Telegram, Google Calendar

Z-Way підтримує мультиканальні пристрої, що мають кілька логічних інтерфейсів (наприклад, розумна розетка з кількома каналами або сенсор з температурним та вологісним датчиками), і дає змогу керувати кожним з них окремо. Крім того, Z-Way дозволяє створювати віртуальні пристрої, які використовуються для сценаріїв або абстрагування логіки (наприклад, створення віртуальної кнопки для виклику певної сцени) [28].

Архітектура Z-Way базується на модульному принципі, де кожен компонент виконує свою специфічну роль. Система має розділення за логічними рівнями – від ядра мережевої взаємодії до інтерфейсів користувача та зовнішніх API. У таблиці 2.1 представлено основні елементи архітектури Z-Way та описано їхні функції.

Таблиця 2.1 Компоненти архітектури Z-Way та їх функціональні ролі

Компонент	Опис
<b>Z-Way Engine</b>	Основне ядро, що відповідає за логіку роботи мережі Z-Wave: обробку команд, маршрутизацію, підтримку асоціацій і сценаріїв.
<b>Device Management</b>	Модуль, що забезпечує виявлення, реєстрацію, ідентифікацію та контроль за пристроями. Містить статуси, журнали подій, типи пристроїв.
<b>REST API / WebSocket</b>	Відкритий інтерфейс для обміну даними з іншими додатками, що дає змогу інтегрувати Z-Way з зовнішніми системами в режимі реального часу.
<b>Apps Middleware</b>	Спеціальні модулі, що додають розширені функції автоматизації (наприклад, інтеграція з Telegram, IFTTT, HTTP-запити, Google Calendar тощо).
<b>Web UI</b>	Веб-інтерфейс для користувача, що дозволяє здійснювати налаштування системи, керувати пристроями та переглядати історію подій.
<b>Scripting Engine</b>	Вбудований JavaScript-движок, який дозволяє створювати кастомні сценарії, обробляти події, змінювати глобальні змінні та реалізовувати логіку дій.

Компоненти Z-Way можна умовно розділити на **три рівні**:

- **Низькорівневий (системний)** – ядро Z-Way Engine та Device Management відповідають за стабільну роботу пристроїв і комунікацію з Z-Wave мережею.
- **Середній (логічний)** – Apps і Scripting Engine дозволяють створювати інтелектуальну поведінку системи через сценарії, таймери, автоматичні реакції.
- **Високорівневий (інтерфейсний)** – Web UI та API реалізують керування з боку кристувача та інтеграцію з зовнішніми сервісами й платформами.

### 2.3. Аналіз апаратних пристроїв, сумісних із Z-Wave

Z-Wave як бездротовий протокол для домашньої автоматизації підтримує широкий спектр пристроїв, які забезпечують керування освітленням, опаленням, безпекою, енергоспоживанням тощо. Основною перевагою цієї технології є її сумісність між пристроями різних виробників, що досягається завдяки суворій сертифікації в рамках Z-Wave Alliance. Згідно з даними альянсу, на ринку доступні понад 3 000 пристроїв із підтримкою Z-Wave, і всі вони можуть взаємодіяти один з одним у межах однієї мережі.

Пристрої можна умовно поділити на кілька категорій: сенсори (датчики руху, температури, вологості, відкриття), виконавчі пристрої (розетки, реле, термостати), контролери (центральні хаби, USB-донгли), а також елементи безпеки (сирени, детектори диму та газу). Завдяки підтримці мережі типу mesh, ці пристрої також функціонують як повторювачі сигналу, посилюючи загальну стабільність системи [21].

Наприклад, Fibaro пропонує широкий асортимент пристроїв з витонченим дизайном: зокрема, датчики руху з сенсорами освітлення й температури, водяні клапани для запобігання протіканню, а також настінні реле, які дозволяють автоматизувати навіть звичайні вимикачі. Компанія Aeotec спеціалізується на багатофункціональних пристроях, як-от Smart Switch 7 – розумна розетка з вимірюванням енергоспоживання.

Також варто згадати про пристрої Qubino та Heatit, які орієнтовані на інтеграцію з системами опалення та енергоменеджменту. Їхні модулі можуть вбудовуватись у розподільчі щитки або монтуватися під стіну, що робить їх зручними для професійного монтажу в рамках комерційних і житлових проектів. Наприклад, модулі Qubino мають вбудовані температурні сенсори та можуть керувати електроприводами, теплими підлогами, вентиляторами тощо. Heatit же пропонує ряд терморегуляторів із сенсорними панелями, які підтримують як ручне керування, так і віддалене управління через Z-Wave-хаб. Обидві компанії активно підтримують сертифікацію Z-Wave Plus, що гарантує високу енергоефективність та розширену дальність дії.

Окрему увагу слід приділити питанню сумісності пристроїв Z-Wave з іншими технологіями автоматизації. Попри те, що протокол Z-Wave має власну специфікацію, включно з частотним діапазоном (в Європі – 868,4 МГц), модуляцією (FSK/GFSK), методами маршрутизації (mesh-мережа) та захистом (S2 Security, AES-128), він здатен ефективно взаємодіяти з іншими технологіями за умови наявності відповідного програмного або апаратного шлюзу. Це стало можливим завдяки розвитку універсальних контролерів і хабів, які підтримують кілька протоколів одночасно.

Серед найбільш поширених рішень – Home Assistant Yellow, Hubitat Elevation, SmartThings Hub v3, Aeotec Z-Stick у поєднанні з платформою Z-Way. Вони дозволяють інтегрувати Z-Wave з протоколами Zigbee, Wi-Fi, Bluetooth LE, а також з новими стандартами на кшталт Matter або Thread. Наприклад, Zigbee працює на частоті 2,4 ГГц, використовує DSSS-модуляцію та також підтримує mesh-топологію. Хоча обидві технології мають схожу архітектуру, їх безпосереднє з'єднання неможливе без проміжного шлюзу. Саме таку функцію виконують сучасні багатопрокольні контролери, які об'єднують пристрої у спільну логіку сценаріїв та автоматизації.

На практиці це означає, що користувач може, наприклад, поєднати Zigbee-освітлення з Z-Wave-термостатами, Wi-Fi-камерами та Bluetooth-браслетами присутності – всі ці компоненти можуть взаємодіяти між собою, наприклад, при

виявленні руху автоматично увімкнути світло й надіслати повідомлення в мобільний застосунок. Це стає можливим завдяки загальній логіці автоматизації, яку задає центральний контролер або програмне середовище типу Z-Way, Node-RED чи Home Assistant.

Крім того, Z-Wave підтримує різні способи зовнішньої інтеграції: за допомогою REST API, WebSocket, MQTT, HTTP-запитів або інтеграцій з хмарними сервісами. Наприклад, за допомогою MQTT-шлюзу можна реалізувати двосторонній обмін даними між сенсорами Z-Wave та платформами моніторингу, такими як Grafana або InfluxDB. REST API Z-Way дозволяє керувати будь-яким пристроєм через стандартні веб-запити, що є особливо зручним для сторонніх розробників або мобільних додатків.

Слід також зазначити, що сучасні Z-Wave пристрої, сертифіковані за стандартом Z-Wave Plus v2, мають покращену підтримку SmartStart – автоматичного налаштування при додаванні в мережу, а також енергоефективніший режим сну для батарейних пристроїв. У порівнянні з попередніми поколіннями, нові моделі забезпечують дальність дії до 150 метрів на відкритому просторі, мають зменшену затримку передачі команд і забезпечують надійніший захист переданих даних.

Інтеграція з іншими екосистемами також реалізується через голосові асистенти – Google Assistant, Amazon Alexa, Apple Siri (через HomeKit). Наприклад, якщо Z-Wave-хаб підтримує інтеграцію з Alexa, користувач може голосом увімкнути або вимкнути освітлення, змінити температуру термостату або перевірити стан дверного сенсора.

## **2.4. Інтеграція Z-Way з іншими платформами та системами**

Однією з головних переваг платформи Z-Way є її відкритість та висока адаптивність до роботи з іншими технологіями розумного дому. Завдяки підтримці різних інтерфейсів – REST API, WebSocket, MQTT, а також можливості

встановлення додаткових модулів, Z-Way легко інтегрується з багатьма сучасними програмними платформами для автоматизації. Це дозволяє створювати складні сценарії взаємодії пристроїв навіть за умов використання різних протоколів передачі даних [28].

Платформа надає зручні засоби для обміну даними з такими популярними рішеннями, як Home Assistant, OpenHAB, Domoticz та Node-RED. Наприклад, інтеграція з Home Assistant можлива через офіційний компонент Z-Wave JS або за допомогою HTTP API. Це дозволяє централізовано керувати всіма пристроями незалежно від їх виробника чи протоколу. У таких конфігураціях Z-Way часто виступає як спеціалізований шлюз для Z-Wave-пристроїв, тоді як Home Assistant забезпечує гнучку автоматизацію і користувацький інтерфейс [16].

Особливо зручною є підтримка Node-RED – системи візуального програмування, яка дозволяє створювати складні логічні сценарії без глибоких знань програмування. Завдяки WebSocket API або MQTT-шлюзу, Z-Way може обмінюватися подіями з Node-RED у реальному часі. Це дає змогу реалізувати складні умови автоматизації, наприклад, включення освітлення при виявленні руху лише в темний час доби, або надсилання сповіщення на телефон при одночасному спрацюванні кількох сенсорів.

Ще одним напрямком інтеграції є зв'язок із голосовими помічниками, такими як Amazon Alexa або Google Assistant. Через проміжні сервіси – IFTTT або Homebridge – можна реалізувати голосове керування пристроями, що підключені до Z-Way. Таким чином, користувач може, наприклад, запустити сценарій “Вечір” лише однією голосовою командою, яка автоматично вимикає освітлення в коридорі, опускає жалюзі та активує охоронну сигналізацію [30].

Також варто зазначити, що Z-Way дозволяє створення систем збору та аналітики даних. Завдяки підтримці REST API можлива інтеграція з такими платформами, як InfluxDB або Grafana. Це дає змогу не лише зберігати історію роботи пристроїв, але й створювати графіки температурних змін, споживання електроенергії чи активності сенсорів. Такі можливості особливо корисні для виявлення тенденцій або оптимізації витрат у системі «розумний будинок».

Крім того, завдяки відкритому коду ядра Z-Way та доступності SDK, розробники можуть самостійно створювати або модифікувати плагіни, що робить платформу надзвичайно гнучкою. Це важливо для складних або нестандартних проєктів, де типових рішень недостатньо.

Таким чином, система Z-Way не тільки ефективно працює з пристроями на базі Z-Wave, а й виступає як універсальний шлюз, здатний поєднувати різні технології, протоколи та програмні середовища в єдину автоматизовану екосистему.

## **2.5. Порівняння Z-Way з іншими альтернативними системами управління**

У світі розумних систем автоматизації існує чимало програмних платформ, які дозволяють створювати інтелектуальні житлові або комерційні середовища. Серед найбільш популярних – Home Assistant, OpenHAB та Domoticz. Платформа Z-Way, у свою чергу, займає особливе місце завдяки своїй глибокій спеціалізації на протоколі Z-Wave, що дає їй як переваги, так і певні обмеження.

На відміну від універсальних систем, таких як Home Assistant, які підтримують десятки різних протоколів і тисячі пристроїв, Z-Way фокусується насамперед на глибокій, нативній підтримці Z-Wave. Це забезпечує кращу сумісність, стабільнішу роботу й повний доступ до всіх функцій пристроїв цього стандарту. У той же час, Home Assistant може бути більш привабливим для користувачів, які хочуть поєднувати Z-Wave з Zigbee, Wi-Fi або Bluetooth-пристроями без додаткових шлюзів [29].

OpenHAB, у свою чергу, є потужною платформою з акцентом на кросплатформеність і стабільну роботу у великих інсталяціях. Завдяки підтримці Java, вона може запускатися практично на будь-якому пристрої – від Raspberry Pi до серверів. У порівнянні з Z-Way, OpenHAB пропонує ширшу гнучкість у налаштуванні логіки та автоматизацій, однак це часто потребує більш глибоких

технічних знань. Z-Way виграє у зручності – завдяки простішому інтерфейсу, інтегрованому магазину додатків і детальному API-доступу до пристроїв [17].

Domoticz – це ще одна платформа з відкритим кодом, орієнтована на простоту й легкість у використанні. Вона добре працює з різноманітними протоколами, включно з Z-Wave, однак підтримка часто обмежується базовими функціями. У цьому контексті Z-Way забезпечує більш повну реалізацію можливостей пристроїв, таких як асоціації, параметри конфігурації чи сцени, що критично важливо для створення ефективної автоматизації [32].

Ще одним аспектом порівняння є підтримка мобільних додатків. Z-Way має офіційний застосунок Z-Way App, який дозволяє здійснювати базове керування пристроями, перевіряти стан сенсорів та отримувати повідомлення. Home Assistant має більш гнучкий мобільний інтерфейс, підтримку локаційного відстеження та push-сповіщень. OpenHAB також надає свій офіційний застосунок, але його конфігурація може бути складнішою для новачків.

Z-Way також відзначається добрим рівнем безпеки. Підтримка S2 Security (включно з шифруванням передачі даних у Z-Wave мережі), авторизація через токени в API та можливість ізоляції мережі – усе це робить платформу відповідною для сценаріїв, де безпека критична. У Home Assistant питання захисту більше перекладене на користувача: необхідно самостійно налаштовувати HTTPS, брандмауери та VPN-доступ. Сумісність пристроїв з Z-Wave можна переглянути у таблиці 2.2 у Додатку В[18].

Для кращого розуміння особливостей та переваг кожної з платформ, доцільно візуалізувати ключові аспекти у вигляді порівняльної інфографіки. На рисунку 2.2 представлено радарну діаграму, яка порівнює чотири популярні системи керування "розумним будинком" – Z-Way, Home Assistant, OpenHAB та Domoticz – за п'ятьма основними критеріями: глибина підтримки Z-Wave, підтримка інших протоколів, зручність інтерфейсу, гнучкість налаштування автоматизацій та рівень безпеки.

