

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЗАПОРІЗЬКА ПОЛІТЕХНІКА»

Факультет комп'ютерних наук та технологій
Кафедра комп'ютерних систем та мереж

Пояснювальна записка

до дипломного проекту (роботи)

магістра

(ступінь вищої освіти)

на тему СИСТЕМА БАГАТОЗАДАЧНОГО КЕРУВАННЯ
ПОБУТОВИМИ ПРИЛАДАМИ

Виконав: студент 2 курсу, групи КНТ-512м
спеціальності _____

123 Комп'ютерна інженерія

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Комп'ютерні системи та мережі

ЖИРКОВА О. С.

(ПРИЗВИЩЕ та ініціали)

Керівник КУДЕРМЕТОВ Р.К.

(ПРИЗВИЩЕ та ініціали)

Рецензент ЄФИМЕНКО М.В.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет Комп'ютерних наук і технологій
Кафедра «Комп'ютерні системи та мережі»
Ступінь вищої освіти магістерський
Спеціальність 123 Комп'ютерна інженерія
(код і найменування)
Освітня програма (спеціалізація) «Комп'ютерні системи та мережі»
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Зав. кафедри Кудерметов Р.К.

“10” грудня 2023 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА

ЖИРКОВА Ольга Сергіївна

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Система багатозадачного керування побутовими
приладами

керівник проєкту (роботи) к. т. н., доцент, КУДЕРМЕТОВ Равіль Камілович
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом вищого навчального закладу від “24” жовтня 2023 року № 400

2. Строк подання студентом проєкту (роботи) 10 грудня 2023 року

3. Вихідні дані до проєкту (роботи) аналоги систем «розумний дім»

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

1) Аналіз предметної області;

2) Теоретична частина;

3) Практична частина.

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

6. Консультанти розділів проєкту (роботи)

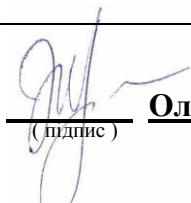
Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-3	КУДЕРМЕТОВ Р. К., доцент		
нормоконтроль	ПОЛЬСЬКА О.В., ст. викл.		

7. Дата видачі завдання 01.10.2023 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Аналіз аналогів системи «розумний дім»	01.10.2023 р.	
2	Постановка завдання	10.10.2023 р.	
3	Розробка вимог до системи	20.10.2023 р.	
4	Розробка загальної структурної схеми системи	25.11.2023 р.	
5	Вибір апаратного забезпечення системи	01.11.2023 р.	
6	Налаштування роботи системи	10.11.2023 р.	
7	Розробка мобільного додатку	20.11.2023 р.	
8	Аналіз результатів роботи системи	01.12.2023 р.	
9	Оформлення пояснювальної записки	10.12.2023 р.	

Студент

Ольга ЖИРКОВА
(ім'я, ПРИЗВИЩЕ)

Керівник проєкту (роботи)

Равіль КУДЕРМЕТОВ
(ім'я, ПРИЗВИЩЕ)

РЕФЕРАТ

ПЗ: 72 с., 33 рис., 25 джерел, 2 додатки.

RASPBERRY PI, PYTHON, WIFI МЕРЕЖА, АВТОМАТИЗАЦІЯ, АНДРОІД ДОДАТОК, КЕРУВАННЯ ЗА ДОПОМОГОЮ СЦЕНАРІЇВ, РОЗУМНИЙ БУДИНОК

Об'єкт розробки – система багатозадачного керування побутовими приладами.

Мета роботи – розробка системи управління автоматизацією розумного будинку і створення зручного мобільного додатка-інтерфейсу до нього.

У першій частині було проведено детальний огляд та аналіз різних наявних рішень у сфері розумних будинків, а також проведено категоризацію технічної структури таких систем. Визначено ключові вимоги.

У другій частині роботи було виконано всебічне дослідження, що включає в себе зіставлення та оцінку поточних технологій для передачі даних, аналіз обладнання для центральних систем розумного будинку, а також вивчення відповідного програмного забезпечення для баз даних і серверів. Проведено аналіз ефективних методів та інструментів для створення серверної частини і частини, призначеної для користувача.

Третій розділ присвячено розробці багатозадачного керування побутовими приладами. Створено серверну частину на Python для Raspberry Pi, налаштовано GPIO для контролю над твердотільними реле, проведено інтеграцію з Bluetooth і Wi-Fi, а також розроблено мобільний застосунок на Android для забезпечення взаємодії користувача з системою. Проведено експериментальні випробування всіх компонентів системи, що підтверджує працездатність розробленої системи розумного будинку.

ABSTRACT

Explanatory note: 72 p., 33 pictures, 25 references, 2 appendices

RASPBERRY PI, PYTHON, WIFI NETWORK, AUTOMATION, ANDROID APPLICATION, SCRIPT CONTROL, SMART HOME

The object of development is a multitasking control system for household appliances.

The goal of the work is the development of a smart home automation control system and the creation of a convenient mobile application-interface for it.

In the first part, a detailed review and analysis of various available solutions in the field of smart buildings was carried out, as well as a categorization of the technical structure of such systems. The key requirements are defined.

In the second part of the work, a comprehensive study was performed, which includes comparison and evaluation of current technologies for data transmission, analysis of hardware for central smart home systems, and study of relevant software for databases and servers. An analysis of effective methods and tools for creating the server part and the part intended for the user was carried out.

The third section is devoted to the development of multitasking control of household appliances. Created a backend in Python for Raspberry Pi, configured GPIO to control solid-state relays, integrated with Bluetooth and Wi-Fi, and developed a mobile application on Android to provide user interaction with the system. Experimental tests of all system components were conducted, which confirms the functionality of the developed smart home system.

ЗМІСТ

Вступ.....	7
1 Аналіз предметної області.....	8
1.1 Загальний огляд системи «Розумний дім».....	8
1.2 Історія домашньої автоматизації: від електроприладів до розумних будинків ..	9
1.3 Аналіз готових систем та підходів розробки розумного будинку	11
1.4 Архітектурна будова обладнання в системах розумного будинку	17
1.5 Постановка завдання.....	19
2 Теоретична частина	21
2.1 Основні вузли функціонування розумного будинку	22
2.2 Технології передачі даних	24
2.3 Апаратні платформи	28
2.4 Програмні інструменти та платформи	35
3 Практична частина.....	39
3.1 Структурна схема системи	39
3.2 Вибір пристрою й програмних засобів.....	40
3.3 Електрична схема системи керування побутовими приладами.....	42
3.4 Налаштування мікрокомп'ютера Raspberry Pi 4B+	43
3.5 Налаштування віддаленого середовища розробки через SSH.....	46
3.6 Розробка програмного забезпечення системи	48
3.7 Функціонування системи	51
3.8 Результат роботи системи	57
Висновки	62
Перелік джерел посилання	63
Додаток А Головний сервер системи.....	66
Додаток Б Головне меню.....	68

ВСТУП

В епоху цифровізації дедалі більше уваги приділяється розвитку інформаційних систем, здатних синхронізувати розумне побутове обладнання, інженерні пристрої та системи управління будівлями в єдину, керовану людиною і машинами мережу.

Прикладом такої інтеграції є концепція «розумного будинку», де технології спрямовані на підвищення комфорту і зручності в повсякденному житті. У таких системах можливе автоматичне керування освітленням, шторами, вікнами і температурою для створення оптимальної атмосфери для відпочинку і сну, а також заварювання кави та розігрів їжі вранці, відстеження фізичної активності і навіть ранкове привітання.

Розробка цих систем вимагає вирішення безлічі завдань, включно з інтеграцією різних пристроїв в єдину інформаційну основу та оптимізацією процесів автоматизації. Це породжує необхідність глибокого дослідження наявних систем і розроблення теоретичної основи для їх створення.

Таким чином, вплив комп'ютерних технологій на наше життя виявився революційним, перетворюючи нашу повсякденність і взаємодію з технологіями. Розвиток концепції розумних будинків - це тільки один із багатьох прикладів цієї еволюції. Необхідність у постійних дослідженнях і розробках для поліпшення цих систем та їхньої доступності для користувачів підкреслює актуальність обраної теми для магістерської роботи.

Метою цієї роботи є детальне дослідження і теоретичне обґрунтування розроблення системи багатозадачного керування побутовими приладами, в рамках концепції розумного будинку, з особливим акцентом на створення зручного інтерфейсу для користувача. Головне завдання - розробити систему, яка не тільки ефективна і функціональна, а й максимально зручна для користувача, даючи йому повною мірою використовувати можливості розумного будинку для поліпшення якості життя.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Наведено огляд де розглядається концепція розумного будинку, історія її розвитку та аналізуються різноманітні рішення, запропоновані за минулі десятиліття для задоволення різних потреб. Основна увага приділяється історичному контексту технологій розумного будинку, а також порівнянню безлічі наявних систем з метою виявлення їхніх сильних і слабких сторін.

Огляд завершується формулюванням ключових критеріїв, яким має відповідати ефективна інформаційна система розумного будинку. Також пропонується стратегія, спрямована на створення системи багатозадачного управління побутовими приладами, яка поєднує в собі доступність за вартістю та високу функціональність.

1.1 Загальний огляд системи «Розумний дім»

Концепція «розумного будинку» перетворює наш побут, надаючи автоматизований контроль над ключовими системами життєзабезпечення, підвищуючи їхню ефективність і надійність, а також значно покращуючи рівень комфорту для мешканців. В епоху розширення ринку розумних пристроїв, включно з розумним освітленням, автономними пилососами, системами потокового передавання аудіо- та відеосигналів, електронними замками та іншим, "розумний дім" інтегрує всі ці елементи в єдину, автоматизовану мережу.

Системи, подібні до Google Smart Home або Xiaomi Smart Home, надають всеосяжні рішення для управління домашньою екосистемою. Однак їхній основний недолік полягає в обмеженій сумісності з пристроями від інших виробників, що знижує гнучкість і масштабованість системи. Крім того, ці

системи часто не надають можливості налаштування складних сценаріїв взаємодії пристроїв, наприклад, автоматичного вимкнення електроживлення під час спрацьовування датчика диму або коли останній мешканець залишає будинок.

«Розумний будинок» розробляють і налаштовують спеціалізовані компанії для задоволення індивідуальних потреб мешканців. Вбудовані програми та алгоритми дають змогу керувати домашніми системами централізовано, адаптуючись до поточних умов і вподобань користувача. Також розумні прилади можуть бути підключені до загальної мережі Plug and Play (PnP), забезпечуючи легкість інтеграції та доступ до Інтернету.

У розумних системах пристрої зазвичай оснащені вбудованими інтерфейсами для взаємодії із зовнішнім середовищем згідно зі встановленими протоколами, що забезпечує гармонійну взаємодію між різними елементами системи. Однак для деяких спеціалізованих функцій і компонентів системи розумного будинку можуть знадобитися додаткові пристрої, як-от під'єднання до електромережі або резервні системи для випадків збоїв. Ці аспекти обладнання виходять за рамки цього дослідження.

У підсумку, система розумного будинку є ефективним рішенням для управління домашніми системами, забезпечуючи при цьому підвищений рівень комфорту для мешканців. Централізація розумних пристроїв в єдиній мережі спрощує управління і підвищує загальну якість життя.

1.2 Історія домашньої автоматизації: від електроприладів до розумних будинків

У 21 столітті будівельний сектор зазнав значних змін завдяки технологічному прогресу. Основоположну роль у цьому зіграли доступність електроенергії та розвиток інформаційних технологій. Це сприяло появі

розумних будинків, де автоматизація побутових завдань і обмін інформацією між людьми, пристроями та мережами стали новою нормою. Інтеграція цих технологій радикально змінила спосіб життя, роблячи його більш комфортним і ефективним.

У період з 1915 по 1920 роки, через дефіцит робочої сили, побутові електроприлади, як-от пилососи та швейні машини, стали невід'ємною частиною домашніх господарств. Ці інновації звільнили час для дозвілля, змінюючи традиційні підходи до домашнього господарства.

Між 1920 і 1940 роками, в епоху феміністичної революції, жінки отримали змогу приділяти менше часу домашнім справам завдяки новим технологіям, таким як пральні машини і пилососи, які спростили і прискорили виконання повсякденних завдань.

Під час Другої світової війни відсутність чоловіків на фронті призвела до зміни робочої сили, і домогосподарки взяли на себе нові обов'язки. У цей період увагу технологічного прогресу було зосереджено на військових потребах, що тимчасово сповільнило розвиток побутових технологій.

З 1945 по 1959 роки, після війни, спостерігалось відродження інтересу до домашньої автоматизації. Це було пов'язано з дефіцитом робочої сили і прагненням людей полегшити свої побутові завдання. У цей період відбулися значні зміни в дизайні будинків, які стали включати сучасні технології, такі як холодильники та електричні плити.

У 1960-1970-ті роки, в епоху капіталізму, ринок побутової техніки активно розвивався, пропонуючи споживачам різноманітні пристрої, що спрощують домашні справи.

У 1980-1990-х роках використання електроприладів стало буденним явищем у багатьох країнах. У Великій Британії, Японії та Китаї поширення кольорових телевізорів, систем відеоспостереження та розважальних пристроїв, як-от PlayStation, віддзеркалило зростаючу доступність електронних пристроїв.

З 1990-х до початку 2000-х Всесвітня мережа справила значний вплив на спосіб життя. Інтернет привніс нові послуги, такі як онлайн-магазини і

банківська справа, а також сприяв поширенню доступу до інформації. До 2000 року понад 40% американських домогосподарств мали доступ до Інтернету.

На початку 21 століття зростання Інтернету призвело до значних змін у багатьох сферах, включно з комунікацією, торгівлею та освітою. Термін "Інтернет речей" (IoT), введений у 1999 році, описує інтеграцію розумних об'єктів у глобальну інформаційну архітектуру. Це не тільки перетворило будинки на розумні будинки, а й сприяло створенню цифрових міст. IoT значно спростив обмін товарами і послугами, надаючи глибокий вплив на наше життя і взаємодію з технологіями [1].

1.3 Аналіз готових систем та підходів розробки розумного будинку

У процесі розробки пристроїв і продуктів, одним із ключових методів прискорення є аналіз і адаптація вже наявних рішень. Це підхід, заснований на вивченні та застосуванні готових систем, які виконують аналогічні або схожі функції. Такий підхід не тільки економить час і ресурси, а й сприяє створенню модульних продуктів, які легко обслуговуються. Використання готових апаратних і програмних платформ може значно спростити розробку, забезпечуючи гнучкість і масштабованість кінцевого продукту. Основна мета систем "розумного будинку" - забезпечення комфорту та автоматизація домашніх процесів, що вимагає інтеграції різноманітних модулів і пристроїв.

Ajax Systems являє собою комплексну систему автоматизації «розумного будинку» і безпеки, розроблену українською компанією (рис. 1.1). Ця система швидко набирає популярності завдяки своєму інноваційному підходу до інтеграції різноманітних пристроїв безпеки та розумного будинку. Ґрунтуючись на сучасних технологіях, Ajax Systems пропонує рішення, які поєднують у собі надійність, зручність використання та гнучкість у масштабуванні.



Рисунок 1.1 – Пристрої системи Ajax Systems

Основною технологічною особливістю систем Ajax є використання власної розробки бездротової технології Jeweller, яка забезпечує двосторонній радіозв'язок між пристроями на відстані до 2000 метрів [2]. Це дає змогу мінімізувати необхідність у дротових з'єднаннях, спрощуючи процес встановлення та розширення системи.

Пристрої Ajax характеризуються високою довговічністю і здатністю функціонувати в різних умовах. Вони оснащені датчиками з тривалим терміном служби батареї та механізмами захисту від несанкціонованого доступу і злому. Більшість пристроїв також мають вбудовані акумулятори для забезпечення безперервної роботи в разі відключення електроенергії.

У контексті користувацького досвіду, система Ajax пропонує мобільний застосунок для управління і моніторингу, що спрощує взаємодію з системою і підвищує її доступність для широкого кола користувачів. Однак, система залежить від центрального контролера (Hub), що обмежує автономність окремих датчиків і може становити певні обмеження в разі збоїв у роботі центрального вузла.

Незважаючи на широкий спектр функціональності, система Ajax не включає власні камери відеоспостереження, хоча передбачає можливість інтеграції із зовнішніми пристроями. Це може бути розглянуто як обмеження в контексті повноцінних систем відеоспостереження.

Система розумного дому BroadLink (рис. 1.2). BroadLink китайський виробник систем розумного будинку, посідає значуще місце на ринку автоматизації житла. Ця система об'єднує широкий спектр цифрових пристроїв, призначених для управління побутовою технікою, освітлювальними, енергетичними, охоронними та іншими системами в будинку. Особливістю BroadLink є можливість роботи кожного елемента як в автономному режимі, так і в рамках взаємодії з іншими пристроями екосистеми.



Рисунок 1.2 – Зовнішній вигляд пристроїв екосистеми BroadLink

BroadLink пропонує різноманітність датчиків (вологості, температури, освітлення, шуму, забруднення повітря) і пристроїв, включно з камерами відеоспостереження [3]. Управління системою здійснюється через Wi-Fi, що дає змогу контролювати пристрої з будь-якої точки світу. Відмінною рисою, на відміну від системи Ajax, є відсутність необхідності в центральному хабі, що забезпечує автономність роботи кожного пристрою. Це робить BroadLink привабливим вибором для споживачів, які шукають надійну й економічно вигідну систему розумного будинку.

