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

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

Індивідуальний дослідницький проєкт

на здобуття ступеня бакалавра
за освітньо-професійною програмою «Інформаційне забезпечення роботехнічних систем»
спеціальності 126 «Інформаційні системи та технології»
на тему: «Мобільний застосунок для керування інтелектуальними пристроями розумного дому на базі операційної системи Android»

Виконала:
студентка IV курсу, групи ІК-82
Герц Таїсія Андріївна _____

Керівник:
Доцент
Ткач Михайло Мартинович _____

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

Київ – 2022 року

Пояснювальна записка
до індивідуального дослідницького проєкту на тему: «Мобільний застосунок для керування інтелектуальними пристроями розумного дому на базі операційної системи Android»

Київ – 2022 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 126 «Інформаційні системи та технології»

Освітньо-професійна програма «Інформаційне забезпечення роботехнічних систем»

ЗАВДАННЯ
на індивідуальний дослідницький проєкт студенту
Герц Таїсії Андріївни

1. Тема проєкту «Мобільний застосунок для керування інтелектуальними пристроями розумного дому на базі операційної системи Android», керівник проекту Ткач Михайло Мартинович, доцент.

2. Термін подання студентом проєкту: 15 червня 2022 року

3. Вихідні дані до проєкту: структура мобільного додатку для управління

пристроями розумного будинку.

4. Зміст пояснювальної записки:

Вступ

1. Огляд і аналіз існуючих мобільних додатків для управління інтелектуальними пристроями розумного дому.

2. Проектування системи

3. Розробка ПЗ

Висновок

5. Перелік графічного матеріалу (із зазначенням обов’язкових креслеників, плакатів, презентацій тощо): Схема архітектури системи (А3), схема взаємодії

між компонентами системи (А1), Алгоритм роботи системи (А1), ER-діаграма сутностей та відношень бази даних (А3), Інтерфейс програми (А1), Блок-схема алгоритмів програми (А3).

6. Дата видачі завдання 1 грудня 2021 року

Календарний план

№ з/п	Назва етапів виконання проєкту	Термін виконання етапів проєкту	Примітка
1	Ознайомлення з завданням	02.05.22 – 08.05.22	
2	Аналіз предметної області	09.05.22 – 15.05.22	
3	Аналіз існуючих рішень	16.05.22 – 22.05.22	
4	Проектування моделі	23.05.22 – 29.05.22	
5	Розробка програмного забезпечення	30.05.22 – 12.06.22	
6	Оформлення документації	13.06.22 – 14.06.22	
7	Норм. контроль		
8	Перевірка на співпадіння		
9	Захист		

Студент Таїсія ГЕРЦ

Керівник Михайло ТКАЧ

АНОТАЦІЯ

Пояснювальна записка проекту складається з трьох розділів, містить 2 таблиці, 8 додатків та 16 джерел – загалом 63 сторінок.

Об`єкт дослідження: мобільні застосунки для керування інтелектуальними пристроями розумного дому на базі операційної системи Android.

Мета проекту – створення додатку для управління датчиками і сенсорами системи «Розумний дім» для мобільних пристроїв.

«Розумний дім» – це система, що включає в себе дві складові, а саме: локальна мережа та її складові (сенсори, датчики і контролери) і хмарна мережа (сервер, що містить у собі бази даних, збереження файлів з логами і більшість обчислень).

Мобільних додаток даного проекту має всі необхідні можливості для забезпечення функціонування та управління компонентами «Розумний дім», а також для агрегації даних, формування статистики за результатами використання пристроїв і можливістю пріоритетних змін в системі з подальшим сповіщенням. Щоб система могла бути реалізована, були розглянуті вже існуючі рішення та визначені як основні недоліки, так і переваги. Внаслідок проведеної порівняльної характеристики за мету було поставлено створити власний мобільних додаток та виправити існуючі недоліки. Також було сформовано основні алгоритми роботи системи, структурна та архітектурна схеми і ER-діаграма сутностей бази даних.

КЛЮЧОВІ СЛОВА: РОЗУМНИЙ ДІМ, МОБІЛЬНИЙ ДОДАТОК, ANDROID, СЕРВЕР, БАЗА ДАНИХ, ХМАРНІ ТЕХНОЛОГІЇ, БЕЗПРОВІДНЕ З’ЄДНАННЯ.

ANNOTATION

The explanatory note of the diploma project consists of four sections, contains 2 tables, 8 applications and 16 sources - a total of 63 pages.

The object of study: mobile applications for managing smart devices for smart homes based on the Android operating system.

The main theme of the project is to create an add-on for controlling the sensors of the «Smart Home» system for mobile devices.

«Smart House» is a system which includes two stores, namely: the local network and its components (sensors, sensors and controllers) and the cloud network (the server which contains the data base, storage of log files and most of the calculations).