З діаграми видно, що платформа Z-Way має найбільшу перевагу у повній реалізації функціоналу Z-Wave та високому рівні безпеки. Home Assistant



демонструє найкращу підтримку інших протоколів і потужні можливості автоматизації, однак потребує більше ручного налаштування. OpenHAB відзначається високою масштабованістю і стабільністю, проте має складнішу конфігурацію. Domoticz, у свою чергу, залишається простим у використанні, але з обмеженою функціональністю.



Рисунок 2.2 – Порівняння платформ Z-Way, Home Assistant, OpenHAB та Domoticz за ключовими технічними критеріями

## РОЗДІЛ 3. РЕАЛІЗАЦІЯ СИСТЕМИ «РОЗУМНИЙ БУДИНОК» НА БАЗІ СЕРВІСУ Z-WAY

### 3.1. Постановка задачі та вимоги до системи

Основна задача автоматизованої системи «Розумний будинок» полягає у створенні єдиного середовища для централізованого керування ключовими підсистемами житла з метою підвищення комфорту, безпеки та енергоефективності. Платформа Z-Way на основі протоколу Z-Wave дає можливість інтегрувати освітлення, кліматичні пристрої та побутову техніку, реалізувати індивідуальні й групові сценарії автоматизації, забезпечити оперативний моніторинг та віддалений доступ.

Система має підтримувати повний цикл керування освітленням у різних приміщеннях – від простого вмикання та вимикання ламп до створення складних сценаріїв освітлення за розкладом або у відповідь на зовнішні фактори, наприклад, рівень освітленості, присутність людей чи час доби. Передбачається інтеграція датчиків руху, освітленості та температури для підвищення ефективності автоматизації та мінімізації зайвого енергоспоживання. Особливу увагу приділено можливості формування груп освітлювальних пристроїв для керування зонами або всією будівлею з одного інтерфейсу.

Вимоги до мікроклімату включають реалізацію автоматичного регулювання температури й вологості на основі заданих користувачем параметрів, сезонних режимів або спеціальних сценаріїв («нічний режим», «відпустка»). Система повинна інтегрувати термостати, кондиціонери та вентилятори, забезпечувати координацію їхньої роботи для досягнення оптимальних кліматичних умов з урахуванням економії енергоресурсів.

Управління побутовими приладами здійснюється через розумні розетки та реле, які дозволяють не лише дистанційно вмикати та вимикати пристрої, а й реалізувати автоматизацію їхньої роботи. Наприклад, запуск пральної машини за розкладом, автоматичне відключення електроприладів у разі відсутності

мешканців або виявлення аварійної ситуації (перевищення споживання струму, задимлення тощо).

Інтерфейс користувача повинен бути максимально інтуїтивним, забезпечувати швидкий доступ до всіх функцій системи, відображати поточний стан пристроїв, журнали подій, попередження про аварії, дозволяти гнучке налаштування сценаріїв, додавання та видалення нових пристроїв. Передбачається підтримка веб-інтерфейсу, мобільного застосунку та можливість підключення голосових асистентів.

До обов'язкових вимог належать масштабованість архітектури, що дозволяє розширювати систему без зупинки її роботи, а також надійність і стійкість до збоїв. Кожен компонент має автоматично відновлювати роботу після втрати живлення або з'єднання, всі дані про стан пристроїв та сценарії мають зберігатися централізовано з резервним копіюванням. В системі необхідно передбачити багаторівневу систему захисту: автентифікацію користувачів, шифрування трафіку, ведення журналів дій та своєчасне оновлення безпеки.

Крім того, система повинна мати можливість віддаленого доступу через захищений канал для керування житлом з будь-якої точки світу. Передбачається інтеграція з іншими протоколами автоматизації (наприклад, ZigBee, Wi-Fi, Bluetooth) для підвищення універсальності й подальшого розширення функціоналу. Система повинна надавати API для зовнішніх розробників, що дозволить створювати додаткові сервіси, мобільні застосунки чи розширення.

У рамках проєктування системи «Розумний будинок» було прийнято рішення зосередитися на трьох основних функціональних підсистемах: керування освітленням, клімат-контролем (опалення та кондиціонування) та побутовими електроприладами (через розетки). Такий вибір обумовлений кількома ключовими факторами:

1. Функціональна базовість. Освітлення, клімат і живлення – це базові елементи, з якими щодня взаємодіє користувач у побуті. Їх автоматизація забезпечує найбільший ефект з точки зору комфорту та енергоефективності.

2. Типовість для більшості житлових об'єктів. На відміну від складніших систем (сигналізація, мультимедіа, штори, полив тощо), освітлення, опалення та електроприлади є присутніми у кожному домогосподарстві, що робить рішення універсальним і легко адаптованим.

3. Наявність усталених рішень у середовищі Z-Wave. Для освітлення, температурного регулювання та розумних розеток існує велика кількість сертифікованих Z-Wave пристроїв із повною підтримкою командних класів, що полегшує розробку системи керування та її подальшу масштабованість.

4. Можливість демонстрації ключових сценаріїв. Навіть на базі трьох підсистем можна реалізувати показові сценарії автоматизації, як-от “Нічний режим”, який об'єднує взаємодію всіх вибраних функцій.

Таким чином, вибір саме цих підсистем дозволяє сконцентруватися на найбільш важливих і універсальних аспектах управління житловим простором, а також забезпечує баланс між складністю реалізації та практичною користю.

### **3.2. Архітектура системи “Розумний будинок”**

Архітектура системи «Розумний будинок» із використанням сервісу Z-Way має багаторівневу структуру, що дозволяє забезпечити гнучкість, масштабованість і простоту керування усіма підсистемами житла. Такий підхід дає змогу ефективно інтегрувати різноманітні пристрої – від освітлення та клімат-контролю до побутової техніки, з можливістю централізованого керування через зручний інтерфейс.

Основні компоненти архітектури:

Користувач – кінцевий оператор системи, який здійснює керування за допомогою спеціалізованого інтерфейсу. Взаємодія з системою може здійснюватися як локально, так і віддалено через мережу Інтернет.

Інтерфейс користувача – програмний модуль, що надає доступ до функціоналу системи через веб-додаток або мобільний застосунок. Інтерфейс

забезпечує ідентифікацію та автентифікацію користувачів, дозволяє здійснювати моніторинг і керування всіма підключеними пристроями, створювати індивідуальні сценарії автоматизації та переглядати історію подій.

Центральний контролер (Z-Way Controller) – апаратно-програмний комплекс, який забезпечує обробку команд від користувача, формування сценаріїв керування та обмін даними з периферійними пристроями за протоколом Z-Wave. Контролер є центральною ланкою системи, яка підтримує комунікацію між усіма складовими.

Комунікаційна мережа – сукупність каналів зв'язку, що забезпечують обмін даними між контролером і пристроями. В якості основного протоколу використовується Z-Wave, який дозволяє формувати масштабовану бездротову мережу з автоматичною маршрутизацією сигналів. Крім того, можливий доступ до системи через локальну мережу (LAN) та Інтернет, що забезпечує віддалене керування.

Z-Wave-пристрої (периферія) – набір "розумних" пристроїв, об'єднаних у підсистеми:

- Освітлення: реле, розумні лампи та вимикачі, які дозволяють здійснювати увімкнення, вимкнення світла, а також реалізовувати автоматизовані сценарії.
- Мікроклімат: термостати, кондиціонери, датчики температури та вологості, що дозволяють підтримувати оптимальні умови у приміщенні та економити енергію.
- Побутова техніка: розумні розетки, реле, контролери побутових пристроїв, що забезпечують можливість віддаленого керування.

Варто зауважити, що в архітектурі системи "Розумний будинок" користувач і інтерфейс користувача розглядаються як окремі логічні компоненти. Такий підхід зумовлений тим, що користувач виконує роль активного елемента, який ініціює дії, приймає рішення та формує сценарії, а інтерфейс – це програмна оболонка, яка забезпечує технічну реалізацію цієї взаємодії. Інтерфейс користувача не є самостійним "актором", але виступає посередником між людиною та системою, забезпечуючи автентифікацію, зворотний зв'язок, зручну візуалізацію стану

пристроїв і доступ до функцій автоматизації. У сучасних архітектурних підходах, зокрема в системах НМІ (Human-Machine Interface), подібне розділення є обґрунтованим і логічним.



Рисунок 3.1 – Опис структури систем «розумного будинку».

Представлена схема (рисунок 3.1) ілюструє взаємозв'язок між усіма ключовими елементами системи. Користувач за допомогою веб-інтерфейсу або мобільного застосунку взаємодіє із центральним контролером, який отримує команди, обробляє їх та передає сигнал до відповідних пристроїв через бездротову мережу Z-Wave.

Для наочного відображення загальної архітектури запропонованої системи «Розумний будинок» на базі Z-Way було розроблено схему комутації (рисунок 3.2). Схема демонструє основні компоненти системи та їх взаємозв'язок на рівні передачі даних і керуючих команд.

У центрі архітектури знаходиться контролер Z-WAY, який виконує функцію центрального вузла обробки даних. Контролер приймає команди з вебінтерфейсу та передає їх на відповідні пристрої, а також здійснює зворотну синхронізацію станів.

До контролера підключено такі компоненти:

- Світильник (з димером) – керується командами типу exact, що дозволяє регулювати рівень яскравості. Передача значення здійснюється в обох напрямках.
- Термостат – отримує значення температури з інтерфейсу та передає назад актуальну температуру для відображення.
- Кондиціонер – аналогічно отримує та повертає значення температури (через глобальні змінні).
- Розетки – керуються стандартними командами on/off.
- Світильник (звичайний) – просте двоступеневе керування (увімк./вимк.).

Окремим логічним блоком виступає сценарій “Нічний режим”, який ініціюється з контролера або вручну через вебінтерфейс.

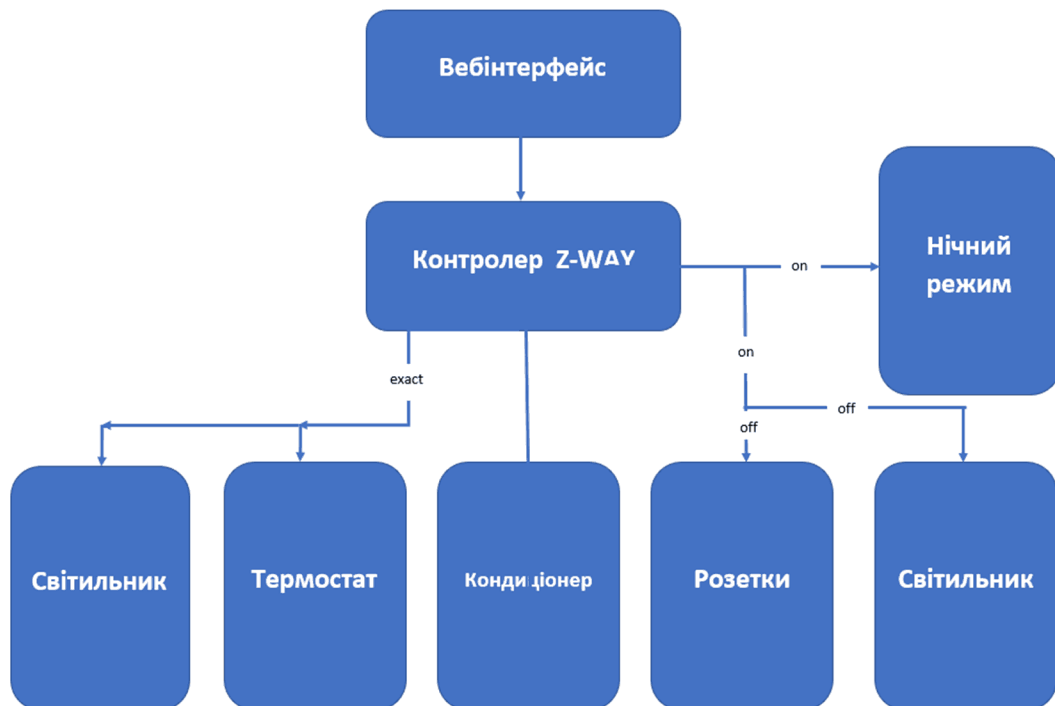


Рисунок 3.2–Схема комутації системи «Розумний будинок» на базі Z-Way

### 3.3. Реалізація модулів управління освітленням, мікрокліматом та побутовими приладами.

Для реалізації практичної частини було здійснено встановлення та

налаштування серверу Z-Way на локальному комп'ютері під управлінням Windows. Після запуску служби користувач отримує доступ до веб-інтерфейсу керування за адресою <http://localhost:8083/smarthome>.

На рисунку 3.3 зображено стартову сторінку інтерфейсу Z-Wave Smart Home, яка свідчить про успішний запуск платформи.

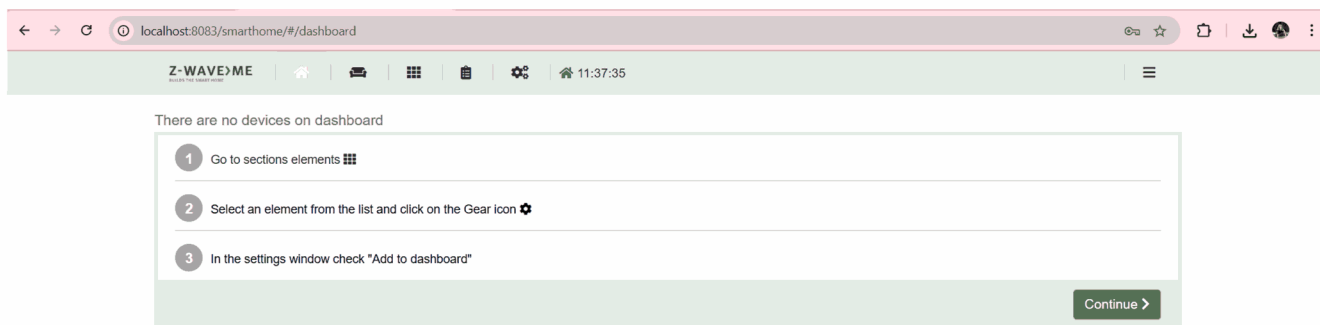


Рисунок 3.3 – Головна сторінка веб-інтерфейсу Z-Way після встановлення системи

Оскільки фізичний Z-Wave контролер у даному проєкті був відсутній, для реалізації та демонстрації роботи системи «Розумний будинок» було вирішено використати віртуальні пристрої, доступні у середовищі Z-Way. Такий підхід дозволив забезпечити повноцінне моделювання ключових функцій системи без потреби у реальному обладнанні.