Простота встановлення та налаштування BroadLink швидко завоювала популярність серед користувачів, що сприяло його високому рейтингу на ринку систем розумного будинку. Автономність пристроїв збільшує надійність системи, оскільки відмова одного пристрою не впливає на роботу інших. Демократична вартість, починаючи від 200 доларів, робить систему доступною для широкого кола споживачів. Гнучкість екосистеми, що дає змогу легко додавати та видаляти пристрої, спрощує масштабування й адаптацію системи під індивідуальні потреби користувача. В Україні доступні такі пристрої, як захисні реле, розумні розетки двох видів (одна з енергомоніторингом, інша без нього) і розумний датчик відкриття Broadlink S1D.

Система BroadLink дає змогу від'єднувати до своєї екосистеми пристрої, які не є частиною їхнього власного бренду, але це пов'язано з деякими обмеженнями і залежить від конкретних характеристик і протоколів зв'язку цих пристроїв. BroadLink відомий своїми універсальними пультами дистанційного керування, які можуть керувати різними пристроями, що використовують інфрачервоні або радіочастотні сигнали. Це означає, що пристрої, які підтримують ці стандарти зв'язку, можуть бути інтегровані в екосистему BroadLink.

Однак важливо зазначити, що інтеграція сторонніх пристроїв може потребувати додаткових налаштувань і не завжди гарантує повну сумісність усіх функцій. Наприклад, деякі просунуті функції розумного будинку можуть не підтримуватися під час під'єднання пристроїв, не розроблених спеціально для екосистеми BroadLink.

Незважаючи на безліч переваг, система BroadLink має низку недоліків. Обмежена дальність дії сигналу до 50 метрів може створювати проблеми у великих будинках або будівлях. Відсутність резервного живлення хаба може стати проблемою в разі відключення електроенергії. Крім того, обмежена функціональність пульта, який працює тільки на приймання сигналів, може обмежувати його використання в деяких сценаріях. Ці недоліки можуть

впливати на загальну зручність та ефективність використання системи в певних умовах.

Ще одним прикладом несумісності модулів з іншими виробниками є система розумного будинку від Samsung SmartThings вона являє собою інтегровану систему для автоматизації будинку, що забезпечує централізоване управління різними розумними пристроями, включно з освітленням, термостатами, замками і побутовою технікою. Ця система доступна через універсальний застосунок, сумісний із більшістю операційних систем, як показано на рисунку 1.3. Основою роботи SmartThings є центральний хаб, який пов'язує всі розумні пристрої в домі та синхронізує їх із мобільним додатком користувача. Це дає змогу створювати різні автоматизовані сценарії, як-от увімкнення світла під час входу в кімнату або адаптація температури залежно від часу доби.

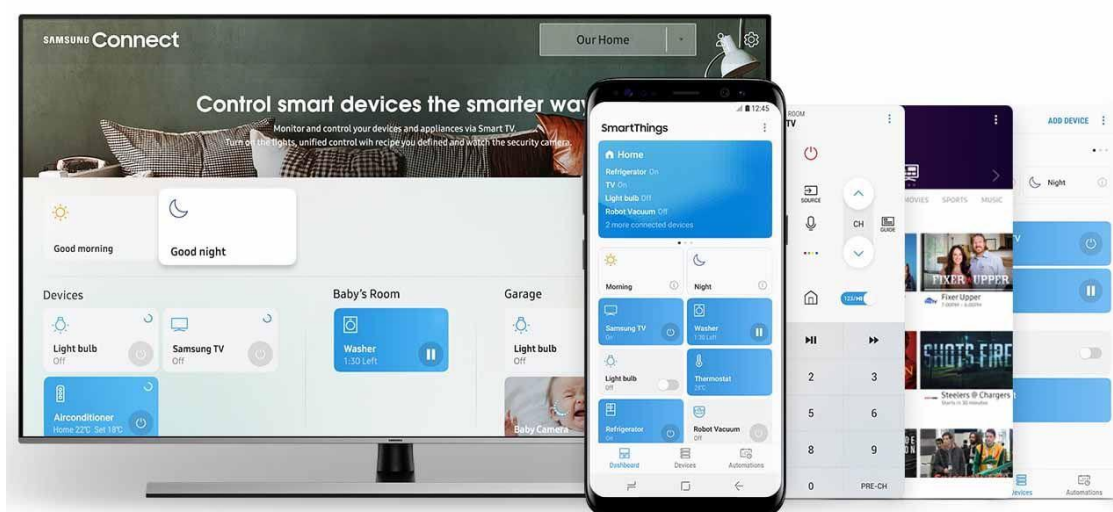


Рисунок 1.3 – Інтерфейс програми SmartThings на різних платформах

Однією з ключових переваг Samsung SmartThings є інтуїтивно зрозумілий застосунок, що спрощує налаштування та управління пристроями, роблячи систему доступною навіть для тих, хто не володіє глибокими технічними знаннями. Крім того, система має просунуті функції безпеки, включно з

двофакторною автентифікацією та зашифрованою передачею даних, забезпечуючи надійний захист особистої інформації та управління пристроями.

Проте, Samsung SmartThings має деякі обмеження. Система може бути несумісна з пристроями старих моделей або продуктами менш відомих брендів, що обмежує її універсальність [4]. Крім того, початкові витрати на придбання хаба SmartThings і сумісних пристроїв можуть бути значними, особливо для тих, хто тільки починає формувати свою систему домашньої автоматизації. Ці фактори слід враховувати під час планування впровадження системи SmartThings у домашнє середовище.

Ще одним об'єктом дослідження стала система розумного дому Fibaro Home Center (рис. 1.4). Ця система розроблена польською компанією Fibaro, являє собою комплексну систему домашньої автоматизації, що характеризується значною гнучкістю в конфігурації та інтеграції. Розробка привертає увагу дослідників і практиків у галузі домашньої автоматизації завдяки своєму зручному інтерфейсу і здатності до ефективної інтеграції з різними пристроями. Відмінною рисою процесу встановлення є його ненав'язливий характер, що дає змогу мінімізувати візуальний вплив на інтер'єр завдяки використанню компактних і естетично привабливих модулів і датчиків.



Рисунок 1.4 – Система Fibaro Home Center

Основна перевага системи Fibaro Home Center полягає в її просунутому графічному інтерфейсі, який забезпечує зручність управління і високий рівень безпеки. Система побудована на модульному принципі, що дає змогу користувачам гнучко налаштовувати й розширювати систему, додаючи нові датчики та модулі відповідно до мінливих потреб. Компанія Fibaro забезпечує широку підтримку клієнтів, маючи представництва в 106 країнах, включно з Україною, і підтримує голосове управління, що збільшує зручність використання системи.

Проте, система Fibaro Home Center має низку обмежень. Одним з основних є висока вартість початкового комплекту, що може бути бар'єром для певних категорій користувачів [5]. Крім того, система обмежена в сумісності, оскільки працює виключно з пристроями, що використовують протокол Z-Wave. Це обмеження сумісності може звужувати спектр доступних для інтеграції пристроїв. Також слід зазначити, що налаштування складних сценаріїв автоматизації може вимагати спеціалізованих знань і досвіду, що може бути перешкодою для деяких користувачів. Незважаючи на ці обмеження, Fibaro Home Center продовжує знаходити застосування в якості надійного, гнучкого та інтуїтивно зрозумілого рішення в галузі домашньої автоматизації.

1.4 Архітектурна будова обладнання в системах розумного будинку

Дослідження наявних систем розумного будинку виявляє три основні підходи до зв'язку між модулями та центральною системою: централізовані, децентралізовані та змішані.

Централізовані системи функціонують через центральний вузол, до якого підключаються компоненти через спеціалізовану мережу. Усі налаштування і зв'язки між модулями централізовано управляються і зберігаються на цьому вузлі (рис. 1.5). Ці системи забезпечують універсальне управління і

підключення пристроїв різних виробників, але в разі відключення центрального вузла весь системний функціонал припиняє роботу, якщо не передбачено резервних механізмів.

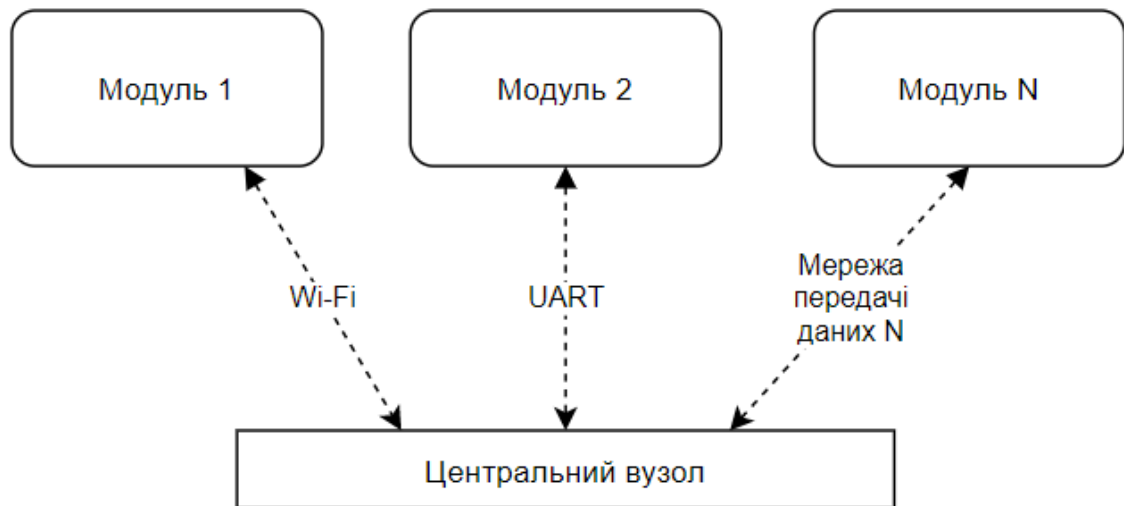


Рисунок 1.5 – Архітектура централізованої системи

Децентралізовані системи відрізняються відсутністю центрального вузла. Тут кожен модуль незалежно виконує завдання, що забезпечує високу надійність і гнучкість у модернізації системи (рис. 1.6). Однак складність інтеграції компонентів і розробки складних сценаріїв може бути проблематичною.

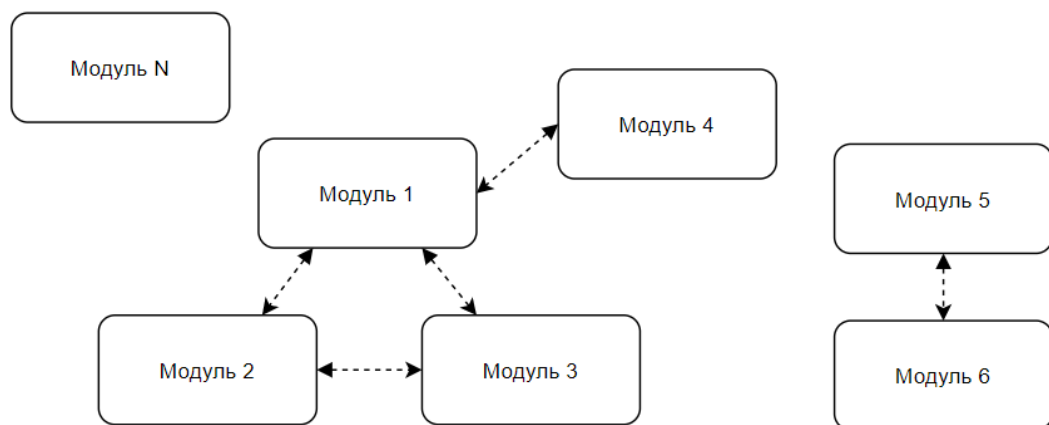


Рисунок 1.6 – Децентралізована система

Змішані системи комбінують елементи обох підходів, використовуючи допоміжні контролери для управління окремими групами функцій, що координуються з центральним вузлом (рис. 1.7). Цей підхід підвищує надійність, але збільшує витрати на розробку та обслуговування. У разі вимкнення центрального вузла, базові функції зберігаються, але складніші сценарії можуть бути порушені.

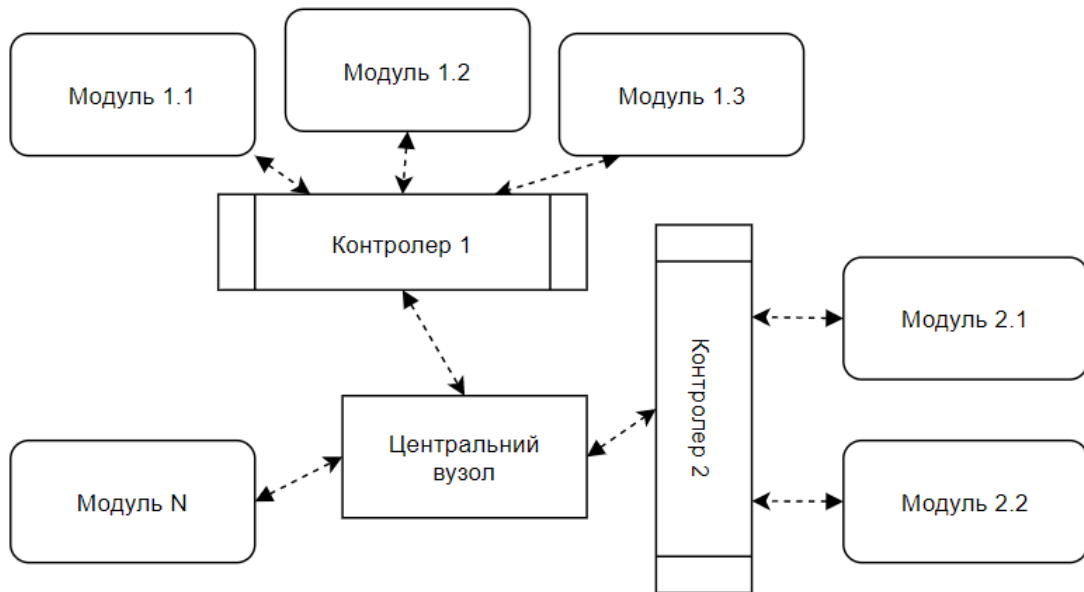


Рисунок 1.7 – Архітектура змішаної системи

1.5 Постановка завдання

Метою даної дипломної роботи є розробка багатозадачної системи керування побутовими приладами в рамках концепції "розумного будинку". Дослідження поточних систем розумного будинку показало, що вони часто стикаються з проблемами сумісності з різними пристроями. Це підкреслює необхідність створення централізованої системи, здатної інтегрувати нове обладнання без необхідності модифікації програмного коду.

Ключовим елементом такої системи має стати центральний хаб, який об'єднує апаратні та програмні компоненти, керуючи пристроями через різні сценарії та події. Цей хаб має також надавати користувацький інтерфейс для активації, модифікації, налаштування та вимкнення сценаріїв.

Для реалізації цієї системи пропонується такий план дій:

- розробка уніфікованого методу доступу до пристроїв, класифікуючи їх за внутрішніми станами і можливими діями;
- створення гнучкого механізму інтеграції нових пристроїв, що дає змогу додавати їх у систему через настроюванні драйвери та конфігурації, які не потребують змін в основному програмному коді;
- розробка просунутої сценарної системи управління, що базується на попередніх дослідженнях і розробках у цій галузі;
- створення інструментів для розроблення, редагування та налаштування користувацьких скриптів управління;
- розроблення інтуїтивно зрозумілого і зручного користувацького інтерфейсу для взаємодії з системою розумного будинку;
- створення високорівневої сценарної системи керування приладами з використанням попередніх розробок;
- розробка системи створення, редагування та налаштування скриптів;
- створення зручного інтерфейсу для системи розумний дім.

Таким чином, метою роботи є створення гнучкої, масштабованої та легко інтегрованої системи розумного будинку, яка за задумкою здатна адаптуватися до різноманітних потреб користувачів і швидко включати в себе нові технології та пристрої.

2 ТЕОРЕТИЧНА ЧАСТИНА

У цій частині дипломної роботи здійснюється глибокий аналіз сучасних технологій у контексті створення багатозадачної системи управління побутовими приладами, у рамках концепції «розумного будинку». Дослідження починається з вивчення різноманітних методів передавання даних, з акцентом на найперспективніші та найширше використовувані підходи на даний момент.

Далі, дослідження переходить до аналізу апаратних платформ, які можуть бути використані в якості основи для центральної системи розумного будинку. Це важливий етап для визначення потенціалу та обмежень різних пристроїв у контексті їхнього використання в проєкті.

Наступний крок - аналіз програмного забезпечення для баз даних. Розглядаються як класичні, так і новітні системи управління базами даних, що дає змогу вибрати найбільш підходящий інструмент для оброблення та зберігання даних, які генеруються системою розумного будинку.

Важливою частиною дослідження є аналіз програмного забезпечення для серверної частини проєкту, яке забезпечує координацію і взаємодію всіх компонентів системи.

Дослідження завершується оглядом програмних платформ та інструментів для розроблення клієнтської частини застосунку, що взаємодіятиме з серверною частиною системи.

Основна мета цього дослідження - виявити найбільш підходящі технології для створення комплексної та ефективної системи розумного будинку, що забезпечує надійне і зручне управління побутовими приладами.

2.1 Основні вузли функціонування розумного будинку

Розумні будинки являють собою гнучкі системи, здатні адаптуватися до різних потреб і переваг користувачів [6]. Відсутність уніфікованого стандарту дає змогу кожному будинку бути унікальним у своєму роді. Наприклад, в одному випадку може бути достатньо базових систем безпеки, в той час як інший будинок може бути оснащений більш складними мультимедійними та автоматизованими функціями. Це розмаїття вимагає універсального програмного забезпечення, здатного інтегрувати і керувати різними пристроями і системами.

Центральним елементом розумного будинку є його програмне забезпечення, яке координує роботу всіх компонентів системи [7]. Це ПЗ обробляє сигнали від датчиків, ініціює відповідні дії, керує процесами і забезпечує комунікацію між різними елементами системи. Для його роботи використовується спеціалізована апаратна платформа, приклад цієї концепції показано на рисунку 2.1.