Mobile extension of this project has all the necessary capabilities to ensure the functioning and management of components of the smart home, as well as data aggregation, generation of statistics on the results of the use of devices and the possibility of prioritized changes in the system for further communication. So that the system could be implemented, the existing solutions were considered and both the main drawbacks and benefits were identified. As a result of the carried out comparative characteristics, the goal was to create our own mobile additive and to correct current drawbacks. The main algorithms of the system operation, structural and architectural schemes and ER-diagram of data base essences were also formed.

KEY WORDS: SMART HOUSE, MOBILE APPLICATION, ANDROID, SERVER, DATABASE, CLOUD TECHNOLOGIES, WIRELESS CONNECTION.

ЗМІСТ

ВСТУП

В останні десятиліття нашого життя спостерігається значний розвиток інформаційних технологій, які проникають у кожну сферу нашого життя. Окрім того, що робототехніка і штучний інтелект займають, що не найважливіше місце в таких галузях, як наука, екологія, історія, також вони покращують якість життя звичайних людей. Наразі складно уявити комфортне життя в суспільстві без використання інформаційних технологій і керування за допомогою їх можливостей значною кількість аспектів свого сучасного життя.

Оскільки людство завжди намагалося покращити умови для свого існування, створити їх більших комфортними, безпечними і простими, то XXI відкриває надзвичайні можливості завдяки розвитку технологій. Прикладом цього є система автоматизованого керування компонентами інфраструктури помешкання «Розумний дім». Дана система найчастіше включає в себе наступні складові:

- датчики та зовнішні пристрої – це сукупність всіх приладів помешкання, якими безпосередньо ведеться управління;
- точки управління – це пристрої, що дають можливість керувати системою (до них відносяться будь-які компоненти, які взаємодіють з системою)
- центральних контролер – це, так зване, «ядро системи», сюди надходить інформація від виконавчих пристроїв та проходить обробка команд від точок управління.

Найважливішим аспектом системи «Розумний дім», що створив значний попит на даний тип продукту, можна вважати можливість забезпечення безпеки не тільки житла, а його мешканців. Завдяки даній технології людина отримала унікальну можливість не перебуваючи у своєму житлі контролювати його. Оскільки система «Розумний дім» забезпечена аварійними датчиками та відеоспостереженням. Також завдяки створенню подібної системи людина може налаштувати і редагувати її під свої потреби.

Тема дипломного проекту набуває особливої актуальності не тільки через стрімкий розвиток інформаційних технологій, а також через ріст та підвищення навичок людей. До негативних аспектів даної платформи відноситься її відносно висока складність і вартість інтеграції. Та все ж з кожним роком це стає все менш помітним, оскільки ще десятиліття тому прототипи подібного програмного забезпечення були можливі лише у межах певних компаній та прив'язані до їх технічного потенціалу. Наразі ж з'являється все більше можливостей реалізувати таки систему власноруч.

Проривом в еволюції розвитку програмного забезпечення «Розумний дім» вважається поява в 2010 році перших версій плати Arduino. Даний девайс дав можливість в реалізації комплексних рішень з підключенням різноманітних датчиків, при цьому не вимагав високих знань в галузі програмування.

Мета дипломного проекту – розробка мобільного застосунку на базі операційної системи Android з можливістю автоматизації більшості функцій автономно. Наприклад, збір та аналіз даних з лічильників помешкання для підвищення комфорту та економії витрат у будинку.

1. АНАЛІЗ ОСНОВНОЇ ХАРАКТЕРИСТИКИ МОБІЛЬНИХ ЗАСТОСУНКІВ СИСТЕМИ «РОЗУМНИЙ ДІМ»

1.1 Огляд предметної області

Найбільш важлива проблема в житті людини – це її житло та все, що пов’язано з комфортним і безпечним існуванням у ньому. Отже коло проблем, які має вирішувати даний мобільний застосунок, засноване на фундаментальних принципах для знаходження найбільш якісного вирішення, що включає в себе забезпечення безпеки, зручності, простоти використання і ефективності.

Щоб отримати якісну реалізацію даної задумки, по перше потрібно дослідити основні принципи логіки роботи подібної системи, а також платформу для розробки інтерфейсу користувача, який має бути зручним і простим у налаштуванні і користуванні.

1.1.1 Система «Розумний дім»

«Розумний дім» (англ. «Smart House») – житловий будинок, вдосконалений для проживання завдяки автоматизації роботи його пристроїв, автоматизованого постачання необхідними для життя ресурсами, а також гарантії безпеки та комфорту для його мешканців[1].

Звернувшись до історії, варто зазначити патентування диммеру (dimmer) у 1961 році двома винахідниками Джоель і Рут Спіра, що і стало початком розвитку «Інтернету речей». Даний пристрій був розроблений для регулювання інтенсивності освітлення.