Завдяки підтримці JavaScript у платформі Z-Way, усі пристрої налаштовувалися вручну: задавався тип пристрою, функціональна поведінка та взаємозв'язок між елементами. Це надало змогу побудувати логіку автоматизації, створити інтерфейс для керування кожним пристроєм та навіть реалізувати сценарії, наприклад, «Нічний режим», без залучення фізичного контролера.

Одним з перших пристроїв, реалізованих у системі, стало керування освітленням у вітальні. Для цього було створено віртуальний пристрій типу switchMultilevel, який дозволяє не лише вмикати або вимикати освітлення, а й регулювати рівень яскравості в межах від 0 до 99 % ( рисунок 3.4).

У налаштуваннях пристрою було вказано ім'я – «Світло у вітальні», що дає змогу легко ідентифікувати його у загальному інтерфейсі. Як тип пристрою обрано switchMultilevel – димер, який забезпечує змінне керування.



Параметр Code for action має значення: `devStatus = %%;`

Це означає, що під час взаємодії з елементом інтерфейсу значення яскравості автоматично присвоюється змінній `devStatus`.

У полі Code to get value вказано: `return devStatus;`

Це дозволяє зчитувати поточний стан пристрою для оновлення візуального інтерфейсу або логіки сценаріїв.

Важливою особливістю є встановлення поля інтервалу опитування у значення 0, а також активація опції «Update value on action». Це гарантує, що стан пристрою оновлюється миттєво після взаємодії з ним, без зайвого навантаження на систему.

The screenshot shows a configuration form for a device named "switchMultilevel". The form includes the following fields and options:

- Name:** "Світло у вітальні"
- Code for action:** `devStatus = %% ;` (with a note: "%% will represent value from 0 to 99")
- Code to get value:** `return devStatus;` (with a note: "Should return value from 0 to 99")
- Interval in seconds between polling requests:** "0" (with a note: "Empty or 0 to disable periodical requests (explicit update command will still initiate request process)")
- Update value on action:**  (checked)

Рисунок 3.4 – Налаштування віртуального пристрою типу `switchMultilevel` для керування освітленням у вітальні.

Для керування освітленням у спальні було створено віртуальний пристрій типу `switchBinary`, що дозволяє реалізувати базову двоступеневу логіку – увімкнення або вимкнення світла. Такий тип пристрою є одним із найпоширеніших у системах автоматизації, коли не потрібне регулювання яскравості, а лише вмикання/вимкання лампи( див. рисунок 3.5).

У полі Name було вказано логічну назву – «Світло у спальні». У полі Device type обрано значення switchBinary, що відповідає класичному перемикачу стану.

Нижче було визначено логіку обробки команд:

У полі Code for action On записано: `devStatus = "on";`

У полі Code for action Off: `devStatus = "off";`

Це означає, що при натисканні на елемент інтерфейсу вмикання або вимикання лампи виконується шляхом зміни значення змінної devStatus, яка виступає логічним прапором стану пристрою.

Команда `return devStatus;` у полі Code to get value забезпечує зчитування актуального стану пристрою, що дозволяє інтерфейсу відображати статус лампи в реальному часі.

Як і в попередньому випадку, періодичне опитування пристрою відключено (інтервал = 0), що оптимізує навантаження на контролер. Всі зміни ініціюються лише в момент дії користувача, що відповідає принципу подійно-орієнтованої логіки.

Name	Світло у спальні
Device type	switchBinary
Code for action On	<code>devStatus = "on";</code> <small>%% will represent value 'on'</small>
Code for action Off	<code>devStatus = "off";</code> <small>%% will represent value 'off'</small>
Code to get value	<code>return devStatus;</code> <small>Should return 'on' or 'off' value</small>
Interval in seconds between polling requests	0

Рисунок 3.5 – Налаштування пристрою switchBinary для двоступеневого керування світлом у спальні.

У рамках підсистеми керування побутовими приладами було реалізовано два віртуальні пристрої типу switchBinary:(див. рисунок 3.6)

«Розетка (чайник)» – умовно імітує електроживлення для чайника.

«Розетка (ТБ)» – відповідає за підключення телевізора у спальні.

Обидва пристрої дають змогу вмикати або вимикати подачу живлення до умовного електроприладу одним натисканням, що забезпечує зручність і безпеку, особливо при використанні у нічному сценарії або за відсутності мешканців.

У кожному з пристроїв задана проста логіка дій:

Code for action On: devStatus = "on";

Code for action Off: devStatus = "off";

Завдяки цьому система може не лише надсилати команду на зміну стану розетки, а й контролювати її актуальний статус у реальному часі. Обидва пристрої мають значення поля інтервалу опитування = 0, що означає відключення періодичних запитів.

Name	<input type="text" value="Розетка(чайник)"/>
	<input type="text" value="switchBinary"/>
Code for action On	<input type="text" value='devStatus = "on";'/> <small>%% will represent value 'on'</small>
Code for action Off	<input type="text" value='devStatus = "off";'/> <small>%% will represent value 'off'</small>
Code to get value	<input type="text" value="return devStatus;"/> <small>Should return 'on' or 'off' value</small>
Interval in seconds between polling requests	<input type="text" value="0"/>

### Рисунок 3.6 – Створення віртуальних розеток для чайника та телевізора в інтерфейсі Z-Way

Для реалізації повного циклу керування мікрокліматом у вітальні було створено два взаємопов'язані віртуальні пристрої:(див. рисунок 3.7)

- «Кондиціонер» – виконавчий пристрій типу `switchMultilevel`, який дозволяє користувачу задавати бажану температуру;
- «Температура кондиціонера» – сенсор типу `sensorMultilevel`, який динамічно зчитує встановлене значення для відображення у системі та використання в логіці сценаріїв.

Для створення пристрою «Кондиціонер» у параметрах пристрою було обрано тип `switchMultilevel`, який ідеально підходить для задачі встановлення температури, оскільки забезпечує можливість плавного регулювання. Це значно розширює функціональність у порівнянні зі стандартним `switchBinary`.

Поле `Code for action` містить наступний скрипт: `global.acTemp = Math.round(18 + ((%% / 99) * (30 - 18)));`

Цей вираз дозволяє трансформувати значення повзунка (від 0 до 99) у температурний діапазон від 18 °С до 30 °С. Результат округлюється і зберігається у глобальній змінній `global.acTemp`, яка виступає єдиним джерелом істини для пов'язаного сенсора.

Поле `Code to get value: return global.acTemp;` - забезпечує зворотну синхронізацію – інтерфейс отримує актуальне значення температури і відображає його на панелі керування. Таким чином, навіть після перезавантаження сторінки чи зміни інтерфейсу користувач завжди бачить правильне значення.

З технічної точки зору, встановлено інтервал опитування = 0, а також активовано опцію “Update value on action”, що дозволяє системі оновлювати значення одразу після дії користувача без додаткових запитів.

Name

Кондеціонер

---

switchMultilevel

---

Code for action

```
global.acTemp = Math.round(18 + (%% / 99) * (30 - 18));
```

**i** %% will represent value from 0 to 99

---

Code to get value

```
return global.acTemp;
```

**i** Should return value from 0 to 99

---

Interval in seconds between polling requests

0

**i** Empty or 0 to disable periodical requests (explicit update command will still initiate request process)

Рисунок 3.7 – Налаштування віртуального пристрою switchMultilevel для кондиціонера.

Другий пристрій – сенсор температури ( див. рисунок 3.8) – використовує тип sensorMultilevel. Його основне завдання – зчитувати значення глобальної змінної global.acTemp і відобразити його у вигляді числового індикатора температури в °C.

Поле Code to get value має захисну перевірку на існування змінної: `if (typeof global !== 'undefined' && global.acTemp !== undefined) {return global.acTemp;} return 0;`

Цей код гарантує стабільну роботу навіть у випадках, коли змінна ще не була ініціалізована або очищена. Значення виводиться у форматі °C, що задається через параметр Sensor scale.

Name	Температура кондиціонера
sensorMultilevel	
Icon	temperature
Code to get value	if (typeof global !== 'undefined' && global.acTemp !== undefined) { return global.acTemp; } return 0;
	<b>i</b> Should return number
Interval in seconds between polling requests	0
	<b>i</b> Empty or 0 to disable periodical requests (explicit update command will still initiate request process)
Sensor scale	°C

Рисунок 3.8 – Налаштування сенсора sensorMultilevel для зчитування температури кондиціонера.

Щоб уникнути помилок при першому запуску системи або при очищенні пам'яті, у полі JavaScript-коду сторінки було реалізовано ініціалізацію змінної global.acTemp: `if (typeof global === 'undefined') global = {};`

`if (global.acTemp === undefined) global.acTemp = 23;`

Це дозволяє за замовчуванням задати температуру на рівні 23 °C, навіть якщо користувач ще не взаємодіяв із повзунком кондиціонера.

Аналогічно до кондиціонера, для забезпечення повного контролю над мікрокліматом у приміщенні було **створено** два віртуальні пристрої:

Пристрій «Опалення» (встановлення температури) має тип пристрою – switchMultilevel, що забезпечує плавну регуляцію температури( див. рисунок 3.9).

Його логіка реалізована через JavaScript у полі Code for action:

`if (%% < 15) { global.heatingTemp = 15; } else if (%% > 30) { global.heatingTemp = 30; } else { global.heatingTemp = %%; }`

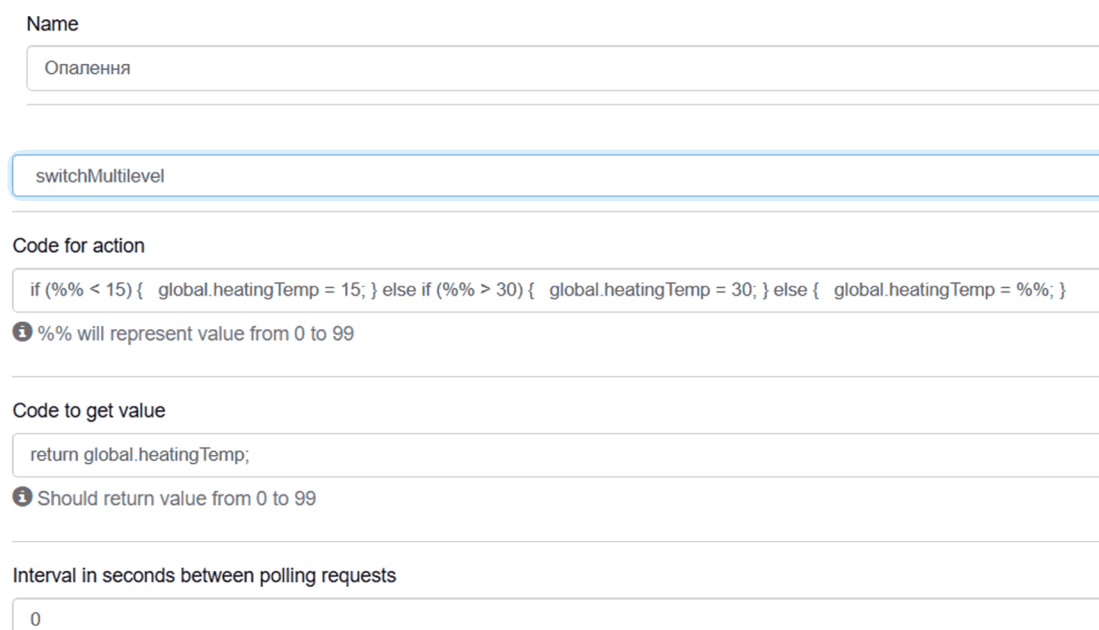
Цей код гарантує, що незалежно від значення, яке встановлює користувач повзунком, температура не вийде за межі 15–30 °C, навіть якщо система отримує

некоректні дані. Також це дозволяє використовувати реальне числове значення повзунка як температуру, а не як відсоткову шкалу.

Для зчитування значення у полі **Code to get value** вказано: `return global.heatingTemp;`

Це дозволяє інтерфейсу автоматично оновлювати значення температури при кожній зміні, що особливо важливо для забезпечення коректної візуалізації та сценаріїв автоматизації.

Параметри опитування встановлені так само, як і у всіх пристроях: Інтервал = 0 (відсутність періодичних запитів) Опція “Update value on action” увімкнена, що забезпечує миттєве оновлення стану.



Name

Опалення

switchMultilevel

Code for action

`if (%% < 15) { global.heatingTemp = 15; } else if (%% > 30) { global.heatingTemp = 30; } else { global.heatingTemp = %%; }`

%% will represent value from 0 to 99

Code to get value

`return global.heatingTemp;`

Should return value from 0 to 99

Interval in seconds between polling requests

0

Рисунок 3.9 – Налаштування пристрою `switchMultilevel` для керування температурою опалення.

Пристрій «Температура опалення»(див. рисунок 3.10) типу **sensorMultilevel** призначений для зчитування значення температури, яке було встановлено пристроєм “Опалення”.

Його логіка зчитування реалізована у полі **Code to get value** наступним чином:

```
if (typeof global !== 'undefined' && global.heatingTemp !== undefined) {
```

```
return global.heatingTemp;}
return 0;
```

Завдяки цьому система коректно реагує на всі зміни та гарантує стабільну роботу навіть у випадку, якщо змінна ще не існує або була видалена. Виведення температури здійснюється у градусах Цельсія завдяки встановленому Sensor scale: °C.

