НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ПРОГРАМНИХ СИСТЕМ ТА ПРИКЛАДНОЇ МАТЕМАТИКИ

Кафедра системного програмування і спеціалізованих комп'ютерних систем

До захисту допущено:

Завідувач кафедри

_____ Віталій РОМАНКЕВИЧ

«___»_____2026 р.

Дипломний проєкт

на здобуття ступеня бакалавра

за освітньо-професійною програмою

«Системне програмування і спеціалізовані комп'ютерні систем»

спеціальності 123 «Комп'ютерна інженерія»

на тему: «Симулятор системи автоматизації розумного будинку з вебінтерфейсом»

Виконав:
студент IV курсу, групи КВ-22
Лисенко Віталій Олександрович _____

Керівник:
доцент каф. СПіСКС, к.т.н., доцент
Щербина Олександр Андрійович _____

Консультант з нормконтролю:
доцент каф. СПіСКС, к.т.н., доцент
Клятченко Ярослав Михайлович _____

Рецензент:
доцент каф. ОТ ФІОТ, к.т.н., доцент
Долголенко Олександр Миколайович _____

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

Київ – 2026 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ПРОГРАМНИХ СИСТЕМ ТА ПРИКЛАДНОЇ МАТЕМАТИКИ

Кафедра системного програмування і спеціалізованих комп'ютерних систем

Рівень вищої освіти – перший (бакалаврський)
Освітньо-професійна програма
«Системне програмування та спеціалізовані
комп'ютерні системи»
Спеціальність 123 «Комп'ютерна інженерія»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Віталій РОМАНКЕВИЧ
(підпис)
«___» _____2026 р.

ЗАВДАННЯ
на дипломний проєкт студента
Лисенко Віталія Олександровича

1. Тема проєкту «Симулятор системи автоматизації розумного будинку з вебінтерфейсом», керівник проєкту доц.каф. СПіСКС, к.т.н., доцент Щербина Олександр Андрійович, затверджені наказом по університету від «23» травня 2026 р. №1984-с

2. Термін подання студентом проєкту «10» червня 2026 р.

3. Вихідні дані до проєкту див. Технічне завдання

4. Зміст пояснювальної записки:

- Аналіз предметної області та постановка задачі;
- Проєктування системи автоматизації розумного будинку;
- Розроблення програмного забезпечення;
- Тестування, аналіз та оцінка роботи системи.

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслеників, плакатів, презентацій тощо):

- Структурна схема функціональних модулів симулятора;

– Структурна схема вебінтерфейсу;
– Схема алгоритму охоронної системи;
– Схема алгоритму клімат-контролю.

6. Консультанти розділів проєкту

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	Доцент кафедри СПіСКС, к.т.н., Клятченко Ярослав Михайлович		

7. Дата видачі завдання 15 грудня 2026 р.

Календарний план

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
1.	Вивчення літератури за тематикою проєкту	10.04.2026	
2.	Розроблення та узгодження технічного завдання	12.04.2026	
3.	Аналіз існуючих рішень	23.04.2026	
4.	Підготовка матеріалів першого розділу проєкту	01.05.2026	
5.	Підготовка матеріалів другого розділу проєкту	08.05.2026	
6.	Розробка програмного забезпечення проєкту	15.05.2026	
7.	Підготовка матеріалів графічної частини проєкту	22.05.2026	
8.	Оформлення документації дипломного проєкту	01.06.2026	
9.	Попередній огляд матеріалів на кафедрі	03.06.2026	

Студент _____ Віталій ЛИСЕНКО
(підпис) (ініціали, прізвище)

Керівник проєкту _____ Олександр ЩЕРБИНА
(підпис) (ініціали, прізвище)

АНОТАЦІЯ

Кваліфікаційна робота складається з пояснювальної записки (67 с., 15 рис., 5 таблиці, , 13 джерел).

Об'єктом дослідження та розробки є системи автоматизації розумного дому та програмні засоби моделювання їх функціонування у віртуальному середовищі.

Метою роботи є проєктування, розробка та тестування програмного симулятора системи автоматизації розумного будинку з вебінтерфейсом користувача, який забезпечує моделювання роботи датчиків, керування виконавчими пристроями та реалізацію алгоритмів автоматизації.

У процесі виконання роботи досліджено принципи побудови сучасних систем класу Smart Home, проаналізовано підходи до автоматизації житлових приміщень та особливості використання вебтехнологій для керування пристроями. На основі проведеного аналізу розроблено структуру програмного комплексу, що включає підсистеми моніторингу, кліматичного контролю, освітлення, безпеки та обліку енергоспоживання.

Створений програмний комплекс забезпечує симуляцію роботи датчиків температури, вологості та руху, автоматичне керування кондиціонером і обігрівачем відповідно до заданих параметрів середовища, підтримує роботу режимів Home, Night та Away, а також ведення журналу подій. Для взаємодії користувача із системою реалізовано вебінтерфейс із можливістю керування обладнанням та відображення поточного стану будинку в режимі реального часу.

У ході тестування підтверджено працездатність програмного забезпечення, коректність реалізації алгоритмів автоматизації та стабільність роботи вебінтерфейсу. Отримані результати показали можливість використання створеного симулятора як навчального засобу для дослідження принципів функціонування систем розумного дому та як основи для подальшого створення реальних програмно-апаратних комплексів автоматизації.

Ключові слова: розумний будинок, автоматизація, симулятор системи, вебінтерфейс користувача, моніторинг, база даних, клімат-контроль, енергоспоживання.

ABSTRACT

The qualification work consists of an explanatory note (67 p., 15 fig., 5 tables, 13 references).

The object of research and development is Smart Home automation systems and software tools for simulating their operation in a virtual environment..

The purpose of the work is to design, develop and test a software simulator of a Smart Home automation system with a web-based user interface that provides sensor simulation, device control and implementation of automation algorithms.

During the research, the principles of modern Smart Home systems construction, approaches to residential automation and the use of web technologies for device management were analyzed. Based on the conducted analysis, a software architecture was developed that includes monitoring, climate control, lighting, security and energy consumption management subsystems

The developed software system provides simulation of temperature, humidity and motion sensors, automatic control of air conditioning and heating devices according to environmental parameters, support for Home, Night and Away operating modes, as well as event logging. A web-based user interface was implemented to control devices and display the current state of the house in real time.

Testing confirmed the correct operation of the software, the reliability of the implemented automation algorithms and the stability of the web interface. The developed simulator can be used as an educational tool for studying Smart Home technologies and as a basis for the development of real automation systems.

Keywords: Smart Home, automation, system simulator, web interface, monitoring, database, climate control, energy consumption.

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
	А4	ІАЛЦ.467200.002 ТЗ	Симулятор системи	2	А4	
			автоматизації розумного			
			будинку з вебінтерфейсом			
			Технічне завдання			
	А4	ІАЛЦ.467200.003 ТП	Симулятор системи	1	А4	
			автоматизації розумного			
			будинку з вебінтерфейсом			
			Відомість технічного			
			проекту			
	А4	ІАЛЦ.467200.004 ПЗ	Симулятор системи	60	А4	
			автоматизації розумного			
			будинку з вебінтерфейсом			
			Пояснювальна записка			
	А4	ІАЛЦ.467200.005 Д1	Функціональні модулі	1	А4	
			симулятора			
			Схема структурна			
	А4	ІАЛЦ.467200.006 Д2	Структура вебінтерфейсу	1	А4	
			Схема структурна			

Змін	Арк.	№ докум.	Підпис	Дата	ІАЛЦ.467200.001 ОА		
Розробив		Лисенко В.О.			Симулятор системи автоматизації розумного будинку з вебінтерфейсом **Опис альбому**	Літ.	Аркуш 1 / Аркушів 2
Перевірив		Щербина О.А.					
Консульт.						КПІ ім. Ігоря Сікорського, ФПСПМ, КВ-22	
Н.контроль		Клятченко Я.М.					
Зав. каф.		Романкевич В.О					

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
	А4	ІАЛЦ.467200.007 Д3	Алгоритм охоронної	1	А4	
			системи			
			Схема алгоритму			
	А4	ІАЛЦ.467200.008 Д4	Алгоритм клімат-контролю	1	А4	
			Схема алгоритму			
		Диск CD-ROM	Текст пояснювальної			
			записки.			
			Графічний матеріал			

ЗМІСТ

НАЙМЕНУВАННЯ ТА ГАЛУЗЬ РОЗРОБКИ

Назва розробки: «Симулятор системи автоматизації розумного будинку з вебінтерфейсом».

Галузь застосування охоплює сферу комп'ютерної інженерії, технологій Інтернету речей (IoT), автоматизації будівель (Smart Home / Smart Building) та розробки кросплатформних розподілених вебсистем. Розробка фокусується на створенні інтерактивного середовища для моделювання кіберфізичних просторів, тестуванні логіки автоматизації (сценаріїв взаємодії пристроїв) та побудові динамічних інтерфейсів користувача для моніторингу й диспетчеризації в реальному часі.

ПІДСТАВА ДЛЯ РОЗРОБКИ

Підставою для розробки є завдання на дипломне проектування на здобуття першого (бакалаврського) рівня вищої освіти, затверджене кафедрою системного програмування і спеціалізованих комп’ютерних систем Національного технічного університету України «Київський Політехнічний Інститут імені Ігоря Сікорського».

МЕТА І ПРИЗНАЧЕННЯ РОБОТИ

Метою проєкту є розробка архітектури, алгоритмічного забезпечення та програмного комплексу для інтерактивної симуляції «розумного будинку». Система має моделювати поведінку датчиків і виконавчих механізмів, забезпечувати асинхронну обробку подій за заданими користувачем правилами (сценаріями) та надавати гнучкі інструменти візуалізації стану приміщень через вебінтерфейс.

Призначенням системи є створення безпечного, масштабованого та апаратного-незалежного середовища для налагодження й тестування алгоритмів автоматизації

житлових просторів, наочної демонстрації концепцій Інтернету речей (IoT) без використання фізичного обладнання, забезпечення користувача централізованим, інтуїтивно зрозумілим інструментом моніторингу мікроклімату, безпеки та енергоспоживання об'єкта в реальному часі.

ДЖЕРЕЛА РОЗРОБКИ

Джерелом інформації є технічна та науково-технічна література в галузі архітектури кіберфізичних систем та IoT, стандарти та протоколи побудови систем розумного дому (MQTT, WebSockets, HTTP), офіційна документація до сучасних вебфреймворків (React, Django/FastAPI, Node.js), публікації у періодичних виданнях, матеріали профільних міжнародних конференцій та електронні ресурси у мережі Інтернет.

ТЕХНІЧНІ ВИМОГИ

Вимоги до програмного продукту:

інтерактивний вебінтерфейс, тобто наявність Single Page Application (SPA) для візуального відображення структури дому (кімнат), поточного стану віртуальних пристроїв (увімкнено/вимкнено, показники сенсорів) та віджетів керування актуаторами;

серверна частина буде асинхронний модуль для генерації подій від сенсорів (динамічна зміна температури, освітленості, фіксація руху) та обробки вхідних команд користувача;

модуль автоматизації (двигун сценаріїв): підсистема обробки логічних правил (тригерів), що забезпечує автоматичне реагування системи на зміну параметрів навколишнього середовища;

зв'язок у реальному часі: реалізація двостороннього обміну даними між клієнтом та сервером (через технологію WebSockets або аналоги) для миттєвого оновлення стану інтерфейсу при спрацюванні датчиків;

адаптивність інтерфейсу: коректне відображення, масштабування та доступність усіх функцій вебпанелі на мобільних пристроях (смартфонах, планшетах) та десктопних моніторах.

Вимоги до апаратного забезпечення

Персональний комп’ютер або ноутбук із процесором не нижче Intel Core i3, AMD Ryzen 3 або аналогічним;
Оперативна пам’ять обсягом не менше 4 ГБ;
Вільне місце на накопичувачі не менше 500 МБ для розміщення програмного забезпечення, бази даних та службових файлів;
Монітор із роздільною здатністю не нижче 1366×768 пікселів для коректного відображення вебінтерфейсу користувача.

Вимоги до програмного та апаратного забезпечення користувача

Встановлений пакетний менеджер pip для інсталяції необхідних бібліотек;
Веббраузер Google Chrome, Mozilla Firefox, Microsoft Edge, Opera або інший сучасний браузер із підтримкою HTML5, CSS3 та JavaScript;
Бібліотека SQLite для зберігання журналу подій та службової інформації;
Доступ до локальної мережі або мережі Internet для взаємодії з вебінтерфейсом через браузер.

ЕТАПИ РОЗРОБКИ

№ з/п	Назва етапів виконання дипломного проекту	Термін виконання етапів
1.	Вивчення літератури за тематикою проекту	03.05.2026
2.	Розроблення та узгодження технічного завдання	07.05.2026
3.	Аналіз існуючих рішень	10.05.2026
4.	Підготовка матеріалів першого розділу дипломного проекту	16.05.2026
5.	Підготовка матеріалів другого розділу дипломного проекту	18.05.2026
6.	Підготовка графічної частини дипломного проекту	21.05.2026
7.	Оформлення документації дипломного проекту	24.05.2026
8.	Попередній огляд матеріалів диплому на кафедрі	27.05.2026

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
	А4	ІАЛЦ.467200.004 ПЗ	Симулятор системи	60	А4	
			автоматизації розумного			
			будинку з вебінтерфейсом			
			Пояснювальна записка			
	А4	ІАЛЦ.467200.005 Д1	Функціональні модулі	1	А4	
			симулятора			
			Схема структурна			
	А4	ІАЛЦ.467200.006 Д2	Структура вебінтерфейсу	1	А4	
			Схема структурна			
	А4	ІАЛЦ.467200.007 Д3	Алгоритм охоронної	1	А4	
			системи			
			Схема алгоритму			
	А4	ІАЛЦ.467200.008 Д4	Алгоритм клімат-контролю	1	А4	
			Схема алгоритму			
		Диск CD-ROM	Текст пояснювальної			
			записки.			
			Графічний матеріал			

Змін	Арк.	№ докум.	Підпис	Дата	ІАЛЦ.467200.003 ТП		
Розробив		Лисенко В.О.			Симулятор системи автоматизації розумного будинку з вебінтерфейсом **Відомість технічного проєкту**	Літ.	Аркуш 1 / Аркушів 1
Перевірив		Щербина О.А.					
Консульт.						КПІ ім. Ігоря Сікорського, ФПСПМ, КВ-22	
Н.контроль		Клятченко Я.М.					
Зав. каф.		Романкевич В.О					

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ

АС – автоматизована система, сукупність програмних та технічних засобів для виконання функцій керування та моніторингу.

БД – база даних, структуроване сховище інформації про події та стан компонентів системи.

ВІ – вебінтерфейс, програмний інтерфейс взаємодії користувача із системою через браузер.

ЖП – журнал подій, підсистема реєстрації дій користувача та змін стану обладнання.

КС – комп’ютерна система, програмно-апаратний комплекс для виконання функцій автоматизації.

ПЗ – програмне забезпечення, сукупність програмних компонентів симулятора.

СКБД – система керування базами даних, програмне забезпечення для роботи з базою даних.

СРБ – система розумного будинку, комплекс засобів автоматизації житлового приміщення.

ТП – температурний показник, параметр середовища, що використовується алгоритмами клімат-контролю.

API – Application Programming Interface (інтерфейс прикладного програмування), набір засобів взаємодії між програмними компонентами.

HTTP – HyperText Transfer Protocol, протокол обміну даними між клієнтом і сервером.

IoT – Internet of Things (інтернет речей), концепція мережі взаємопов’язаних пристроїв, що здійснюють обмін даними.

SQL – Structured Query Language (мова структурованих запитів), використовується для роботи з базами даних.

URL – Uniform Resource Locator (уніфікований покажчик ресурсу), адреса вебресурсу в мережі.

UI – User Interface (інтерфейс користувача), сукупність елементів взаємодії користувача із системою.

ВСТУП

Сучасний етап розвитку інформаційних технологій характеризується активним впровадженням засобів автоматизації у повсякденне життя людини. Зростання обчислювальних можливостей цифрових пристроїв, розвиток мережевих технологій та поширення концепції взаємодії пристроїв у межах єдиного інформаційного середовища сприяли появі нових підходів до організації житлового простору. Одним із найбільш перспективних напрямів є автоматизація житлових приміщень, що реалізується через системи розумного будинку, здатні забезпечувати керування освітленням, кліматичними параметрами, побутовими пристроями та елементами безпеки.

Система автоматизації розумного будинку являє собою комплекс технічних та програмних компонентів, взаємодія яких забезпечує виконання визначених сценаріїв роботи без постійного втручання користувача. Такі системи дозволяють контролювати стан обладнання, змінювати параметри функціонування пристроїв, здійснювати дистанційне керування та реалізовувати механізми адаптації до умов навколишнього середовища. Наприклад, автоматичне регулювання освітлення залежно від часу доби, підтримка комфортної температури або керування електроприладами за попередньо визначеним розкладом підвищують рівень комфорту та сприяють ефективнішому використанню енергетичних ресурсів.

Водночас практичне впровадження та валідація алгоритмів автоматизації часто лімітуються потребою у спеціалізованій елементній базі, датчиках, мікроконтролерах і телекомунікаційному інструментарії. Даний фактор створює додаткові бар'єри для всебічного вивчення принципів побудови таких систем під час навчання або концептуального проєктування. У цьому контексті особливої ваги набуває створення програмних симуляційних платформ. Їхнє функціональне призначення полягає у детермінованому відтворенні робочих циклів систем розумного будинку в ізольованому програмному просторі, аналізі взаємозв'язків між підсистемами та верифікації апаратно-незалежних рішень.