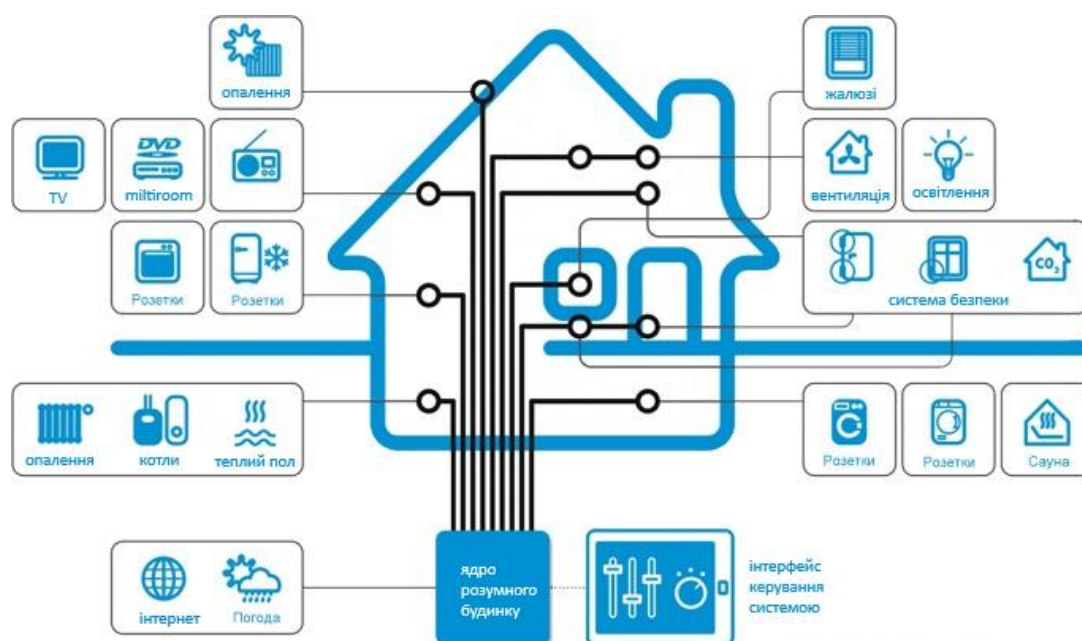


Рисунок 2.1 – Інфографік змішаної системи

У царині прихованого контролю основна увага приділяється системам безпеки та моніторингу навколишнього середовища. Це охоплює датчики руху, відеоспостереження, системи пожежної безпеки, а також датчики температури, вологості та якості повітря. Ці системи зазвичай непомітні і не впливають на загальний дизайн інтер'єру [8].

Підхід комплексного управління охоплює інтеграцію систем управління інженерними системами та побутовою технікою. Це може охоплювати елементи управління освітленням, клімат-контролем, розумними шторами і вікнами, а також побутовою технікою. Цей процес інтеграції досі представляє певні труднощі і може обмежувати вибір конкретних пристроїв.

У контексті ілюзорного мультимедіа особлива увага приділяється пристроям, які візуально або аудіо трансформують простір. Прикладами можуть слугувати проектори, голографічні системи та аудіо/відео трансляції з можливістю багатокімнатного відтворення.

Динамічний підхід передбачає поступове і гнучке додавання різних компонентів у систему відповідно до бажань користувача. Це може включати в себе унікальні функції, такі як автоматизовані меблі або персоналізовані сценарії управління. Цей підхід вимагає більшої гнучкості з боку програмного забезпечення і найчастіше створюється на замовлення.

Кожен із цих підходів має свої особливості та може перетинатися з іншими, створюючи унікальні та багатофункціональні системи розумного будинку. Важливо зазначити, що для ефективної роботи такої системи потрібне гнучке і потужне програмне забезпечення, здатне інтегрувати різноманітні компоненти і пристрої.

2.2 Технології передачі даних

У процесі розроблення системи багатозадачного керування побутовими приладами, що є концепцією розумного будинку, було проведено глибокий аналіз сучасних технологій передавання даних. Ці технології важливі для забезпечення надійної та ефективної комунікації між компонентами системи. Аналіз охоплював широкий спектр апаратних платформ і програмних інструментів, необхідних для створення і підтримки як серверної, так і клієнтської частини розумного будинку.

Найпопулярнішою технологією передачі даних на сьогоднішній день є бездротова технологія Wi-Fi, заснована на стандартах IEEE 802.11. Ця технологія надає можливість пристроям підключатися до мережі без фізичних проводів, використовуючи роутери або точки доступу. Для забезпечення безпеки передавання даних Wi-Fi застосовуються різні методи шифрування, включно з WPA2 і WEP, що важливо для захисту персональних даних користувачів або екосистеми дипломного проєкту [9].

Wi-Fi технологія була винайдена корпорацією NCR/AT&T в Нідерландах в 1991 році і спочатку розроблялася для мобільних обчислювальних пристроїв, таких як ноутбуки. Відтоді вона набула широкого поширення і зараз використовується в безлічі пристроїв, включно зі смартфонами, телевізорами, колись DVD-плеєрами та цифровими камерами.

Wi-Fi технологія підтримує два основні режими зв'язку: точка доступу до клієнтського з'єднання і з'єднання клієнта з клієнтом. Ці режими забезпечують гнучкість у налаштуванні мережевих з'єднань усередині домашнього середовища, що особливо важливо для систем розумного будинку, де безліч пристроїв мають взаємодіяти між собою.

Крім того, Wi-Fi технологія забезпечує високу швидкість під'єднання до Інтернету та мережеве з'єднання без використання будь-яких кабелів або дротів, що робить її кращим вибором для багатьох користувачів. У рамках

систем розумного будинку ця технологія дає змогу локальним мережам працювати без кабелів і дротів, що значно спрощує встановлення та підтримку таких систем.

Важливим аспектом Wi-Fi є його радіус дії, який залежить від довкілля і може досягати 30-45 метрів від маршрутизатора. Це забезпечує достатнє покриття для більшості домашніх умов. Однак для збільшення зони покриття можуть використовуватися розширювачі сигналу.

Наступною популярною технологією передачі даних у контексті розумних будинків є Bluetooth. Ця технологія, спочатку розроблена як бездротовий метод передачі даних на короткі відстані, знаходить своє застосування в різних аспектах автоматизації будинку. На відміну від WiFi, який орієнтований на забезпечення широкосмугового доступу до інтернету, і мережі між пристроями, Bluetooth фокусується на створенні локальних мереж між пристроями для управління різними модулями розумного будинку.

Bluetooth працює на основі пікомереж, які можуть містити до восьми пристроїв: один ведучий і до семи ведених. Ведучий пристрій контролює мережу, керуючи каналом зв'язку і трафіком між підключеними пристроями. Це дає змогу ефективно координувати роботу різних елементів системи розумного будинку, як-от датчики, контролери та виконавчі пристрої.

Однією з ключових особливостей Bluetooth є його здатність до частотного хопінгу, що підвищує стійкість мережі до перешкод і забезпечує більш безпечне з'єднання. Кожен пристрій у пікомережі синхронізує свої перемикання частот із провідним пристроєм, що мінімізує ризик перетину сигналів і втрати даних [10].

У контексті розумного будинку, Bluetooth може використовуватися для управління освітленням, термостатами, системами безпеки та іншими пристроями. Наприклад, автомобільна система Bluetooth може інтегруватися з домашньою автоматизацією для активації певних функцій під час наближення автомобіля до будинку. Аналогічно, Bluetooth-принтери та веб-камери можуть

бути частиною ширшої системи розумного будинку, забезпечуючи зручність і підвищену функціональність.

Технічні характеристики Bluetooth, включно з різними рівнями протоколів, як-от LMP (Link Manager Protocol) і L2CAP (Logical Link Control and Adaptation Protocol), забезпечують гнучкість і масштабованість під час інтеграції з різними пристроями. Це дає змогу створювати складні, але керовані системи, в яких кожен пристрій може бути точно налаштований для виконання певних функцій у межах розумного будинку.

Безпека в Bluetooth-мережах також є ключовим аспектом, особливо в контексті розумного будинку. Заходи безпеки, такі як часта зміна радіочастот і можливість налаштування рівнів доступу, допомагають захистити систему від несанкціонованого доступу та атак. Такі атаки, як bluesnarfing і blue bugging, хоча і становлять потенційну загрозу, можуть бути нейтралізовані за допомогою правильної конфігурації та оновлення програмного забезпечення.

Версії Bluetooth, починаючи від v1.0 і до останніх розробок, як-от v4.0 з низьким енергоспоживанням, надають широкий спектр можливостей для розробників модулів і користувачів розумних будинків. Ці версії постійно еволюціонують, пропонуючи поліпшені характеристики в плані швидкості передачі даних, дальності дії та енергоефективності.

Технологія Bluetooth може бути невід'ємним компонентом в арсеналі технологій розроблення власної системи розумного будинку, забезпечуючи гнучкість у деяких рішеннях, безпеку та зручність в управлінні безліччю пристроїв. Його здатність до швидкої та стандартизованої комунікації на коротких відстанях робить його непоганим рішенням для створення інтелектуального та взаємопов'язаного домашнього середовища.

У екосистемі розумного будинку, поряд із бездротовими технологіями, як-от Wi-Fi і Bluetooth, широко використовують і дротові комунікаційні інтерфейси, серед яких виділяються I²C, UART і SPI.

I²C, розроблений Philips Semiconductors, є багатомастерним серійним інтерфейсом, який використовує дві лінії: SDA для даних і SCL для тактування. Цей протокол забезпечує координовану передачу даних між близько розташованими пристроями, даючи змогу під'єднувати безліч пристроїв у мережу з високою швидкістю передачі даних. Це робить його кращим вибором для складних систем розумного будинку, де потрібна ефективна взаємодія між безліччю компонентів.

UART, або Універсальний асинхронний приймач-передавач, служить основою для послідовної передачі даних в електронних системах. Він перетворює паралельні дані з CPU в послідовну форму для передачі і назад після прийому. Цей інтерфейс додає стартовий і стоповий біти до кожного байта даних, що забезпечує цілісність переданої інформації. Простота реалізації UART з використанням лише двох дротів (TX і RX) робить його ідеальним для інтеграції в системи розумного будинку, де потрібна надійна двостороння комунікація.

SPI, або Інтерфейс послідовного периферійного інтерфейсу, являє собою синхронний протокол, що працює за принципом "майстер-раб". Він включає чотири основні лінії: MISO, MOSI, SCK і CS, що дає змогу майстру контролювати обмін даними з одним або кількома рабами. Ця технологія підходить для під'єднання різних датчиків (температури, вологості, відстані тощо) у системах розумного будинку, забезпечуючи швидкий обмін даними завдяки відносно високій швидкості та можливості роботи в режимі повного дуплексу.

2.3 Апаратні платформи

2.3.1 Апаратні платформи ядра системи

Метою дипломної роботи є створення системи багатозадачного керування побутовими приладами, де центральний пристрій системи керування інтелектуальною будівлею виконує роль сполучної ланки між апаратними компонентами та програмним забезпеченням. Цей центральний пристрій має забезпечувати підтримку різних технологій передавання даних, включно з тими, що були описані вище в цьому розділі. Під час дослідження було розглянуто різні апаратні платформи, такі як, Arduino, Raspberry Pi, ODROID-C4, ASUS Tinker board і настільний комп'ютер, кожна з яких пропонує свої унікальні можливості для реалізації дипломного проєкту керування побутовими приладами в контексті інтелектуальної будівлі.

Arduino є відкритою апаратною та програмною платформою, спеціально розробленою для проектування та створення прототипів електронних пристроїв (рис. 2.2). Ця платформа, заснована на одноплатних мікроконтролерах, надає доступний і зручний шлях у світ електроніки та програмування для широкого кола користувачів, включно зі студентами, інженерами та дослідниками. Arduino має гнучкість і відкритість, що робить її ідеальним вибором для експериментів і навчання в галузі електроніки.

Raspberry Pi, розроблений Raspberry Pi Foundation з Великобританії, являє собою серію компактних одноплатних комп'ютерів (рис. 2.3). Ці пристрої, вперше представлені в 2012 році, пройшли значний розвиток: від простих моделей з одноядерним процесором до потужніших версій з чотирьохядерними процесорами і збільшеним об'ємом оперативної пам'яті. Raspberry Pi виділяється своєю доступністю та багатофункціональністю, пропонуючи широкі можливості для програмування, автоматизації та реалізації промислових рішень. Цей комп'ютер на базі Linux з портами GPIO підходить для контролю різних електронних компонентів і роботи у сфері IoT. Raspberry

Pi підтримує різні технології передачі даних, включно з UART і 1-Wire, і завдяки своєму ARM-процесору може запускати сервери та бази даних, а також підтримує платформу .NET core.

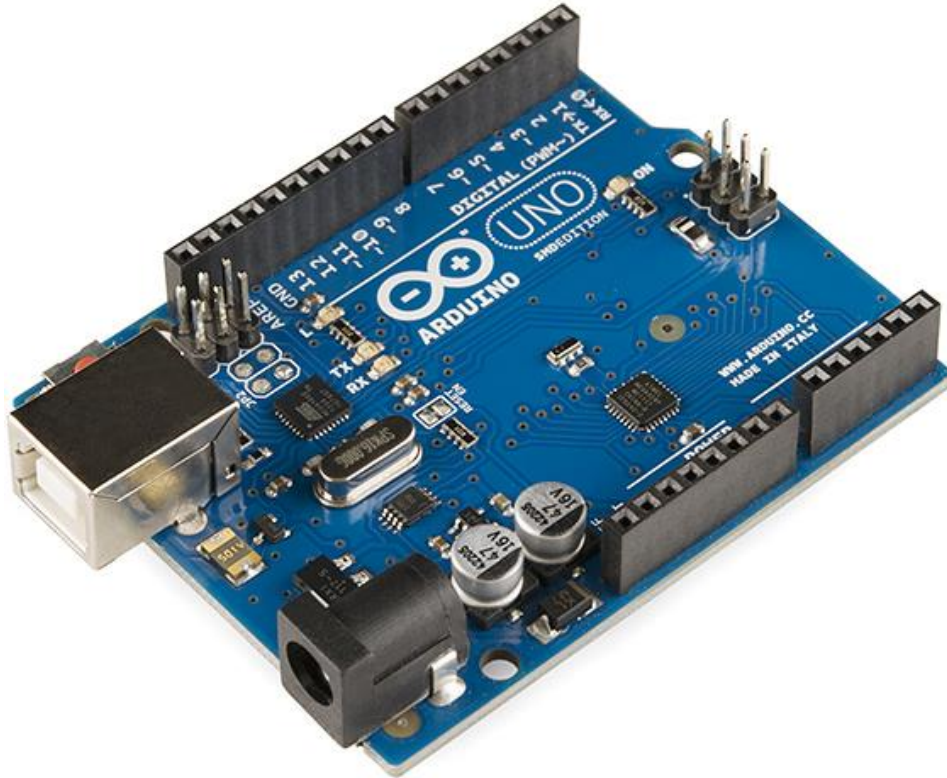


Рисунок 2.2 – Плата платформа Arduino UNO



Рисунок 2.3 – Мікрокомп'ютер Raspberry Pi 4

ODROID-C4 являє собою нове покоління обчислювальних пристроїв, яке значно перевершує свого попередника ODROID-C2 за продуктивністю (рис. 2.4). Цей мікрокомп'ютер заснований на процесорі Amlogic S905X3 з ядром Cortex-A55 і графічним прискорювачем Mali-G31 MP2, забезпечуючи високу продуктивність при низькому рівні шуму. З 4 ГБ оперативної пам'яті та сумісністю з різними версіями Linux і Android, ODROID-C4 пропонує потужні можливості для розроблення та реалізації складних проєктів. Він оснащений сучасними інтерфейсами для обміну даними, включно з eMMC 5.0, USB 3.0 і Gigabit Ethernet, що робить його непоганим вибором для вимогливих застосунків і проєктів.

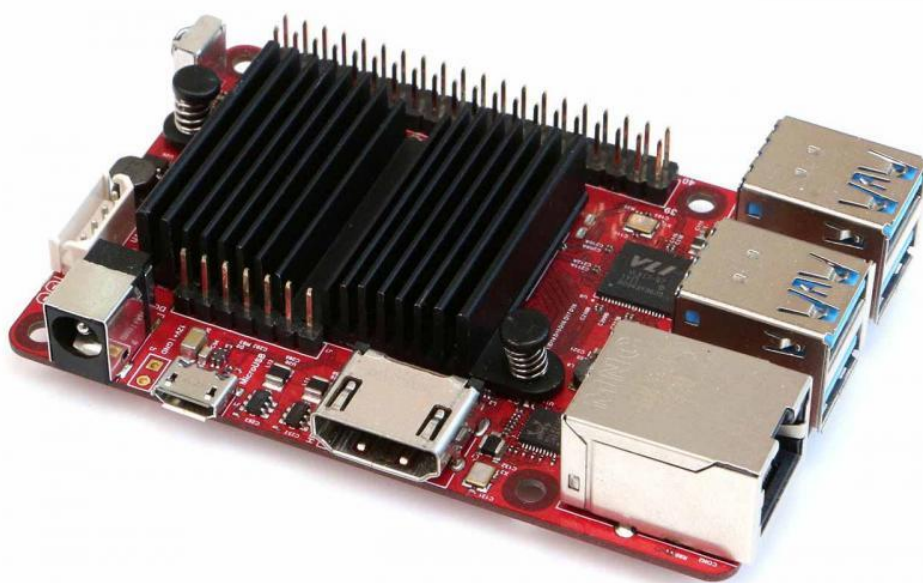


Рисунок 2.4 – Зовнішній вигляд плати ODROID-C4

ASUS Tinker Board, розроблений корпорацією ASUS, є одноплатним комп'ютером, який поєднує в собі високу продуктивність і компактні розміри. Ця платформа, представлена як альтернатива популярним Raspberry Pi та ODROID-C4, вирізняється своїми поліпшеними обчислювальними та графічними характеристиками. Оснащена чотирьохядерним процесором Rockchip RK3288 з частотою до 1,8 ГГц і графічним ядром Mali-T764, ASUS Tinker Board пропонує підвищену продуктивність, що робить її ідеальною для

завдань, які потребують великих обчислювальних ресурсів, як-от оброблення мультимедіа або машинне навчання.