Зв'язок між двома пристроями забезпечується за допомогою глобальної змінної `global.heatingTemp`, яка:

- записується пристроєм “Опалення”;
- зчитується пристроєм “Температура опалення”.

Така модель дозволяє імітувати двосторонню комунікацію, яка в реальних умовах реалізується апаратним термостатом і температурним датчиком. Це також дозволяє швидко адаптувати систему до реальних пристроїв у майбутньому – достатньо буде замінити `sensorMultilevel` на справжній сенсор через Z-Wave.

Name	Температура опалення
sensorMultilevel	
Icon	temperature
Code to get value	if (typeof global !== 'undefined' && global.heatingTemp !== undefined) { return global.heatingTemp; } return 0;
Should return number	<input checked="" type="checkbox"/>
Interval in seconds between polling requests	0
Empty or 0 to disable periodical requests (explicit update command will still initiate request process)	<input checked="" type="checkbox"/>
Sensor scale	°C

Рисунок 3.10 – Налаштування сенсора `sensorMultilevel` для зчитування температури опалення.

Для інтеграції декількох пристроїв у спільну дію було реалізовано сценарій



типу `toggleButton` під назвою «Нічний режим». Цей сценарій дозволяє в один клік активувати комплекс дій: вимкнути освітлення, розетки, перевести кондиціонер та опалення у режим сну (енергозберігаючий стан).

Сценарій «Нічний режим» виконує такі дії: Вимикає світло у вітальні та спальні; Вимикає розетку з телевизором і розетку з чайником; Встановлює температуру кондиціонера на 23 °С; Встановлює температуру опалення на 18 °С.

У полі `Code for action On` сценарію було вказано JavaScript-код, що напряму звертається до пристроїв системи через API Z-Way. Повний код реалізації сценарію «Нічний режим» наведено у Додатку Б.

#### Принцип роботи

- `commandClasses[37]` – клас команд керування `Switch Binary`, який дозволяє передавати значення 0 (вимкнено) або 255 (увімкнено);
- `Set(0)` – команда вимкнення пристрою;
- `zway.devices[ID]` – ідентифікатор відповідного пристрою (відповідає порядку їх створення у системі);
- `global.acTemp` і `global.heatingTemp` – змінні, які використовуються для встановлення температури в пристроях керування кліматом.

Після натискання кнопки «Нічний режим», усі команди виконуються миттєво й одночасно що зменшує витрати часу користувача;

Завдяки гнучкості JavaScript, цей сценарій легко розширюється – можна додати команди на основі розкладу, наявності руху, або з урахуванням зовнішньої погоди (у майбутній інтеграції).

Для впорядкування структури розумного будинку та забезпечення зручного візуального керування всі пристрої були розподілені по кімнатах ( рисунок 3.11) , відповідно до їхнього функціонального призначення. Це дозволяє не лише логічно структурувати інтерфейс, але й забезпечує основу для сценаріїв, які можуть застосовуватись до окремих зон помешкання (наприклад, нічний режим для спальні та вітальні).

У середовищі Z-Way було створено наступні кімнати:

- Вітальня – містить освітлення, кондиціонер, температурний сенсор кондиціонера;
- Кухня – включає розетку для електрочайника;
- Спальня – включає світло, опалення, сенсор температури, розетку для телевізора;

Усі кімнати створено вручну через інтерфейс «Add new room» з подальшим налаштуванням іконок, назв та зображень для зручності користувача. Після цього відповідні пристрої було закріплено за кімнатами, що дозволило:

- отримати зручну **візуалізацію пристроїв за локаціями**;
- швидко перемикатися між підсистемами;



Рисунок 3.11 – Інтерфейс системи Z-Way з прикладом створених кімнат та розподілу пристроїв по зонах.

Після створення та налаштування всіх пристроїв у середовищі Z-Way було сформовано основну панель керування (рисунки 3.12), яка дозволяє користувачу взаємодіяти з підсистемами «розумного будинку» в єдиному інтерфейсі. На панелі відображено:

- елементи освітлення (вітальня, спальня);
- розетки для побутових приладів (чайник, телевізор);
- блоки керування та зчитування температури (опалення, кондиціонер);
- сенсорні значення температури у градусах Цельсія;
- сценарій «Нічний режим», який об'єднує логіку взаємодії всіх підсистем.

Інтерфейс є візуально простим і функціональним – усі елементи згруповано відповідно до приміщень, що дозволяє швидко орієнтуватися в системі. Кожен пристрій має інтуїтивне позначення (пиктограми, назви, температурні значення), а сценарії легко активуються одним натисканням.

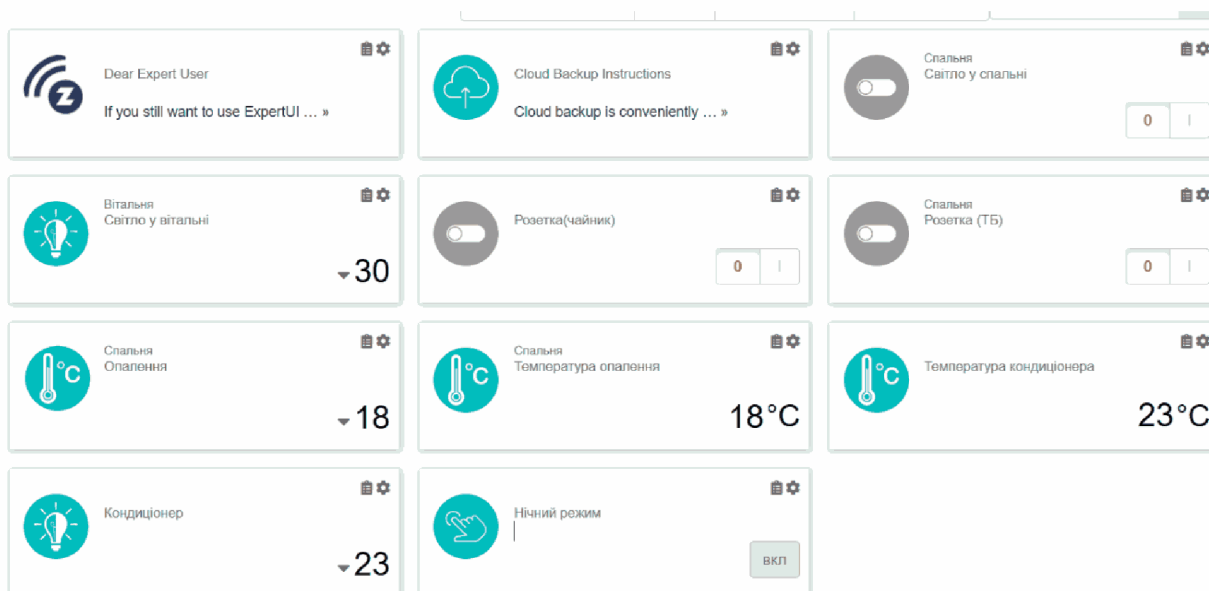


Рисунок 3.12 – Основна панель керування системою «Розумний будинок» у середовищі Z-Way після повного налаштування.

Окрім стандартного вебінтерфейсу, Z-Way надає можливість віддалено керувати пристроями через REST API. Це відкриває перспективи створення окремого клієнтського сайту або мобільного застосунку для персоналізованого доступу до функцій системи. Кожен пристрій може бути керований через HTTP-запити, наприклад:

`http://localhost:8083/ZWaveAPI/Run/devices[ID].instances[0].commandClasses[37].Set(255)` – для вмикання пристрою.

Це дозволяє інтегрувати систему у більш складні архітектури управління або розширити функціональність за межі стандартного інтерфейсу Z-Way.

Для забезпечення повноцінного уявлення про архітектуру системи також було деталізовано специфікацію використаних пристроїв (навіть у випадку симуляції у віртуальному середовищі Z-Way). У таблиці 3.1 наведено перелік основних компонентів системи, їх функціональне призначення та тип пристрою,

обраний при створенні

Таблиця 3.1 Специфікація пристроїв системи «Розумний будинок» та їх логічне підключення

№	Назва пристрою	Тип (в Z-Way)	Функція
1	Світло у вітальні	switchMultilevel	Керування яскравістю (дімер)
2	Світло у спальні	switchBinary	Вмикання/вимикання освітлення
3	Розетка (чайник)	switchBinary	Вмикання/вимикання електроживлення
4	Розетка (ТБ)	switchBinary	Вмикання/вимикання живлення телевізора
5	Кондиціонер	switchMultilevel	Встановлення температури (18–30°C)
6	Температура кондиціонера	sensorMultilevel	Відображення встановленої температури
7	Опалення	switchMultilevel	Керування температурою обігріву
8	Температура опалення	sensorMultilevel	Зчитування заданої температури обігріву
9	Нічний режим	toggleButton	Сценарій керування всіма підсистемами

Таким чином, у межах даного проекту була повністю реалізована функціональна модель системи «Розумний будинок» з використанням віртуальних пристроїв у середовищі Z-Way.

### 3.4. Розробка інтерфейсу користувача

Після завершення конфігурації пристроїв у середовищі Z-Way, наступним кроком стала розробка користувацького інтерфейсу, що дозволяє взаємодіяти з системою в інтерактивному режимі.

У межах реалізації системи керування елементами «Розумного будинку» було розроблено вебдодаток, який забезпечує інтерактивне керування основними пристроями – освітленням, опаленням, кондиціонером, розетками, а також дозволяє запускати заздалегідь визначені сценарії, зокрема «Нічний режим». Інтерфейс побудований відповідно до логіки функціонування системи автоматизації: кожна дія користувача (натискання кнопки, переміщення повзунка яскравості, зміна температури тощо) ініціює зміну відповідних змінних стану.

Вебінтерфейс спроектовано з використанням елементів керування (кнопки, слайдери, регулятори), що відповідають кожному окремому пристрою. Кожен блок інтерфейсу пов'язаний із конкретним ID пристрою, зареєстрованим у системі Z-Way, а дії користувача транслюються у вигляді HTTP-запитів до відповідного API-методу контролера. Отримані у відповідь дані – поточний стан пристрою, рівень яскравості, температура тощо – відображаються в інтерфейсі у режимі реального часу.

У розробленому вебінтерфейсі реалізовано повноцінну логіку керування фізичними пристроями через контролер Z-Wave, зокрема Raspberry або USB Z-Wave Stick, який функціонує під керуванням програмного забезпечення Z-Way. Взаємодія вебдодатку з контролером відбувається за допомогою відкритого REST API, що надається платформою Z-Way. Кожна дія користувача, здійснена через інтерфейс, автоматично генерує відповідний HTTP-запит до контролера, що активує певну команду.

Наприклад, натискання кнопки «Увімкнути світло у вітальні» викликає запит до адреси на зразок:

`http://<іпконтролера>:8083/ZAutomation/api/v1/devices/<device_id>/command/on`, внаслідок чого фізичний пристрій отримує команду на вмикання. Подібним чином реалізовано керування температурою, живленням розеток, запуском сценаріїв, таких як «Нічний режим» тощо.

Завдяки зворотньому зв'язку, який також надає API Z-Way, вебінтерфейс може в режимі реального часу отримувати актуальний стан кожного пристрою. Це дозволяє підтримувати динамічне оновлення інтерфейсу, синхронізоване з фактичним станом системи: індикатори змінюють колір, значення яскравості чи температури оновлюються автоматично після виконання дій.

Функціонально інтерфейс охоплює всі основні підсистеми «розумного будинку», зокрема:

- Керування освітленням – вмикання, вимикання, диммування світильників з індикацією поточного рівня яскравості;

- Регулювання мікроклімату – контроль температури за допомогою інтерфейсу опалення та кондиціонера;
- Управління побутовими приладами – віддалене керування розумними розетками;
- Автоматизовані сценарії – запуск типових режимів, як-от «Нічний режим», що централізовано керує кількома пристроями.

Всі елементи інтерфейсу структуровані у вигляді окремих інтерактивних блоків, що забезпечує швидкий доступ до функцій керування. Кожен блок оснащено графічними іконками, текстовими підписами та індикаторами поточного стану, що робить інтерфейс зручним і зрозумілим навіть для початкових користувачів.

На рисунку 3.13 зображено загальний вигляд розробленого веб-інтерфейсу системи “розумний будинок”.

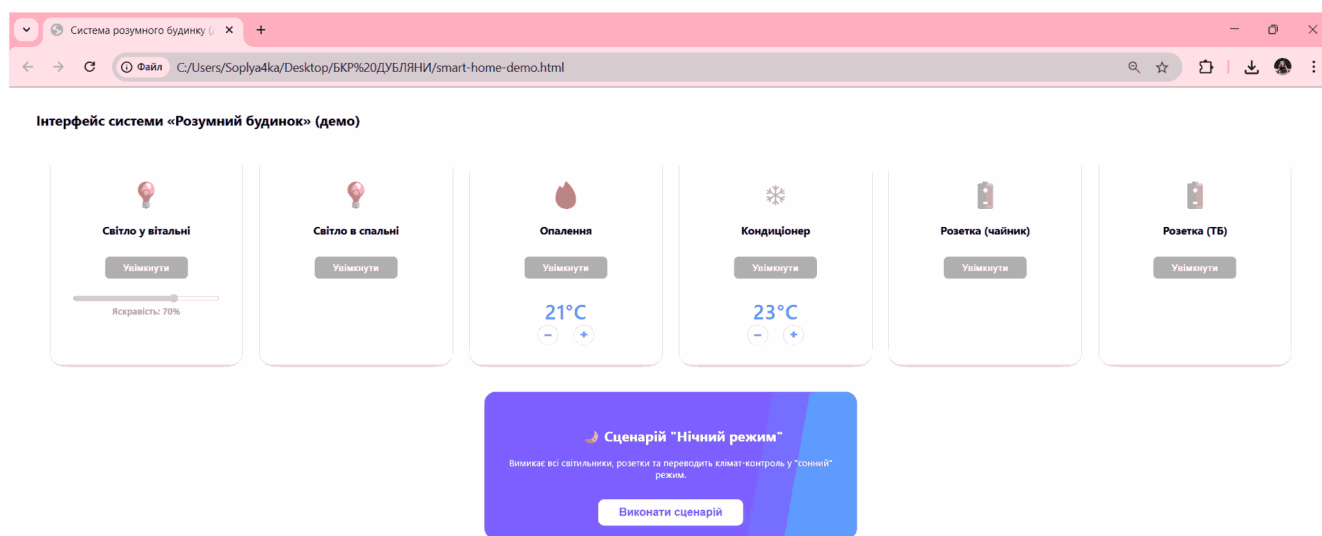


Рисунок 3.13 – Головна сторінка розробленого веб-інтерфейсу

Інтерфейс системи побудовано у вигляді набору інтерактивних “карток”, кожна з яких відображає окремий пристрій або функціональну підсистему розумного будинку. Така структура забезпечує логічну організацію керування та зручну навігацію для користувача. При взаємодії з елементами інтерфейсу – натисканні кнопок, зміні повзунків або інших параметрів – генерується відповідний запит до API контролера, що змінює стан пристрою у фізичній мережі.