Застосування методів імітаційного моделювання дозволяє суттєво оптимізувати етапи проєктування, верифікації та налагодження алгоритмів автоматизованого керування. В ізольованому програмному середовищі уможливлюється апробація різноманітних сценаріїв функціонування інженерних підсистем, зокрема інтелектуального освітлення, комплексів безпеки, кліматичного контролю та побутової апаратури. Це дозволяє аналізувати відгук системи на флуктуації вхідних параметрів та модернізувати керуючу логіку. Додатковою архітектурною перевагою є інтеграція користувацького вебінтерфейсу, який реалізує централізоване адміністрування емулятора через браузер і підвищує ергономічність людино-машинної взаємодії.

Створення веборієнтованого симулятора автоматизації житлового середовища дозволяє не лише моделювати роботу окремих підсистем, але й досліджувати принципи організації взаємодії між користувачем і системою керування. Завдяки використанню сучасних вебтехнологій з'являється можливість реалізувати зручні механізми зміни режимів роботи, перегляду поточного стану пристроїв та запуску автоматизованих сценаріїв. Це робить дослідження систем розумного будинку більш доступним і дозволяє виконувати моделювання без значних матеріальних витрат.

Актуальність дослідження

Поширення систем автоматизації житлових приміщень зумовлює необхідність розробки ефективних засобів дослідження, проєктування та тестування логіки їх функціонування. У сучасних умовах користувачі дедалі частіше використовують інтелектуальні системи керування освітленням, кліматом, системами безпеки та побутовими пристроями, що потребує гнучких і доступних рішень для моделювання їхньої роботи.

Однією з практичних проблем під час вивчення або створення систем розумного будинку є складність організації повноцінного тестового середовища

із застосуванням реального обладнання. Використання фізичних контролерів, датчиків і виконавчих механізмів потребує додаткових фінансових витрат, часу на налаштування та забезпечення сумісності компонентів.

У зв’язку з цим виникає потреба у створенні програмного середовища, яке дозволить імітувати функціонування систем автоматизації без залучення реальних пристроїв.

Актуальність роботи також визначається зростанням ролі вебтехнологій у керуванні технічними системами. Реалізація вебінтерфейсу забезпечує можливість централізованого доступу до елементів керування, візуалізації стану пристроїв і взаємодії із симулятором через браузер. Це сприяє підвищенню зручності використання системи та розширює можливості її практичного застосування для навчальних і дослідницьких цілей.

Мета та завдання

Метою дипломної роботи є розробка симулятора системи автоматизації розумного будинку з вебінтерфейсом користувача, який забезпечує моделювання роботи основних підсистем житлової автоматизації, реалізацію механізмів керування пристроями та підтримку сценаріїв автоматизованої взаємодії. Для досягнення поставленої мети необхідно виконати аналіз принципів функціонування систем розумного будинку та сучасних технологій автоматизації, дослідити особливості організації взаємодії між компонентами smart home-систем, провести аналіз існуючих програмних рішень у сфері автоматизації житла, сформувати вимоги до симулятора, спроєктувати структуру програмної системи, реалізувати вебінтерфейс користувача та виконати тестування функціонування створеного програмного засобу.

Об’єкт та предмет дослідження

Об’єктом дослідження є процеси автоматизації житлового середовища та механізми взаємодії між компонентами систем розумного будинку Предметом

дослідження є методи моделювання роботи систем автоматизації розумного будинку, програмні механізми керування пристроями, а також вебтехнології реалізації інтерфейсу користувача.

Наукова та практична значущість

Наукова значущість полягає у дослідженні принципів функціонування систем автоматизації житлового середовища, аналізі підходів до моделювання поведінки пристроїв та реалізації сценаріїв взаємодії між компонентами розумного будинку у програмному середовищі. Практичне значення роботи полягає у створенні симулятора системи автоматизації розумного будинку з вебінтерфейсом користувача, який може використовуватися для демонстрації принципів роботи smart home-систем, тестування алгоритмів керування, моделювання сценаріїв автоматизації та навчальних цілей без необхідності використання реального обладнання.

1. АНАЛІЗ ПРЕДМЕНТОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1. Поняття та принципи функціонування систем розумного дому

У сучасних умовах стрімкого розвитку інформаційних технологій значна увага приділяється автоматизації повсякденних процесів, що спрямована на підвищення рівня комфорту, безпеки та ефективності використання ресурсів. Одним із важливих напрямів у цій сфері є створення та впровадження систем розумного будинку. Такі системи дозволяють інтегрувати різноманітні технічні засоби в єдине інформаційне середовище, у межах якого забезпечується централізоване або автоматизоване керування обладнанням житлового приміщення. Розвиток концепції розумного будинку став наслідком поєднання телекомунікаційних технологій, засобів автоматизації, сенсорних систем та програмного забезпечення, що здатне аналізувати події та приймати рішення на основі заданих алгоритмів.

Поняття «розумний дім» використовується для опису комплексу програмно-апаратних засобів, які забезпечують автоматичне або дистанційне керування різними функціональними підсистемами житлового приміщення [1]. До таких підсистем, як правило, належать освітлення, кліматичне обладнання, системи безпеки, відеоспостереження, електропостачання, мультимедійні пристрої та побутова техніка. Основною особливістю подібних рішень є можливість взаємодії між окремими компонентами, що дозволяє створювати автоматизовані сценарії функціонування відповідно до потреб користувача.

Застосування систем автоматизації житлового середовища спрямоване не лише на забезпечення зручності керування пристроями, а й на оптимізацію використання енергетичних ресурсів. Наприклад, автоматичне вимкнення освітлення у приміщеннях за відсутності людей, регулювання температури залежно від часу доби або погодних умов, а також керування електроприладами відповідно до заданих сценаріїв дають змогу знизити рівень споживання

електроенергії [2]. Таким чином, системи розумного будинку стають інструментом не лише підвищення комфорту, але й економічної ефективності.

Слід зазначити, що концепція автоматизованого житла не є принципово новою. Перші спроби впровадження автоматизованих систем у побуті з’явилися ще в другій половині XX століття, коли почали використовуватись програмовані контролери та елементи електронного керування. Проте обмежені технічні можливості того часу, висока вартість обладнання та відсутність стандартизованих протоколів обміну даними стримували масове впровадження подібних рішень. Суттєвий розвиток цього напряму став можливим із появою бездротових мереж, мікроконтролерів, доступного програмного забезпечення та глобального поширення концепції Інтернету речей [3].

Сучасний розумний будинок функціонує на основі постійної взаємодії між датчиками, виконавчими механізмами, системою обробки даних та користувачем. Основною ланкою такої системи виступає центр керування, який виконує роль координатора роботи окремих елементів. Він може бути реалізований у вигляді локального сервера, мікроконтролера, персонального комп’ютера або хмарної платформи. Саме центр керування приймає сигнали від сенсорів, аналізує поточний стан системи та надсилає команди виконавчим пристроям.

Датчики відіграють ключову роль у забезпеченні функціонування системи розумного будинку, оскільки саме вони дозволяють отримувати інформацію про стан навколишнього середовища або обладнання. У практичних реалізаціях часто використовуються датчики температури, вологості, руху, освітленості, відкриття дверей і вікон, витоку води або газу. Отримані від них дані використовуються для формування рішень системи автоматизації.

1.2. Огляд сучасних технологій автоматизації розумного будинку

Активний розвиток цифрових технологій сприяв появі великої кількості рішень, призначених для автоматизації житлових приміщень. Системи розумного будинку поступово переходять із категорії дорогих спеціалізованих рішень до більш доступних технологій, які можуть використовуватися у квартирах, приватних будинках або офісних приміщеннях [4]. Основною причиною популярності таких систем є можливість автоматизованого керування побутовими процесами, оптимізація використання енергоресурсів та підвищення рівня комфорту для користувачів. Сучасні технології автоматизації дозволяють інтегрувати різні пристрої в єдине середовище, де вони можуть взаємодіяти між собою та виконувати визначені сценарії.

Одним із ключових елементів сучасних систем автоматизації є сенсорні технології. Датчики забезпечують отримання інформації про стан приміщення, поведінку користувача або параметри навколишнього середовища [5]. Залежно від функціонального призначення можуть використовуватися датчики температури, вологості, руху, освітленості, диму, відкриття дверей і вікон, рівня споживання електроенергії та інші. Наприклад, датчики руху часто застосовуються для автоматичного вмикання освітлення, тоді як температурні сенсори дозволяють підтримувати комфортний мікроклімат шляхом автоматичного керування системами опалення або кондиціонування.

Не менш важливу роль у функціонуванні розумного будинку відіграють виконавчі пристрої, які реалізують команди системи керування. До таких пристроїв належать інтелектуальні вимикачі, електромеханічні реле, сервоприводи, смарт-розетки, системи автоматичного відкривання вікон або штор, а також регулятори освітлення та температури. Саме ці компоненти забезпечують фізичну взаємодію між програмною частиною системи та реальним середовищем, виконуючи необхідні дії відповідно до заданих алгоритмів.

Для забезпечення взаємодії між пристроями використовуються різноманітні технології передачі даних. У сучасних системах найбільш поширеними є бездротові мережі, серед яких особливу популярність мають Wi-Fi, Bluetooth, Zigbee та Z-Wave [6]. Технологія Wi-Fi забезпечує високу швидкість передачі даних та зручність інтеграції з локальними мережами й вебсервісами. Водночас Bluetooth часто використовується для локальної взаємодії між пристроями на невеликих відстанях. Zigbee та Z-Wave, логотипи яких зображені на рисунку 1.1, орієнтовані на побудову енергоефективних мереж, що можуть об'єднувати значну кількість сенсорів і виконавчих механізмів у межах єдиної інфраструктури.

Рисунок 1.1 – Логотипи Zigbee та Z-Wave

Значного поширення в автоматизації житлових приміщень набули технології Інтернету речей, які передбачають об'єднання фізичних пристроїв у цифрове середовище. У межах концепції Інтернету речей кожен пристрій здатний передавати інформацію, отримувати команди та взаємодіяти з іншими компонентами системи через мережу [7]. Це дозволяє організувати централізоване керування та віддалений доступ до обладнання. Користувач може здійснювати контроль за роботою системи через вебінтерфейс або

мобільний застосунок, отримувати повідомлення про події та змінювати налаштування навіть за межами дому.

Важливим аспектом сучасних систем є використання програмних платформ керування, які виконують функції координації роботи всіх компонентів [8]. Такі платформи дозволяють створювати сценарії автоматизації, налаштовувати правила взаємодії пристроїв та забезпечувати візуальне представлення інформації. Наприклад, користувач може сформувати правило, за яким у вечірній час автоматично вмикатиметься освітлення, а система клімат-контролю підтримуватиме задану температуру. Наявність графічного інтерфейсу спрощує процес взаємодії з системою та робить її використання більш зрозумілим.

1.3. Аналіз існуючих систем та програмних рішень

Розвиток технологій автоматизації житлового середовища сприяв появі великої кількості програмних платформ і готових рішень для керування компонентами розумного будинку. Сучасні системи відрізняються функціональними можливостями, способами інтеграції обладнання, рівнем складності налаштування та вартістю впровадження. Частина рішень орієнтована на використання в межах окремого житлового приміщення, тоді як інші дозволяють будувати масштабовані системи з великою кількістю взаємопов’язаних пристроїв.

Серед найбільш поширених рішень можна виділити платформи, що підтримують централізоване керування освітленням, системами безпеки, клімат-контролем та електроприладами [9]. Такі системи, як правило, надають користувачеві графічний інтерфейс для перегляду поточного стану обладнання, зміни параметрів і створення сценаріїв автоматизації. Деякі платформи підтримують інтеграцію з голосовими помічниками та мобільними

застосунками, що спрощує процес взаємодії із системою.

Попри значні переваги готових рішень, вони часто мають певні обмеження. До основних недоліків можна віднести високу вартість обладнання, залежність від конкретного виробника, обмежені можливості модифікації та складність тестування алгоритмів без фізичних пристроїв [10]. Саме тому створення симулятора системи автоматизації розумного будинку є доцільним підходом, оскільки дозволяє моделювати роботу компонентів, аналізувати поведінку системи та тестувати сценарії керування без необхідності використання реального обладнання.

Таким чином, аналіз існуючих програмних рішень свідчить про актуальність розробки власного симулятора з вебінтерфейсом користувача, який забезпечуватиме візуалізацію станів пристроїв, зручність керування та можливість перевірки логіки роботи системи автоматизації.

1.4. Постановка задачі та формування вимог до симулятора

Розробка симулятора системи автоматизації розумного будинку потребує чіткого визначення функціональних можливостей майбутньої системи, принципів її взаємодії з користувачем та вимог до програмної реалізації. Постановка задачі є важливим етапом проєктування, оскільки саме на її основі визначається структура програмного забезпечення, логіка функціонування окремих компонентів і способи взаємодії між ними [11]. У межах даної роботи передбачається створення програмного середовища, яке дозволить імітувати роботу елементів розумного будинку та забезпечить керування ними через вебінтерфейс користувача.

Основним призначенням симулятора є відтворення процесів, характерних для систем автоматизації житлового середовища. Це передбачає моделювання роботи пристроїв освітлення, систем кліматичного контролю, елементів безпеки та побутових електроприладів [12]. Користувач повинен мати можливість змінювати стан пристроїв, активувати режими роботи, запускати сценарії автоматизації та переглядати поточний стан системи у реальному часі.

Реалізація таких можливостей дозволяє оцінити принципи функціонування розумного будинку без необхідності використання фізичного обладнання.

Оскільки система орієнтована на керування великою кількістю компонентів, важливим завданням є створення вебінтерфейсу, який забезпечуватиме швидкий доступ до основних функцій, наочне відображення параметрів і логічну структуру навігації. Інтерфейс повинен дозволяти користувачу отримувати інформацію про стан пристроїв, керувати ними через інтерактивні елементи та змінювати налаштування відповідно до потреб.

Крім функціональних вимог, важливу роль відіграють нефункціональні характеристики системи. До них можна віднести простоту використання, стабільність роботи, швидкість реакції інтерфейсу та можливість подальшого розширення функціоналу. Симулятор повинен бути побудований таким чином, щоб у майбутньому можна було додавати нові модулі, сценарії автоматизації або підтримку додаткових типів пристроїв без суттєвої зміни загальної структури системи.

2. ПРОЄКТУВАННЯ СИСТЕМИ АВТОМАТИЗАЦІЇ РОЗУМНОГО БУДИНКУ

2.1. Розробка структури симулятора розумного будинку

Створення програмного засобу для моделювання роботи розумного будинку розпочинається з формування загальної структури системи та визначення взаємодії між її складовими. На етапі проєктування було поставлено завдання створити середовище, яке дозволяє імітувати роботу основних підсистем житлового будинку, забезпечувати керування ними через вебінтерфейс та відображати поточний стан обладнання в режимі реального часу.

На відміну від систем що працюють із фізичними датчиками та контролерами, розроблюваний програмний продукт виконує моделювання процесів програмними засобами. Такий підхід дає можливість перевіряти алгоритми автоматизації без використання додаткового обладнання, а також значно спрощує процес налагодження та тестування функціональних модулів.

Під час розробки структури системи було визначено перелік основних компонентів, які повинні брати участь у роботі симулятора. До них належать модуль симуляції датчиків, модуль керування пристроями, підсистема автоматизації, база даних журналу подій та вебінтерфейс користувача. Кожен із зазначених елементів виконує окремі функції та взаємодіє з іншими компонентами через програмні інтерфейси.

Сервер виконує роль координатора між усіма елементами програмного комплексу. Він обробляє запити користувача, запускає алгоритми симуляції, формує відповіді для вебінтерфейсу та забезпечує збереження інформації про події [13].

Модуль симуляції датчиків призначений для генерації даних, які в реальних умовах надходили б від фізичних пристроїв. У процесі роботи створюються значення температури повітря, відносної вологості та стану датчика руху. Для підвищення достовірності моделі було відмовленося від випадкової генерації незалежних значень. Натомість кожне нове значення формується на основі попереднього стану із невеликим випадковим відхиленням. Завдяки цьому зміни параметрів мають плавний характер і більше відповідають реальним умовам експлуатації житлових приміщень.

Отримані дані передаються до підсистеми автоматизації. Саме вона аналізує поточний стан середовища та визначає необхідність увімкнення або вимкнення окремих пристроїв. Наприклад, при перевищенні встановленого температурного порогу активується кондиціонер, а при зниженні температури нижче допустимого рівня запускається система обігріву. Таке рішення дозволяє відтворити принцип роботи сучасних кліматичних систем без використання спеціалізованого обладнання.

Окремий блок структури становить модуль керування пристроями. Його функцією є підтримка поточного стану освітлення, дверей, сигналізації, кондиціонера та обігрівача. Для кожного пристрою передбачено два основні стани: активний та неактивний. Зміна стану може здійснюватися як користувачем через вебінтерфейс, так і автоматично під впливом алгоритмів керування.

У системі також реалізовано механізм сценаріїв роботи будинку. Сценарій являє собою набір правил, що визначають поведінку декількох пристроїв одночасно. Під час проєктування було передбачено три режими функціонування: *Home, Night та Away*. Режим *Home* відповідає звичайній експлуатації будинку, Night активує охоронні функції та вимикає освітлення, а Away використовується у випадках відсутності мешканців та передбачає

ввімкнення системи сигналізації. У таблиці 2.1 показано основні компоненти програми.