ASUS Tinker Board знайшла своє застосування в різних сферах, від створення мультимедійних центрів до промислової автоматизації та розробки прототипів нових пристроїв (рис. 2.5). Завдяки вбудованим інтерфейсам, включно з HDMI, Gigabit Ethernet, Wi-Fi і Bluetooth, ця плата легко інтегрується в різноманітні системи та мережі.



Рисунок 2.5 – Зовнішній вигляд плати Asus Tinker Board

У контексті розроблення дипломного проєкту зі створення системи багатозадачного керування побутовими приладами, було розглянуто використання настільного комп'ютера як центрального елемента системи. Цей вибір був обумовлений високими обчислювальними можливостями настільних комп'ютерів, що перевершують апаратні характеристики інших розглянутих платформ. Основною перевагою використання настільного комп'ютера є можливість застосування x86 програмного забезпечення, різноманітних інструментів програмування і систем управління базами даних.

Однак, попереднє дослідження виявило, що для реалізації деяких інтерфейсів, як-от 1-Wire, SPI і UART, знадобиться інтеграція додаткових плат розширення. Також слід враховувати потенційні проблеми, пов'язані з високим енергоспоживанням і шумом, характерними для настільних комп'ютерів, що може вплинути на загальну ефективність і зручність використання системи розумного будинку.

2.3.2 Пристрої розв'язки

Для розроблення мікрокомп'ютерного ядра системи багатозадачного керування побутовими приладами, було ухвалено рішення вибрати реле як систему розв'язки. Реле є електромеханічним пристроєм, що діє як перемикач, який дає змогу керувати великими електричними струмами за допомогою малого керуючого сигналу. У системах розумного будинку реле забезпечує можливість контролю за різними побутовими приладами, включно з освітленням, обігрівачами, кондиціонерами та іншими пристроями, що потребують значної потужності.

Одним із ключових аспектів під час вибору реле є їхня номінальна напруга і струм, які визначають максимальну потужність, що комутується реле. Важливо, щоб обрані реле відповідали електричним навантаженням, з якими вони працюватимуть. Також слід враховувати тип керування реле: вони можуть активуватися безпосередньо від мікроконтролера або через додаткові драйвери, залежно від необхідного рівня струму для активації та характеристик вихідних портів мікроконтролера.

Кількість каналів у реле також відіграє важливу роль у розробленні та інтеграції розроблюваної системи в електросистему будинку, оскільки багатоканальні реле дають змогу керувати кількома навантаженнями незалежно, що підвищує гнучкість і масштабованість системи. Важливим фактором є вибір типу контактів реле: нормально відкриті (NO), нормально закриті (NC) або контакти, що перемикаються, що визначається необхідною логікою управління і безпеки системи.

2.3.3 Датчики

Датчики відіграють ключову роль у системі багатозадачного керування побутовими приладами, забезпечуючи збір даних про стан довкілля системи. Вони слугуватимуть основою для ухвалення рішень і автоматизації процесів в інтелектуальному будинку. Датчики відіграють ключову роль у створенні сценаріїв для автоматизації керування побутовими приладами.

Для забезпечення базової функціональності системи багатозадачного керування побутовими приладами, що не обмежується лише часовими параметрами, було розглянуто датчики руху і температури.

Інфрачервоний датчик руху HC-SR501 (рис. 2.6). Цей датчик здатний виявляти рух людини або тварин на відстані до 7 метрів, що робить його гарним вибором для моніторингу та управління системами освітлення, безпеки та іншими автоматизованими функціями. З кутом спрацьовування 110° , він охоплює широку зону, що збільшує його ефективність у різних сценаріях використання [11]. Датчик працює від напруги живлення 4.5 - 12 В, що робить його сумісним з більшістю мікроконтролерів і мікрокомп'ютерів. Вихідна напруга логічного рівня варіюється від 0 до 3.3 В. Час затримки на цьому датчику регулюється від 0.3 до 300 секунд, що дає змогу налаштувати його відповідно до конкретних вимог проекту. Датчик пропонує два режими спрацьовування: неповторюване (L) і повторюване (H) перемикання, що дає гнучкість у його застосуванні.

Датчик температури і вологості DHT11 являє собою цифровий датчик, який вимірює температуру і вологість повітря (рис. 2.7). Він підходить для систем розумного будинку, де потрібно контролювати або реагувати на кліматичні умови. Датчик забезпечує визначення вологості в діапазоні від 5 до 95% RH з точністю $\pm 5\%$, що забезпечує достатню точність для створення сценаріїв автоматизації. Температурний діапазон датчика становить від -20 до $+60$ $^\circ\text{C}$ з точністю $\pm 2\%$, що дає змогу моніторити широкий діапазон температур. DHT11 працює від живлення 3.5-5.5 В, що робить його сумісним із різними мікроконтролерами та системами [12]. Частота опитування датчика не

перевищує 1 Гц, що підходить для більшості додатків моніторингу. Компактні розміри датчика (15.5 x 12 x 5.5 мм) полегшують його інтеграцію в різні пристрої та системи.



Рисунок 2.6 – Зовнішній вигляд датчику HC-SR501

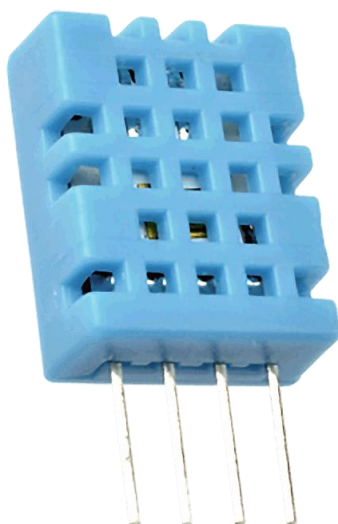


Рисунок 2.7 – Зовнішній вигляд датчику DHT11

2.4 Програмні інструменти та платформи

2.4.1 Серверна частина

Для створення серверної частини системи багатозадачного керування побутовими приладами, необхідно вибрати відповідні програмні інструменти та платформи, які забезпечать функціонування системи, а також масштабованість і гнучкість у розробці. У дипломній роботі розглядаються такі технології, як .NET, Python і Java.

.NET: Ця платформа від Microsoft пропонує потужний набір інструментів для створення серверних додатків. Вона підтримує безліч мов програмування, включно з C# і F#, і надає широку бібліотеку класів для розроблення вебсервісів і API. .NET особливо підходить для створення надійних і масштабованих серверних застосунків, завдяки своїй продуктивності та підтримці асинхронного програмування.

Python: Ця мова програмування відома своєю читабельністю і простотою у використанні, що робить її чудовим вибором для швидкого розроблення та прототипування. Python підтримує безліч фреймворків для веброботки, як від Django і Flask, які можуть бути використані для створення серверної частини системи. Ці фреймворки пропонують готові рішення для багатьох поширених завдань, як-от маршрутизація, аутентифікація та взаємодія з базами даних [13].

Java: Ця мова програмування і платформа вирізняються високою продуктивністю і переносимістю, що робить їх придатними для створення серверних додатків для великих і складних систем. Java пропонує потужні інструменти та бібліотеки для розроблення вебдодатків, включно з Spring Framework, який широко використовується для створення масштабованих і надійних серверних додатків.

2.4.2 Клієнтська частина

Для розробки клієнтської частини системи багатозадачного керування побутовими приладами, призначеної для керування через смартфон, були

розглянуті різні програмні інструменти, що дають змогу розробляти мобільні додатки для Android. Кожен із цих інструментів має унікальні особливості та переваги, які роблять їх придатними для певних завдань і вимог проєкту.

Android Studio є офіційним інтегрованим середовищем розробки для Android-платформи, надаючи великий набір інструментів і функцій для розробки, тестування та налагодження мобільних додатків. Вона включає в себе емулятори пристроїв, інструменти для профілювання продуктивності додатків і візуальний редактор інтерфейсу, що робить її особливо зручною для розробників, які прагнуть створювати високоякісні та продуктивні додатки для Android.

React Native являє собою фреймворк, що дає змогу розробляти кросплатформні мобільні додатки, використовуючи JavaScript і React. Це рішення дає змогу розробникам використовувати один і той самий код для створення застосунків як для Android, так і для iOS, що значно скорочує час розробки і спрощує підтримку застосунків. React Native підходить для проєктів, де важлива швидкість розробки і можливість швидко впроваджувати зміни.

Visual Studio з Xamarin пропонує можливість розробки кросплатформних додатків на C#. Xamarin інтегрується з Visual Studio, забезпечуючи плавну розробку і підтримку як Android, так і iOS додатків. Це рішення ідеально підходить для тих, хто вважає за краще працювати в екосистемі Microsoft і використовувати C# для створення мобільних додатків.

Eclipse з ADT Plugin був одним із перших інструментів для розробки Android-додатків. Хоча він поступається за функціональністю сучасним інструментам, Eclipse з ADT Plugin, як і раніше, використовується деякими розробниками, особливо тими, хто звик до цього середовища розробки. Він пропонує базові інструменти для розробки, налагодження та тестування Android-додатків.

2.4.3 Програмне забезпечення бази даних

У контексті розроблення системи багатозадачного керування побутовими приладами, важливо вибрати відповідне програмне забезпечення для баз даних, яке забезпечить ефективне зберігання та оброблення інформації.

Oracle Database 21c (рис.2.8), розроблено для оптимальної роботи в хмарних системах, її можна масштабувати для використання на різних пристроях [14]. Oracle 21c здатна керувати масивними базами даних, що містять мільярди записів, завдяки своїй мережевій архітектурі та гнучкому поєднанню фізичних і логічних елементів. Ця система вирізняється своєю безпекою, де кожна транзакція ізольована для запобігання взаємних впливів. Основними перевагами Oracle є її інноваційні рішення та стабільність, проте висока вартість і вимоги до ресурсів можуть бути перешкодою для менших проєктів.



Рисунок 2.8 – Oracle Database 21c logo

Microsoft SQL Server (рис 2.9) містить значні поліпшення, як-от вдосконалена інтеграція з хмарними технологіями та розширене машинне навчання для глибшого аналізу даних. Ця система підтримує динамічне маскування даних і забезпечує високий рівень безпеки. Переваги Microsoft SQL Server включають в себе високу продуктивність та інтеграцію з іншими продуктами Microsoft, що робить його ідеальним вибором для підприємств, які активно використовують продукти Microsoft. Однак, як і у випадку з Oracle, значні вимоги до ресурсів можуть бути обмежувальним фактором для деяких проєктів.



Рисунок 2.9 – Logo microsoft SQL Server 22

SQLite представляє собою вбудовану систему управління реляційними базами даних, яка вирізняється своєю автономністю, оскільки для її роботи не потрібен серверний процес (рис. 2.10). Ця особливість робить SQLite вкрай портативним і зручним для інтеграції, оскільки всі операції здійснюються на рівні локальних файлів. Крім того, SQLite не вимагає складного налаштування чи адміністрування, що спрощує його використання.

Завдяки своєму невеликому розміру, SQLite є ідеальним варіантом для застосування в вбудованих системах, мобільних додатках та пристроях з обмеженими ресурсами. Незважаючи на компактність, SQLite забезпечує повноцінну підтримку реляційної моделі даних і може обробляти більшість SQL-запитів [15].

SQLite є відмінним вибором для проектів та систем, де важлива простота та незалежність від великих серверних рішень. Це особливо актуально для вбудованих систем, де потрібна висока надійність при мінімальній складності. SQLite легко інтегрується з багатьма програмними платформами і підтримується широким спектром мов програмування, включаючи Python, C, C++, Java та інші



Рисунок 2.10– Logo SQLite

3 ПРАКТИЧНА ЧАСТИНА

3.1 Структурна схема системи

В основі розробленої структурної схеми багатозадачної системи керування побутовими приладами (рис. 3.1) лежить центральний керуючий модуль, який складається з кількох ключових елементів.

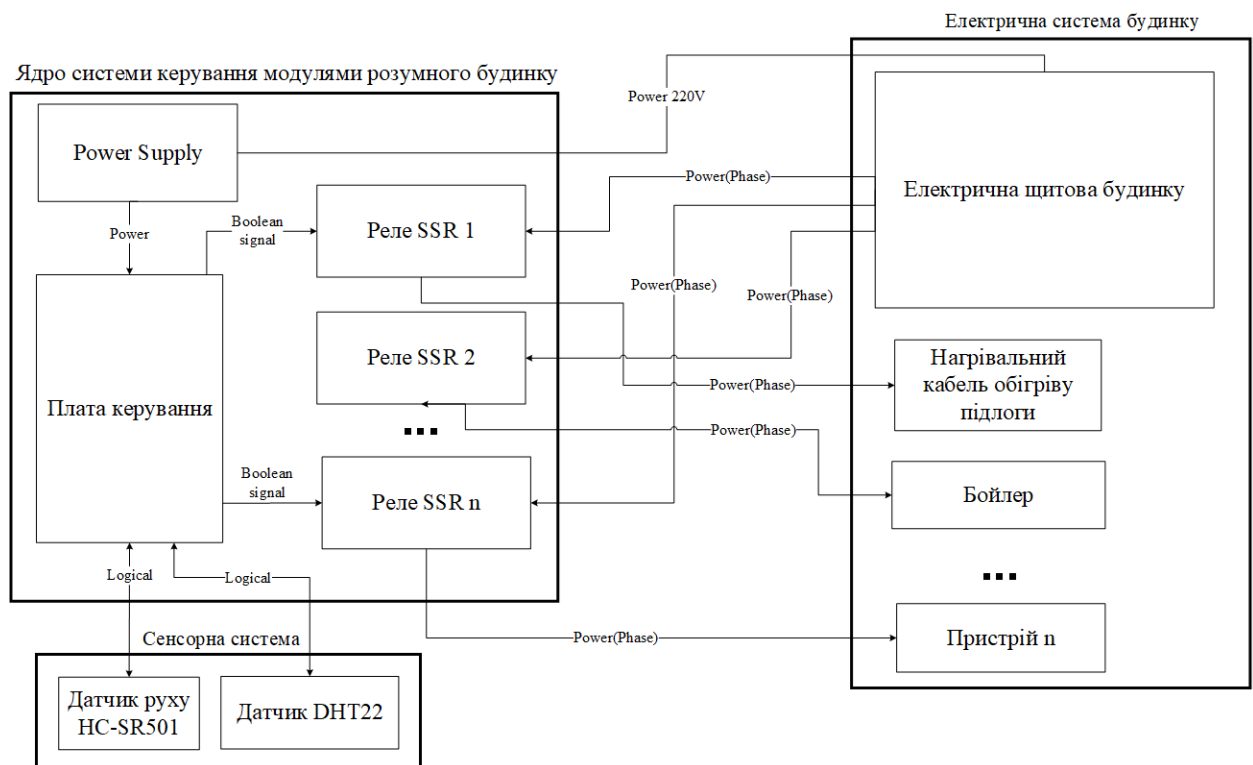


Рисунок 3.1 – Структурна схема багатозадачної системи керування побутовими приладами

Керуюча плата. Цей компонент є мозком системи. Вона відповідає за виконання програмних сценаріїв, які керують роботою побутових приладів, а також за опрацювання даних, одержуваних від різних датчиків і сенсорів.

Блок живлення для керуючої плати. Цей елемент критично важливий для забезпечення стабільної роботи керуючої плати. Він підтримує необхідний рівень напруги та захищає систему від можливих збоїв і перепадів в електромережі.

Твердотільні однофазні реле. Ці компоненти відіграють ключову роль в інтеграції керуючої плати з електричною системою будинку. Вони не тільки забезпечують розв'язку, а й дають змогу керувати фазами змінного струму, що є важливим аспектом для контролю роботи побутових приладів.

Блок сенсорної системи, у системі розумного будинку відповідає за збір інформації із зовнішнього середовища і перетворення її в електронні сигнали для подальшого передавання на керуючу плату. Ці датчики підключаються до плати з використанням методів передавання даних, описаних у другому розділі. Відповідно до структурної схеми, система містить два основні типи датчиків: датчик руху і комбінований датчик вологості і температури, необхідні для автоматизації функцій розумного будинку. Крім того, в систему можна інтегрувати й інші типи датчиків.

Електрична система будинку складається зі стандартної електричної панелі та різних споживачів електроенергії, таких як система обігріву підлоги, водонагрівач та інші пристрої. Ця частина системи відіграє важливу роль у створенні комфортних умов і функціональності житла. Автоматизація та оптимізація споживання електроенергії, а також підвищення рівня комфорту досягаються за рахунок контролю з боку керуючої плати через елементи розв'язки. Такий підхід дає змогу ефективно керувати споживанням енергії електричних пристроїв або модулів, відключаючи їх від основного джерела живлення за відсутності потреби в їхній роботі.

3.2 Вибір пристрою й програмних засобів

Центральним елементом системи багатозадачного керування побутовими приладами було обрано мікрокомп'ютер Raspberry Pi. Цей вибір обумовлений його відповідністю вимогам щодо передачі даних, описаним у розділі 2.3, а також його простотою в налаштуванні та експлуатації. Raspberry Pi має

достатню продуктивність, працює безшумно і має компактні розміри, що робить його ідеальним для безперервної роботи в системі розумного будинку, при цьому залишаючись енергоефективним і доступним.