Особливий акцент зроблено на підсистемі освітлення. У вебінтерфейсі реалізовано окремі картки для керування світильниками, наприклад, “Світло у вітальні” та “Світло в спальні”(рисунок 3.14). Кожен світильник оснащено кнопками увімкнення та вимкнення, а для деяких передбачено додатковий елемент керування – диммер, реалізований у вигляді повзунка. Переміщення повзунка автоматично формує запит з параметром яскравості у відсотках (від 0% до 100%) і надсилає його контролеру для негайного виконання. Відповідь контролера оновлює стан інтерфейсу, синхронізуючи відображення з реальним освітленням у приміщенні.

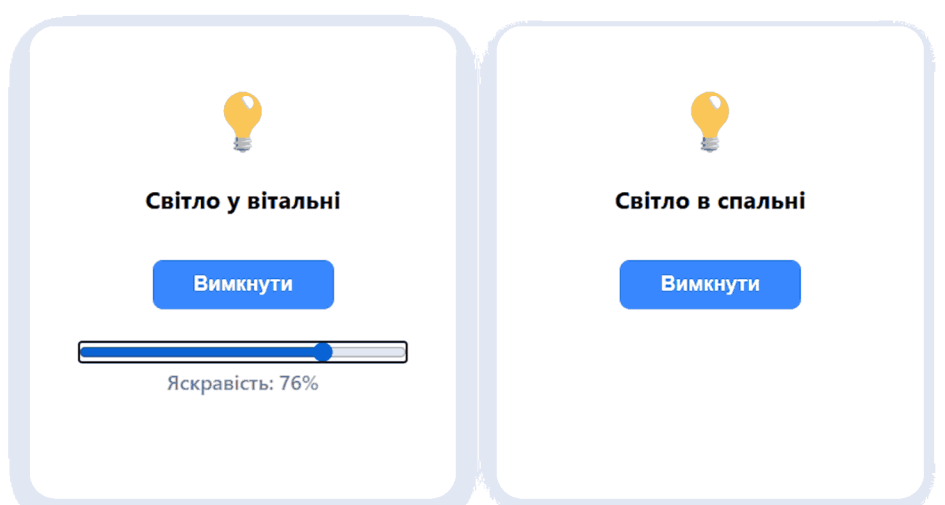


Рисунок 3.14 – Керування світлом

Регулювання яскравості освітлення здійснюється за допомогою інтерактивного повзунка, який формує значення у діапазоні від 0% до 100%. Після зміни положення повзунка, у систему генерується HTTP-запит до Z-Wave контролера через відкритий API інтерфейс Z-Way. Наприклад, якщо користувач виставляє яскравість світильника на 75%, формується запит: <http://<ip-адреса-контролера>:8083/ZAutomation/api/v1/devices/<device-id>/command/exact?level=75>

Цей запит викликає метод `command/exact` для пристрою з відповідним `device-id`, передаючи параметр `level=75`, який задає необхідний рівень яскравості. Значення яскравості паралельно відображається у вигляді числового індикатора та змінює колір іконки світильника в інтерфейсі. Це дозволяє користувачеві одразу оцінити зміни стану освітлення без додаткових дій.

Керування мікрокліматом – опаленням і кондиціонуванням – реалізовано через окремі блоки керування з кнопками вмикання/вимикання та індикаторами поточного стану(рисунок 3.15). Наприклад, натискання кнопки вмикання кондиціонера надсилає запит:

<http://<ip-адреса-контролера>:8083/ZAutomation/api/v1/devices/<device-id>/command/on>

Аналогічно, кнопки + та – змінюють температуру, формуючи запит з новим значенням, наприклад:

<http://<ip-адреса-контролера>:8083/ZAutomation/api/v1/devices/<device-id>/command/exact?level=23>

де level=23 означає встановлення температури на 23°C. Колірна індикація (червоний – опалення, синій – кондиціонер) забезпечує наочне відображення активних режимів.

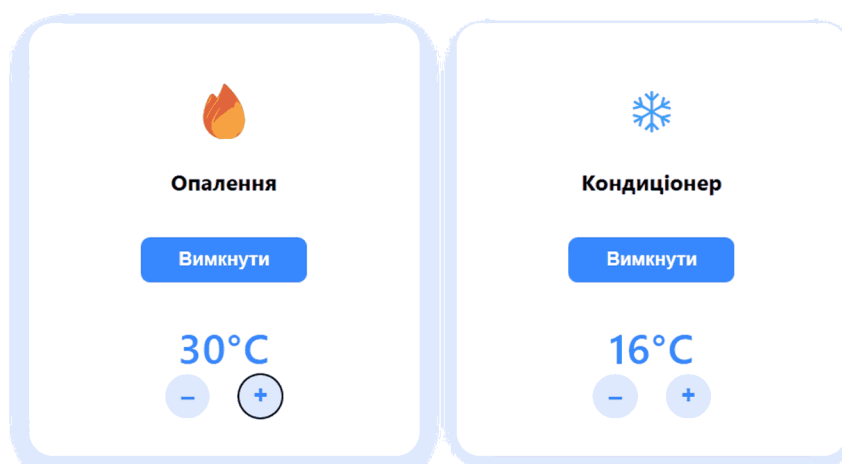


Рисунок 3.15 – Регулювання температури опалення та кондиціонера

У межах інтерфейсу реалізовано окремі інтерактивні картки для дистанційного керування розетками, зокрема для таких побутових приладів як чайник або телевізор(рисунок 3.16). Кожна картка містить кнопку вмикання/вимикання, яка при натисканні ініціює відповідний HTTP-запит до контролера Z-Way. Наприклад, натискання кнопки "увімкнути чайник" формує запит: <http://<ip-адреса-контролера>:8083/ZAutomation/api/v1/devices/<device-id>/command/on>



де <device-id> – це унікальний ідентифікатор пристрою, зареєстрованого у мережі Z-Wave. У випадку вимкнення розетки формується аналогічний запит з параметром off.

Відповідь контролера містить поточний стан пристрою, який миттєво відображається в інтерфейсі через зміну кольору індикатора (наприклад, зелений для увімкненого стану, сірий – для вимкненого). Це забезпечує зворотній зв'язок у реальному часі та надає користувачу наочну інформацію про активність побутових приладів у системі “Розумний будинок”.

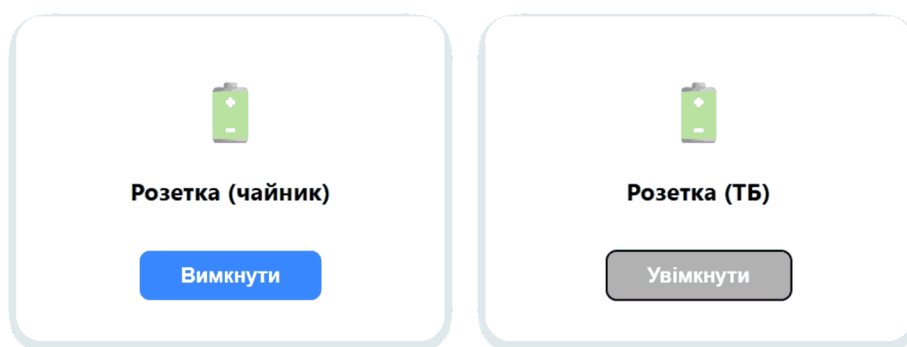


Рисунок 3.16 - Керування побутовими приладами через розумні розетки

Додатковою функціональністю розробленого інтерфейсу є підтримка сценаріїв автоматизації – логічно структурованих послідовностей дій, що виконуються при настанні певної події або за ініціативою користувача. Одним із реалізованих сценаріїв є "Нічний режим", який активується натисканням відповідної кнопки у вебінтерфейсі (рисунок 3.17). При цьому генерується HTTP-запит до контролера на виконання попередньо збереженого сценарію: <http://<ip-адреса-контролера>:8083/ZAutomation/api/v1/scenes/<scene-id>/run>

Після активації сцени виконуються команди на вимкнення всіх світильників (command/off), розеток, а також надсилаються відповідні параметри до пристроїв клімат-контролю – наприклад, зниження температури до економного значення або переведення кондиціонера у сплячий режим. Увесь процес проходить автоматично, без необхідності вручного втручання, а зміни одразу відображаються в інтерфейсі за допомогою кольорових індикаторів і зміни стану елементів керування.

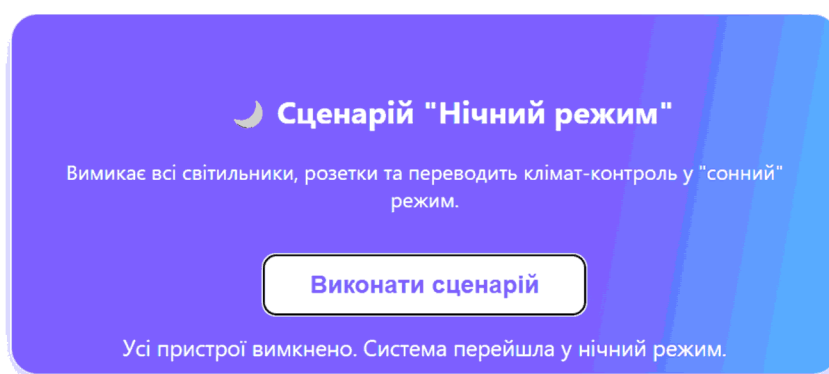


Рисунок 3.17 – Сценарій автоматизації «Нічний режим».

Інтерфейс є адаптивним: завдяки використанню адаптивної верстки (HTML5, CSS3), він коректно масштабується під розміри екранів як настільних комп'ютерів, так і мобільних пристроїв (планшети, смартфони), що забезпечує повноцінну функціональність незалежно від платформи доступу.

Код реалізації вебінтерфейсу розміщено у **Додатку А**. Він написаний з використанням HTML5, CSS3 та JavaScript і охоплює основну логіку керування інтерфейсом системи “розумний будинок”.

Інтерфейс розроблявся як **односторінковий вебдодаток** (single-page application), що складається з одного HTML-файлу з вбудованими стилями та скриптами. Такий підхід був обраний для забезпечення простоти структури та швидкого тестування без додаткових залежностей або серверної логіки.

У структурі HTML-файлу виокремлено три основні блоки:

1. <div id="lighting"> – керування освітленням.
2. <div id="climate"> – керування температурою.
3. <div id="sockets"> – управління розетками.

Кожен блок містить окремі елементи керування (кнопки, повзунки, індикатори стану), оформлені стилями CSS для зручності сприйняття.

JavaScript використано для:

- реакції на події користувача (натискання кнопок, зміна положення повзунка);
- зміни DOM-елементів (оновлення текстових значень, стилів);

- імітації реального стану пристроїв (наприклад, зміна кольору іконки при ввімкненні освітлення).

Ключові функції, реалізовані у кодї:

- `toggleLight()` – перемикає стан лампи між "увімкнено" і "вимкнено", змінюючи зовнішній вигляд кнопки та супровідного тексту.
- `setBrightness(value)` – відповідає за встановлення рівня яскравості за допомогою повзунка; значення відображається у вигляді відсотків.
- `increaseTemperature()` та `decreaseTemperature()` – інкремент/декремент температури на 1°C при натисканні кнопок "+" і "-"; поточне значення відображається у відповідному полі.
- `toggleSocket(id)` – універсальна функція для перемикання стану розетки (використовується для чайника, телевізора тощо).
- `activateNightMode()` – сценарій «Нічний режим», який однією дією вимикає освітлення, знижує температуру до умовного значення (наприклад, 18°C), відключає живлення побутових приладів.

Робота вебінтерфейсу перевірялась у сучасних браузерях (Google Chrome, Firefox). Код написаний із можливістю розширення: наприклад, замість симуляції стану можна підключити REST API Z-Way і керувати реальними пристроями через HTTP-запити.

У разі підключення до реального контролера Z-Way вебінтерфейс виконує HTTP-запити до його REST API для керування фізичними пристроями. Коли користувач взаємодіє з елементами інтерфейсу, JavaScript-код формує відповідний запит на основі ідентифікатора пристрою або сцени. Цей запит передається на адресу контролера і обробляється вбудованим сервером Z-Way. Після обробки команда транслюється у форматі протоколу Z-Wave і надсилається відповідному фізичному пристрою. Пристрій виконує команду, змінює свій стан та передає зворотний сигнал до контролера. Отримані дані використовуються для оновлення інтерфейсу в режимі реального часу.

Таким чином, розроблений вебінтерфейс повністю забезпечує взаємодію користувача з системою «розумний будинок». Він дозволяє керувати пристроями,

переглядати стан системи в реальному часі, а також реалізовувати автоматизовані сценарії.