Таблиця 2.1 – Основні компоненти симулятора

Компонент	Призначення
Модуль симуляції	Формування показників датчиків
Підсистема автоматизації	Аналіз параметрів та прийняття рішень
Модуль керування пристроями	Зміна стану обладнання
База даних	Збереження журналу подій
Вебінтерфейс	Взаємодія користувача із системою

Для фіксації результатів роботи було передбачено ведення журналу подій. Кожна дія користувача та кожне автоматичне спрацювання системи реєструються в базі даних. Збереження історії дає змогу відстежувати зміни стану обладнання, аналізувати поведінку системи та використовувати накопичені дані під час тестування.

Структура сховища даних побудована на основі SQLite. Обраний підхід забезпечує простоту розгортання програмного продукту та не потребує встановлення окремого серверного програмного забезпечення. Для задач симулятора продуктивності SQLite достатньо, оскільки кількість одночасних операцій запису є невеликою.

Взаємодія між сервером та клієнтською частиною здійснюється за допомогою HTTP-запитів. Користувач працює через веббраузер, який надсилає команди на сервер та отримує актуальну інформацію про стан будинку. Передача даних виконується у форматі JSON, що забезпечує компактність повідомлень та спрощує обробку інформації засобами JavaScript.

Для реалізації поставлених завдань було сформовано структуру програмного комплексу, яка включає модуль симуляції датчиків, модуль автоматизації, підсистему зберігання даних та вебінтерфейс користувача. Взаємозв'язок основних функціональних компонентів симулятора наведено на рисунку 2.1.

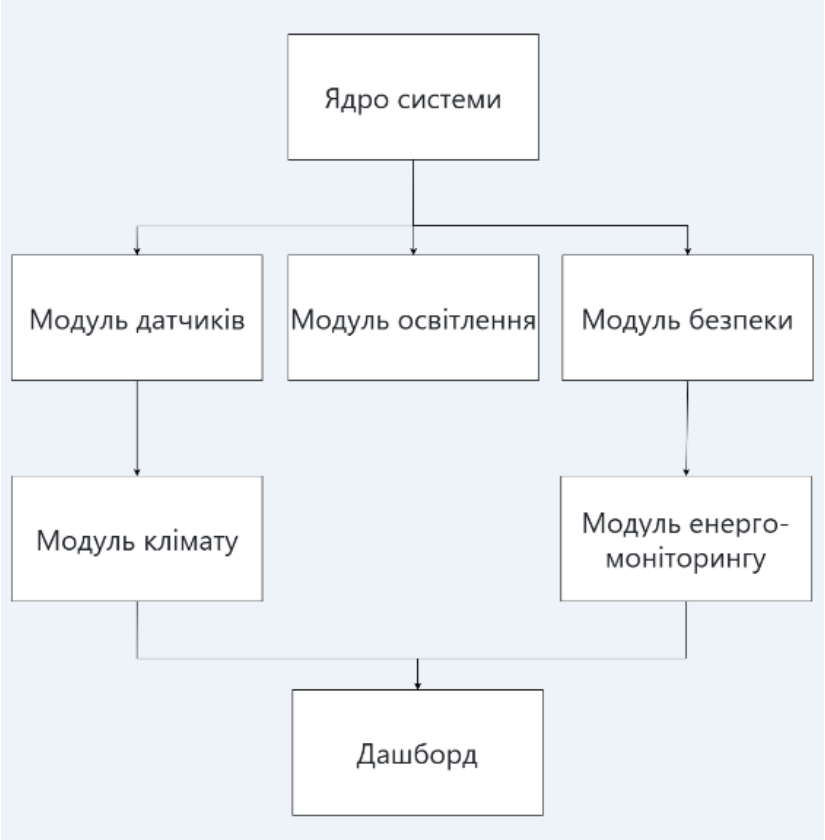


Рисунок 2.1 – Структурна схема функціональних модулів симулятора

Під час проєктування структури особлива увага приділялася можливості подальшого розширення функціональності. Передбачена архітектура дає можливітсь без значних змін додавати нові типи датчиків, виконавчих механізмів та сценаріїв автоматизації. За необхідності система може бути доповнена модулями моніторингу енергоспоживання, відеоспостереження або дистанційного доступу.

Загальна структура симулятора формує єдине програмне середовище, у межах якого реалізуються процеси моделювання, автоматичного керування та взаємодії користувача із системою.

2.2 Моделювання підсистем розумного будинку

Реальна система автоматизації житлового приміщення складається з великої кількості взаємопов'язаних пристроїв, що виконують різні функції. Для забезпечення зрозумілої архітектури та спрощення подальшої розробки було прийнято рішення розділити функціональність симулятора на декілька логічно незалежних підсистем.

Основу моделі складають підсистема освітлення, підсистема кліматичного контролю, підсистема безпеки та підсистема моніторингу стану середовища. Кожна з них виконує окремий набір функцій, але водночас бере участь у спільному процесі автоматизації житлового будинку.

Підсистема освітлення є однією з найбільш поширених складових сучасних систем Smart Home. Її призначення полягає в керуванні джерелами світла в окремих приміщеннях. У межах розробленого симулятора передбачено керування освітленням трьох кімнат: вітальні, кухні та спальні. Для кожного джерела освітлення реалізовано два стани функціонування: увімкнений та вимкнений.

Модель освітлення не обмежується простим перемиканням станів. Кожна дія користувача фіксується в журналі подій, що дозволяє відстежувати історію використання системи. Такий підхід наближає симулятор до принципів функціонування реальних систем автоматизації, де всі зміни конфігурації реєструються для подальшого аналізу.

Особливістю підсистеми освітлення є її інтеграція зі сценаріями роботи будинку. Під час переходу до нічного режиму освітлення автоматично

вимикається незалежно від попереднього стану пристроїв. Подібна поведінка відповідає логіці роботи більшості комерційних рішень у сфері домашньої автоматизації.

Наступною складовою є підсистема кліматичного контролю. Вона забезпечує підтримання комфортних параметрів мікроклімату шляхом аналізу температури навколишнього середовища та автоматичного керування виконавчими пристроями. До складу моделі входять кондиціонер та обігрівач.

На відміну від освітлення, робота кліматичної підсистеми значною мірою залежить від показників датчиків. Температура повітря розглядається як основний параметр, на основі якого приймаються рішення щодо ввімкнення або вимкнення обладнання. Для моделювання процесів регулювання були визначені граничні значення температури.

Якщо температура перевищує встановлений верхній поріг, система автоматично активує кондиціонер. У випадку зниження температури нижче мінімально допустимого значення запускається обігрівач. За нормальних умов обидва пристрої залишаються вимкненими. Такий принцип керування дозволяє підтримувати температуру в заданому діапазоні та відтворює роботу автоматичних систем клімат-контролю.

Підсистема безпеки відповідає за контроль доступу до будинку та реагування на потенційно небезпечні ситуації. У рамках розробленої моделі до її складу включено датчик руху, охоронну сигналізацію та електронний замок дверей.

Моделювання датчика руху здійснюється шляхом генерації подій появи або відсутності руху в зоні контролю. Отримані сигнали використовуються для перевірки роботи охоронних алгоритмів. При цьому особливу увагу приділено взаємодії датчика руху із системою сигналізації.

Якщо режим охорони вимкнений, спрацювання датчика не викликає жодних додаткових дій. У випадку активованої сигналізації виявлення руху розглядається як потенційне порушення безпеки. За таких умов система формує тривожне повідомлення та записує відповідний запис до журналу подій.

Двері в моделі розглядаються як окремий керований об'єкт. Користувач може змінювати їхній стан через вебінтерфейс. Поточний стан дверей відображається у вигляді текстового індикатора. Подібне рішення дає змогу імітувати роботу електронних систем доступу, які широко використовуються в сучасних житлових комплексах.

Завданням підсистеми моніторингу параметрів середовища є формування та передача даних про поточний стан будинку. До набору контрольованих параметрів входять температура повітря, вологість та наявність руху.

Значення температури та вологості не формуються випадково під час кожного запиту. Новий стан розраховується на основі попереднього значення із невеликим випадковим відхиленням. Завдяки цьому вдається уникнути різких стрибків показників та отримати більш реалістичну поведінку системи. Основні функції симулятора показані в таблиці 2.2.

Таблиця 2.2 – Основні функції симулятора розумного будинку

Підсистема	Основні функції
Освітлення	Керування освітленням приміщень
Клімат-контроль	Підтримка комфортної температури
Безпека	Контроль доступу та охорона
Моніторинг	Збір і відображення параметрів середовища

Крім функціональних модулів, у процесі моделювання було передбачено підсистему обліку енергоспоживання. Її призначення полягає у визначенні сумарної потужності активних пристроїв. Для кожного елемента обладнання встановлюється умовне значення споживаної потужності, після чого виконується розрахунок загального навантаження системи.

Наявність модуля енергоспоживання дає змогу оцінювати вплив різних режимів роботи будинку на використання електроенергії. Наприклад, одночасна робота кондиціонера та обігрівача призводить до значного збільшення споживаної потужності порівняно з режимом, у якому активними залишаються лише освітлювальні прилади.

Важливим результатом моделювання стало формування зв'язків між окремими підсистемами. Освітлення, клімат-контроль та безпека не функціонують ізольовано. Їхня робота координується механізмами автоматизації, які аналізують поточний стан системи та забезпечують узгоджену поведінку всіх компонентів.

Запропонована модель дозволяє відтворити основні процеси, характерні для сучасних систем автоматизації житлових приміщень.

2.3 Проєктування алгоритмів автоматизації

Для симулятора розумного будинку алгоритми формують поведінку освітлення, кліматичного обладнання, охоронних засобів та допоміжних

сервісів. Саме вони забезпечують перехід від простого відображення даних до автоматичного реагування на зміни стану середовища.

У створеній системі застосовано комбінований підхід до керування. Частина рішень приймається користувачем через вебінтерфейс, а частина виконується автоматично після аналізу поточних параметрів. Така схема відповідає принципам роботи реальних систем Smart Home, де автоматичні сценарії доповнюють ручне керування.

Побудова алгоритмів починалася з визначення переліку подій, які можуть впливати на роботу будинку. До таких подій належать зміна температури, спрацювання датчика руху, відкриття або закриття дверей, перемикання режимів роботи та зміна стану обладнання користувачем. Для кожної події визначено набір дій, які повинні виконуватися системою.

Найбільш активною частиною програмного комплексу є алгоритм обробки даних датчиків. Після надходження нових значень температури та вологості сервер виконує перевірку встановлених порогів. Отримані результати використовуються для керування кліматичним обладнанням та оновлення інформації на сторінці користувача.

Для моделювання роботи температурних датчиків використовується послідовна зміна показників. Кожне нове значення обчислюється на основі попереднього стану із невеликим випадковим відхиленням. Цей підхід дозволяє отримати більш реалістичну поведінку системи порівняно з незалежною генерацією випадкових чисел.

Алгоритм кліматичного контролю ґрунтується на підтриманні допустимого температурного діапазону. У програмі встановлено верхню та нижню межі температури. Після перевищення верхнього порогу система активує кондиціонер. Якщо температура знижується нижче мінімального

значення, запускається обігрівач. Одночасна робота цих пристроїв не допускається.

Логіка роботи кліматичного контролю наведена в таблиці 2.3.

Таблиця 2.3 – Правила керування температурою

Температура	**Стан кондиціонера**	**Стан обігрівача**
Менше 19 °C	Вимкнений	Увімкнений
19–26 °C	Вимкнений	Вимкнений
Більше 26 °C	Увімкнений	Вимкнений

Після зміни стану обладнання інформація негайно передається до клієнтської частини системи. Користувач може спостерігати результати роботи алгоритму без оновлення сторінки.

Керування освітленням реалізовано за допомогою окремого алгоритму зміни станів пристроїв. Для кожної кімнати зберігається поточний стан освітлення. Натискання відповідної кнопки у вебінтерфейсі викликає серверний маршрут, який змінює значення логічної змінної та записує подію до журналу.

Окрім ручного керування, освітлення бере участь у роботі сценаріїв автоматизації. Наприклад, після активації нічного режиму система примусово вимикає всі освітлювальні прилади незалежно від їхнього попереднього стану. Такий механізм дозволяє забезпечити єдину логіку поведінки будинку при зміні режиму роботи.

Алгоритм підтримання температурного режиму базується на автоматичному керуванні кондиціонером та обігрівачем залежно від поточних показників температури. Послідовність виконання даного алгоритму наведено на рисунку 2.2.

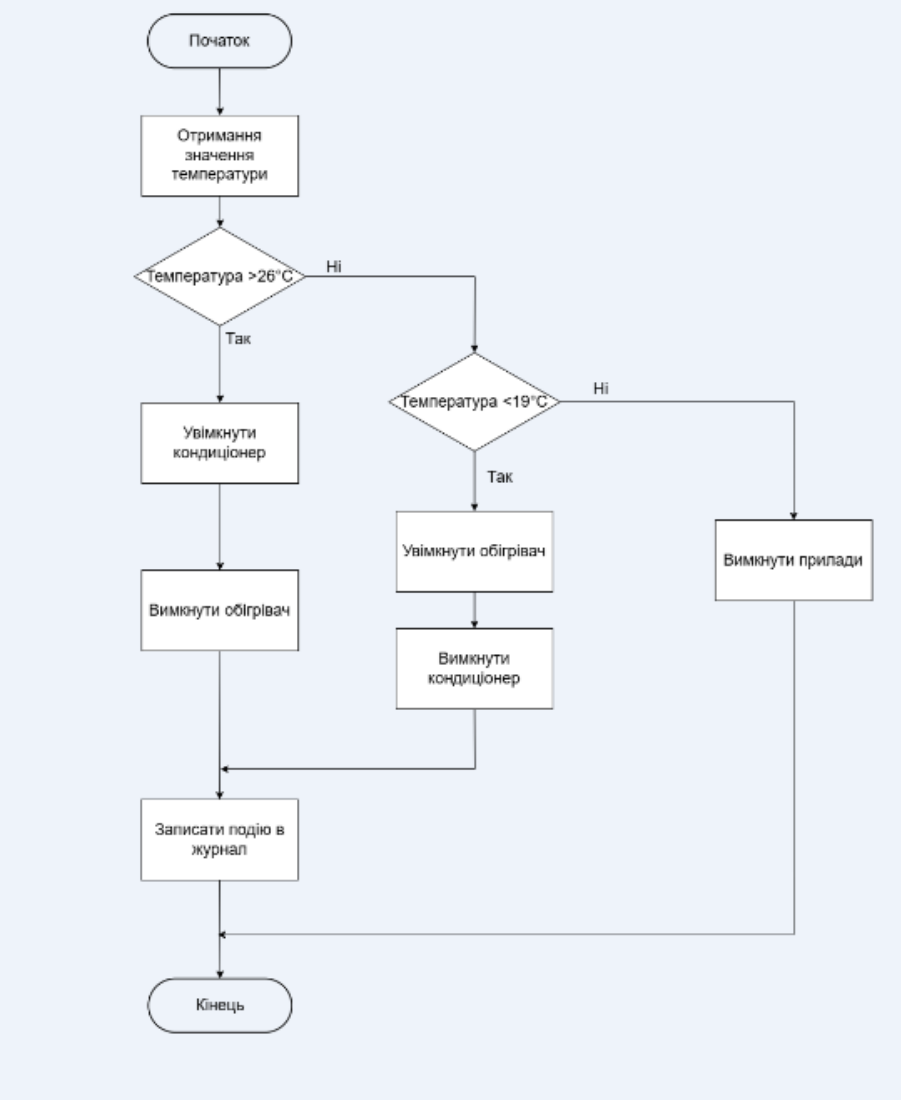


Рисунок 2.2 – Блок-схема алгоритму клімат-контролю

Для охоронної підсистеми було розроблено алгоритм контролю доступу та виявлення потенційних порушень. Основою його роботи є взаємодія між датчиком руху та сигналізацією. Датчик може перебувати у двох станах: рух виявлено або рух відсутній.

Поява руху не завжди повинна викликати тривогу. Якщо система охорони вимкнена, інформація про рух використовується лише для відображення поточного стану середовища. Інша ситуація виникає після активації сигналізації. У такому випадку поява руху розглядається як небажана подія.

Після спрацювання датчика руху система виконує перевірку стану сигналізації. Якщо охоронний режим активний, формується повідомлення про тривогу та створюється запис у журналі подій. Подібний механізм широко застосовується в реальних системах охоронної автоматизації.

Окремий алгоритм відповідає за керування дверима. У моделі двері можуть перебувати у відкритому або закритому стані. Перемикання виконується користувачем через вебінтерфейс. Зміна стану супроводжується оновленням інформації на сторінці та внесенням відповідного запису до бази даних.

Для підвищення наочності роботи системи було реалізовано кольорові індикатори. Алгоритм відображення визначає колір елемента залежно від його поточного стану. Активний пристрій відображається зеленим кольором, а вимкнений – червоним. Такий спосіб представлення інформації дозволяє швидко оцінити стан усіх компонентів без необхідності аналізу текстових повідомлень.

Для забезпечення контролю безпеки реалізовано алгоритм обробки подій датчика руху та формування повідомлень про тривогу. Схему роботи охоронної підсистеми наведено на рисунку 2.3.