Згідно з аналізом програмного забезпечення для серверних рішень було ухвалено рішення використовувати мову програмування Python. Python виділяється своєю популярністю і гнучкістю, підтримується великою спільнотою розробників і пропонує багатий вибір бібліотек і фреймворків. Ці якості роблять його ідеальним для створення серверних додатків, включно з тими, що працюють на Raspberry Pi. Простота синтаксису Python і можливості швидкого прототипування сприяють розробці інноваційних рішень.

Для зберігання даних у системі розумного будинку було обрано базу даних SQLite. Її легковажність і ефективність роблять SQLite ідеальною для використання в невеликих додатках. SQLite легко інтегрується з Python, забезпечуючи зручне зберігання даних у локальному файлі без необхідності встановлення зовнішніх серверів баз даних.

Для розробки користувацького інтерфейсу для клієнтських пристроїв було обрано Android Studio, який є провідним інструментом для розробки на Java для Android-пристроїв. Ця комбінація з Python на серверній стороні створює ефективну взаємодію між сервером і клієнтом. Python з його потужними бібліотеками та простим синтаксисом забезпечує швидке та надійне виконання серверних операцій, що сприяє оптимізації загальної продуктивності системи.

Використання Python у системі розумного будинку забезпечує стабільність та оптимізацію завдяки його виразності та широкій екосистемі бібліотек. Застосування SQLite спрощує управління даними, а Android Studio полегшує розробку інтуїтивно зрозумілого користувацького інтерфейсу. Крім того, гнучкість Python в інтеграції різноманітних IoT-пристроїв і модулів розширює можливості автоматизації системи розумного будинку.

3.3 Електрична схема системи керування побутовими приладами

На рисунку 3.2 представлено електричну схему системи багатозадачного керування побутовими приладами, заснованої на використанні мікрокомп'ютера Raspberry Pi. Ця схема інтегрує ядро системи управління у звичайну конфігурацію домашнього електрощита з робочою напругою 230V.

Електроживлення подається на електролічильник через дроти різних кольорів: червоний (фаза), синій (нуль) і зелений (заземлення). Від лічильника електрика надходить на головний двополіусний автоматичний вимикач, який дає змогу відключати як фазу, так і нуль. Далі в схемі розташоване реле напруги для захисту від стрибків напруги. Після реле нульовий дріт іде до нульових шин, а фазний - до групових пристроїв захисного відключення (ПЗВ), позначених цифрами від 0 до 3. За ПЗВ розташовані автоматичні вимикачі, до яких під'єднані різні домашні навантаження, такі як розетки та освітлювальні прилади.

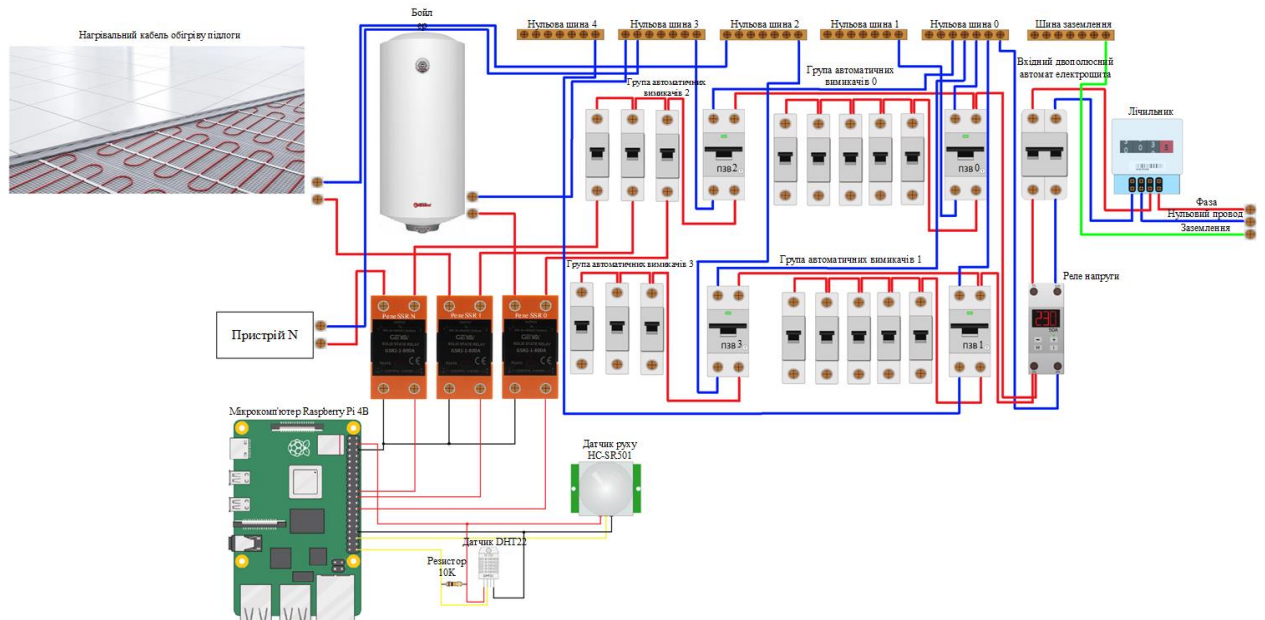


Рисунок 3.2 – Електрична схема системи

До деяких із цих автоматичних вимикачів під'єднано модулі твердотільних реле, контрольовані мікрокомп'ютером Raspberry Pi. У цій схемі до твердотільних реле приєднані бойлер і елемент обігріву підлоги, що дає змогу керувати ними через ядро системи розумного будинку. Керування цими елементами може здійснюватися за різними сценаріями, наприклад, залежно від температури в приміщенні, за заздалегідь заданим розкладом або за іншими параметрами, встановленими користувачем. Сценарії можуть бути розроблені та налаштовані користувачем відповідно до його індивідуальних вимог.

Завдяки широкому спектру інтерфейсів Raspberry Pi, включно з Bluetooth, Wi-Fi і GPIO, до нього можна підключити безліч різноманітних пристроїв. Це дає змогу керувати додатковими пристроями відповідно до користувацьких сценаріїв. Усе це створює можливості для оптимізації споживання електроенергії та реалізації ефективного керування різними побутовими приладами в будинку.

3.4 Налаштування мікрокомп'ютера Raspberry Pi 4B+

Для реалізації програмно-апаратної інтеграції в рамках проєкту зі створення системи багатозадачного керування побутовими приладами було обрано одноплатний мікрокомп'ютер Raspberry Pi, що працює під управлінням операційної системи Raspberry Pi OS Lite, заснованої на Linux Debian x64. Цей вибір був продиктований спеціалізацією даної ОС для ефективної роботи з Raspberry Pi, а також наявністю вбудованих інструментів для налаштування апаратних інтерфейсів. Значною перевагою є включення в ОС мови програмування Python, що полегшує розробку програмного забезпечення.

Для встановлення операційної системи на Raspberry Pi переважно використовувати утиліту Raspberry Pi Imager, яка дає змогу легко записати

образ системи на microSD-карту. Рекомендується вибирати карти пам'яті з класом швидкості 10 і вище для забезпечення оптимальної продуктивності.

У процесі встановлення (рис. 3.3) ОС за допомогою Raspberry Pi Imager, необхідно вибрати потрібний образ ОС у розділі CHOOSE OS і відповідну microSD-карту в розділі CHOOSE STORAGE, після чого почати запис, натиснувши на кнопку WRITE. Після завершення запису, карта готова до використання в Raspberry Pi 4 B.

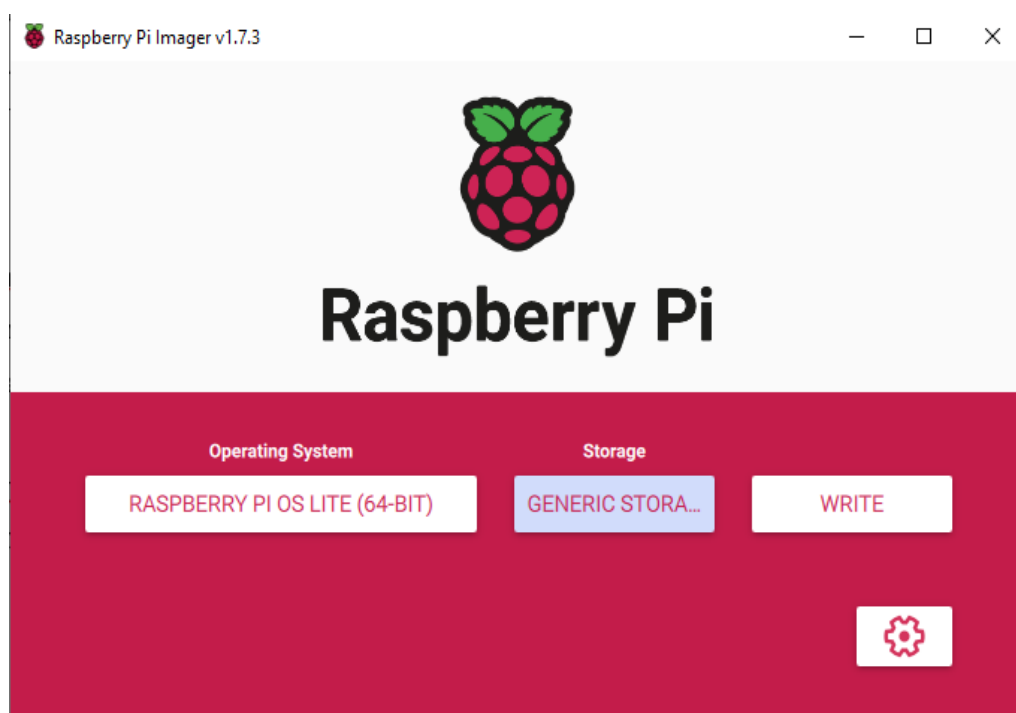


Рисунок 3.3 –Процес запису ОС за допомогою програмами Raspberry Pi Imager

Початковий запуск системи на Raspberry Pi вимагає під'єднання зовнішніх пристроїв, як-от клавіатура, миша та монітор. На етапі ініціалізації передбачено створення облікового запису, налаштування Wi-Fi і активація SSH-сервера для зручності адміністрування. Ці та інші налаштування виконуються через команду:

```
$> sudo raspi-config
```

Ця команда відкриває інтерфейс програми конфігурування (рис. 3.4), де можна налаштувати Wi-Fi в розділі System Options, вибравши Wireless LAN і

вказавши необхідні дані для підключення. Активація SSH-сервера здійснюється в розділі Interface options, вибравши SSH і підтвердивши свій вибір [16]. Після завершення налаштувань виходимо з програми.

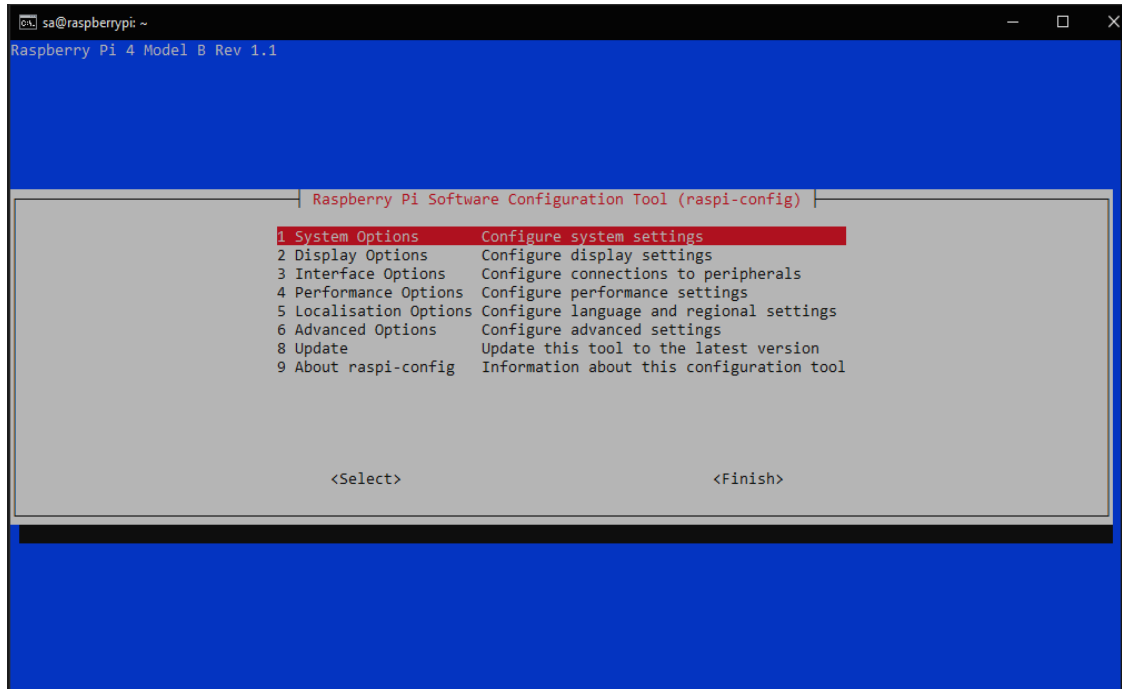


Рисунок 3.4 – Інтерфейс меню конфігурації

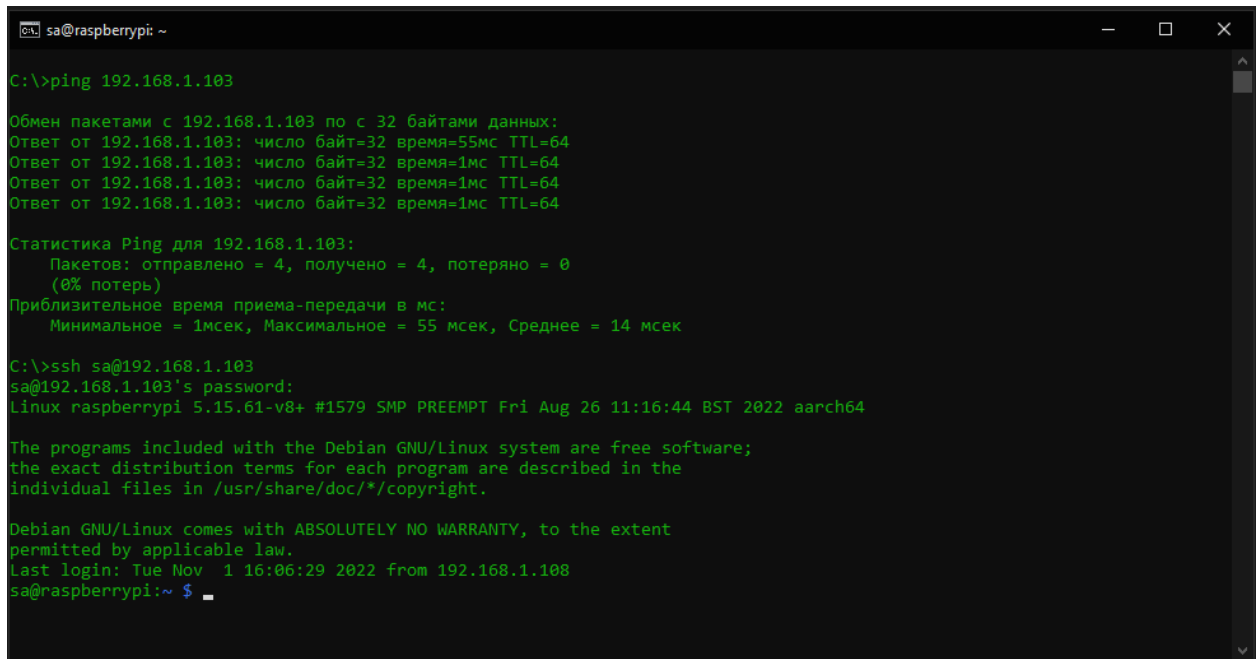
Для визначення IP-адреси Raspberry Pi, присвоєної через DHCP, використовується команда:

```
$> hostname -I
```

Потім слід перевірити доступність Raspberry Pi з робочого комп'ютера. У моєму випадку, це робиться через командний рядок Windows:

```
C:\> ping 192.168.1.103  
C:\> ssh sa@192.168.1.103
```

Успішне виконання команди ping підтверджується відповіддю від 4 пакетів (рис. 3.5). При успішному SSH-підключенні з'явиться запит на введення пароля, після чого відкриється віддалений доступ до терміналу Raspberry Pi OS Lite [17].



```

sa@raspberrypi: ~
C:\>ping 192.168.1.103

Обмен пакетами с 192.168.1.103 по 32 байтами данных:
Ответ от 192.168.1.103: число байт=32 время=55мс TTL=64
Ответ от 192.168.1.103: число байт=32 время=1мс TTL=64
Ответ от 192.168.1.103: число байт=32 время=1мс TTL=64
Ответ от 192.168.1.103: число байт=32 время=1мс TTL=64

Статистика Ping для 192.168.1.103:
    Пакетов: отправлено = 4, получено = 4, потеряно = 0
    (0% потерь)
Приблизительное время приема-передачи в мс:
    Минимальное = 1мсек, Максимальное = 55 мсек, Среднее = 14 мсек

C:\>ssh sa@192.168.1.103
sa@192.168.1.103's password:
Linux raspberrypi 5.15.61-v8+ #1579 SMP PREEMPT Fri Aug 26 11:16:44 BST 2022 aarch64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Nov  1 16:06:29 2022 from 192.168.1.108
sa@raspberrypi:~ $

```

Рисунок 3.5 – Результат виконання команд ping та ssh

3.5 Налаштування віддаленого середовища розробки через SSH

Інструмент для створення програмного коду було використано Visual Studio Code, використовуючи плагіни, що дають змогу створювати віддалене робоче оточення для розробки програмного забезпечення системи розумного будинку. Visual Studio Code, розроблений Microsoft, пропонує широкий спектр можливостей для інтеграції різних розширень, включно з тими, що призначені для віддаленої роботи через SSH. Цей редактор коду, що вирізняється своєю кросплатформеністю і відкритим вихідним кодом, доступний для скачування на офіційному сайті Microsoft.