Щодо комплексної системи «Розумний дім», то вона включає в себе не тільки датчики і пристрої, що інтегровані в житлову інфраструктуру, а і носії для збору та аналізу даних повсякденного життя. Незважаючи на те, що система

використовує передові технології, вона має і ряд не вирішених проблем на даний момент, що пов’язані з управлінням і обробкою великих об’ємів даних, що надходять з різних датчиків, відмовостійкійсть є найбільшою з них.

В даному дипломному проекті використовується проста, але ефективна схема планування даних для управління вхідними пакетами даних із системи, що заснована на основі їх пріоритетів і типів. Контекстно-усвідомлена схема була досліджена в реальних умовах програмного забезпечення «Розумний дім». Дослідження проводилось за двома сценаріями: централізована (центральне ядро і сенсори, що мають зв'язок лише з ним) та розподілена (зв'язок підтримується в режимі «павутини», тобто кожен вузол має зв'язок як мінімум з двома іншими вузлами).

В ході даного експерименту було встановлено, що розподільна система має більше можливостей для реалізації поставлених задач. Оскільки гарантує високу пропускну здатність до даних з високим пріоритетом (Рис.1.1.), і в той же час, надає достатній доступ до даних з низьким пріоритетом без затримки.

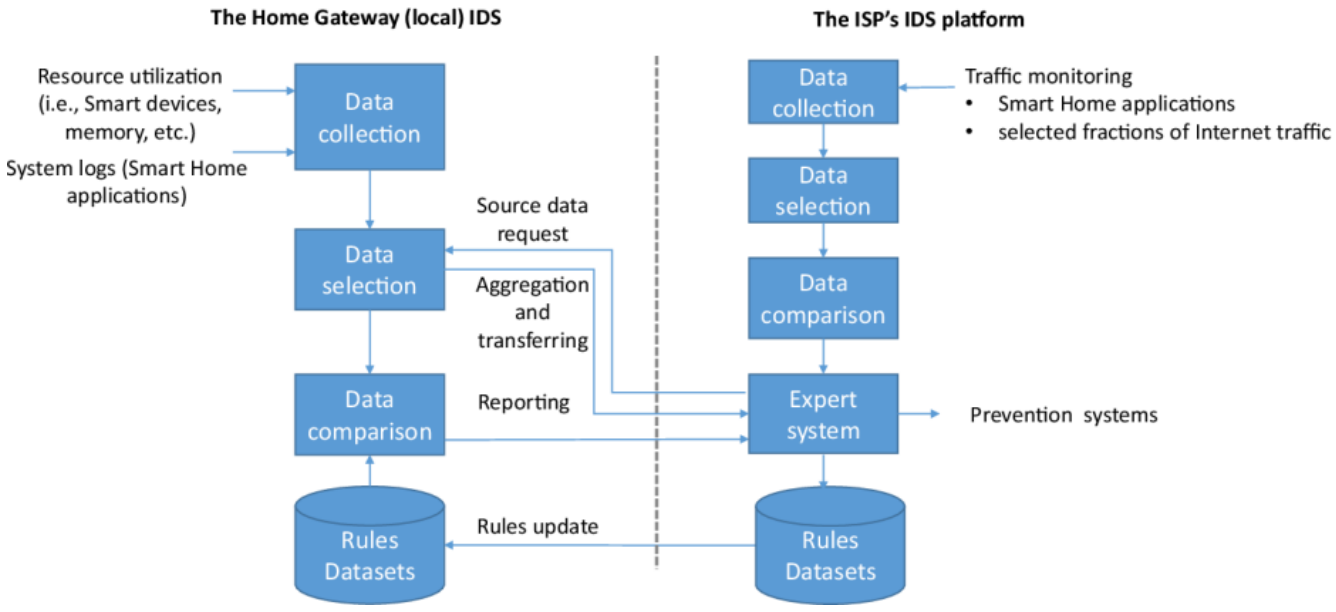

Рис. 1.1. Алгоритм пропуску даних в розподілених системах

Остаточно концепція «Інтернет речей» була сформована лише у 1999 р. Суть даної концепції полягає в тому, щоб за допомогою Інтернету, надати можливість взаємодії різних пристроїв, кожен з яких отримає свою власну унікальну IP-адресу [9]. Таким чином всі пристрої можна буде пов'язати в єдину мережу для взаємодії. Визначення «пристрій» використовується в даному випадку для всіх компонентів дому, включаючи вимикачі, замки, лампочки, а не тільки девайси з реалізованою логікою.

Потрібно відмітити, що одним з найважливіших факторів в роботі даного застосунку є його надійність і відмовостійких при виникненні критичних умов, що можуть надходити ззовні системи, наприклад, недостатня кількість електроенергії для нормального функціонування всіх компонентів системи.