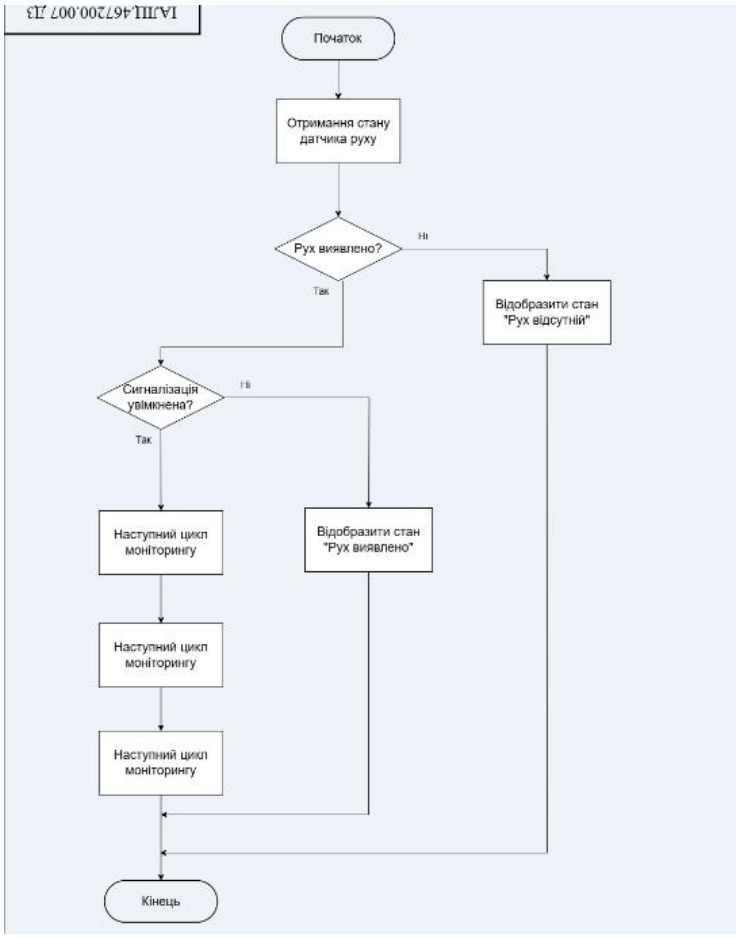


Рисунок 2.3 – Блок-схема алгоритму охоронної системи

Під час розробки симулятора було передбачено декілька режимів роботи будинку. Кожен режим являє собою набір заздалегідь визначених налаштувань, які застосовуються одночасно до декількох пристроїв.

Режим *Home* призначений для повсякденного використання будинку. Після його активації сигналізація вимикається, а користувач отримує можливість керувати обладнанням вручну.

Режим *Night* орієнтований на нічний час. Освітлення автоматично вимикається, а система охорони переходить до активного стану. Такий підхід дозволяє поєднати економію електроенергії та контроль безпеки житла.

Режим *Away* використовується під час відсутності мешканців. Основною його функцією є активація сигналізації та підготовка системи до виявлення несанкціонованого доступу.

Додатковим елементом автоматизації став механізм обліку енергоспоживання. Для кожного пристрою визначено значення споживаної потужності. Під час роботи системи виконується перевірка стану обладнання та обчислюється сумарне навантаження. Результат відображається у вебінтерфейсі та оновлюється разом із іншими параметрами.

Всі алгоритми працюють незалежно один від одного, але використовують спільний набір даних. Така організація дозволяє зменшити складність програмного коду та спрощує подальше розширення функціональних можливостей системи. Додавання нового датчика або виконавчого пристрою не потребує зміни вже реалізованих механізмів керування, що позитивно впливає на масштабованість програмного продукту.

2.4 Проєктування структури вебінтерфейсу користувача

Взаємодія користувача із системою автоматизації здійснюється через вебінтерфейс, загальний вигляд якого показано на рисунку 2.1. Саме цей компонент забезпечує доступ до функцій керування обладнанням, відображення поточного стану будинку та перегляд результатів роботи алгоритмів автоматизації. Від зручності інтерфейсу значною мірою залежить ефективність використання програмного комплексу, тому під час проєктування особлива увага приділялася простоті навігації, зрозумілому розташуванню елементів та швидкому доступу до основних функцій.

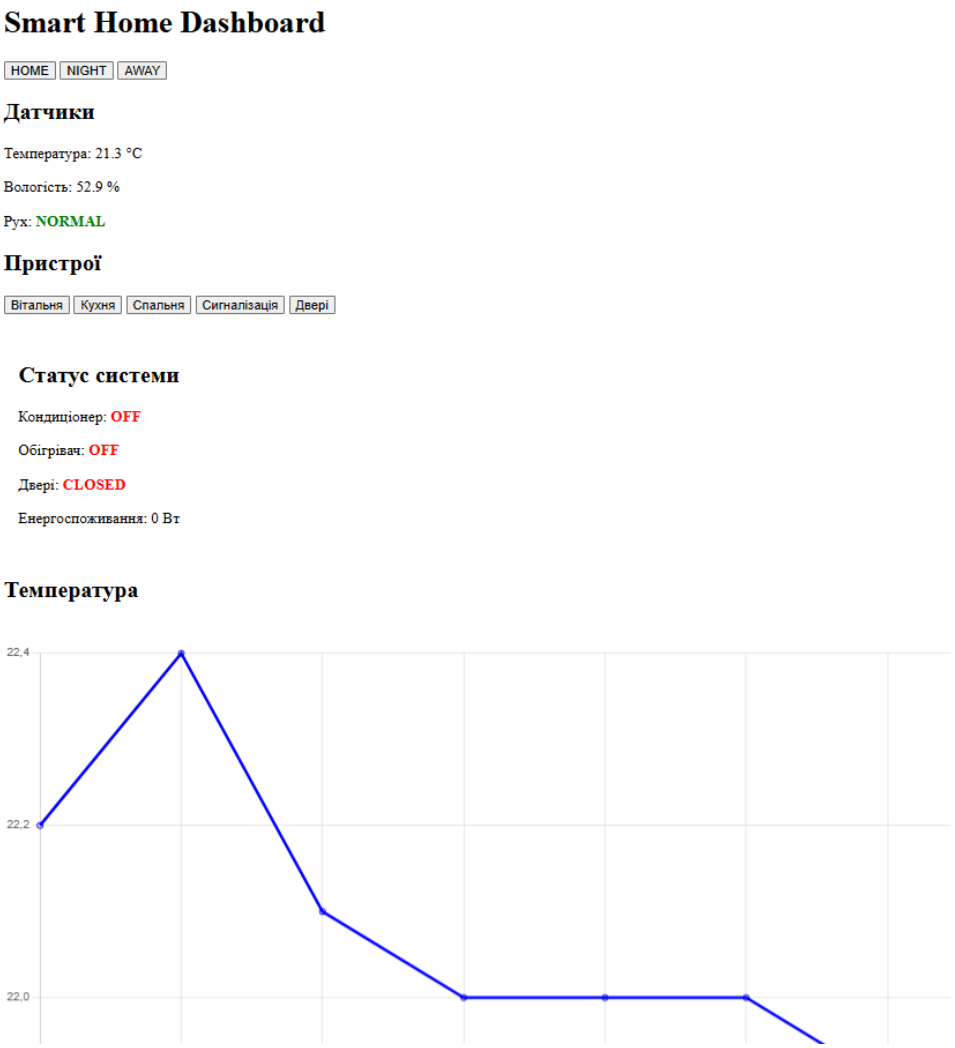


Рисунок 2.4 – Вигляд інтерфейсу користувача

У процесі розробки було обрано односторінкову структуру інтерфейсу. Всі основні компоненти системи розміщуються на одній вебсторінці, що дозволяє уникнути переходів між різними вікнами та спрощує роботу користувача. Такий підхід добре підходить для систем моніторингу, де необхідно одночасно контролювати значну кількість параметрів.

Вебінтерфейс розроблено за модульним принципом, що забезпечує зручне відображення параметрів системи та швидкий доступ до функцій керування. Структура основних елементів інтерфейсу наведена на рисунку 2.5.

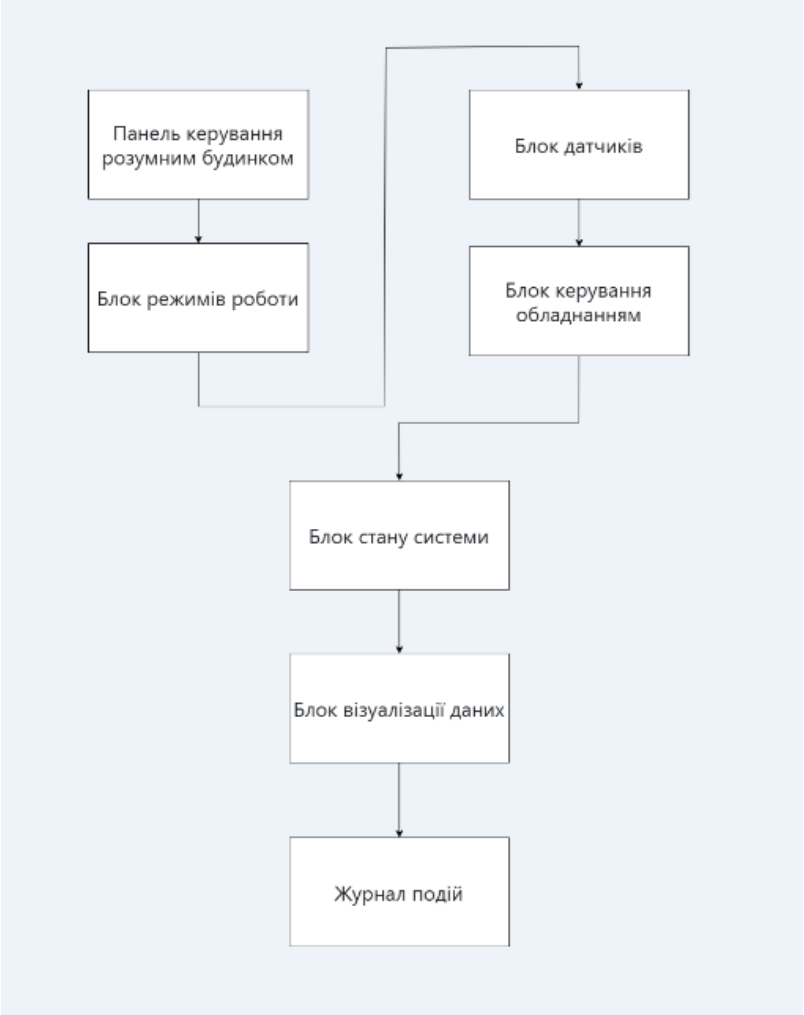


Рисунок 2.5 – Структурна схема вебінтерфейсу

Інтерфейс умовно поділено на декілька функціональних зон. Верхня частина сторінки містить назву системи та блок вибору режимів роботи будинку. Нижче розташовані інформаційні панелі, призначені для відображення показників датчиків, керування пристроями, перегляду журналу подій та візуалізації даних.

У верхній частині сторінки розміщується блок режимів роботи будинку. Він містить кнопки Home, Night та Away. Кожна кнопка відповідає окремому сценарію автоматизації та дозволяє виконати групову зміну налаштувань системи.

Після натискання відповідної кнопки браузер надсилає HTTP-запит до серверної частини. Сервер обробляє команду, виконує необхідні зміни станів пристроїв та повертає оновлену інформацію. Такий механізм забезпечує централізоване керування системою без необхідності окремого налаштування кожного елемента обладнання.

Наступний блок призначений для відображення показників датчиків. Користувач отримує інформацію про температуру повітря, вологість та стан датчика руху. Дані оновлюються автоматично через певні проміжки часу без перезавантаження сторінки.

Для реалізації такого механізму використовується асинхронна взаємодія між браузером та сервером. JavaScript надсилає запити до маршруту отримання даних датчиків, після чого оновлює відповідні елементи інтерфейсу. Завдяки цьому сторінка постійно відображає актуальний стан системи.

Окремий блок відведено для керування пристроями. У поточній версії симулятора користувач може змінювати стан освітлення у трьох приміщеннях, керувати сигналізацією та відкривати або закривати двері. Кожна дія виконується натисканням відповідної кнопки.

При розробці структури інтерфейсу було враховано необхідність швидкого визначення поточного стану обладнання. Для цього застосовано кольорові індикатори. Активний стан відображається зеленим кольором, а неактивний – червоним. Подібне рішення дозволяє оцінити ситуацію в будинку за декілька секунд навіть при великій кількості елементів керування.

Показники роботи кондиціонера та обігрівача також відображаються за допомогою кольорових індикаторів. Користувач може бачити результати роботи алгоритму кліматичного контролю без необхідності перегляду журналу подій або аналізу програмних повідомлень.

Для відображення інформації про доступ до будинку використовується індикатор стану дверей. Відкрите положення супроводжується відповідним текстовим повідомленням та кольоровим позначенням. Закриті двері відображаються іншим кольором, що дозволяє швидко визначити поточний стан системи доступу.

Наступним елементом інтерфейсу є блок енергоспоживання. Його призначення полягає у відображенні сумарної потужності всіх активних пристроїв. Значення розраховується на сервері та передається до браузера разом з іншими параметрами системи.

Відображення енергоспоживання дає можливість оцінювати вплив різних режимів роботи на використання електроенергії. Наприклад, увімкнення кліматичного обладнання суттєво збільшує загальне навантаження порівняно з роботою лише освітлювальних приладів.

Графічне подання даних має декілька переваг порівняно зі звичайним текстовим відображенням. Користувач може оцінити динаміку зміни температури, виявити тенденції та простежити вплив роботи кліматичного обладнання на мікроклімат приміщення.

Ще одним компонентом сторінки є журнал подій. У ньому відображаються останні дії користувача та результати роботи алгоритмів автоматизації. Інформація завантажується з бази даних та регулярно оновлюється.

3.РОЗРОБЛЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Вибір засобів і технологій розробки

Розробка програмного забезпечення для симулятора системи автоматизації розумного будинку потребує використання технологій, які забезпечують зручну реалізацію серверної логіки, взаємодію з користувачем через вебінтерфейс та зберігання службової інформації. Вибір інструментів безпосередньо впливає на складність реалізації проєкту, швидкість розробки та можливість подальшого розвитку програмного комплексу.

Система повинна підтримувати вебдоступ через браузер, забезпечувати оновлення даних без перезавантаження сторінки, виконувати алгоритми автоматизації та зберігати інформацію про події. Крім функціональних вимог, враховувалися доступність інструментів, обсяг документації, простота налагодження та сумісність із різними операційними системами.

Для реалізації серверної частини було обрано мову програмування Python. Це середовище широко використовується під час створення вебзастосунків, систем автоматизації та програм для обробки даних. Однією з причин вибору Python стала наявність великої кількості готових бібліотек, які дають змогу зосередитися на реалізації логіки програми, а не на низькорівневих аспектах роботи мережевих сервісів.

Серед можливих альтернатив розглядалися Java, C# та JavaScript. Використання Java забезпечує високу продуктивність та широкі можливості масштабування, проте потребує більшого обсягу програмного коду. Платформа .NET також надає значний набір інструментів для створення вебсервісів, однак її застосування ускладнює процес розгортання на різних платформах. Технології Node.js дозволяють реалізувати серверну та клієнтську частини

однією мовою програмування, проте для поставлених завдань переваги такого підходу не є визначальними.

Таблиця 3.1 – Порівняння мов програмування для серверної частини

Критерій	**Python**	**Java**	**C#**	**JavaScript**
Простота розробки	Висока	Середня	Середня	Висока
Швидкість створення прототипу	Висока	Низька	Середня	Висока
Документація	Великий обсяг	Великий обсяг	Великий обсяг	Великий обсяг
Поріг входження	Низький	Високий	Середній	Середній

Для побудови серверної частини обрано фреймворк Flask. Його використання дозволяє швидко створювати вебзастосунки та організовувати взаємодію між клієнтом і сервером через HTTP-запити. Кожен маршрут вебсервера може бути пов'язаний із певною функцією програми, що спрощує реалізацію. У розробленому симуляторі маршрути використовуються для отримання показників датчиків, зміни стану обладнання, завантаження журналу подій та перемикання режимів роботи будинку.

Зберігання даних реалізовано за допомогою SQLite. Для даного проєкту використання повноцінного серверного рішення на кшталт PostgreSQL або MySQL є недоцільним через невелику кількість записів та відсутність значного навантаження на систему.

SQLite працює у вигляді окремого файлу бази даних та не потребує встановлення додаткових служб. Така особливість спрощує розгортання програмного забезпечення та дозволяє запускати систему на будь-якому комп'ютері, де встановлено Python.

У базі даних зберігається журнал подій, який використовується для аналізу роботи системи. Кожна дія користувача та автоматичне спрацювання алгоритмів реєструються із зазначенням часу виконання. Отримані записи можуть використовуватися під час тестування та аналізу функціонування програмного комплексу.

Для створення користувацького інтерфейсу використовуються HTML, C# та JS. Таке поєднання технологій є стандартним рішенням для розробки вебзастосунків і підтримується всіма сучасними браузерами.

HTML відповідає за структуру сторінки та визначає розташування інформаційних блоків. За його допомогою створено області відображення датчиків, кнопки керування обладнанням, графік температури та журнал подій.

Оформлення сторінки реалізовано засобами C#. Стилі дозволяють формувати зовнішній вигляд елементів інтерфейсу, задавати кольорове оформлення та керувати розташуванням блоків на сторінці. Для відображення станів пристроїв використовуються кольорові індикатори, які дозволяють швидко визначити активність обладнання.

Передача інформації між серверною та клієнтською частинами виконується у форматі JSON. Такий формат є компактним, легко обробляється засобами JavaScript та широко використовується під час створення вебсервісів.

3.2 Розроблення програмної логіки симулятора

Програмна логіка симулятора забезпечує взаємодію між окремими компонентами системи та визначає порядок обробки даних, що надходять від модулів моделювання. У розробленому програмному забезпеченні основні функції зосереджені в серверній частині, яка відповідає за керування пристроями, виконання алгоритмів автоматизації, зберігання журналу подій та обмін інформацією з вебінтерфейсом.