Для ввімкнення можливостей роботи по SSH, необхідно встановити розширення Remote Development і Remote - SSH. Після їх додавання рекомендується перезавантажити інтерфейс Visual Studio Code. Для початку віддаленої роботи користувач має натиснути на синю іконку в нижньому лівому кутку інтерфейсу (рис. 3.6), вибрати опцію "Open a Remote Window",

потім "Connect to Host" і ввести дані для підключення SSH `sa@192.168.1.103`. У процесі аутентифікації слід вибрати операційну систему Linux, після чого система почне встановлення необхідних бібліотек на віддалений сервер.

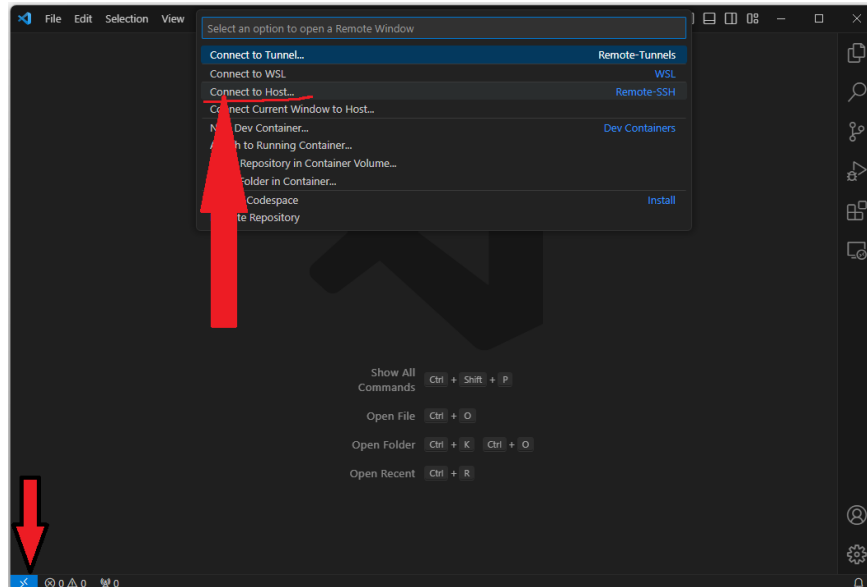


Рисунок 3.6 – Вигляд встановлення remotedev через SSH

Під час інсталяції проекту HomeSmartControl було створено структуру робочої папки, розділеної на дві піддиректорії: `server` і `scenarios`. Після успішного завершення інсталяції, адреса віддаленого сервера відобразиться поруч із зеленою іконкою в інтерфейсі Visual Studio Code. Під час використання функції "Open Folder", користувачеві надається можливість вибору директорії для роботи у файловому менеджері редактора, що полегшує процес розробки та управління кодом.

3.6 Розробка програмного забезпечення системи

3.6.1 Серверна частина

За допомогою платформи Python і його мережових бібліотек було розроблено сервер для управління "розумним будинком". Мову Python було обрано через її простоту, потужність і широкий спектр доступних інструментів, що робить її привабливою для створення складних систем управління. Сервер використовує TCP/IP протокол для забезпечення стабільного з'єднання з клієнтськими пристроями, його протоколи гарантують безперебійне і безпечне передавання даних [18, 19].

Як базу даних для зберігання налаштувань користувача та інформації про побутові пристрої було обрано SQLite [20]. Цей вибір зумовлений її невеликим розміром і високою надійністю, що робить її оптимальним рішенням для систем, які працюють на пристроях з обмеженими ресурсами, наприклад, Raspberry Pi, що використовується в дипломному проєкті. Використання SQLite забезпечує передбачувану роботу системи та можливість масштабування.

За допомогою бібліотеки RPi.GPIO, встановленої на Raspberry Pi, сервер керує різноманітними пристроями "розумного будинку". Ця бібліотека дає змогу керувати портами GPIO, що забезпечує точний контроль над кожним під'єднаним пристроєм [21].

Безпека системи забезпечується суворою автентифікацією. Під час спроби підключення клієнта до сервера відбувається перевірка IP-адреси та пароля на відповідність попередньо зареєстрованим даним. Після успішної автентифікації клієнт може надсилати команди для керування різними функціями "розумного будинку", включно з керуванням освітленням, температурою, музичними пристроями, бойлером, електричними вимикачами та шторами. Команди обробляються відповідно до попередньо заданих сценаріїв, забезпечуючи гнучкість і функціональність системи.

3.6.2 Розробка мобільного додатку. Клієнтська частина системи

У серверній частині мобільного застосунку для забезпечення поставленого в аналітичній частині рівня безпеки системи, було створено сторінку авторизації. Ця сторінка забезпечує підтвердження прав користувача на доступ до системи [22].

Ключовим елементом у дизайні інтерфейсу є компоувальник RelativeLayout (рис. 3.7). Вибір цього інструменту диктується його гнучкістю в розташуванні елементів інтерфейсу відносно один одного і рамок батьківського елемента [23]. Переваги RelativeLayout включають спрощення структури вкладеності, що сприяє більш швидкому відображенню інтерфейсу та підвищенню ефективності програми [24].

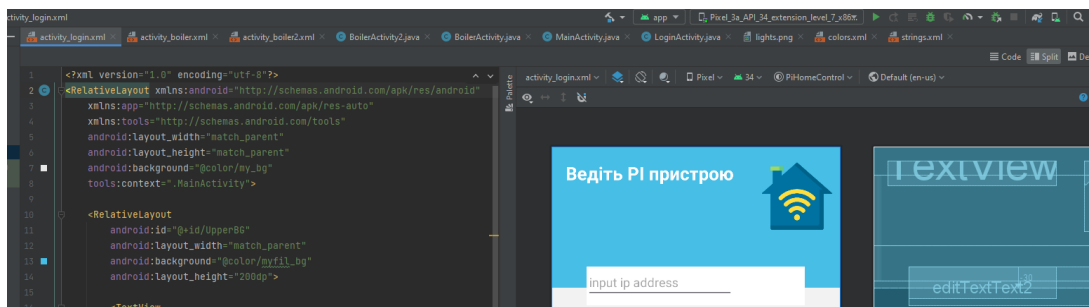


Рисунок 3.7 – Використання RelativeLayout

Центральний інтерфейс системи багатозадачного керування побутовими приладами:

а) у верхній частині інтерфейсу, під маркуванням UpperBG, розташовується візуальний блок, що представляє концепцію системи та іконку керування приладами.

б) середня частина інтерфейсу виділена для введення даних автентифікації, включно з двома полями:

- 1) поле для зазначення IP-адреси сервера;
- 2) поле для пароля, створене з урахуванням стандартів безпеки (атрибут `android:inputType="textPassword"` забезпечує приховування даних, що вводяться) [25].

Кнопка "Connect" у застосунку запускає процес під'єднання до сервера і передачу даних для автентифікації.

Після успішної автентифікації користувач перенаправляється на головний екран застосунку, де отримує змогу обирати та керувати побутовими приладами.

Головний екран програми, або Main Activity, є основним компонентом користувацького інтерфейсу. Він розроблений для забезпечення інтуїтивно зрозумілого та зручного керування різними функціями та приладами.

Як і в попередніх розділах, дизайн ґрунтується на використанні RelativeLayout, що дає змогу створювати складні та адаптивні інтерфейси. Така організація забезпечує гнучкість і масштабованість, що критично важливо для додатків, призначених для різноманітних пристроїв.

Графічний інтерфейс головного екрана (рис. 3.8) містить низку кнопок та індикаторів, розміщених у лінійних компонентах (LinearLayout). Це спрощує організацію елементів керування і робить інтерфейс доступнішим для користувача.

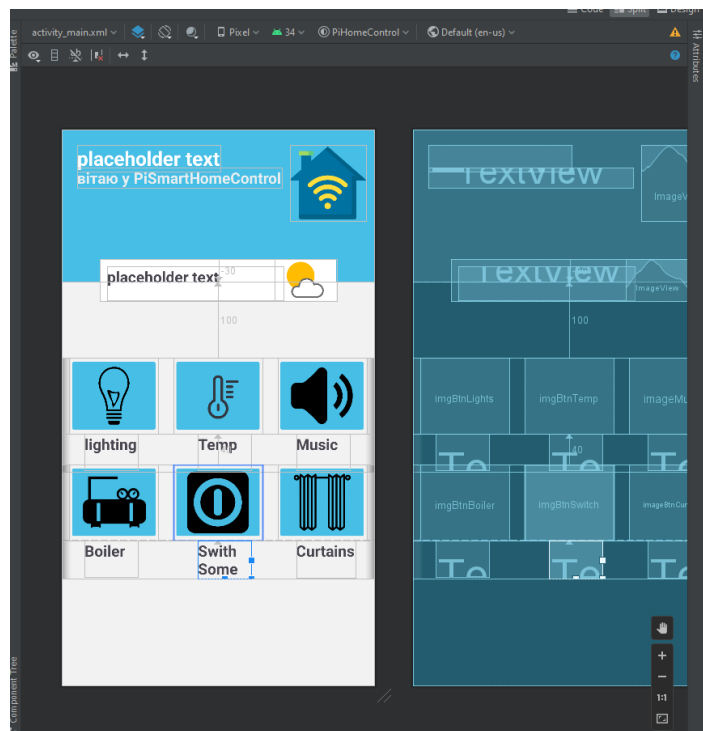


Рисунок 3.8 – Головна сторінка інтерфейсу додатка

Кожна кнопка на головному екрані відповідає певній функції або побутовому приладу, наприклад, освітленню, системі опалення, музичній системі або управлінню шторами. Ці іконографічні елементи слугують не тільки для візуалізації, а й для надсилання команд пристроям через застосунок.

3.7 Функціонування системи

Центром управління системою багатозадачного управління побутовими приладами служить мікропроцесор Raspberry Pi, інтегрований в електричний щит будинку за допомогою однофазних твердотільних реле, розташованих після автоматичних вимикачів. Це дає змогу керувати під'єднанням і від'єднанням різних пристроїв, як-от система підігріву підлоги, бойлер, масляний обігрівач, а також пристрій N - це будь-який пристрій, який користувач бажає інтегрувати в систему автоматизації будинку.

Розроблена система автоматизації не тільки контролює подачу живлення до побутових пристроїв, а й має інтерфейси Wi-Fi і Bluetooth. Це дає змогу керувати smart-лампами та іншими подібними пристроями дистанційно.

Управління всією системою здійснюється через спеціально розроблений Android-додаток Smart Home Control. Після проходження процесу авторизації, користувач отримує доступ до головного інтерфейсу додатка (рис. 3.9), через який він може керувати всіма підключеними до системи пристроями. Сюди входять як ті, що під'єднані через електрощит за допомогою однофазного твердотільного реле, так і пристрої, які під'єднуються через Wi-Fi або Bluetooth.

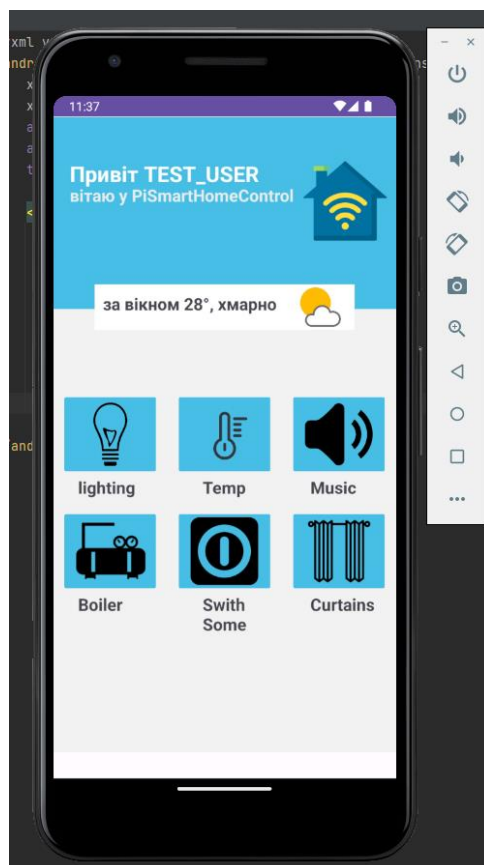


Рисунок 3.9 – Головне меню керування пристроями додатку
SmartHomeControl

Керування в системі багатозадачного керування побутовими приладами здійснюється на основі індивідуально створених користувачем сценаріїв, заснованих на чотирьох ключових параметрах: часі, температурі, вологості та виявленні руху в будинку. Як приклад, розглянемо користувача, який регулярно приймає душ на роботі в будні дні і повертається додому пізно, через що не потрібно нагрівати воду в бойлері щодня. Для оптимізації споживання електроенергії він може налаштувати бойлер так, щоб він вмикався тільки на вихідних.

У розробленій системі, користувач входить у додаток, використовуючи свій пароль, і потрапляє на головний екран графічного інтерфейсу (рис. 3.10). Потім, вибравши іконку бойлера, він переходить на сторінку управління цим пристроєм (рисунок 3.9). На цій сторінці користувач може встановити сценарій роботи бойлера, що ґрунтується на часових параметрах, тим самим

раціоналізуючи використання електроенергії відповідно до своїх повсякденних потреб.

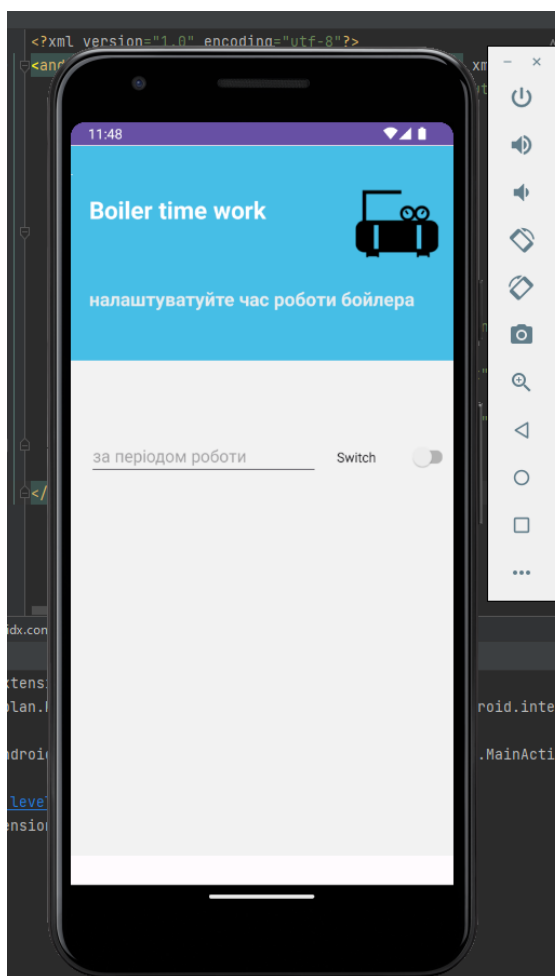


Рисунок 3.10 – Меню керування періодом роботи бойлеру

Для формування сценарію автоматичного керування бойлером (або іншим пристроєм) користувачеві пропонується використовувати такі кодування: «WE» (Weekend) – вказує на те, що бойлер має працювати тільки у вихідні.

Формат «HH:MM-HH:MM» – для задавання часу початку і завершення роботи пристрою.

Приклад сценарію, за якого бойлер працюватиме з 6:00 ранку до 20:00 вечора у вихідні, має такий вигляд:

```
> WE 06:00-20:00
```

Користувач вводить цей сценарій у спеціальне поле, яке це виглядає можна побачити на рисунку 3.11, і активує перемикач. Після цього сформоване повідомлення надсилається на сервер Raspberry Pi. У разі активації перемикача без попереднього введення сценарію, сервер увімкне бойлер. Якщо перемикач деактивувати, клієнт надішле команду на сервер для вимкнення бойлера. Таким чином, користувач може керувати пристроями без заздалегідь заданих сценаріїв, використовуючи віртуальний перемикач.

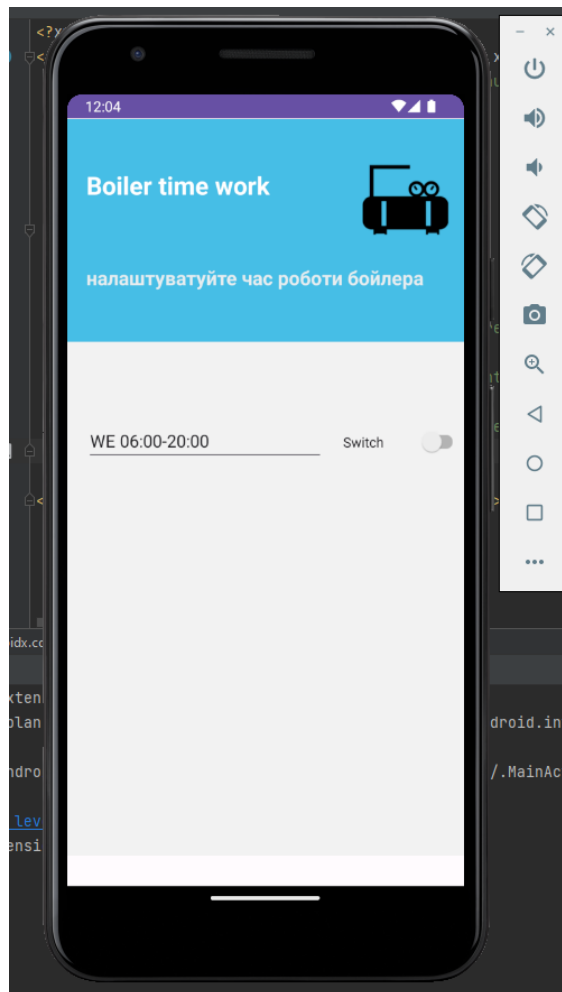


Рисунок 3.11 – Приклад написання сценарію для бойлера

Загальний алгоритм функціонування системи представлений на рисунку 3.12.