### 3.5. Тестування системи та результати

Після завершення розгортання демонстраційної версії системи управління «Розумним будинком» на базі Z-Way було проведено тестування її основних функціональних модулів. Тестування відбувалося в умовах симуляції реального середовища за допомогою віртуальних пристроїв, що дозволило перевірити логіку взаємодії між елементами системи, сценаріями автоматизації та користувацьким інтерфейсом.

Було протестовано такі функції:

- Керування освітленням: перевірено можливість вмикання, вимикання та регулювання яскравості світла у віртуальній вітальні та спальні. Реакція системи на зміну параметрів була стабільною, з моментальним оновленням значень у інтерфейсі.
- Регулювання мікроклімату: успішно реалізовано окреме управління опаленням та кондиціонером із можливістю встановлення температури вручну. Значення температури синхронізуються між повзунком керування та температурним відображенням.
- Керування побутовими приладами: протестовано керування віртуальними розетками, зокрема для умовного «чайника» та телевізора. Вмикання й вимикання працювали коректно.
- Сценарій автоматизації: створено сценарій «Нічний режим», що вимикає всі пристрої та переводить клімат-контроль у «сонний» режим. Сценарій спрацював без збоїв при кожному виклику.

Щоб переконатися в тому, що закладена логіка системи працює коректно було змодельовано кілька умовних ситуацій:

- Перед запуском сценарію «Нічний режим» усі пристрої були вручну активовані: ввімкнено світло у вітальні та спальні, активовано розетки,

температура опалення – 22°C, кондиціонер – 26°C. Після натискання кнопки «Нічний режим» в інтерфейсі, система автоматично виконала запрограмовану логіку:

- освітлення вимкнено у двох кімнатах;
- побутові розетки (чайник, телевізор) вимкнені;
- температура опалення зменшена до 18°C;
- температура кондиціонера виставлена на 23°C.

Всі дії супроводжувалися візуальними оновленнями в інтерфейсі та записами у журналі подій (Log Viewer), де чітко відображалися мітки часу, ID пристрою, тип події та її результат. Таким чином підтверджено, що сценарій працює не як фіксована команда, а як узгоджена серія дій залежно від поточного стану системи.

- Тестування зворотного зв'язку: зміна стану пристроїв у сценаріях миттєво відображалася в інтерфейсі без потреби ручного оновлення або перезавантаження сторінки. Це свідчить про коректну роботу WebSocket-каналу та правильну реалізацію зворотної передачі даних.

- Поведінка системи у випадку повторного запуску сценарію: перевірено, що повторне натискання на кнопку «Нічний режим» не викликало помилок, а лише фіксувалося в журналі як повторна команда без зміни станів пристроїв, які вже перебували у потрібному режимі. Це підтверджує ідемпотентність сценарію, тобто його стійкість до повторного виконання без негативних наслідків.

Таке тестування дозволило оцінити не лише функціональність окремих дій, а й узгодженість логіки в цілому – з урахуванням змін середовища, умов виконання, часу доби та команд користувача.

## РОЗДІЛ 4. ОХОРОНА ПРАЦІ

### 4.1. Аналіз стану виробничої санітарії і гігієни праці

У процесі розробки та впровадження інтелектуальної системи керування «Розумний будинок» основні роботи виконуються у приміщенні з офісними або лабораторними умовами. Основними видами діяльності є розробка програмного забезпечення, тестування інтерфейсів керування, робота з документацією, налаштування системи управління через Z-Wave контролери та, за потреби, збирання й підключення електронних компонентів. Хоча фізичні навантаження у процесі роботи мінімальні, важливо враховувати низку чинників, що впливають на санітарно-гігієнічні умови праці та безпеку персоналу.

Виконання робіт відбувається за персональними комп'ютерами, у закритих приміщеннях із постійним перебуванням працівників. Основні шкідливі фактори включають:

- електромагнітне випромінювання моніторів та периферійних пристроїв;
- тривале статичне сидіння, що впливає на опорно-руховий апарат;
- зорове навантаження, пов'язане з тривалою роботою з екранами;
- електричне обладнання, що потребує дотримання електробезпеки;
- низька рухова активність, що сприяє розвитку професійних захворювань при тривалій експлуатації.

Відповідно до ДСН 3.3.6.042-99, в приміщеннях повинні бути забезпечені такі умови:

- температура повітря: від 20 °С до 24 °С;
- відносна вологість: від 40 % до 60 %;
- рівень загальної освітленості: не менше 300 лк;
- рівень шуму: до 50 дБ у робочій зоні;
- яскравість моніторів та контрастність – відповідно до вимог ДСТУ ГОСТ 12.1.019-2009;

Забезпечення належного мікроклімату та освітлення здійснюється за допомогою систем вентиляції, природного та комбінованого освітлення. Для захисту зору рекомендовано використання фільтрів екранів або режимів зниження яскравості.

Для забезпечення ефективної праці персоналу слід впроваджувати:

- ергономічне облаштування робочого місця – регульовані столи та крісла, підставки для рук і моніторів;
- перерви кожні 1–2 години для відпочинку очей і зміни положення тіла;
- використання засобів індивідуального захисту під час роботи з живленням або модулями керування
- обов'язковий інструктаж з охорони праці та пожежної безпеки;
- журнали реєстрації інструктажів, відповідальність за ведення яких покладається на керівника проєкту або адміністратора лабораторії;

своєчасне провітрювання приміщень та контроль за мікрокліматом.

Тривалість робочого часу повинна відповідати нормам трудового законодавства – не більше 40 годин на тиждень. Рекомендовано організувати гнучкий графік або плаваючий робочий день, що враховує ментальне та зорове навантаження. Важливо дотримуватися балансу між розумовою активністю та фізичною мобільністю, включаючи короткі фізичні вправи або зарядку під час перерв.

Умови праці під час розробки системи «Розумний будинок» характеризуються як помірно шкідливі в аспекті гігієни праці, але при правильній організації робочого місця та дотриманні санітарних норм ризику можуть бути мінімізовані до прийняттого рівня. Основну увагу слід приділити ергономіці, освітленню, електробезпеці, а також регулярному моніторингу факторів середовища.

## 4.2. Обґрунтування організаційно-технічних рекомендацій з охорони праці

У межах реалізації системи «Розумний будинок» важливо не лише забезпечити функціональність програмного забезпечення та апаратної частини, але й створити безпечні умови праці для осіб, які залучені до проєктування, тестування та монтажу системи. Заходи щодо поліпшення виробничої санітарії та гігієни праці мають на меті зменшення впливу шкідливих і небезпечних факторів середовища, що можуть виникати у процесі виконання робіт [38; 39].

Зокрема, рекомендовано впровадження ергономічного облаштування робочих місць, застосування регламентованих перерв при тривалій роботі за комп'ютером, використання сучасного комп'ютерного обладнання з мінімальним рівнем випромінювання, контроль за освітленістю та температурним режимом згідно з нормами ДСН 3.3.6.042-99 [40]. У роботі з електронними компонентами (контролерами, реле, сенсорами) обов'язковим є використання заземлення, антистатичних засобів індивідуального захисту та дотримання Правил безпечної експлуатації електроустановок споживачів [38].

До організаційно-технічних заходів належить також обов'язкове проведення інструктажів з охорони праці, фіксація їх у відповідних журналах, підготовка персоналу до можливих надзвичайних ситуацій – зокрема пожеж, ураження електричним струмом, перегріву обладнання. Відповідно до вимог пожежної безпеки [40], приміщення має бути оснащене вогнегасниками, засобами пожежогасіння та вентиляційною системою, а шляхи евакуації – позначені відповідними знаками.

Оцінка ефективності впроваджених заходів ґрунтується на технічних, соціальних та економічних критеріях. Технічні зміни забезпечують відповідність умов праці встановленим стандартам – зменшення рівнів шуму, освітленість, температурний режим, мікроклімат тощо. Соціальні переваги проявляються у зниженні рівня професійного вигорання, зменшенні психоемоційного навантаження, підвищенні задоволеності працею, зниженні плинності кадрів [39]. Економічні результати заходів полягають у зменшенні втрат, пов'язаних з



аваріями, виходом обладнання з ладу, лікарняними листками, витратами на компенсації та простої.

Комплексне впровадження організаційно-технічних заходів дозволяє забезпечити не лише відповідність умов праці чинному законодавству [42; 43], а й підвищити ефективність реалізації проєкту завдяки створенню безпечного та комфортного середовища для персоналу.

### **4.3. Пожежна безпека**

У приміщеннях, де здійснюється розробка, тестування або часткове впровадження елементів системи «Розумний будинок», важливим фактором безпеки є дотримання вимог пожежної безпеки. Система протипожежного захисту розглядається як сукупність організаційних заходів і технічних засобів, спрямованих на запобігання виникненню пожеж, обмеження їх поширення, захист життя і здоров'я людей, а також зменшення матеріальних збитків [40].

Приміщення, що використовується для виконання робіт, належить до категорії з помірним пожежним навантаженням, оскільки у ньому розміщено офісну комп'ютерну техніку, периферійне обладнання, джерела живлення та монтажні пристрої. За класифікацією Правил пожежної безпеки в Україні [41], таке середовище може бути віднесене до категорії В або Г – залежно від об'єму електрообладнання та горючих матеріалів у приміщенні.

Вимоги до безпеки передбачають наявність вогнестійких конструкцій, забезпечення проходів і евакуаційних шляхів шириною не менше 1 м, обладнання приміщень первинними засобами пожежогасіння (порошковими або вуглекислотними вогнегасниками, наприклад, типу ВП-5 або ВВК-2), розміщеними у доступних місцях згідно з нормами [40]. Усі електричні прилади повинні бути справними, відповідати умовам безпечної експлуатації та мати захисне заземлення. Необхідно контролювати навантаження на електромережу, виключити можливість короткого замикання та перегріву електронних модулів.

Рекомендовано встановлення автономного пожежного сповіщувача з акустичним сигналом або централізованої пожежної сигналізації, залежно від призначення приміщення. При підключенні систем Z-Wave або інших інтелектуальних пристроїв необхідно забезпечити їх автоматичне відключення або перемикання в аварійний режим у разі пожежної небезпеки. Усі працівники повинні бути ознайомлені з правилами протипожежної безпеки та пройти відповідний інструктаж.

Організаційні та технічні заходи повинні враховувати вимоги чинних нормативних документів, зокрема Правил пожежної безпеки в Україні [43], а також відповідних стандартів щодо утримання електроустановок та облаштування систем автоматизації [42].

Таким чином, дотримання вимог пожежної безпеки в рамках проєкту «Розумний будинок» є необхідною умовою забезпечення безперервності роботи, захисту обладнання та створення безпечного середовища для персоналу.

## ВИСНОВКИ

У межах виконання бакалаврської кваліфікаційної роботи було розглянуто актуальну науково-прикладну проблему – створення системи автоматизації житлового простору із застосуванням технологій «Розумного будинку». Було досягнуто головної мети дослідження – розробка, моделювання та тестування демонстраційної версії Smart Home-системи на основі протоколу Z-Wave та платформи Z-Way. Поставлені завдання виконано у повному обсязі.

У теоретичному розділі проведено глибокий аналіз сучасних технологій автоматизації житла, розглянуто архітектуру найпоширеніших протоколів зв'язку (Z-Wave, Zigbee, Wi-Fi), а також проведено порівняльний аналіз існуючих програмних платформ управління (Home Assistant, OpenHAB, Z-Way). Особлива увага приділена протоколу Z-Wave, що характеризується високою надійністю, енергоефективністю та міжсумісністю пристроїв різних виробників.

Практична частина роботи реалізована шляхом побудови демонстраційної моделі системи керування без використання фізичного контролера, із застосуванням віртуальних пристроїв у середовищі Z-Way. Створено функціональну модель освітлення, кліматичного контролю, розеток та реалізовано автоматизований сценарій «Нічний режим». За допомогою JavaScript-кодування реалізовано логіку обробки подій, глобальні змінні та взаємозв'язки між пристроями, що повністю імітує роботу реальної системи.

Крім цього, створено власний вебінтерфейс для керування системою, який відображає базові функції Smart Home: регулювання яскравості світла, зміна температури, запуск сценаріїв. Незважаючи на те, що вебдодаток наразі не взаємодіє з фізичними пристроями, реалізована архітектура цілком готова до розширення та інтеграції через API Z-Way у разі наявності відповідного контролера.

Отримані результати демонструють гнучкість та функціональну повноту платформи Z-Way, її придатність для реалізації як простих, так і складних сценаріїв автоматизації у побуті. Проект має високу прикладну цінність – як з точки зору

розгортання в реальних умовах (за наявності обладнання), так і як основа для навчання, досліджень або подальшої модернізації. Розроблена структура системи дозволяє легко інтегрувати додаткові пристрої, розширювати сценарії та підтримує міжплатформену взаємодію.

У перспективі подальші дослідження можуть бути спрямовані на:

- підключення до реального контролера Z-Wave та тестування з фізичними пристроями;
- інтеграцію з іншими протоколами (Zigbee, Wi-Fi, Bluetooth, Matter);
- реалізацію голосового керування через Alexa, Google Assistant або Siri;
- створення повноцінного мобільного додатку для зручності користувача;
- впровадження алгоритмів машинного навчання для інтелектуального аналізу поведінки мешканців і автоматичної адаптації системи до їхніх потреб.

Таким чином, мета та завдання, поставлені у межах роботи, повністю реалізовані. Робота має як теоретичну, так і практичну значущість, а також демонструє високий рівень підготовки до подальшої професійної діяльності у сфері розробки інтелектуальних систем керування.