Реалізація серверної частини виконана засобами фреймворку Flask. Робота програми побудована на основі маршрутів, кожен з яких обслуговує певний тип запиту. Такий підхід дозволяє логічно розділити функціональність системи та спростити супровід програмного коду.

Після запуску сервера виконується ініціалізація основних структур даних. Для зберігання поточного стану обладнання використовується словник devices, який містить інформацію про освітлення приміщень, сигналізацію, двері, кондиціонер та обігрівач. Кожен елемент приймає логічне значення, що відображає поточний стан пристрою.

Для роботи з журналом подій створено окремі функції ініціалізації бази даних та додавання записів. Під час першого запуску програми автоматично формується таблиця logs. Подальша робота із журналом не потребує додаткових налаштувань і виконується засобами SQL-запитів.

Кожна дія користувача супроводжується створенням нового запису. Наприклад, увімкнення освітлення або активація режиму охорони фіксуються в базі даних із зазначенням часу виконання. Наявність журналу дозволяє відтворити послідовність подій та оцінити коректність роботи алгоритмів.

Модуль моделювання датчиків займається тим, що, після отримання запиту від клієнтської частини сервер генерує нові значення температури та вологості.

Отримані значення проходять перевірку на відповідність допустимим межам. Температура обмежується діапазоном від 18 до 30 градусів Цельсія, а вологість підтримується в межах від 35 до 80 відсотків. Такий підхід запобігає появі нереалістичних результатів під час тривалої роботи симулятора.

Після оновлення параметрів середовища виконується аналіз умов роботи кліматичного обладнання. Якщо температура перевищує встановлене граничне

значення, система активує кондиціонер. Зниження температури нижче мінімального рівня призводить до запуску обігрівача. Робота цих пристроїв взаємно виключається, що відповідає принципам функціонування реальних систем клімат-контролю.

Окремий блок програмної логіки відповідає за керування режимами роботи будинку. При переході до режиму *Home* сигналізація деактивується та система переходить до звичайного режиму експлуатації. Режим *Night* змінює стан освітлення та переводить охоронну систему в активний стан. Під час вибору режиму Away пріоритет надається функціям безпеки, тому сигналізація залишається увімкненою незалежно від інших параметрів.

Перемикання станів обладнання здійснюється через окремий серверний маршрут. Перед внесенням змін виконується перевірка існування пристрою в системі. Якщо користувач намагається звернутися до неіснуючого елемента, сервер повертає повідомлення про помилку. Така перевірка підвищує надійність роботи програмного забезпечення та запобігає некоректній обробці запитів.

Для розрахунку енергоспоживання використовується таблиця умовних значень потужності. Кожному пристрою відповідає певне навантаження, яке враховується лише за умови його активного стану. Після аналізу всіх елементів обладнання обчислюється сумарне значення потужності, що відображається у вебінтерфейсі.

Під час роботи системи можуть виникати ситуації, які потребують негайного реагування. До таких випадків належить спрацювання датчика руху при активній сигналізації. Сервер аналізує стан охоронної системи та формує повідомлення про тривогу. Одночасно створюється запис у журналі подій, що дозволяє зберегти інформацію про інцидент.

Передача даних між серверною та клієнтською частинами виконується у форматі JSON, що забезпечує компактне представлення інформації та спрощує її подальшу обробку засобами JavaScript. Всі маршрути повертають структуровані дані, які можуть використовуватися для оновлення окремих елементів вебсторінки без повного перезавантаження інтерфейсу.

Для підтвердження практичної реалізації алгоритмів автоматизації в роботі наведено фрагменти програмного коду серверної частини.

Серверна частина отримує поточне значення температури, після чого виконує перевірку встановлених порогів. Якщо температура перевищує 26 °C, система вмикає кондиціонер та вимикає обігрівач. Якщо температура нижча за 19 °C, активується обігрівач, а кондиціонер вимикається. У проміжному діапазоні обидва пристрої залишаються вимкненими.

```
if temperature > 26:
    devices["conditioner"] = True
    devices["heater"] = False

elif temperature < 19:
    devices["heater"] = True
    devices["conditioner"] = False

else:
    devices["conditioner"] = False
    devices["heater"] = False
```

Наведений фрагмент відповідає схемі алгоритму клімат-контролю, поданій на рисунку 2.2. У схемі відображено послідовність перевірки температури та вибір дії залежно від поточного значення параметра.

Охоронна підсистема реалізована через перевірку стану сигналізації та датчика руху. Якщо сигналізація активна і датчик руху повідомляє про виявлення руху, система створює запис у журналі подій про спрацювання тривоги.

```
motion = random.choice([0, 0, 0, 1])

if devices["alarm"] and motion:
    add_log("ALARM TRIGGERED! Motion detected.")
```

Цей фрагмент відповідає схемі алгоритму охоронної системи, наведеній на рисунку 2.3. У програмній реалізації датчик руху моделюється випадковою подією, а реакція системи залежить від поточного стану сигналізації.

Керування пристроями виконується через окремий маршрут серверної частини. Перед зміною стану система перевіряє, чи існує пристрій із заданою назвою. Це дозволяє уникнути помилок при некоректних запитах до сервера.

```
@app.route("/api/device/<device_name>")
def toggle_device(device_name):

    if device_name not in devices:
        return jsonify({"error": "Device not found"}), 404

    devices[device_name] = not devices[device_name]

    add_log(f"{device_name} -> {devices[device_name]}")

    return jsonify(devices)
```

Реалізація журналу подій забезпечує збереження інформації про дії користувача та автоматичні спрацювання системи. Дані записуються в таблицю бази даних SQLite із зазначенням часу події.

```
def add_log(event):
    conn = sqlite3.connect("smart_home.db")
    cursor = conn.cursor()

    cursor.execute(
        "INSERT INTO logs(event, created_at) VALUES (?, ?)",
        (
            event,
            datetime.now().strftime("%Y-%m-%d %H:%M:%S")
        )
    )

    conn.commit()
    conn.close()
```

Наведені фрагменти коду підтверджують практичну реалізацію основних алгоритмів симулятора. Розроблені схеми у розділі 2 відображають логіку роботи системи на етапі проєктування, а приклади коду у розділі 3 демонструють її реалізацію засобами мови Python та фреймворку Flask.

3.3 Створення вебінтерфейсу користувача

Через користувацький інтерфейс здійснюється керування обладнанням, перегляд поточних параметрів та контроль роботи алгоритмів автоматизації. Для симулятора розумного будинку було обрано підхід, який дозволяє працювати із системою за допомогою браузера без встановлення додаткового програмного забезпечення.

Користувач повинен мати можливість отримати інформацію про стан будинку одразу після відкриття сторінки, не виконуючи додаткових дій для переходу між різними розділами системи.

Всі елементи сторінки розміщені у вигляді окремих логічних блоків. Кожен блок відповідає певній функціональній області та містить інформацію, пов'язану з конкретною підсистемою. Такий підхід спрощує сприйняття даних та забезпечує впорядковане відображення інформації.

У верхній частині сторінки, яка показана на рисунку 3.1, розміщується назва системи та блок вибору режимів роботи. Кнопки Home, Night та Away забезпечують швидке перемикання між заздалегідь визначеними сценаріями функціонування будинку. Після натискання відповідної кнопки формується запит до сервера, який виконує необхідні зміни конфігурації системи.

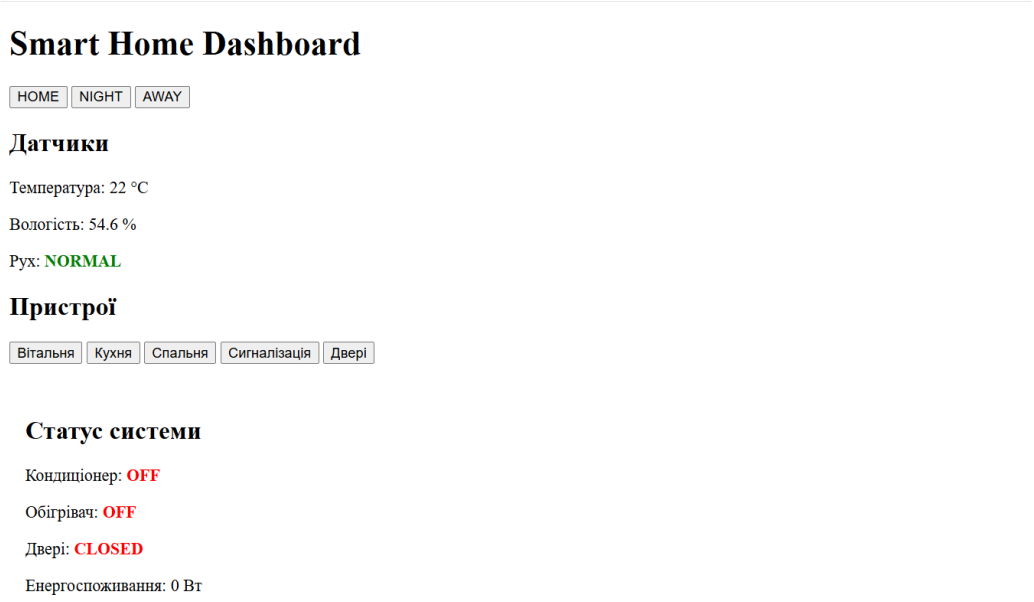


Рисунок 3.1 – Верхня частина сторінки інтерфейсу

Наступна частина інтерфейсу відведена для відображення показників датчиків. У цьому блоці користувач бачить поточну температуру повітря,

значення вологості та стан датчика руху. Дані надходять від серверної частини та автоматично оновлюються через визначені проміжки часу.

Відображення параметрів середовища в реальному часі дозволяє контролювати роботу алгоритмів кліматичного регулювання та оцінювати поведінку симулятора під час тестування різних режимів функціонування. Для датчика руху додатково використовується кольорове виділення, яке спрощує виявлення моментів спрацювання охоронної системи.

Окремий блок сторінки містить елементи керування обладнанням. Для кожного пристрою передбачена окрема кнопка. Натискання кнопки призводить до надсилання команди на сервер та зміни стану відповідного елемента. Після обробки запиту інформація на сторінці оновлюється автоматично. У таблиці 3.2 зображено елементи інтерфейсу та функції, які вони виконують.

Таблиця 3.2 – Елементи керування вебінтерфейсу

Елемент	**Функція**
HOME	Перехід до базового режиму
NIGHT	Активація нічного сценарію
AWAY	Увімкнення режиму відсутності
Вітальня	Керування освітленням вітальні
Кухня	Керування освітленням кухні
Спальня	Керування освітленням спальні
Сигналізація	Зміна стану охоронної системи
Двері	Відкриття або закриття дверей

Для оформлення сторінки використовується C#. За допомогою стилів визначаються кольори, розміри елементів, відступи та розташування інформаційних блоків. Застосування стилів дозволило сформувати єдиний візуальний стиль та зробити інтерфейс більш зрозумілим для користувача.

Окреме значення має використання кольорових індикаторів. Зелений колір використовується для відображення активних пристроїв та нормального стану системи. Червоний сигналізує про вимкнений пристрій або спрацювання окремих елементів безпеки. Подібний спосіб представлення інформації значно пришвидшує сприйняття даних у порівнянні з текстовими повідомленнями.

Асинхронний механізм обміну інформацією дає змогу оновлювати лише необхідні елементи сторінки без її повного перезавантаження. Користувач продовжує працювати з інтерфейсом, тоді як дані у фоновому режимі отримуються із сервера та відображаються в актуальному вигляді.

Для візуалізації зміни температури використано графічний компонент. Після кожного оновлення даних нове значення додається до набору точок графіка. У результаті формується крива, яка відображає зміну температури в часі.

Журнал подій, який продемонстровано на рисунку 3.2, також інтегрований безпосередньо до вебінтерфейсу. Після виконання будь-якої дії відповідний запис додається до бази даних і відображається на сторінці. Користувач може переглядати останні зміни станів обладнання, активацію режимів та спрацювання охоронної системи.

Журнал подій

- 2026-06-06 21:01:19 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 21:00:19 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:58:19 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:56:19 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:41:19 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:40:19 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:38:19 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:36:19 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:31:19 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:29:19 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:27:00 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:26:56 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:26:36 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:26:26 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:26:20 : away mode activated
- 2026-06-06 20:26:17 : home mode activated
- 2026-06-06 20:26:15 : night mode activated
- 2026-06-06 20:26:14 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:26:12 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:26:09 : bedroom_light -> False

Рисунок 3.2 – Приклад журналу подій

Запропонована структура вебінтерфейсу забезпечує швидкий доступ до основних функцій симулятора, наочне відображення стану системи та зручний механізм взаємодії з обладнанням.

3.4 Реалізація механізму симуляції та сценаріїв роботи

Формування температури та вологості виконується програмним шляхом. Початкові значення задаються під час запуску сервера, після чого кожне нове значення обчислюється з урахуванням попереднього стану системи. Такий підхід дозволяє отримати плавну зміну параметрів та уникнути різких стрибків, які характерні для звичайної випадкової генерації чисел.

Для температури використовується невелике випадкове відхилення в межах заданого діапазону. Після розрахунку нового значення виконується перевірка на відповідність встановленим обмеженням. Це дозволяє підтримувати реалістичну поведінку системи протягом тривалого часу роботи.

Подібний принцип застосовується і для вологості повітря. Значення змінюється поступово та не виходить за межі допустимого інтервалу. У результаті користувач отримує дані, які нагадують реальні показники житлового приміщення.

Імітація датчика руху реалізована дещо інакше. Його стан формується випадковим чином під час кожного циклу оновлення. В більшості випадків система повідомляє про відсутність руху, проте з певною ймовірністю генерується подія виявлення активності. Такий механізм дозволяє перевіряти роботу охоронних алгоритмів без втручання користувача.

Після отримання нових даних запускаються процедури автоматичного керування обладнанням. Першим виконується аналіз температури повітря.

Якщо значення перевищує встановлений поріг, система активує кондиціонер. При зниженні температури нижче допустимого рівня вмикається обігрівач. У проміжному діапазоні обидва пристрої залишаються вимкненими.

Застосований підхід відтворює принцип роботи звичайного термостата. Система самостійно підтримує комфортні умови та не потребує постійного втручання користувача. Результати роботи алгоритму відображаються у вебінтерфейсі за допомогою текстових та кольорових індикаторів.

Для підвищення наочності роботи програмного комплексу було реалізовано набір сценаріїв функціонування будинку. Кожен сценарій являє собою певну конфігурацію обладнання, яка відповідає конкретним умовам експлуатації.

Сценарій *Home* використовується як базовий режим роботи. У цьому випадку охоронна система перебуває у вимкненому стані, а користувач отримує повний контроль над освітленням та іншими пристроями. Даний режим відповідає ситуації, коли мешканці знаходяться вдома та можуть самостійно керувати обладнанням.

Сценарій *Night* орієнтований на використання у вечірній та нічний час. Після його активації освітлення автоматично вимикається, а сигналізація переходить у режим контролю. Така конфігурація дозволяє зменшити споживання електроенергії та одночасно забезпечити додатковий рівень безпеки.

Сценарій *Away* використовується під час відсутності мешканців. Основним завданням цього режиму є захист приміщення від несанкціонованого доступу. Після його активації система переходить до стану постійного моніторингу датчика руху.

Механізм перемикання режимів роботи будинку також реалізовано на рівні серверної частини. У програмі передбачено три режими: *Home, Night* та *Away*. Кожен режим змінює стан окремих пристроїв відповідно до заданого сценарію.

```
@app.route("/api/mode/<mode>")
def set_mode(mode):

    if mode == "home":
        devices["alarm"] = False

    elif mode == "night":
        devices["living_light"] = False
        devices["kitchen_light"] = False
        devices["bedroom_light"] = False
        devices["alarm"] = True

    elif mode == "away":
        devices["alarm"] = True

    else:
        return jsonify({"error": "Unknown mode"}), 404

    add_log(f"{mode} mode activated")

    return jsonify(devices)
```

Фрагмент коду демонструє, що сценарії роботи будинку реалізовані не тільки на рівні опису алгоритмів, а й у вигляді серверної логіки. При виборі

режиму користувачем вебінтерфейс надсилає запит до сервера, після чого змінюється стан відповідних пристроїв і створюється запис у журналі подій.

При спрацюванні датчика руху система виконує перевірку стану сигналізації. Якщо охоронний режим активний, формується повідомлення про тривогу. Одночасно створюється запис у журналі подій із зазначенням часу інциденту. Такий підхід дозволяє аналізувати поведінку системи та відстежувати всі випадки спрацювання охоронних механізмів.

Для моделювання роботи електронного замка реалізовано механізм відкриття та закриття дверей. Поточний стан дверей зберігається на сервері та може змінюватися через вебінтерфейс. Всі зміни також фіксуються в журналі подій.

Окремим елементом симулятора є підсистема розрахунку енергоспоживання. Для кожного пристрою встановлено умовне значення потужності. Після визначення поточного стану обладнання виконується підрахунок сумарного навантаження системи.

Розраховане значення використовується для відображення поточного споживання електроенергії. Користувач може спостерігати зміну навантаження при ввімкненні освітлення, запуску кліматичного обладнання або переході між режимами роботи.

4.ТЕСТУВАННЯ, АНАЛІЗ ТА ОЦІНКА РОБОТИ СИСТЕМИ

4.1 Методика тестування симулятора

Процес тестування охоплював усі ключові елементи симулятора. Перевірялися механізми отримання даних від модулів симуляції, робота алгоритмів кліматичного контролю, функціонування охоронної підсистеми, перемикання режимів роботи будинку, обробка користувацьких команд та правильність відображення інформації у вебінтерфейсі.