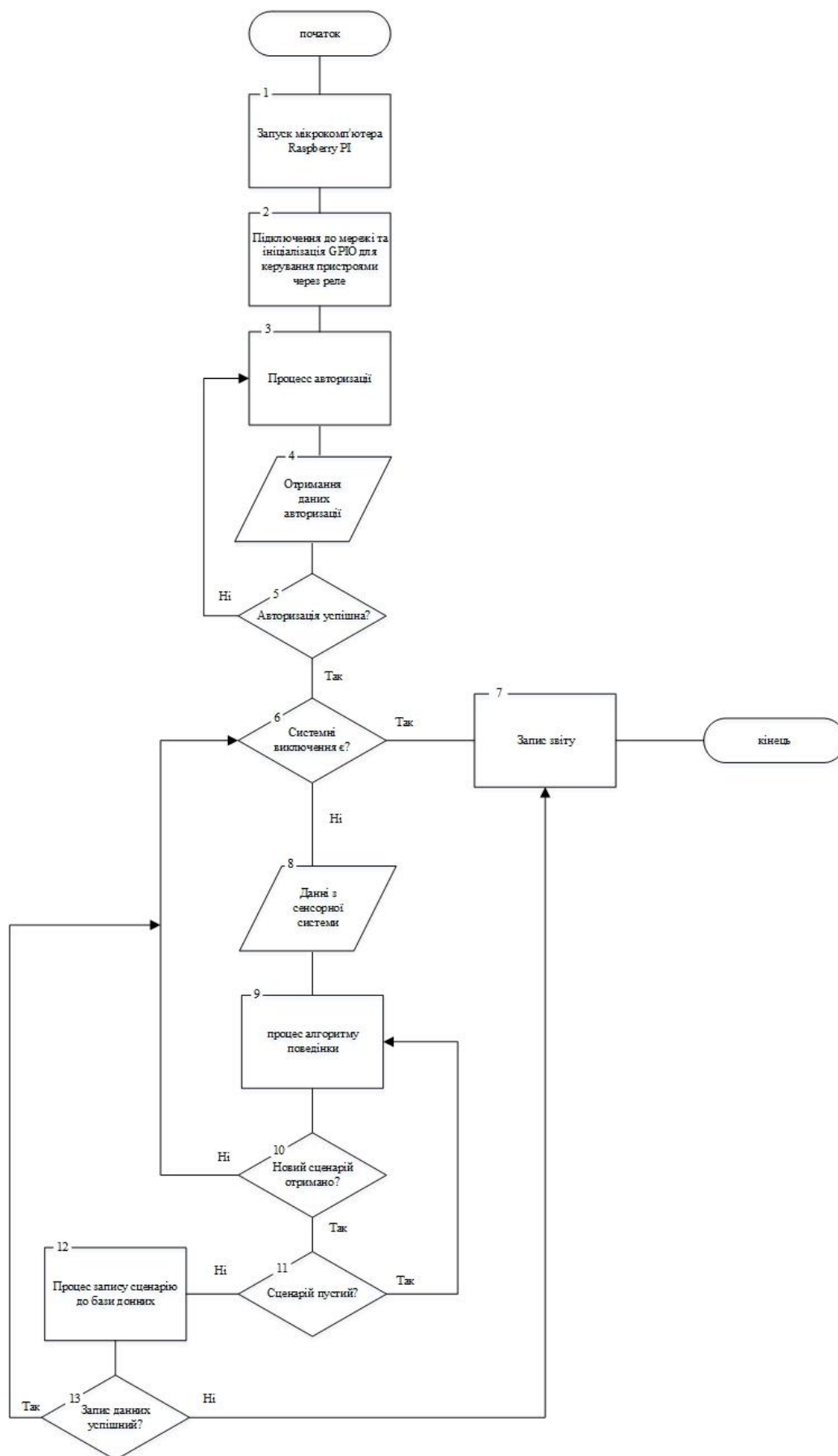


Рисунок 3.12 – Загальний алгоритм функціонування системи

У вихідному стані алгоритм ініціюється шляхом активації мікрокомп'ютера Raspberry Pi, який виступає як центральний керуючий вузол в архітектурі інтелектуального будинку. Підключення до локальної мережі та конфігурація GPIO здійснюються для забезпечення управління пристроями через реле, інтегровані в електрощитову будинку.

На наступному етапі реалізується процес авторизації користувача в системі. Система запитує дані для авторизації, що забезпечує безпеку доступу до керування пристроями будинку. У разі невдалої спроби авторизації передбачено механізм повтору запиту даних.

Після успішної авторизації алгоритм виконує перевірку на предмет системних виключень, що є важливим кроком для забезпечення стабільної роботи системи. У разі виявлення критичних помилок система фіксує звіт про збій і завершує свою роботу. Це необхідно для запобігання можливого неправильного керування пристроями або шкоди системі.

За відсутності системних винятків алгоритм продовжує свою роботу, переходячи до зчитування даних із датчиків. Інформація, отримана від датчиків температури, вологості та руху, використовується для визначення поточного стану параметрів домашнього середовища.

Дев'ятий блок алгоритму відповідає за аналіз отриманих даних і визначення поведінки системи. Система моніторить наявність нових сценаріїв, що дає змогу адаптувати поведінку системи відповідно до мінливих потреб користувача. Нові сценарії задаються користувачем через мобільний додаток.

При отриманні нового сценарію алгоритм перевіряє його вміст. Порожній сценарій призведе до стандартної поведінки системи (вмикання\ вимикання), тоді як непорожній сценарій буде занесено в базу даних. Це забезпечує гнучкість керування пристроями будинку відповідно до заданих параметрів часу, температури, вологості та виявлення руху.

У разі успішного внесення сценарію в базу даних алгоритм повторно перевіряє систему на наявність винятків і переходить до процесу загальної обробки даних. Цей процес охоплює аналіз користувацьких даних і взаємодію

із зовнішніми сервісами та інтерфейсами для інтеграції з іншими системами і пристроями.

Таким чином, розроблена система розумного будинку на базі Raspberry Pi володіє високим ступенем автоматизації і надає користувачеві інтуїтивно зрозумілі інструменти для управління побутовими пристроями і сценаріями, що сприяє підвищенню комфорту та економії ресурсів.

3.8 Результат роботи системи

У рамках реалізації та тестування системи багатозадачного керування побутовими приладами, що базується на мікрокомп'ютері Raspberry Pi та синхронізована з Android-додатком, було виконано випробування для оцінки ефективності та функціональності системи. Ці тести були спрямовані на перевірку можливості дистанційного керування побутовими пристроями та на підтвердження ефективності розроблених алгоритмів автоматизації та користувацьких сценаріїв.

Першим кроком у тестуванні системи було створення нового облікового запису TEST_USER, що дало змогу використовувати всі функції мобільного застосунку. Реєстрацію користувача було виконано за допомогою спеціально розробленого скрипта на Python, який давав змогу вводити ім'я користувача та пароль через консольний інтерфейс. Цей скрипт, розташований в одній директорії з сервером, реєстрував дані нового користувача в базі даних, після чого завершував свою роботу. У разі, якщо сервер не було запущено до реєстрації користувача, необхідно було його ініціювати. Сервер автоматично запускається під час увімкнення мікрокомп'ютера завдяки службовій програмі crontab, налаштованій на активацію сервера під час кожного запуску системи.

Для авторизації в мобільному застосунку після успішної реєстрації потрібно ввести IP-адресу пристрою і пароль (рис. 3.13).

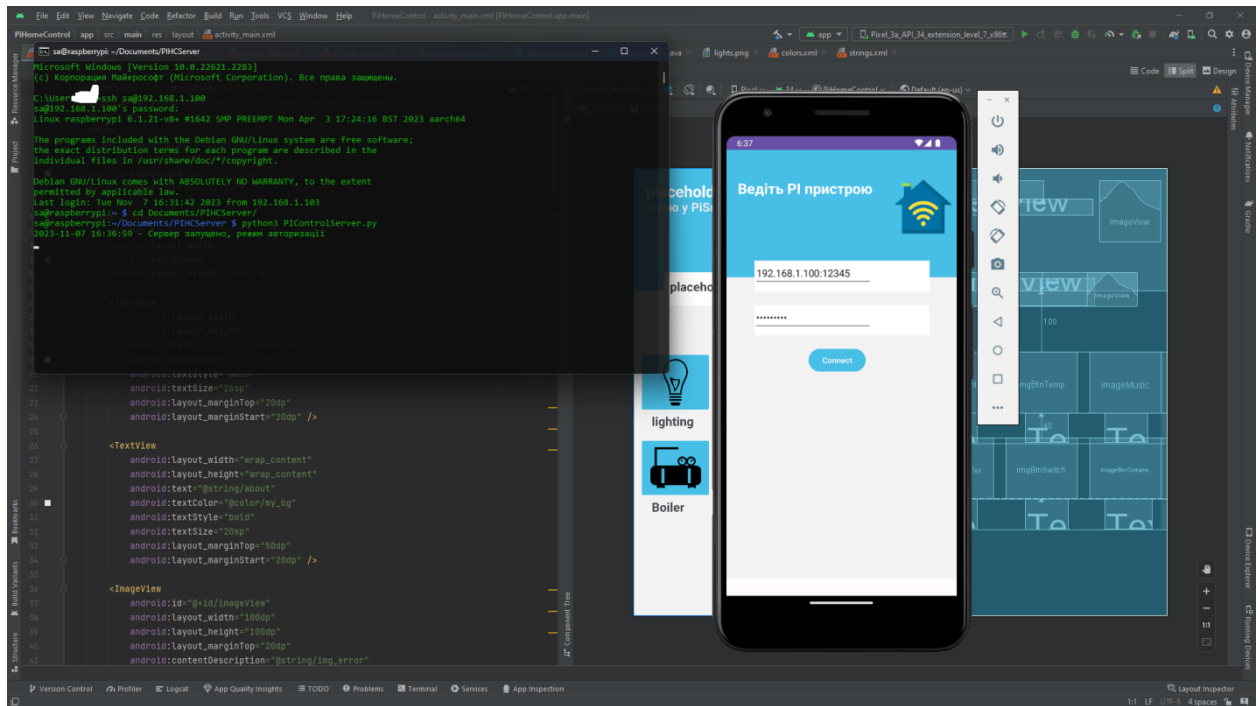


Рисунок 3.13 –Запуск сервера і введення даних авторизації у клієнті

Якщо пароль присутній у БД, система ідентифікує користувача і перенаправляє його в головне меню управління пристроями будинку. Успішну авторизацію та підключення до системи розумного будинку через мобільний застосунок було підтверджено: зв'язок із основними процесами серверу встановлено (рис. 3.14), і дані про поточний стан системи відображено на екрані смартфона, що свідчить про успішне підключення.

Далі було проведено тест управління роботою бойлера. За допомогою мобільного застосунку користувач створив сценарій, що наказує ввімкнення бойлера в певний час. Введення сценарію «WE 16:53-16:54» ініціювало надсилання команди на сервер, де її було перехоплено і занесено в базу даних. Підтвердження про прийняття сценарію до виконання відображалося у вигляді ехо-повідомлення, спеціально запрограмованого для тестування системи. На рисунку 3.15 демонструється функціонування механізму запису налаштувань користувача у БД.

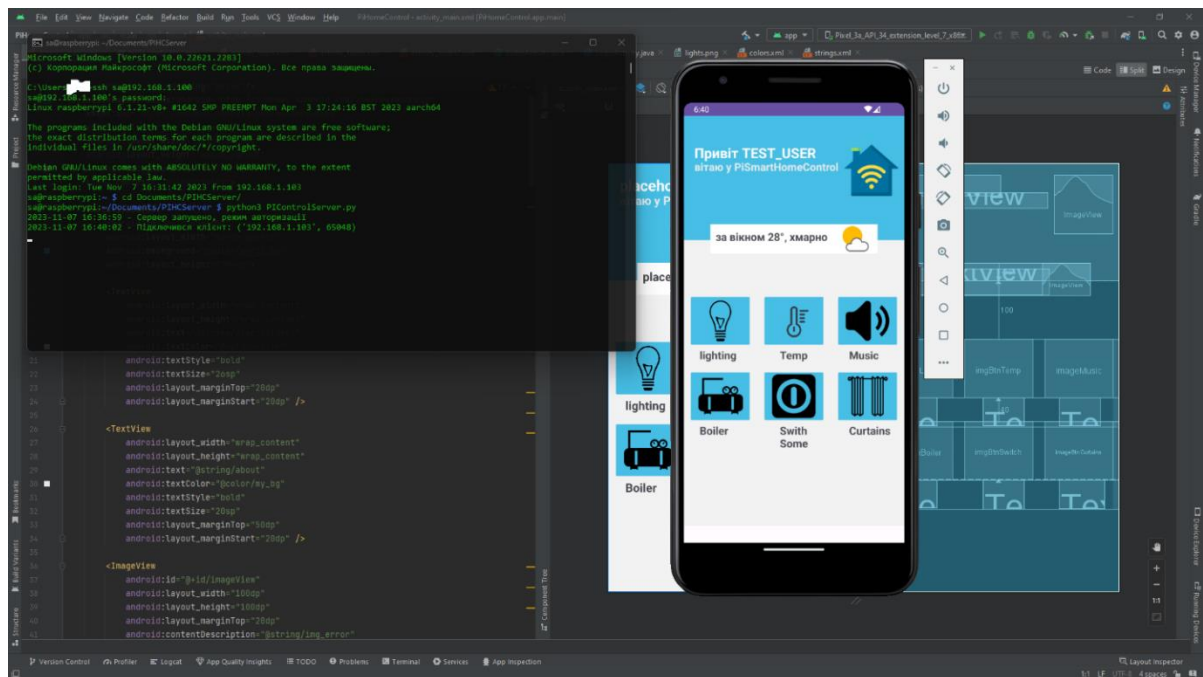


Рисунок 3.14 – Результат успішної авторизації

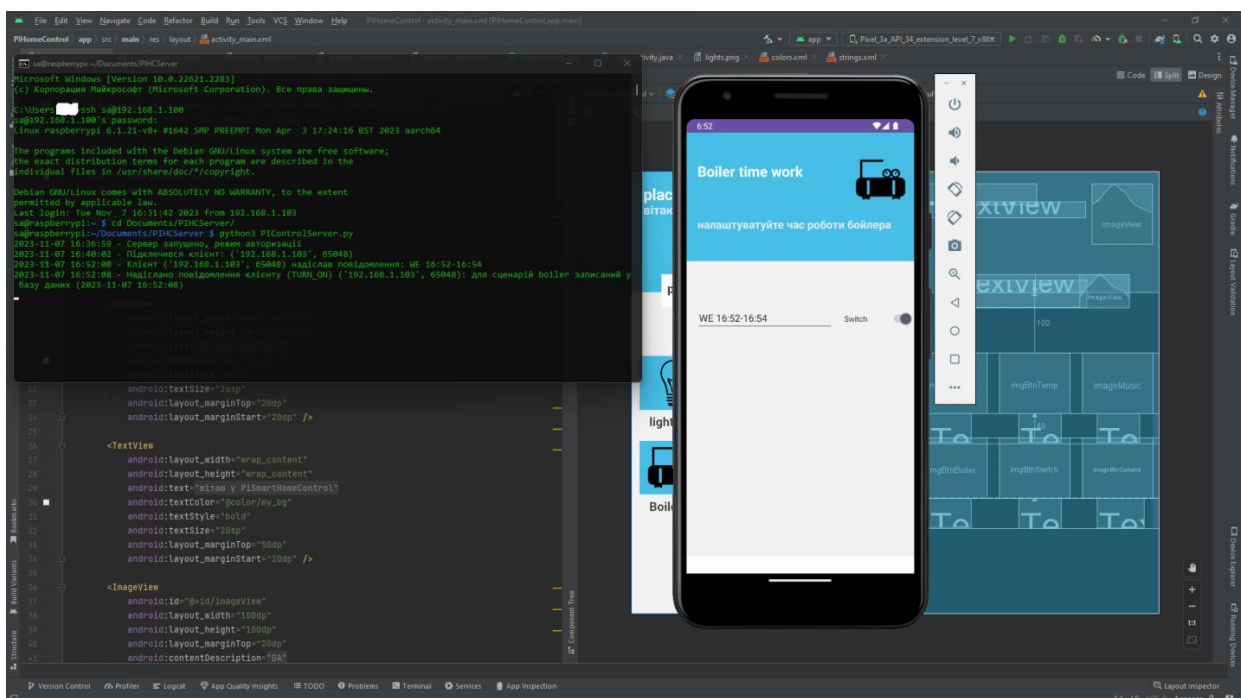


Рисунок 3.15 – Запис сценарію до БД сервером написаного користувачем android додатка

У встановлений для експерименту час, визначений сценарієм "«WE 16:53-16:54», система автоматично активувала бойлер рівно на дві хвилини. Цей час було обрано спеціально, щоб перевірити періодичну роботу сценарію

управління бойлером. Моніторинг активності бойлера здійснювався через відстеження подачі логічного сигналу "одиниці" на відповідній ніжці GPIO мікрокомп'ютера. Таким чином, точно в призначений інтервал часу система демонструвала планомірне увімкнення і подальше вимкнення бойлера, підтверджуючи реалізацію функції завдання періодично виконуваних операцій (рис. 3.16).