## БІБЛОГРАФІЧНИЙ СПИСОК

1. Розумний дім [Електронний ресурс] // Вікіпедія : вільна енциклопедія. – Режим доступу: [https://uk.wikipedia.org/wiki/%D0%A0%D0%BE%D0%B7%D1%83%D0%BC%D0%BD%D0%B8%D0%B9\\_%D0%B4%D1%96%D0%BC](https://uk.wikipedia.org/wiki/%D0%A0%D0%BE%D0%B7%D1%83%D0%BC%D0%BD%D0%B8%D0%B9_%D0%B4%D1%96%D0%BC) – Назва з екрана. (Дата звернення 01.04.2025р.)
2. The Evolution of the Smart Home: How It Started – Part 1 [Електронний ресурс] // Ubuntu Blog. – Режим доступу: <https://ubuntu.com/blog/the-evolution-of-the-smart-home-how-it-started-part-1> – Назва з екрана. (Дата звернення 01.04.2025р.)
3. Tcaflisch, C. The Evolution of Smart Home Technology [Електронний ресурс] // Medium. – Режим доступу: <https://tcaflisch.medium.com/the-evolution-of-smart-home-technology-c3b3f918fc8c> – Назва з екрана. (Дата звернення 05.04.2025р.)
4. Smart devices pose cyber security risk [Електронний ресурс] // The Times. – Режим доступу: <https://www.thetimes.co.uk/article/smart-devices-pose-cyber-security-risk-kl0zhzlff> – Назва з екрана. (Дата звернення 07.04.2025р.)
5. Yadav, V., & Singh, D. (2020). Internet of Things (IoT): A Review of Enabled Technologies and Future Challenges [Електронний ресурс] // International Journal of Technology and Engineering Sciences. – Vol. 1, No. 2. – Режим доступу: <https://www.arpweb.com/journal/7/archive/01-2020/1/2> – Назва з екрана. (Дата звернення 07.04.2025р.)
6. Що таке розумний дім: функції, види, складові та екосистеми [Електронний ресурс] // ЕК.ua. – 2022. – Режим доступу: <https://ek.ua/ua/post/1990/618-chto-takoe-umnyy-dom-funkcii-vidy-sostavlyayuschie-i-ekosistemy/> – Назва з екрана.
7. Smart home connectivity: Zigbee, Z-Wave or Wi-Fi? [Електрон. ресурс] // TechTarget. – Режим доступу: <https://www.techtarget.com/iotagenda/feature/Smart-home-connectivity-Zigbee-Z-Wave-or-Wi-Fi> – Назва з екрана. (Дата звернення 07.04.2025р.)
8. Smart Home | Google Assistant [Електрон. ресурс] // Google Developers. – Режим доступу: <https://developers.google.com/assistant/smarthome/overview> (Дата звернення 10.04.2025р.)

9. Digital Demand-Driven Electricity Services [Електрон. ресурс] // International Energy Agency. – Режим доступу: <https://www.iea.org/reports/digital-demand-driven-electricity-services> – Назва з екрана. (Дата звернення 10.04.2025р.)
10. Elshrkawey M., Iqbal M. A., Abdelsalam M., Mohamed A., Elshrkawey M. E. Intelligent Smart Home Automation System Based on Internet of Things [Електрон. ресурс] // IEEE Xplore. – Режим доступу: <https://ieeexplore.ieee.org/document/9246763> – Назва з екрана. (Дата звернення 15.04.2025р.)
11. Patel K., Patel S., Nagineni S. M., Pathan M. K. A Survey on Smart Home System Technologies [Електрон. ресурс] // IEEE Xplore. – Режим доступу: <https://ieeexplore.ieee.org/document/8568781> – Назва з екрана. (Дата звернення 10.04.2025р.)
12. Zigbee for Consumers [Електрон. ресурс] // Zigbee Alliance. – Режим доступу: [https://zigbeealliance.org/zigbee\\_for\\_consumers/](https://zigbeealliance.org/zigbee_for_consumers/) – Назва з екрана. (Дата звернення 16.04.2025р.)
13. Learn about Z-Wave [Електрон. ресурс] // Z-Wave Alliance. – Режим доступу: <https://z-wavealliance.org/learn-about-z-wave/> – Назва з екрана. (Дата звернення 18.04.2025р.)
14. Smart Home Solutions [Електрон. ресурс] // Cisco. – Режим доступу: <https://www.cisco.com/c/en/us/solutions/internet-of-things/smart-home.html> – Назва з екрана. (Дата звернення 20.04.2025р.)
15. Matter – The Foundation for Connected Things [Електрон. ресурс] // Connectivity Standards Alliance. – Режим доступу: <https://csa-iot.org/all-solutions/matter/> – Назва з екрана. (Дата звернення 19.04.2025р.)
16. Smart Home – Worldwide Market Value [Електрон. ресурс] // Statista. – Режим доступу: <https://www.statista.com/statistics/682204/worldwide-smart-home-market-value/> – Назва з екрана. (Дата звернення 10.05.2025р.)
17. Home Assistant – Open Source Home Automation [Електрон. ресурс] // Home Assistant. – Режим доступу: <https://www.home-assistant.io/> – Назва з екрана. (Дата звернення 30.04.2025р.)

18. openHAB Documentation [Електрон. ресурс] // openHAB. – Режим доступу: <https://www.openhab.org/docs/> – Назва з екрана. (Дата звернення 30.04.2025р.)
19. Z-Way – Smart Home Controller Software [Електрон. ресурс] // Z-Wave.Me. – Режим доступу: <https://z-wave.me/z-way/> – Назва з екрана. (Дата звернення 25.04.2025р.)
20. Best Smart Home Platforms [Електрон. ресурс] // SmartHomeStart. – Режим доступу: <https://smarthomestart.com/best-smart-home-platforms/> – Назва з екрана. (Дата звернення 19.04.2025р.)
21. Z-Wave Technology Overview [Електрон. ресурс] // Silicon Labs. – Режим доступу: <https://www.silabs.com/wireless/z-wave> – Назва з екрана. (Дата звернення 03.05.2025р.)
22. Z-Wave Alliance [Електрон. ресурс] // Z-Wave Alliance. – Режим доступу: <https://z-wavealliance.org/> – Назва з екрана. (Дата звернення 05.05.2025р.)
23. Z-Wave White Paper [Електрон. ресурс] // Silicon Labs. – Режим доступу: <https://www.silabs.com/documents/public/white-papers/z-wave-white-paper.pdf> – Назва з екрана. (Дата звернення 30.04.2025р.)
24. Z-Wave Protocol for Smart Homes [Електрон. ресурс] // DigiKey. – Режим доступу: <https://www.digikey.com/en/articles/z-wave-protocol-for-smart-homes> – Назва з екрана. (Дата звернення 27.04.2025р.)
25. Z-Wave S2 Security Overview [Електрон. ресурс] // Silicon Labs. – Режим доступу: <https://www.silabs.com/documents/public/application-notes/AN0331-Z-Wave-S2-Security.pdf> – Назва з екрана. (Дата звернення 10.04.2025р.)
26. Z-Wave Certified Product Catalog [Електрон. ресурс] // Z-Wave Alliance. – Режим доступу: <https://products.z-wavealliance.org/> – Назва з екрана. (Дата звернення 20.04.2025р.)
27. Zigbee vs Z-Wave: What's the Difference? [Електрон. ресурс] // MakeUseOf. – Режим доступу: <https://www.makeuseof.com/tag/zigbee-vs-zwave/> – Назва з екрана. (Дата звернення 15.04.2025р.)
28. Z-Way Developer Documentation [Електрон. ресурс] // Z-Wave.Me. – Режим доступу: <https://developer.z-wave.me> – Назва з екрана. (Дата звернення

- 01.05.2025р.)
29. Z-Wave JS Integration [Електрон. ресурс] // Home Assistant. – Режим доступу: [https://www.home-assistant.io/integrations/zwave\\_js/](https://www.home-assistant.io/integrations/zwave_js/) – Назва з екрана. (Дата звернення 21.04.2025р.)
  30. openHAB Documentation [Електрон. ресурс] // openHAB. – Режим доступу: <https://www.openhab.org/docs/> – Назва з екрана. (Дата звернення 20.04.2025р.)
  31. Порівняння Zigbee та Z-Wave. Що обрати? [Електрон. ресурс] // xterm. – Режим доступу: <https://xterm.com.ua/novosti/porivniannia-zigbee-ta-z-wave-shcho-obrati> – Назва з екрана [homesmart.com.ua](https://www.homesmart.com.ua)+[xterm.com.ua](https://www.xterm.com.ua)+[smarty.ninja](https://www.smarty.ninja)+4. (Дата звернення 27.04.2025р.)
  32. IEEE-огляд: ZigBee vs Z-Wave [Електрон. ресурс] // Smarty Ninja. – Режим доступу: <https://www.smarty.ninja/zigbee/zigbee-vs-z-wave/> – Назва з екрана [kspu.edu](https://www.kspu.edu)+14[smarty.ninja](https://www.smarty.ninja)+14[xterm.com.ua](https://www.xterm.com.ua)+14. (Дата звернення 29.04.2025р.)
  33. Порівняння Zigbee, Z-Wave і WiFi: що найкраще? [Електрон. ресурс] // MyLikeLed. – Режим доступу: <https://mylikeled.com/uk/porivnannia-zigbee-z-wave-ios-wifi%2C-%D1%8F%D0%BA%D0%B8%D0%B9-%D0%BD%D0%B0%D0%B9%D0%BA%D1%80%D0%B0%D1%89%D0%B8%D0%B9/> – Назва з екрана [mylikeled.com](https://www.mylikeled.com). (Дата звернення 30.04.2025р.)
  34. Z-Wave, Zigbee, HomeKit: особливості протоколів [Електрон. ресурс] // NeoSmart. – Режим доступу: <https://neosmart.com.ua/> – Назва з екрана [its.kpi.ua](https://www.its.kpi.ua)+9[neosmart.com.ua](https://www.neosmart.com.ua)+9[homesmart.com.ua](https://www.homesmart.com.ua)+9. (Дата звернення 01.05.2025р.)
  35. Як облаштувати «розумний будинок» – огляд сетапів айтівців [Електрон. ресурс] // DOU. – Режим доступу: <https://dou.ua/lenta/articles/overview-of-smart-home/> – Назва з екрана [dou.ua](https://www.dou.ua). (Дата звернення 03.05.2025р.)
  36. Протоколи розумного дому [Електрон. ресурс] // HomeSmart. – Режим доступу: <https://homesmart.com.ua/protokoly-umnoho-doma/> – Назва з екрана [homesmart.com.ua](https://www.homesmart.com.ua). (Дата звернення 07.05.2025р.)
  37. Розробка інформаційної системи управління електроспоживанням у розумному будинку [Електрон. ресурс] // «Комп'ютерно-інтегровані технології». –



Режим доступу: <https://cit.lntu.edu.ua/index.php/cit/article/view/225/314> – Назва з екрана. (Дата звернення 05.05.2025р.)

38. Тимочко В. О., Городецький І. М., Березовецький А. П., Мазур І. Б. та ін. *Безпека життєдіяльності та охорона праці: навч. посібник*. Львів: Сполом, 2022. 376 с.

39. Пістун І. П., Березовецький А. П., Тимочко В. О., Городецький І. М. *Охорона праці (гігієна праці та виробнича санітарія): навч. посіб. / за ред. І. П. Пістуна*. Львів: Тріада плюс, 2017. Ч. I. 620 с.

40. ДСН 3.3.6.042-99. *Санітарні норми мікроклімату виробничих приміщень*. URL: <https://zakon.rada.gov.ua/rada/show/va042282-99#Text> – Назва з екрана (Дата звернення 20.05.2025р.)

41. *Правила пожежної безпеки в Україні (ред. 22.01.2022 р.)*. URL: <https://zakon.rada.gov.ua/laws/show/z0252-15#Text> – Назва з екрана. (Дата звернення 20.05.2025р.)

42. Закон України “Про охорону праці”. URL: <http://zakon2.rada.gov.ua/laws/show/2694-12> – Назва з екрана. (Дата звернення 20.05.2025р.)

# ДОДАТКИ

**Додаток А:**

```

<!DOCTYPE html>
<html lang="uk">
<head>
  <meta charset="UTF-8">
  <title>Система розумного будинку (демо)</title>
  <style>
    /* Загальні стилі для сторінки */
    body {
      font-family: 'Segoe UI', Arial, sans-serif; /* Встановлення основного шрифту
*/
      background: #f5f6fa; /* Світлий фон для приємного вигляду */
    }
    /* Заголовок */
    h2 {
      margin-left: 40px; /* Відступ зліва для заголовку */
      margin-top: 40px; /* Відступ зверху */
    }
    /* Контейнер для блоків пристроїв */
    .container {
      display: flex; /* Встановлення flex-контейнера для розміщення карток */
      flex-wrap: wrap; /* Дозволяє переносити картки на наступний рядок */
      justify-content: center; /* Центрування карток по горизонталі */
      gap: 32px; /* Відстань між картками */
      margin-top: 40px; /* Відступ зверху */
    }
    /* Картка одного пристрою */
    .card {

```

```

background: #fff; /* Білий фон для картки */
border-radius: 20px; /* Заокруглені кути */
box-shadow: 0 4px 16px #abb1e155; /* Тінь для об'єму */
padding: 32px 24px; /* Відступи всередині картки */
width: 285px; /* Фіксована ширина */
text-align: center; /* Центрування тексту */
transition: box-shadow .2s; /* Плавна анімація тіні при наведенні */
}
.card:hover {
    box-shadow: 0 8px 28px #6d8cff40; /* Підсвітка при наведенні */
}
/* Іконка пристрою */
.icon {
    font-size: 48px; /* Великий розмір іконки */
    margin-bottom: 8px; /* Відступ знизу */
    margin-top: 8px; /* Відступ зверху */
}
/* Класи для кольору іконок в залежності від стану */
.lamp.on { color: #ffe066; filter: drop-shadow(0 2px 8px #ffe06688);} /* Світло
увімкнено */
.lamp.off { color: #b1b1b1;} /* Світло вимкнено */
.temp { color: #3887fe;} /* Колір для температури */
.heating.on { color: #e8594a;} /* Опалення увімкнено */
.heating.off { color: #b1b1b1;} /* Опалення вимкнено */
.cooling.on { color: #43a2f9;} /* Кондиціонер увімкнено */
.cooling.off { color: #b1b1b1;} /* Кондиціонер вимкнено */
.socket.on { color: #67e86b;} /* Розетка увімкнена */
.socket.off { color: #b1b1b1;} /* Розетка вимкнена */
/* Стили для кнопок */
.btn {