Тестування виконувалося покроково. На першому кроці кожного маршруту серверної частини здійснювалося надсилання тестових запитів з подальшим аналізом отриманих відповідей. Особлива увага приділялася коректності обробки помилкових даних та реакції системи на нестандартні ситуації.

Під час перевірки функції керування пристроями здійснювалися спроби звернення як до існуючих елементів обладнання, так і до неіснуючих об'єктів. У випадках звернення до відсутнього пристрою система повинна повертати повідомлення про помилку та не виконувати жодних змін у поточній конфігурації будинку.

Другий етап тестування стосувався роботи механізму симуляції датчиків. Протягом тривалого часу виконувалося спостереження за зміною температури та вологості. Отримані результати аналізувалися з точки зору плавності зміни параметрів та дотримання встановлених меж.

Окремо перевірялася робота датчика руху. Для цього виконувалося багаторазове оновлення даних із фіксацією моментів появи сигналу про виявлення руху. Отримані результати використовувалися для оцінки роботи охоронних алгоритмів. Як можна побачити на рисунку 4.1, в даний момент стан руху горить зеленим кольором.

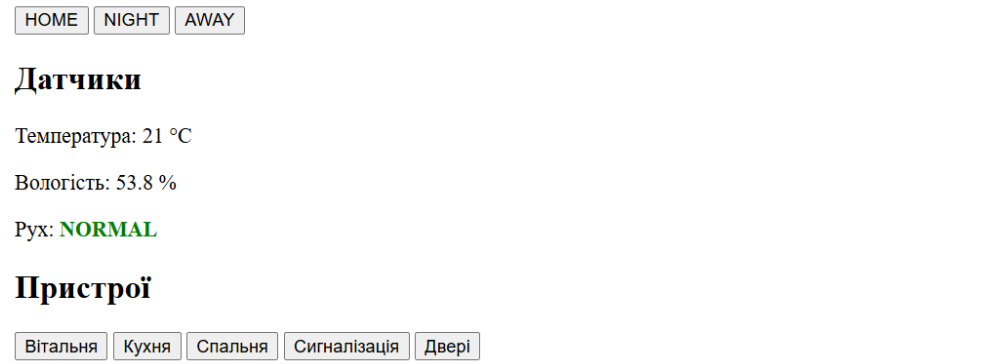


Рисунок 4.1 – Перевірка роботи датчика руху

Наступним кроком стала перевірка функціонування кліматичного контролю. Під час випробувань змінювалися значення температури, після чого аналізувалася реакція системи. Перевірялося автоматичне увімкнення кондиціонера при перевищенні встановленого порогу та запуск обігрівача при зниженні температури нижче допустимого рівня.

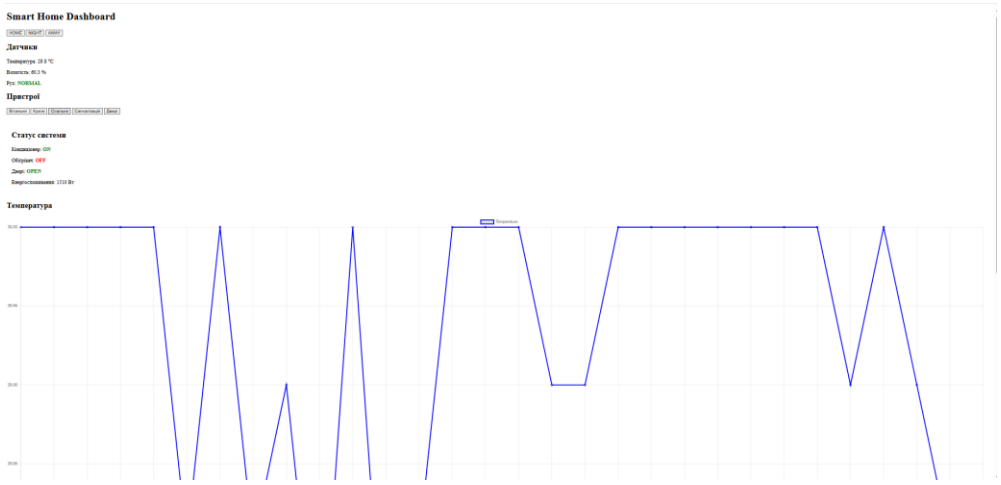


Рисунок 4.2 – Перевірка роботи кліматичного оконтролю

Як видно з рисунку 4.2, у разі вибору Спальні програма, аналізуючи температуру, ввімкнула кондиціонер та вимкнула обігрівач.

Для оцінки роботи сценаріїв автоматизації виконувалося послідовне перемикання режимів *Home, Night* та *Away*. Після активації кожного режиму контролювалися зміни станів освітлення, сигналізації та інших пристроїв. На рисунках 4.3 - 4.5 показана робота розробленої системи в усіх трьох режимах. Отримані результати порівнювалися з алгоритмами, визначеними на етапі проєктування.

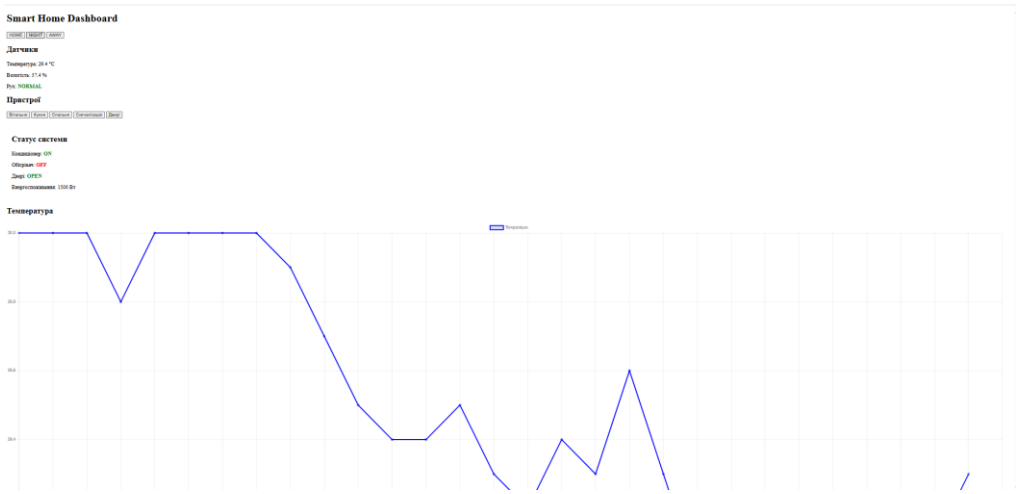


Рисунок 4.3 – Вигляд роботи програми в режимі *Home*

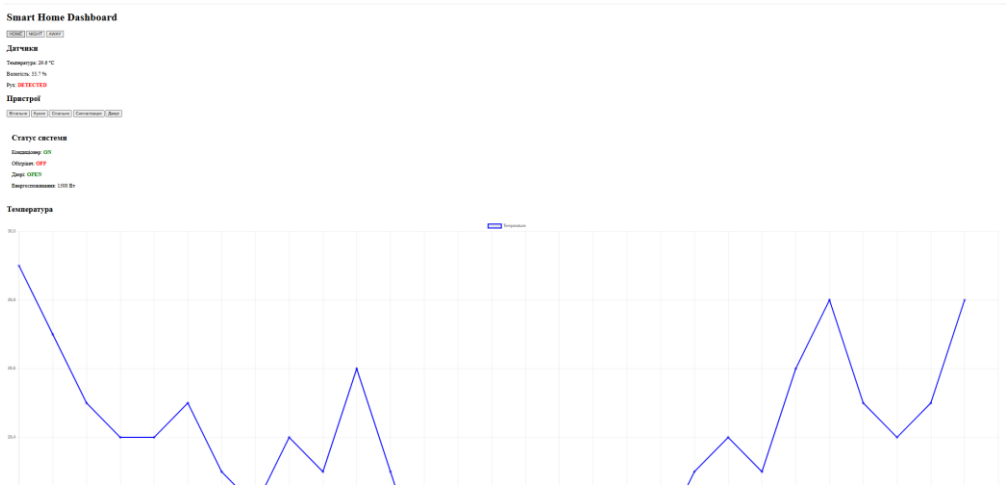


Рисунок 4.4 – Вигляд роботи програми в режимі *Night*

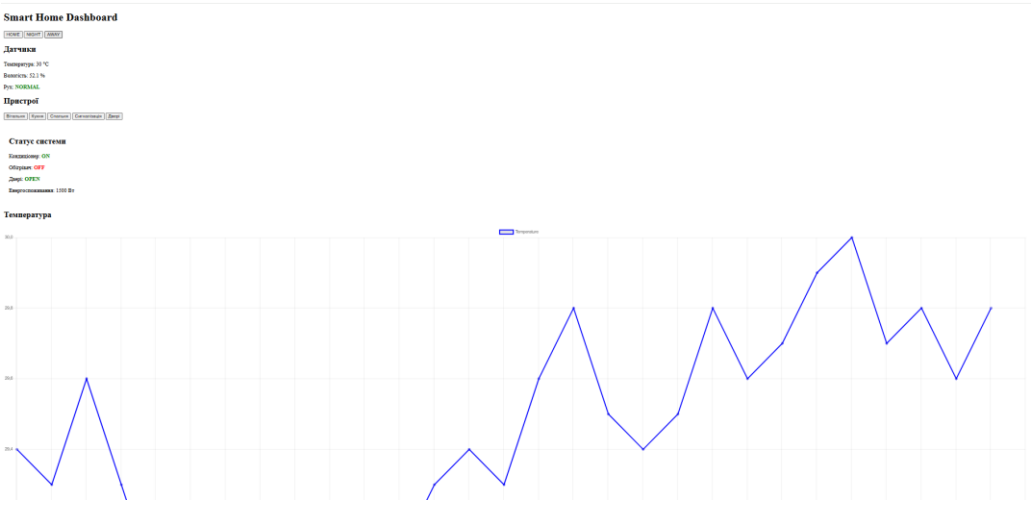


Рисунок 4.5 – Вигляд роботи програми в режимі *Away*

Наступна група випробувань була пов'язана з охоронною підсистемою. Під час тестування активувалася сигналізація, після чого імітувалося спрацювання датчика руху. У відповідь система повинна була сформувати повідомлення про тривогу та додати відповідний запис до журналу подій.

Перевірка журналу подій виконувалася шляхом послідовного виконання різних дій користувача. Після кожної зміни стану обладнання аналізувалася наявність нового запису в базі даних. Крім факту створення запису перевірялася правильність відображення часу та змісту події. На рисунку 4.6 продемонстровано правильність внесення різних даних у журнал подій в інтерфейсі користувача.

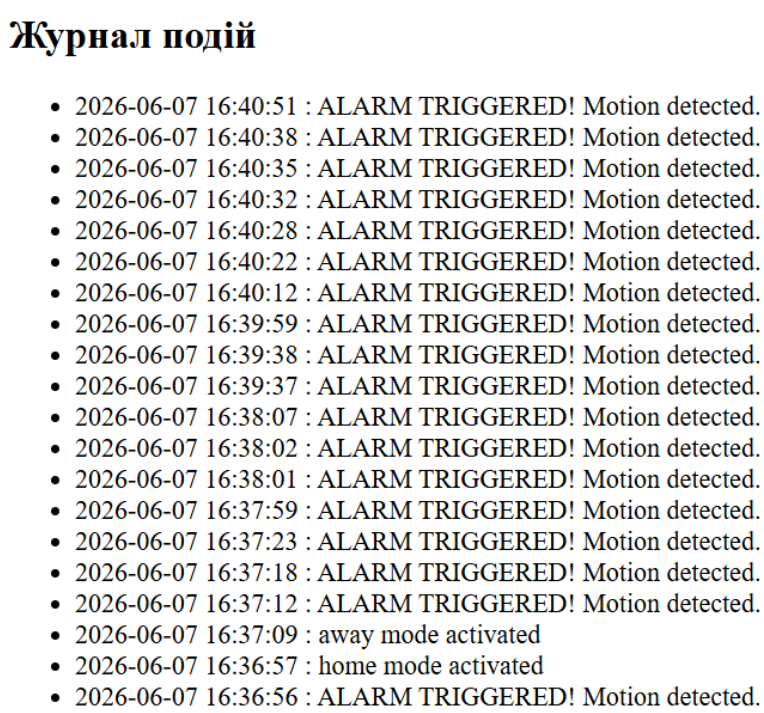


Рисунок 4.6 – Перевірка правильності внесення даних у журнал подій

Тестування вебінтерфейсу передбачало перевірку відображення всіх елементів сторінки у браузері. Аналізувалася робота кнопок керування, коректність оновлення показників датчиків, функціонування графіка температури та відображення журналу подій.

Окремо оцінювалася швидкість оновлення інформації на сторінці. Оскільки взаємодія між браузером та сервером реалізована за допомогою асинхронних запитів, необхідно було переконатися у відсутності затримок під час отримання нових даних.

Для перевірки стабільності програмного забезпечення симулятор працював безперервно протягом тривалого часу. У процесі тестування контролювалися коректність зміни параметрів середовища, робота алгоритмів автоматизації та відсутність критичних помилок у роботі сервера.

Проведені випробування дозволили підтвердити працездатність усіх функціональних компонентів програмного комплексу. Отримані результати показали, що симулятор коректно виконує покладені на нього функції та забезпечує стабільне функціонування в різних режимах експлуатації.

4.2 Аналіз роботи вебінтерфейсу користувача

Під час випробувань було перевірено коректність відображення всіх елементів сторінки після завантаження браузером. Ця перевірка показана на рисунку 4.7. Аналіз показав, що інформаційні блоки відображаються відповідно до структури, визначеної на етапі проєктування. Дані про температуру, вологість та стан датчика руху завантажуються автоматично без необхідності додаткових дій з боку користувача.

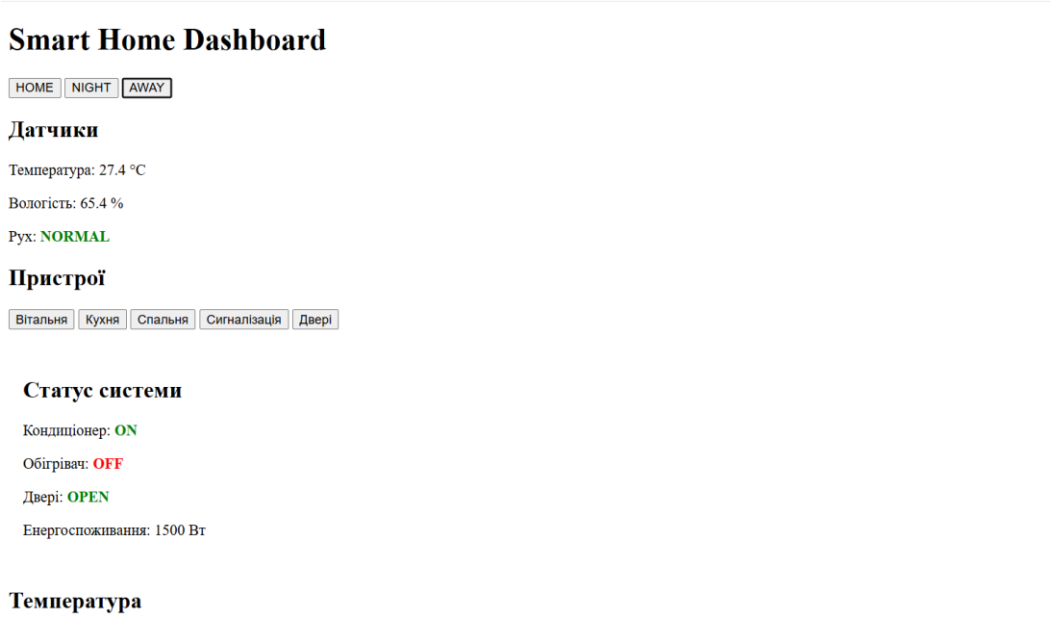


Рисунок 4.7 – Відображення елементів сторінки після завантаження браузера

Особливу увагу було приділено роботі механізму оновлення інформації. Після відкриття сторінки браузер періодично надсилає запити до серверної частини та отримує нові дані про стан системи.

У процесі тестування не було зафіксовано випадків втрати даних або припинення оновлення показників датчиків. Навіть при тривалому використанні сторінки значення температури та вологості продовжували змінюватися відповідно до алгоритмів симуляції.

Перевірка елементів керування виконувалася шляхом багаторазового натискання кнопок керування обладнанням. Після надсилання команди зміни станів пристроїв відображалися практично миттєво. Затримка між виконанням дії та оновленням інтерфейсу залишалася незначною і не впливала на зручність використання системи.

Позитивний вплив на сприйняття інформації забезпечило використання кольорових індикаторів. Під час аналізу було встановлено, що визначення стану обладнання займає значно менше часу порівняно з текстовим представленням даних. Користувач може швидко оцінити поточну ситуацію в системі без необхідності детального перегляду кожного параметра.

Для оцінки ефективності інтерфейсу було проаналізовано роботу блоку відображення кліматичного обладнання. Стан кондиціонера та обігрівача змінювався автоматично залежно від температури середовища. Інформація про зміни миттєво відображалася на сторінці, що дозволяло спостерігати реакцію системи на зміну параметрів.

Аналогічна перевірка проводилася для охоронної підсистеми. Після активації сигналізації та появи події руху система коректно відображала інформацію про спрацювання охоронних механізмів. Одночасно оновлювався журнал подій, що підтверджувало правильність роботи серверної логіки та клієнтської частини.