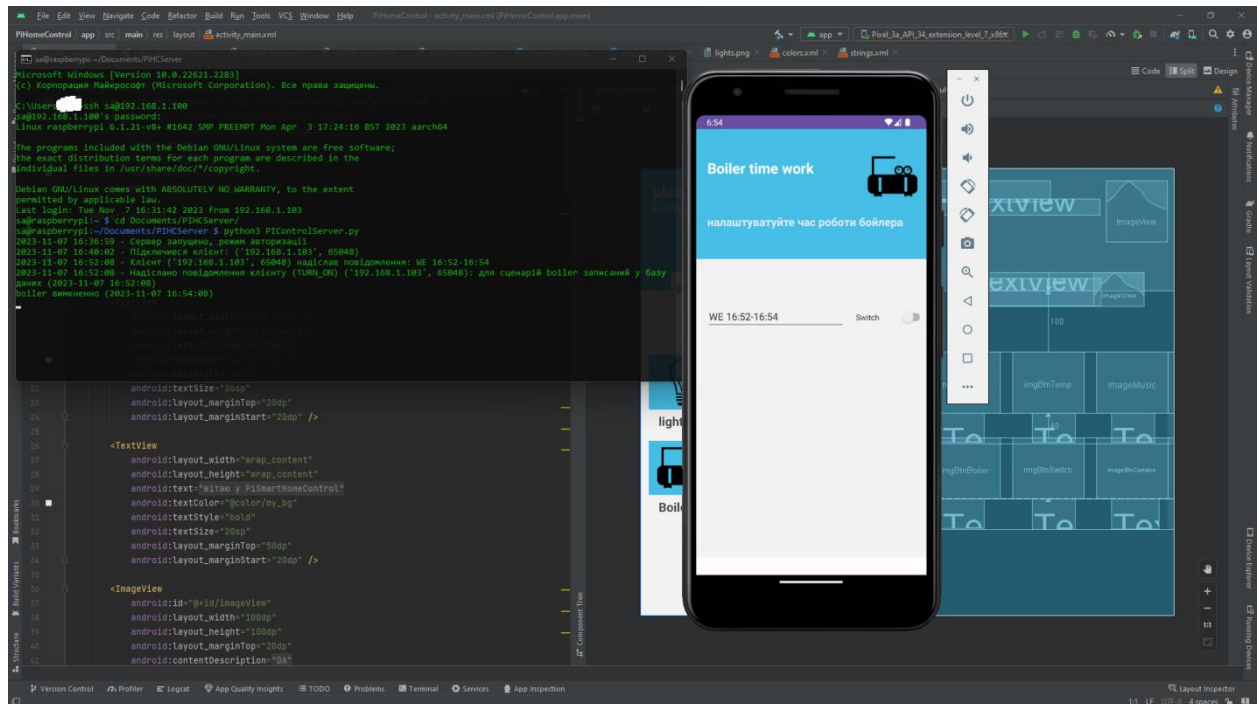


Рисунок 3.16 – Відключення бойлеру за прописаним сценарієм

Заключний етап демонстрації роботи системи, полягав у демонстрації можливості миттєвого керування пристроями без використання попередньо заданих сценаріїв. Під час натискання кнопки керування на екрані мобільного застосунку команда передавалася на сервер і оброблялася головним алгоритмом системи. Це призводило до зміни логічного сигналу на виході відповідного піна GPIO мікрокомп'ютера, який у разі керування однофазним твердотільним реле означав би ввімкнення або вимкнення подачі живлення на бойлер.

Статус зміни стану пристрою відображався на екрані смартфона зміною статусу перемикача з певною затримкою, оскільки оновлення статусу перемикача

здійснювалося після отримання підтвердження про зміну стану піна на мікрокомп'ютері, що доводило контрольованість системи розумного будинку.

Таким чином, результати експериментальної перевірки підтвердили високу ефективність і надійність розробленої системи багатозадачного керування побутовими приладами. Система здатна оперативно реагувати на дії користувача і коректно виконувати задані сценарії, забезпечуючи тим самим підвищений рівень комфорту та економічності в домашньому господарстві.

ВИСНОВКИ

У рамках магістерської роботи було розроблено інноваційну систему для керування побутовими приладами в домашній автоматизації з використанням мікрокомп'ютера Raspberry Pi. Ця система унікально об'єднує інтерфейси Bluetooth і Wi-Fi для зв'язку, а також містить у собі підключення твердотільних реле до GPIO портів. Створений спеціалізований Android-додаток демонструє високу надійність і ефективність в управлінні, що було підтверджено під час тестування.

Процес реєстрації користувачів у системі реалізовано через спеціалізований Python-скрипт. Крім того, серверна частина системи, що відповідає за виконання ключових алгоритмів управління, також розроблена на Python. Це забезпечує гладку інтеграцію і високу продуктивність.

Аналізуючи наявні рішення в галузі розумних будинків та інтерфейсів передавання даних, було обрано найефективніші та оптимальніші способи керування. Система гармонійно вписується в екосистему розумних пристроїв, полегшуючи процес встановлення та налаштування. Завдяки використанню стандартних протоколів та інтерфейсів, система легко адаптується до різних електрощитів і пристроїв будинку.

Важливим досягненням цієї роботи є створення універсальної та масштабованої системи, що об'єднує різноманітні модулі в єдине ціле. Це забезпечує не тільки базове керування побутовими приладами, а й дає змогу розширювати функціональність системи шляхом інтеграції додаткових модулів і сервісів. Такий підхід значно підвищує загальну ефективність і зручність використання системи, роблячи її адаптивною до індивідуальних потреб користувача.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. The history of smart homes (from the 1900s to today). Smart Home Point. URL: <https://www.smarthomepoint.com/history/> (date of access: 02.11.2023).
2. Система Ajax – сигналізація та smart-системи. ajax.systems. URL: <https://ajax.systems/ua/> (дата звернення: 15.11.2023).
3. BroadLink global store. Broadlink Global Store. URL: <https://ebroadlink.com/> (date of access: 17.11.2023).
4. Products. smarthings. URL: <https://www.smarthings.com/> (date of access: 15.09.2023).
5. Laan N. Review: fibaro home center 3. The Smart Home Blog. URL: <https://www.thesmarthomeblog.com/post/review-fibaro-home-center-3/> (date of access: 18.09.2023).
6. Juniper A. The smart smart home handbook: control your home with your voice : підручник. 2nd ed. London : Ilex Press, 2018. 160 p.
7. Vandome N. Smart Homes in easy steps: master smart technology for your home : підручник. 2nd ed. London : In Easy Steps Limited, 2018. 192 p.
8. Howe A. The Smarthome Book: simple ideas to assist with your smarthome renovation : підручник. 2nd ed. London : Andrew Howe, 2018. 122 p.
9. Zhang L. Fault detection method for wi-fi-based smart home devices. Wireless communications and mobile computing. 2022. Vol. 2, no. 2. P. 12. URL: <https://doi.org/10.1155/2022/4328307> (date of access: 02.10.2023).
10. Riman C. Heap bluetooth solution for smart controlled home devices. Journal of mobile multimedia. 2019. Vol. 2, no. 2. P. 7. URL: <https://doi.org/10.3844/JMRSP.2019.589.595> (date of access: 01.11.2023).
11. How HC-SR501 PIR sensor works & interface it with arduino. Last Minute Engineers. URL: <https://lastminuteengineers.com/pir-sensor-arduino-tutorial/> (date of access: 13.09.2023).

12. DHT11 humidity & temperature sensor : datasheet. 10 p. URL: <https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf> (date of access: 20.09.2023).
13. Lutz M. Learning python, : підручник. 3rd ed. O'Reilly Media, 2013. 1643 p.
14. Oracle database 21c. docs.oracle.com. URL: <https://docs.oracle.com/en/database/oracle/oracle-database/21/> (date of access: 13.09.2023).
15. Explore SQL server 2022 capabilities. microsoft. URL: <https://www.microsoft.com/en-us/sql-server/sql-server-2022> (date of access: 19.09.2023).
16. SSH extension for visual studio code. Visual Studio Code. URL: <https://code.visualstudio.com/docs/remote/ssh> (date of access: 09.08.2023).
17. How to SSH into Raspberry Pi. OnLogic. URL: <https://www.onlogic.com/company/io-hub/how-to-ssh-into-raspberry-pi/> (date of access: 28.07.2023).
18. Verhagen J. "Socket programming in python (guide). Real Python. URL: <https://realpython.com/python-sockets/> (date of access: 13.09.2023).
19. Pandey P. A complete guide to socket programming in python. DataCamp. URL: <https://www.datacamp.com/tutorial/a-complete-guide-to-socket-programming-in-python> (date of access: 06.10.2023).
20. Beaulieu A. Learning SQL: generate, manipulate, and retrieve data 3rd edition : підручник. 3rd ed. Sebastopol : O'Reilly Media, 2020. 377 p.
21. Basu S. Learn sqlite with python in 24 hours for beginners - simple, concise & easy guide to using database with python : підручник. 2nd ed. 2021. 80 p.
22. Burd B. Android application development all-in-one for dummies : підручник. 2nd ed. For Dummies, 2020. 800 p.
23. Effective java : підручник. 3rd ed. Addison-Wesley Professional, 2017. 461 p.

24. Loy M. Learning java: an introduction to real-world programming : підручник. 6th ed. Cambridge : O'Reilly Media, 2023. 549 p.
25. Samoylov N. Learn Java 17 Programming: Learn the fundamentals of Java Programming with this updated guide with the latest features, 2nd Edition : підручник. 2nd ed. Packt Publishing, 2022. 748 p.

ДОДАТОК А

Головний сервер системи

Лістинг А.1 – Програма server.py

```
import socket
import sqlite3
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)
DEVICES = {
    "lighting": 17,
    "temp": 18,
    "Music": 19,
    "Boiler": 20,
    "SwitchHome": 21,
    "Curtains": 22
}

for pin in DEVICES.values():
    GPIO.setup(pin, GPIO.OUT)

# Створення бази даних
conn = sqlite3.connect('smart_home.db')
cursor = conn.cursor()
cursor.execute('''
CREATE TABLE IF NOT EXISTS users (
    ip TEXT PRIMARY KEY,
    password TEXT
)
''')
conn.commit()

def handle_client(client_socket):
    # Аутентифікація користувача
    ip = client_socket.recv(1024).decode('utf-8')
    password = client_socket.recv(1024).decode('utf-8')

    cursor.execute("SELECT * FROM users WHERE ip=? AND password=?", (ip, password))
    user = cursor.fetchone()

    if not user:
        client_socket.send("Authentication
failed.".encode('utf-8'))
        client_socket.close()
        return
```

```

        client_socket.send("Authentication
successful.".encode('utf-8'))

    # Прослуховування команд від клієнта
    while True:
        command = client_socket.recv(1024).decode('utf-
8').split()

        if not command:
            break

        device, action = command[0], command[1:]

        if device in DEVICES:
            if "WE" in action:
                GPIO.output(DEVICES[device], GPIO.HIGH)
            else:
                GPIO.output(DEVICES[device], GPIO.LOW)

    client_socket.close()

def main():
    server = socket.socket(socket.AF_INET,
socket.SOCK_STREAM)
    server.bind(('0.0.0.0', 9999))
    server.listen(5)
    print("Server listening on port 9999")

    while True:
        client, addr = server.accept()
        print(f"Accepted connection from
{addr[0]}:{addr[1]}")
        handle_client(client)

if __name__ == '__main__':
    try:
        main()
    except KeyboardInterrupt:
        print("\nShutting down...")
        GPIO.cleanup()

```

ДОДАТОК Б

Головне меню

Лістинг Б.1 – Програма MainActivity.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/my_bg"
    tools:context=".MainActivity">

    <RelativeLayout
        android:id="@+id/UpperBG"
        android:layout_width="match_parent"
        android:background="@color/myfil_bg"
        android:layout_height="200dp">

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/placeholderT"
            android:textColor="@color/white"
            android:textStyle="bold"
            android:textSize="26sp"
            android:layout_marginTop="20dp"
            android:layout_marginStart="20dp" />

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/about"
            android:textColor="@color/my_bg"
            android:textStyle="bold"
            android:textSize="20sp"
            android:layout_marginTop="50dp"
            android:layout_marginStart="20dp" />

        <ImageView
            android:id="@+id/imageView"
            android:layout_width="100dp"
            android:layout_height="100dp"
            android:layout_marginTop="20dp"
            android:contentDescription="@string/img_error"
            android:layout_marginStart="300dp"
```

```

        app:srcCompat="@drawable/smart_home" />

</RelativeLayout>

<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="55dp"
    android:outlineAmbientShadowColor="@color/white"
    android:layout_marginHorizontal="50dp"
    android:layout_marginTop="-30dp"
    android:background="@color/white"
    android:contentDescription="@string/img_error"
    android:layout_below="@+id/UpperBG">

    <TextView
        android:id="@+id/tempOutDorText"
        android:layout_width="232dp"
        android:layout_height="48dp"
        android:text="@string/placeholderT"
        android:textStyle="bold"
        android:textSize="20sp"
        android:layout_marginStart="10dp"
        android:layout_marginTop="10dp"

        />

    <ImageView
        android:id="@+id/imageView2"
        android:layout_width="100dp"
        android:layout_height="100dp"
        android:layout_marginStart="230dp"
        android:contentDescription="@string/img_error"
        app:srcCompat="@drawable/weather" />

</RelativeLayout>

<LinearLayout
    android:id="@+id/contentBTN_ONE"
    android:layout_width="match_parent"
    android:layout_height="100dp"
    android:layout_below="@+id/UpperBG"
    android:layout_marginTop="100dp"
    android:orientation="horizontal">

    <ImageButton
        android:id="@+id/imgBtnLights"
        android:layout_marginHorizontal="10dp"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:backgroundTint="@color/myfil_bg"
        android:scaleType="centerInside"
        android:contentDescription="@string/img_error"

```

```

        app:srcCompat="@drawable/lights" />

<ImageButton
    android:id="@+id/imgBtnTemp"
    android:layout_marginHorizontal="10dp"
    android:layout_width="match_parent"
    android:backgroundTint="@color/myfil_bg"
    android:layout_height="match_parent"
    android:layout_weight="1"
    android:scaleType="centerInside"
    android:contentDescription="@string/img_error"
    app:srcCompat="@drawable/temperature" />

<ImageButton
    android:id="@+id/imageMusic"
    android:layout_marginHorizontal="10dp"
    android:backgroundTint="@color/myfil_bg"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_weight="1"
    android:scaleType="centerInside"
    android:contentDescription="@string/img_error"
    app:srcCompat="@drawable/sounds" />

</LinearLayout>

<LinearLayout
    android:layout_below="@id/contentBTN_ONE"
    android:layout_width="match_parent"
    android:layout_height="50dp"
    android:orientation="horizontal">

    <TextView
        android:layout_width="70dp"
        android:layout_height="48dp"
        android:layout_marginHorizontal="30dp"
        android:text="lighting"
        android:textStyle="bold"
        android:textSize="20sp"
        />

    <TextView
        android:layout_width="70dp"
        android:layout_height="48dp"
        android:layout_marginHorizontal="49dp"
        android:text="Temp"
        android:textStyle="bold"
        android:textSize="20sp"
        />

    <TextView
        android:layout_width="70dp"
        android:layout_height="48dp"

```

```

        android:layout_marginHorizontal="10dp"
        android:text="Music"
        android:textStyle="bold"
        android:textSize="20sp"
    />

</LinearLayout>

<LinearLayout
    android:id="@+id/contentBTN_TWO"
    android:layout_width="match_parent"
    android:layout_marginTop="40dp"
    android:layout_height="100dp"
    android:layout_below="@id/contentBTN_ONE"
    android:orientation="horizontal">

    <ImageButton
        android:id="@+id/imgBtnBoiler"
        android:layout_marginHorizontal="10dp"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:backgroundTint="@color/myfil_bg"
        android:scaleType="centerInside"
        android:contentDescription="@string/img_error"
        app:srcCompat="@drawable/boiler3" />

    <ImageButton
        android:id="@+id/imgBtnSwitch"
        android:layout_marginHorizontal="10dp"
        android:layout_width="match_parent"
        android:backgroundTint="@color/myfil_bg"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:scaleType="centerInside"
        android:contentDescription="@string/img_error"
        app:srcCompat="@drawable/light_switch" />

    <ImageButton
        android:id="@+id/imageBtnCurtains"
        android:layout_marginHorizontal="10dp"
        android:backgroundTint="@color/myfil_bg"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:scaleType="centerInside"
        android:contentDescription="@string/img_error"
        app:srcCompat="@drawable/curtains" />

</LinearLayout>

<LinearLayout
    android:layout_below="@id/contentBTN_TWO"
    android:layout_width="match_parent"

```

```
        android:layout_height="50dp"
        android:orientation="horizontal">
```

```
<TextView
    android:layout_width="70dp"
    android:layout_height="48dp"
    android:layout_marginHorizontal="30dp"
    android:text="Boiler"
    android:textStyle="bold"
    android:textSize="20sp"
/>
```

```
<TextView
    android:layout_width="70dp"
    android:layout_height="50dp"
    android:layout_marginHorizontal="49dp"
    android:text="Swith Some"
    android:textStyle="bold"
    android:textSize="20sp"
/>
```

```
<TextView
    android:layout_width="78dp"
    android:layout_height="49dp"
    android:layout_marginHorizontal="10dp"
    android:text="Curtains"
    android:textStyle="bold"
    android:textSize="20sp"
/>
```

```
</LinearLayout>
</RelativeLayout>
```