```

```

padding: 10px 32px; /* Відступи всередині кнопки */
font-size: 1rem; /* Розмір тексту */
border: none; /* Без обводки */
border-radius: 8px; /* Заокруглення */
background: #3887fe; /* Синій фон */
color: #fff; /* Білий текст */
font-weight: bold; /* Жирний шрифт */
margin-top: 16px; /* Відступ зверху */
cursor: pointer; /* Курсор у вигляді руки */
box-shadow: 0 2px 8px #b2c3ff44; /* Тінь для кнопки */
transition: background 0.2s, transform 0.08s; /* Плавна анімація */
}
.btn:active { transform: scale(0.97); } /* Зменшення кнопки при натисканні */
.btn.off { background: #b1b1b1; color: #fff; } /* Сірий фон для відключених
кнопок */
/* Стилі для повзунка яскравості */
.slider {
width: 90%; /* Ширина повзунка */
margin-top: 12px; /* Відступ зверху */
}
.slider::-webkit-slider-thumb { background: #ffe066; } /* Колір ручки повзунка */
.slider-value {
font-weight: 500; /* Напівжирний шрифт */
color: #8a8a8a; /* Сірий текст */
font-size: 1rem; /* Розмір шрифту */
}
/* Стилі для кнопок регулювання температури */
.temp-controls button {
margin: 0 10px; /* Відступи по боках */
background: #e0eaff; /* Світло-блакитний фон */

```

```

color: #3887fe; /* Синій текст */
font-size: 1.3rem; /* Розмір тексту */
border-radius: 50%; /* Круглі кнопки */
width: 38px; height: 38px; /* Розмір кнопок */
border: none; /* Без обводки */
font-weight: bold; /* Жирний текст */
box-shadow: 0 1px 6px #b2c3ff22; /* Тінь */
cursor: pointer; /* Курсор-рука */
transition: background .2s; /* Анімація зміни фону */
}
.temp-controls button:active {
    background: #c4d6ff; /* Темніший фон при натисканні */
}
.temp-val {
    display: inline-block; /* Відображення блоку в лінію */
    margin: 14px 0 0 0; /* Відступ зверху */
    font-size: 2.2rem; /* Великий розмір тексту */
    font-weight: 600; /* Напівжирний */
    color: #3887fe; /* Синій */
    letter-spacing: 1px; /* Проміжок між літерами */
}
/* Стили для сценарію "Нічний режим" */
.scenario {
    background: linear-gradient(100deg, #7d5fff 70%, #4ebfff 100%); /*
Градiєнтний фон */
    color: #fff; /* Білий текст */
    border-radius: 20px; /* Заокруглення */
    box-shadow: 0 4px 20px #b2c3ff77; /* Тінь */
    margin-top: 16px; /* Відступ зверху */
    padding: 20px 24px 10px 24px; /* Відступи */

```

```

width: 600px; /* Фіксована ширина */
text-align: center; /* Центрування тексту */
}
.scenario-btn {
background: #fff; /* Білий фон */
color: #7d5fff; /* Фіолетовий текст */
font-size: 1.2rem; /* Розмір шрифту */
font-weight: 600; /* Напівжирний */
padding: 12px 36px; /* Відступи */
border: none; /* Без обводки */
border-radius: 10px; /* Заокруглення */
margin-top: 18px; /* Відступ зверху */
cursor: pointer; /* Курсор-рука */
box-shadow: 0 2px 14px #4ebfff33; /* Тінь */
transition: background 0.16s, color 0.16s; /* Анімація */
}
.scenario-btn:active {
background: #e6ebff; /* Світлий фон при натисканні */
color: #4ebfff; /* Синій текст */
}
</style>
</head>
<body>
<h2>Інтерфейс системи «Розумний будинок» (демо)</h2>
<div class="container">
<!-- Картка: Світло у вітальні -->
<div class="card">
<!-- Іконка лампи з початковим станом "вимкнено" -->
<div class="icon lamp off" id="livingLampIcon">&#128161;</div>
<h3>Світло у вітальні</h3>

```

```

<!-- Кнопка увімкнення/вимкнення світла -->
<button          class="btn          off"          id="livingLightBtn"
onclick="toggleLamp('living')">Увімкнути</button>
<div style="margin-top:14px;">
  <!-- Повзунок для диммування (регулювання яскравості) -->
  <input type="range" min="0" max="100" value="70" class="slider"
id="livingDimmer" oninput="updateDimmer('living')" disabled>
  <!-- Текстовий індикатор поточної яскравості -->
  <div class="slider-value" id="livingDimmerValue">Яскравість: 70%</div>
</div>
</div>
<!-- Картка: Світло в спальні -->
<div class="card">
  <div class="icon lamp off" id="bedroomLampIcon">&#128161;</div>
  <h3>Світло в спальні</h3>
  <button          class="btn          off"          id="bedroomLightBtn"
onclick="toggleLamp('bedroom')">Увімкнути</button>
</div>
<!-- Картка: Опалення -->
<div class="card">
  <div class="icon heating off" id="heatingIcon">&#128293;</div>
  <h3>Опалення</h3>
  <button          class="btn          off"          id="heatingBtn"
onclick="toggleHeating()">Увімкнути</button>
<div style="margin-top:18px;">
  <!-- Показник поточної температури -->
  <span class="temp-val" id="heatingTempValue">21°C</span>
  <!-- Кнопки для регулювання температури -->
  <div class="temp-controls">
    <button onclick="changeTemp('heating',-1)">-</button>

```



```

        <button onclick="changeTemp('heating',1)">+</button>
    </div>
</div>
</div>
<!-- Картка: Кондиціонер -->
<div class="card">
    <div class="icon cooling off" id="coolingIcon">#10052;</div>
    <h3>Кондиціонер</h3>
    <button class="btn off" id="coolingBtn"
onclick="toggleCooling()">Увімкнути</button>
    <div style="margin-top:18px;">
        <span class="temp-val" id="coolingTempValue">23°C</span>
        <div class="temp-controls">
            <button onclick="changeTemp('cooling',-1)">-</button>
            <button onclick="changeTemp('cooling',1)">+</button>
        </div>
    </div>
</div>
</div>
<!-- Картка: Розетка для чайника -->
<div class="card">
    <div class="icon socket off" id="kettleSocketIcon">#128267;</div>
    <h3>Розетка (чайник)</h3>
    <button class="btn off" id="kettleSocketBtn"
onclick="toggleSocket('kettle')">Увімкнути</button>
</div>
<!-- Картка: Розетка для телевізора -->
<div class="card">
    <div class="icon socket off" id="tvSocketIcon">#128267;</div>
    <h3>Розетка (ТБ)</h3>
    <button class="btn off" id="tvSocketBtn"

```

```

onclick="toggleSocket('tv')">Увімкнути</button>
  </div>
</div>
<!-- Сценарій "Нічний режим" -->
<div class="scenario" style="margin: 48px auto;">
  <h2> Сценарій "Нічний режим"</h2>
  <p>Вимикає всі світильники, розетки та переводить клімат-контроль у
"сонний" режим.</p>
  <button      class="scenario-btn"      onclick="nightMode()">Виконати
сценарій</button>
  <div id="scenarioMsg" style="margin-top:14px; font-size:1.08rem;"></div>
</div>
<script>
  /* Функція для перемикання стану лампи в заданій кімнаті */
  function toggleLamp(room) {
    // Визначаємо елементи кнопки, іконки та повзунок за ID залежно від
кімнати
    const btn = document.getElementById(room + "LightBtn");
    const icon = document.getElementById(room + "LampIcon");
    const dimmer = document.getElementById(room + "Dimmer");

    if (btn.classList.contains("off")) {
      // Якщо лампа вимкнена, вмикаємо
      btn.classList.remove("off");
      btn.textContent = "Вимкнути";
      icon.classList.remove("off");
      icon.classList.add("on");
      // Увімкнути повзунок диммера, якщо є
      if (dimmer) dimmer.disabled = false;
      // Тут можна додати код для надсилання команди на контролер Z-Way

```

```

} else {
    // Якщо лампа увімкнена, вимикаємо
    btn.classList.add("off");
    btn.textContent = "Увімкнути";
    icon.classList.remove("on");
    icon.classList.add("off");
    if (dimmer) dimmer.disabled = true;
    // Тут можна додати код для надсилання команди на контролер Z-Way
}
}

/* Функція для оновлення значення диммера та текстового індикатора */
function updateDimmer(room) {
    const dimmer = document.getElementById(room + "Dimmer");
    const valueText = document.getElementById(room + "DimmerValue");
    valueText.textContent = "Яскравість: " + dimmer.value + "%";
    // Тут можна додати код для надсилання рівня яскравості на контролер Z-
Way
}

/* Функція для перемикання опалення */
function toggleHeating() {
    const btn = document.getElementById("heatingBtn");
    const icon = document.getElementById("heatingIcon");
    if (btn.classList.contains("off")) {
        btn.classList.remove("off");
        btn.textContent = "Вимкнути";
        icon.classList.remove("off");
        icon.classList.add("on");
        // Додати код для увімкнення опалення через контролер

```

```
} else {  
    btn.classList.add("off");  
    btn.textContent = "Увімкнути";  
    icon.classList.remove("on");  
    icon.classList.add("off");  
    // Додати код для вимкнення опалення через контролер  
}  
}  
  
/* Функція для перемикавання кондиціонера */  
function toggleCooling() {  
    const btn = document.getElementById("coolingBtn");  
    const icon = document.getElementById("coolingIcon");  
    if (btn.classList.contains("off")) {  
        btn.classList.remove("off");  
        btn.textContent = "Вимкнути";  
        icon.classList.remove("off");  
        icon.classList.add("on");  
        // Додати код для увімкнення кондиціонера через контролер  
    } else {  
        btn.classList.add("off");  
        btn.textContent = "Увімкнути";  
        icon.classList.remove("on");  
        icon.classList.add("off");  
        // Додати код для вимкнення кондиціонера через контролер  
    }  
}  
  
/* Функція для зміни температури (плюс або мінус) */  
function changeTemp(device, delta) {
```

```

const tempValue = document.getElementById(device + "TempValue");
let currentTemp = parseInt(tempValue.textContent);
currentTemp += delta;
// Обмеження, наприклад від 10 до 30 градусів
if (currentTemp < 10) currentTemp = 10;
if (currentTemp > 30) currentTemp = 30;
tempValue.textContent = currentTemp + "°C";
// Тут можна додати код для надсилання нового значення температури на
контролер
    }

/* Функція для перемикання стану розетки */
function toggleSocket(type) {
    const btn = document.getElementById(type + "SocketBtn");
    const icon = document.getElementById(type + "SocketIcon");
    if (btn.classList.contains("off")) {
        btn.classList.remove("off");
        btn.textContent = "Вимкнути";
        icon.classList.remove("off");
        icon.classList.add("on");
        // Додати код для увімкнення розетки через контролер
    } else {
        btn.classList.add("off");
        btn.textContent = "Увімкнути";
        icon.classList.remove("on");
        icon.classList.add("off");
        // Додати код для вимкнення розетки через контролер
    }
}
}

```

```
/* Функція для виконання сценарію "Нічний режим" */  
function nightMode() {  
    const scenarioMsg = document.getElementById("scenarioMsg");  
    scenarioMsg.textContent = "Виконуємо сценарій...";  
    // Тут можна додати код для запуску сценарію через контролер  
  
    // Симуляція затримки виконання  
    setTimeout(() => {  
        scenarioMsg.textContent = "Сценарій 'Нічний режим' виконано";  
        // Оновити стан елементів інтерфейсу відповідно  
        // Наприклад, вимкнути всі лампи, розетки, знизити температуру і т.д.  
    }, 1500);  
}  
</script>  
</body>  
</html>  
Додаток Б:  
if (%% < 15) {  
    global.heatingTemp = 15;  
} else if (%% > 30) {  
    global.heatingTemp = 30;  
} else {  
    global.heatingTemp = %%;  
}
```

## Додаток Б

// Вимкнення освітлення

zway.devices[2].instances[0].commandClasses[37].Set(0); // Світло у вітальні

zway.devices[3].instances[0].commandClasses[37].Set(0); // Світло у спальні

// Вимкнення розеток

zway.devices[4].instances[0].commandClasses[37].Set(0); // Розетка (чайник)

zway.devices[5].instances[0].commandClasses[37].Set(0); // Розетка (ТБ)

// Встановлення температури кондиціонера на 23 °C

global.acTemp = 23;

// Встановлення температури опалення на 18 °C

global.heatingTemp18; ЧЧ

## Додаток В

Таблиця 2.2 Пристрої сумісні з Z-Wave

Характеристика	Z-Wave	Home Assistant	OpenHAB	Domoticz
Основний протокол	Z-Wave	Усі популярні (Zigbee, Z-Wave, Wi-Fi тощо)	Багато протоколів (через Binding'и)	Основні (Z-Wave, RF, Wi-Fi)
Підтримка Z-Wave	Глибока, повний контроль	Добра (через Z-Wave JS)	Добра (через Z-Wave Binding)	Обмежена
Мобільний застосунок	Є, базовий функціонал	Є, розширений функціонал	Є, середній функціонал	Є, базовий
Рівень складності налаштування	Низький	Середній	Високий	Низький
Безпека	Підтримка S2, API токени	Залежить від користувача	Висока, залежно від налаштувань	Обмежені можливості
Гнучкість налаштувань	Середня	Висока	Висока	Середня
Інтеграція з іншими системами	Через API, MQTT	Велика кількість інтеграцій	Через binding'и	Обмежена