Окремо аналізувалася робота графічних елементів інтерфейсу. Для побудови графіка температури використовується бібліотека Chart.js. Під час

тестування перевірялося додавання нових значень та оновлення графіка в режимі реального часу. Усі отримані дані коректно відображалися на діаграмі, а сама побудова графіка не створювала помітного навантаження на браузер.

Журнал подій також продемонстрував стабільну роботу. Нові записи з'являлися одразу після виконання відповідних дій користувача або автоматичних алгоритмів. Інформація про час виконання подій відображалася правильно, що дозволяло відстежувати послідовність роботи системи.

Під час аналізу було оцінено зручність структури сторінки. Розташування елементів забезпечує швидкий доступ до основних функцій без необхідності прокручування великої кількості інформації. Найбільш важливі компоненти системи знаходяться у видимій області сторінки та доступні одразу після її відкриття.

Отримані результати свідчать про те, що розроблений вебінтерфейс забезпечує ефективну взаємодію користувача із симулятором розумного будинку. Реалізовані механізми відображення та оновлення інформації працюють стабільно, а структура сторінки дозволяє зручно керувати обладнанням та контролювати роботу системи автоматизації.

4.3 Результати моделювання роботи системи

Після завершення тестування було проведено серію експериментів, спрямованих на оцінку поведінки симулятора в різних режимах роботи. Отримані результати дозволили перевірити правильність реалізації алгоритмів автоматизації та визначити реакцію системи на зміну параметрів середовища.

Перший етап дослідження передбачав спостереження за роботою підсистеми кліматичного контролю. Під час моделювання температура середовища поступово змінювалася відповідно до алгоритму генерації даних. У процесі роботи було зафіксовано декілька випадків перевищення верхнього

температурного порогу та зниження температури до мінімально допустимого рівня.

Після перевищення встановленого значення система автоматично активувала кондиціонер. Інформація про зміну стану обладнання відображалася у вебінтерфейсі та фіксувалася у журналі подій. Аналогічна ситуація спостерігалася при зниженні температури нижче допустимого рівня, коли система запускала обігрівач.

Отримані результати підтвердили правильність реалізації алгоритму кліматичного контролю. У всіх випадках обладнання реагувало відповідно до заданих умов, а одночасне ввімкнення кондиціонера та обігрівача не відбувалося.

Другий етап дослідження був присвячений аналізу роботи режимів автоматизації. Для перевірки використовувалися сценарії Home, Night та Away. Після активації кожного режиму контролювалися зміни станів освітлення та сигналізації.

У режимі *Home* система переходила до звичайного стану експлуатації. Освітлення залишалося доступним для ручного керування, а охоронна підсистема вимикалася. Під час перевірки не було зафіксовано відхилень від заданого алгоритму.

Перехід до режиму *Night* призводив до автоматичного вимкнення освітлення та активації сигналізації. Після запуску сценарію всі освітлювальні прилади змінювали свій стан незалежно від попередньої конфігурації системи. Така поведінка повністю відповідала проєктним вимогам.

Під час використання режиму *Away* система переводилася до стану підвищеного контролю безпеки. Основний акцент у цьому режимі робився на роботі сигналізації та моніторингу подій датчика руху.

Наступна серія випробувань стосувалася роботи охоронної підсистеми. Для цього активувалася сигналізація та виконувалася симуляція появи руху в контрольованій зоні. Після спрацювання датчика система формувала повідомлення про тривогу та створювала відповідний запис у журналі подій.

Під час багаторазового повторення експерименту всі випадки виявлення руху були коректно оброблені серверною частиною. Втрата подій або некоректне відображення повідомлень не спостерігалися.

Окремо оцінювалася робота підсистеми керування дверима. Користувач багаторазово змінював стан дверей через вебінтерфейс. Сервер успішно обробляв усі запити та відображав актуальний стан системи. Кожна зміна також фіксувалася у журналі подій.

Під час експлуатаційного тестування було перевірено роботу механізму накопичення журналу подій. Протягом декількох циклів роботи виконувалися різні операції: зміна режимів, керування освітленням, відкриття дверей та спрацювання сигналізації. Усі події успішно зберігалися в базі даних та відображалися у вебінтерфейсі.

Ще одним напрямом дослідження стала оцінка роботи підсистеми обліку енергоспоживання. Після активації різних пристроїв змінювалися значення сумарної потужності, що відображалися на сторінці користувача. Найбільший вплив на навантаження системи створювали кондиціонер та обігрівач, тоді як освітлювальні прилади споживали значно менше енергії.

Результати розрахунків підтвердили правильність роботи алгоритму підрахунку потужності. Показники змінювалися відповідно до поточного стану обладнання та відображали реальну структуру навантаження системи.

Під час тривалого тестування симулятор працював безперервно протягом декількох годин. У цей період контролювалися стабільність серверної частини,

коректність оновлення даних та робота вебінтерфейсу. Ознак втрати з'єднання, зависання програми або припинення оновлення параметрів виявлено не було.

Аналіз отриманих результатів свідчить про коректне функціонування всіх основних компонентів програмного комплексу. Реалізовані алгоритми автоматизації забезпечують узгоджену роботу обладнання, а система успішно реагує на зміни параметрів середовища та дії користувача. Проведене моделювання підтвердило працездатність розробленого симулятора та можливість його використання для демонстрації принципів роботи систем автоматизації розумного будинку.

4.4 Оцінка ефективності та перспективи розвитку системи

Розроблений симулятор системи автоматизації розумного будинку дозволив реалізувати комплекс функцій, характерних для сучасних систем керування житловими приміщеннями. У процесі роботи було створено програмне середовище, яке забезпечує моделювання параметрів навколишнього середовища, автоматичне керування обладнанням, моніторинг стану пристроїв та взаємодію користувача із системою через вебінтерфейс.

Оцінювання ефективності виконувалося на основі результатів тестування окремих компонентів та аналізу роботи системи в цілому. Основними критеріями стали стабільність функціонування, коректність виконання алгоритмів автоматизації, швидкість оновлення даних та зручність взаємодії користувача із програмним забезпеченням.

Під час експлуатації система демонструвала стабільну роботу всіх реалізованих підсистем. Серверна частина успішно обробляла запити клієнтів, виконувала моделювання параметрів середовища та забезпечувала роботу алгоритмів автоматичного керування. Випадки втрати даних або порушення логіки роботи програмного комплексу під час тестування не спостерігалися.

Застосування веборієнтованої архітектури дозволило забезпечити доступ до системи через будь-який сучасний браузер. Відсутність необхідності встановлення спеціалізованого клієнтського програмного забезпечення спрощує використання системи та створює можливість її подальшого розгортання на різних програмно-апаратних платформах.

Реалізований механізм симуляції датчиків дозволив відтворити зміну температури, вологості та стану датчика руху без використання фізичного обладнання. Такий підхід виявився достатнім для перевірки алгоритмів автоматизації та аналізу поведінки системи в різних умовах експлуатації.

Алгоритми кліматичного контролю забезпечили автоматичне реагування на зміну температури середовища. Кондиціонер та обігрівач активувалися відповідно до встановлених порогових значень, що дозволило підтримувати параметри в заданому діапазоні. Отримані результати свідчать про коректну реалізацію механізму автоматичного керування.

Позитивні результати були отримані також під час оцінювання роботи охоронної підсистеми. Реакція на спрацювання датчика руху виконувалася відповідно до поточного режиму роботи системи. У випадках активованої сигналізації формувалися повідомлення про тривогу та створювалися записи у журналі подій.

Ефективність вебінтерфейсу підтвердилася під час практичного використання системи. Інформація про стан обладнання оновлювалася своєчасно, а структура сторінки забезпечувала швидкий доступ до всіх основних функцій. Використання кольорових індикаторів дозволило спростити процес контролю стану пристроїв та підвищити наочність представлення інформації.

Під час аналізу програмного комплексу було визначено низку напрямів подальшого розвитку системи. Поточна версія симулятора виконує функції програмного моделювання, проте її архітектура дозволяє інтегрувати реальні апаратні компоненти без суттєвої зміни загальної структури програмного забезпечення.

Одним із можливих напрямів модернізації є підключення фізичних датчиків температури, вологості та руху. У такому випадку система зможе працювати не лише як симулятор, а й як повноцінний центр керування реальним обладнанням.

Подальший розвиток може передбачати використання мікроконтролерів для збору даних та керування виконавчими пристроями. Як приклад можуть бути використані плати STM32, ESP32 або Arduino, які широко застосовуються у проєктах автоматизації будівель.

Ще одним перспективним напрямом є впровадження системи авторизації користувачів. Реалізація механізмів аутентифікації дозволить розмежувати права доступу та забезпечити додатковий рівень захисту інформації.

Підвищити функціональність програмного комплексу можна також шляхом впровадження розширених сценаріїв автоматизації. Наприклад, система може враховувати час доби, рівень освітленості приміщення або наявність людей у будинку. Це дозволить зробити алгоритми керування більш адаптивними та наблизити їх до реальних умов експлуатації.

Інтерес становить і використання технологій машинного навчання. Аналіз накопичених даних може застосовуватися для прогнозування споживання електроенергії, автоматичного налаштування параметрів клімат-контролю або формування індивідуальних сценаріїв роботи будинку.

Розширення системи за рахунок мобільного застосунку також може стати перспективним напрямом розвитку. У цьому випадку користувач отримає можливість керувати обладнанням та контролювати стан будинку безпосередньо зі смартфона.

Проведена робота підтвердила можливість створення програмного симулятора системи автоматизації розумного будинку із використанням сучасних вебтехнологій. Отриманий програмний комплекс забезпечує виконання поставлених завдань та демонструє принципи функціонування систем Smart Home.

ВИСНОВКИ

У результаті виконання дипломної роботи було розроблено програмний симулятор системи автоматизації розумного будинку з вебінтерфейсом користувача. Створений програмний комплекс дозволяє моделювати роботу основних підсистем сучасного житлового середовища, забезпечує керування обладнанням через браузер та надає можливість аналізувати результати роботи алгоритмів автоматизації в режимі реального часу.

Спочатку було проведено дослідження принципів функціонування систем автоматизації житлових приміщень, розглянуто особливості побудови сучасних рішень класу Smart Home та визначено перелік функцій, необхідних для реалізації в програмному симуляторі. Аналіз предметної області дозволив сформувати вимоги до структури майбутньої системи та визначити перелік підсистем, що повинні бути реалізовані в межах проєкту.

У процесі проєктування було сформовано структуру симулятора, яка включає модуль моделювання датчиків, підсистему автоматизації, блок керування обладнанням, журнал подій та вебінтерфейс користувача. Визначено принципи взаємодії між окремими компонентами системи та розроблено алгоритми обробки даних, отриманих у результаті моделювання.

Для відтворення роботи розумного будинку було реалізовано підсистеми освітлення, кліматичного контролю, безпеки та моніторингу параметрів середовища. Передбачено підтримку декількох режимів функціонування будинку, що дозволяють автоматично змінювати конфігурацію обладнання відповідно до поточних умов експлуатації.

У ході розробки було реалізовано механізм симуляції датчиків температури, вологості та руху. На основі отриманих даних система автоматично керує роботою кондиціонера та обігрівача, підтримуючи

параметри середовища в заданих межах. Додатково реалізовано алгоритми роботи охоронної підсистеми, керування дверима, підрахунку енергоспоживання та ведення журналу подій.

Проведене тестування підтвердило працездатність усіх функціональних компонентів програмного комплексу. Перевірка роботи вебінтерфейсу, алгоритмів автоматизації та механізмів взаємодії між серверною та клієнтською частинами не виявила критичних помилок. Отримані результати засвідчили коректне виконання сценаріїв керування обладнанням та стабільне функціонування системи в різних режимах роботи.

Під час моделювання було підтверджено правильність роботи алгоритмів кліматичного контролю, охоронної сигналізації та механізму керування режимами *Home, Night* та *Away*. Система своєчасно реагувала на зміни параметрів середовища та забезпечувала виконання визначених правил автоматизації.

Розроблений симулятор може використовуватися для демонстрації принципів побудови систем автоматизації розумного будинку, а також як основа для створення повноцінного програмно-апаратного комплексу. Структура програмного забезпечення дозволяє розширювати функціональність шляхом підключення реальних датчиків, мікроконтролерів та додаткових модулів керування.

Поставлена мета дипломної роботи досягнута й усі завдання, визначені на етапі постановки задачі, виконані в повному обсязі.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Home Automation and Smart Home Technologies [Електронний ресурс]. – Режим доступу: https://www.techtarget.com/iotagenda/definition/home-automation. – Дата доступу: травень 2026.
2. Building Automation Systems Overview [Електронний ресурс]. – Режим доступу: : https://www.automatedbuildings.com. – Дата доступу: травень 2026.
3. Internet of Things Global Standards Initiative [Електронний ресурс]. – Режим доступу: https://www.itu.int/en/ITU-T/gsi/iot. – Дата доступу: травень 2026.
4. Smart Home Market and Technology Overview [Електронний ресурс]. – Режим доступу: https://www.statista.com/topics/2430/smart-homes/. – Дата доступу: травень 2026.
5. Home Assistant Documentation [Електронний ресурс]. – Режим доступу: https://www.home-assistant.io/docs/. – Дата доступу: травень 2026.
6. OpenHAB Documentation [Електронний ресурс]. – Режим доступу: https://www.openhab.org/docs/. – Дата доступу: травень 2026.
7. Eclipse SmartHome Project [Електронний ресурс]. – Режим доступу: https://www.eclipse.org/smarthome/. – Дата доступу: травень 2026.
8. KNX Association – Smart Home and Building Solutions [Електронний ресурс]. – Режим доступу: https://www.knx.org. – Дата доступу: травень 2026.
9. Zigbee Alliance Documentation [Електронний ресурс]. – Режим доступу: https://csa-iot.org/all-solutions/zigbee/. – Дата доступу: травень 2026.
10. IEEE Internet of Things Resource Center [Електронний ресурс]. – Режим доступу: https://iot.ieee.org. – Дата доступу: травень 2026.
11. Cisco Internet of Things Fundamentals [Електронний ресурс]. – Режим доступу: https://www.cisco.com/c/en/us/solutions/internet-of-things/overview.html. – Дата доступу: травень 2026.

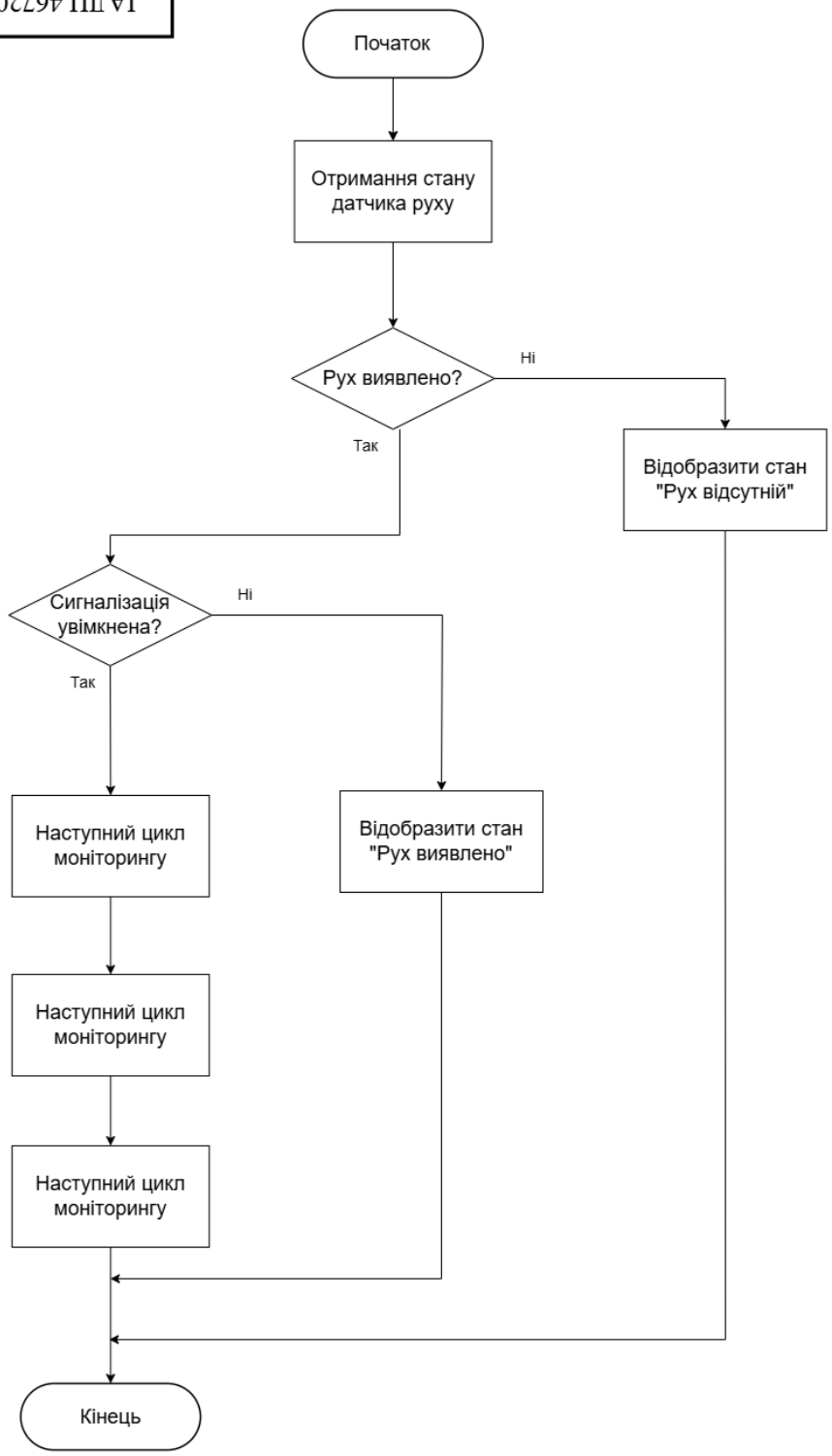
12. Schneider Electric Smart Building Solutions [Електронний ресурс]. – Режим доступу: https://www.se.com/ww/en/work/solutions/buildings/. – Дата доступу: травень 2026.
13. Siemens Building Technologies [Електронний ресурс]. – Режим доступу: https://www.siemens.com/global/en/products/buildings.html. – Дата доступу: травень 2026.

ДОДАТОК А
КОПІЇ ГРАФІЧНОГО МАТЕРІАЛУ

ІАЛЦ.467200.005 Д1

					ІАЛЦ.467200.005 Д1	
Зм.	Арк.	№ докум.	Підпис	Дата		
Розробив		Лисенко В.О.			Функціональні модулі симулятора. Схема структурна.	Літ. / Маса / Масштаб
Перевірив		Щербина О.А.				Арк. / Аркушів
Н. контр.		Клятченко Я.М.				КПІ ім. Ігоря Сікорського, ФПСПМ, КВ-22
Затв.		Романкевич В.О.				

					ІАЛЦ.467200.006 Д2	
Зм.	Арк.	№ докум.	Підпис	Дата		
Розробив		Лисенко В.О.			Вебінтерфейс симулятора. Схема структурна.	Літ. / Маса / Масштаб
Перевірив		Щербина О.А.				Арк. / Аркушів
Н. контр.		Клятченко Я.М.				КПІ ім. Ігоря Сікорського, ФПСПМ, КВ-22
Затв.		Романкевич В.О.				



					ІАЛЦ.467200.007 Д3	
Зм.	Арк.	№ докум.	Підпис	Дата		
Розробив		Лисенко В.О.			Алгоритм роботи сигналізації при спрацюванні датчика руху. Схема алгоритму.	Літ. / Маса / Масштаб
Перевірив		Щербина О.А.				Арк. / Аркушів
Н. контр.		Клятченко Я.М.				КПІ ім. Ігоря Сікорського, ФПСПМ, КВ-22
Затв.		Романкевич В.О.				

Зм.	Арк.	№ докум.	Підпис	Дата	ІАЛЦ.467200.008 Д4		
Розробив		Лисенко В.О.			Алгоритм автоматичного керування температурою. Схема алгоритму.	Літ.	Маса	Масштаб
Перевірив		Щербина О.А.						
						Арк.	Аркушів	
Н. контр.		Клятченко Я.М.				КПІ ім. Ігоря Сікорського, ФПСПМ, КВ-22		
Затв.		Романкевич В.О.						

ДОДАТОК Б
ФРАГМЕНТИ КОДУ

smart_home_server.py

```
from flask import Flask, render_template, jsonify
import sqlite3
import random
from datetime import datetime

app = Flask(__name__)

temperature = 22.0
humidity = 55.0

devices = {
    "living_light": False,
    "kitchen_light": False,
    "bedroom_light": False,
    "alarm": False,
    "door": False,
    "conditioner": False,
    "heater": False
}

POWER = {
    "living_light": 10,
    "kitchen_light": 10,
    "bedroom_light": 10,
    "conditioner": 1500,
    "heater": 2000
}
```

```
def init_database():
    conn = sqlite3.connect("smart_home.db")
    cursor = conn.cursor()

    cursor.execute("""
    CREATE TABLE IF NOT EXISTS logs(
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        event TEXT,
        created_at TEXT
    )
    """)

    conn.commit()
    conn.close()

def add_log(event):
    conn = sqlite3.connect("smart_home.db")
    cursor = conn.cursor()

    cursor.execute(
        "INSERT INTO logs(event, created_at) VALUES (?, ?)",
        (
            event,
            datetime.now().strftime("%Y-%m-%d %H:%M:%S")
        )
    )

    conn.commit()
```

```
    conn.close()


init_database()


@app.route("/")
def dashboard():
    return render_template("dashboard.html")


@app.route("/api/sensors")
def sensors():

    global temperature
    global humidity

    temperature += random.uniform(-0.3, 0.3)
    humidity += random.uniform(-1, 1)

    temperature = max(18, min(30, temperature))
    humidity = max(35, min(80, humidity))

    motion = random.choice([0, 0, 0, 1])

    if temperature > 26:
        devices["conditioner"] = True
        devices["heater"] = False

    elif temperature < 19:
        devices["heater"] = True
        devices["conditioner"] = False
```

```
    else:
        devices["conditioner"] = False
        devices["heater"] = False

    if devices["alarm"] and motion:
        add_log("ALARM TRIGGERED! Motion detected.")

    return jsonify({
        "temperature": round(temperature, 1),
        "humidity": round(humidity, 1),
        "motion": motion,
        "devices": devices
    })

@app.route("/api/device/<device_name>")
def toggle_device(device_name):

    if device_name not in devices:
        return jsonify({"error": "Device not found"}), 404

    devices[device_name] = not devices[device_name]

    add_log(f"{device_name} -> {devices[device_name]}")

    return jsonify(devices)

@app.route("/api/power")
def power():
```

```
    total = 0

    for device, status in devices.items():

        if status and device in POWER:
            total += POWER[device]

    return jsonify({
        "power": total
    })

@app.route("/api/mode/<mode>")
def set_mode(mode):

    if mode == "home":

        devices["alarm"] = False

    elif mode == "night":

        devices["living_light"] = False
        devices["kitchen_light"] = False
        devices["bedroom_light"] = False
        devices["alarm"] = True

    elif mode == "away":

        devices["alarm"] = True
```

```
    else:
        return jsonify({"error": "Unknown mode"}), 404

    add_log(f"{mode} mode activated")

    return jsonify(devices)

@app.route("/api/logs")
def logs():
    conn = sqlite3.connect("smart_home.db")
    cursor = conn.cursor()

    cursor.execute("""
    SELECT event, created_at
    FROM logs
    ORDER BY id DESC
    LIMIT 20
    """)

    data = cursor.fetchall()

    conn.close()

    return jsonify(data)

if __name__ == "__main__":
    app.run(debug=True)
```

dashboard.html

```
<!DOCTYPE html>
<html lang="uk">
<head>
<meta charset="UTF-8">

<title>Smart Home Dashboard</title>

<link rel="stylesheet" href="/static/css/style.css">

<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>

<body>

<div class="container">

<h1>Smart Home Dashboard</h1>

<div class="modes">
<button onclick="setMode('home')">HOME</button>
<button onclick="setMode('night')">NIGHT</button>
<button onclick="setMode('away')">AWAY</button>
</div>

<div class="sensors">
```

```
<h2>Датчики</h2>

<p>Температура:
<span id="temperature">--</span> °C
</p>

<p>Вологість:
<span id="humidity">--</span> %
</p>

<p>Рух:
<span id="motion">--</span>
</p>

</div>

<div class="devices">

<h2>Пристрої</h2>

<button onclick="toggleDevice('living_light')">Вітальня</button>

<button onclick="toggleDevice('kitchen_light')">Кухня</button>

<button onclick="toggleDevice('bedroom_light')">Спальня</button>

<button onclick="toggleDevice('alarm')">Сигналізація</button>

<button onclick="toggleDevice('door')">Двері</button>
```

```
</div>

<div class="status">

<h2>Статус системи</h2>

<p>Кондиціонер:
<span id="conditioner"></span>
</p>

<p>Обігрівач:
<span id="heater"></span>
</p>

<p>Двері:
<span id="doorState"></span>
</p>

<p>Енергоспоживання:
<span id="power"></span> Вт
</p>

</div>

<div class="chart-block">
<h2>Температура</h2>
<canvas id="tempChart"></canvas>
</div>
```

```
<div class="logs">
<h2>Журнал подій</h2>
<ul id="logs"></ul>
</div>

</div>

<script src="/static/js/app.js"></script>

</body>
</html>
```

app.js

```
let labels = [];
let temperatures = [];

const chart = new Chart(
    document.getElementById("tempChart"),
    {
        type: "line",
        data: {
            labels: labels,
            datasets: [{
                label: "Temperature",
                data: temperatures,
                borderColor: "blue"
            }]
        }
    }
);

async function updateSensors() {

    const response =
        await fetch("/api/sensors");

    const data =
        await response.json();
```

```
document.getElementById("temperature").innerText =
    data.temperature;

document.getElementById("humidity").innerText =
    data.humidity;

if(data.motion){

    document.getElementById("motion").innerHTML =
        "<span class='red'>DETECTED</span>";

}else{

    document.getElementById("motion").innerHTML =
        "<span class='green'>NORMAL</span>";
}

document.getElementById("conditioner").innerHTML =
    data.devices.conditioner
    ? "<span class='green'>ON</span>"
    : "<span class='red'>OFF</span>";

document.getElementById("heater").innerHTML =
    data.devices.heater
    ? "<span class='green'>ON</span>"
    : "<span class='red'>OFF</span>";
```

```
    document.getElementById("doorState").innerHTML =
        data.devices.door
        ? "<span class='green'>OPEN</span>"
        : "<span class='red'>CLOSED</span>";

    labels.push(
        new Date().toLocaleTimeString()
    );

    temperatures.push(
        data.temperature
    );

    if(labels.length > 30){

        labels.shift();
        temperatures.shift();
    }

    chart.update();

    updatePower();
}

async function updatePower(){
```

```
    const response =
        await fetch("/api/power");

    const data =
        await response.json();

    document.getElementById("power").innerText =
        data.power;
}

async function toggleDevice(device){

    await fetch("/api/device/" + device);

    loadLogs();
}

async function setMode(mode){

    await fetch("/api/mode/" + mode);

    loadLogs();
}

async function loadLogs(){

    const response =
```

```
        await fetch("/api/logs");

    const logs =
        await response.json();

    let html = "";

    logs.forEach(log => {

        html += `
        <li>${log[1]} : ${log[0]}</li>
        `;
    });

    document.getElementById("logs").innerHTML =
        html;
}

setInterval(updateSensors, 2000);
setInterval(loadLogs, 5000);

updateSensors();
loadLogs();
```

ДОДАТОК В
ПРЕЗЕНТАЦІЯ

Симулятор системи автоматизації розумного будинку з вебінтерфейсом

Виконав : студент IV курсу
групи КВ-22
Лисенко Віталій Олександрович

Актуальність теми

- стрімкий розвиток технологій Smart Home
- поширення автоматизації житлових приміщень
- зростання кількості IoT-пристроїв
- необхідність тестування алгоритмів без реального обладнання
- попит на навчальні та дослідницькі симулятори

Мета та завдання роботи

Мета: розробка симулятора системи автоматизації розумного будинку з вебінтерфейсом.

Завдання:

- дослідити системи Smart Home
- розробити структуру симулятора
- реалізувати моделювання датчиків
- створити алгоритми автоматизації
- розробити вебінтерфейс
- провести тестування системи

Об'єкт і предмет дослідження

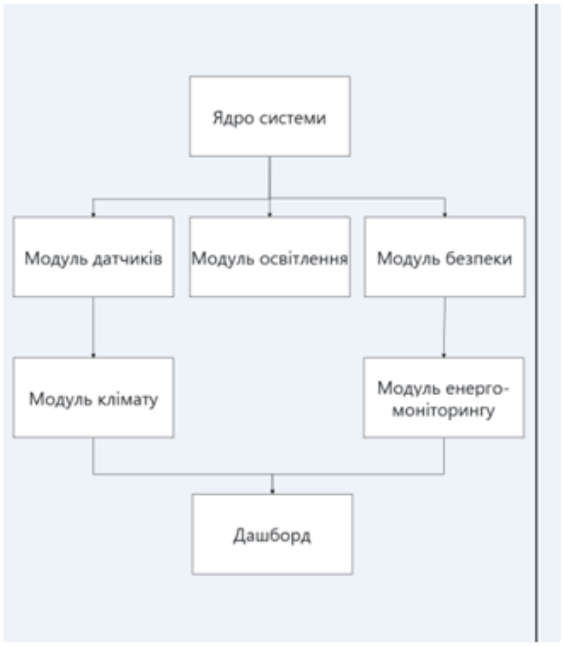
Об'єкт: системи автоматизації розумного будинку.

Предмет: методи програмного моделювання датчиків, виконавчих пристроїв та алгоритмів автоматизації.

Аналіз предметної області

1. Smart Home
2. IoT-технології
3. Клімат-контроль
4. Розумне освітлення
5. Системи безпеки
6. Вебмоніторинг

Структурна схема функціональних модулів симулятора



Структурна схема вебінтерфейсу

Схема алгоритму клімат-контролю

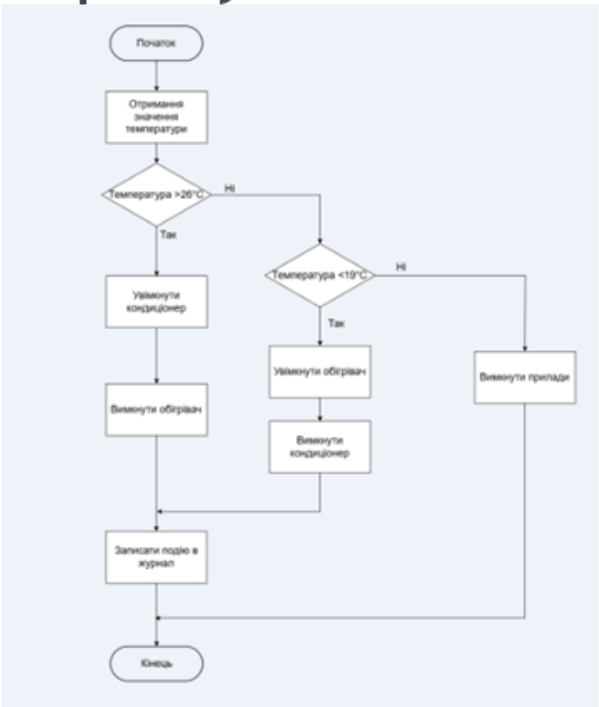
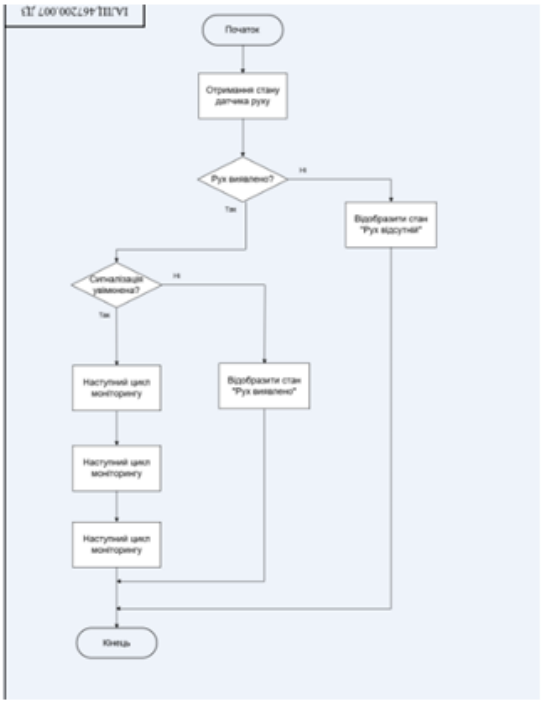


Схема алгоритму охоронної системи



Технології розробки

Серверна частина

- Python
- Flask

База даних

- SQLite

Клієнтська частина

- HTML
- CSS
- JavaScript

Візуалізація

- Chart.js

Архітектура програмного забезпечення

1. Клієнтська частина
2. Flask-сервер
3. SQLite
4. Модуль симуляції
5. Алгоритми автоматизації

Реалізація симуляції датчиків

Реалізовано:

- Температура
- Вологість
- Датчик руху

Особливість:

- Показники змінюються автоматично в режимі реального часу.

Реалізація клімат-контролю

Функції

- Контроль температури
- Керування кондиціонером
- Керування обігрівачем

Пороги

- < 19°C → обігрів
- 26°C → охолодження

Реалізація системи безпеки

Функції

- Датчик руху
- Сигналізація
- Контроль дверей
- Журнал тривог

Режими роботи будинку

Home

- Звичайний режим

Night

- Освітлення вимкнене
- Сигналізація активна

Away

- Контроль безпеки
- Сигналізація активна

Вебінтерфейс

Журнал подій та енергоспоживання

Журнал подій

- 2026-06-06 21:01:19 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 21:00:19 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:58:19 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:56:19 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:41:19 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:40:19 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:38:19 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:36:19 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:31:19 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:29:19 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:27:00 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:26:56 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:26:36 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:26:26 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:26:20 : away mode activated
- 2026-06-06 20:26:17 : home mode activated
- 2026-06-06 20:26:15 : night mode activated
- 2026-06-06 20:26:14 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:26:12 : ALARM TRIGGERED! Motion detected.
- 2026-06-06 20:26:09 : bedroom_light -> False

Відображається:
дії користувача
робота автоматизації
поточне навантаження системи

Практична цінність

1. Навчальний симулятор Smart Home
2. Відпрацювання алгоритмів автоматизації
3. Тестування без фізичного обладнання
4. Основа для подальшого розвитку системи

Висновки

1. Розроблено симулятор розумного будинку
2. Реалізовано вебінтерфейс
3. Створено алгоритми автоматизації
4. Проведено тестування системи
5. Поставлена мета досягнута

Дякую за увагу!