Також потрібно зазначити, що акцент ставиться і на ефективну та швидку взаємодію всіх компонентів системи. При розробці подібного типу програмного забезпечення, важливим є і варіанти комунікації пристроїв між собою, оскільки вони створюють між собою мережу, не важливо, працюючи командно чи автоматично[3].

Також важливо зауважити про створення такого проекту як «Connected Home over IP» (сумісні зусилля Apple, Amazon, Google та Zigbee Aliance) [10]. Суть даної розробки – створення єдиного стандарту підключення для збільшенні сумісності продуктів різних торгових марок, що полягає у використання хмарних сховищ, де основних керуючий елемент – це смартфон з встановленим застосунком, що взаємодіє з API всіх елементів системи: підключених пристроїв і хмарного сховища, використовуючи зашифровані повідомлення (Рис.1.2.).

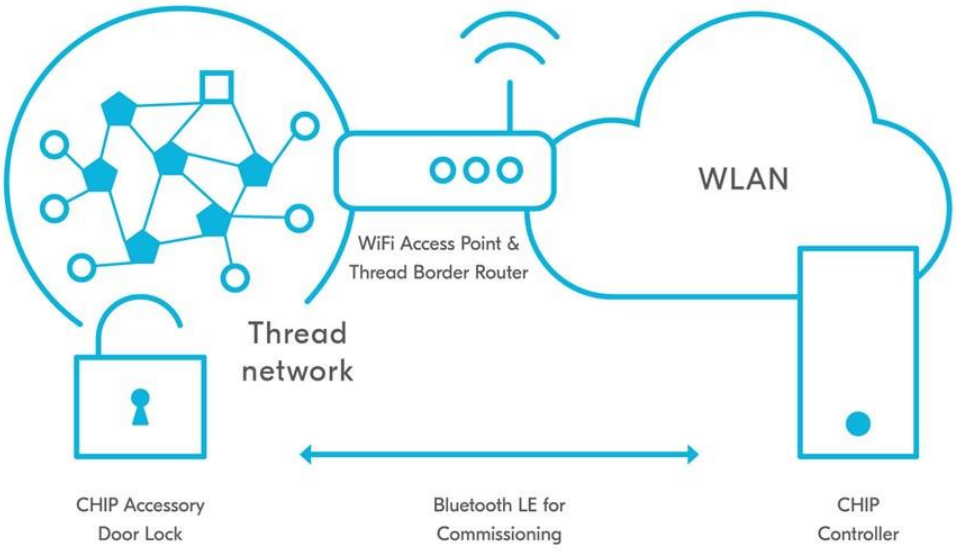

Рис. 1.2. Схема взаємодії між застосунком та пристроями «Connected Home over IP»

Вже у 2017 р. до єдиної мережі у загальному було підключено понад 20 млрд. пристроїв, а вже у 2020 р. їх кількість перевищила 50 млрд.

На даний момент використовується декілька варіантів мережевих протоколів, що дають можливість комунікації між пристроями у системі. Далі в роботі буде розглянуто їх переваги і недоліки:

- Bluetooth – це одна з найстаріших бездротових технологій, що використовується на всіх мобільних девайсах [11]. На жаль, даний протокол рідко використовується у системі «Розумний дім», оскільки має невеликий діапазон з'єднання (до 30м), а також використовує частоту передачі даних 2.4гГц, яка тягне за собою проблеми у перешкоджанні проходження інформації іншими пристроями;
- Wi-Fi – це технологія баздротової локальної мережі, де основними перевагами даної технології є низька ціна пристроїв, однак

найчастіший діапазон роботи, як і у Bluetooth, є традиційні 2.4гГц, тому також сприятливий для перешкод у вигляді інших пристроїв [12]. До недоліків технології Wi-Fi можна віднести і не висока захищеність сигналу, незважаючи на постійні оновлення функцій безпеки мережі Wi-Fi, постійно виникають і нові шляхи їх обходу. Також потрібно зазначити, що інші девайси не можуть повторно перенести сигнал з маршрутизатора, оскільки мережа Wi-Fi має обмежене покриття;

- Z-Wave - у 2001 році даний протокол був розроблений компанією Z-Wave Alliance[4] спеціально для системи «Розумний дім» і активно підтримується до сьогоднішнього часу. Його основною перевагою є можливість пов'язувати для комунікації між собою всі пристрої системи без необхідного зв'язку з центральним концентратором. Таким чином до мережі Інтернет буде підключений лише смартфон, а всі інші пристрої системи будуть працювати на протоколах Z-Wave, щоб надсилати пакети даних на смартфон (концентратор).

Таким чином технологія «Розумний дім» – це ланка, що пов'язує людину з пристроями в її помешканні, що збирають, обробляють та передають інформацію з них до бази даних для генерації статистики у подальшому (Рис. 1.3.). Висновком даної частини можна визначити не тільки шалений розвиток інформаційних технологій, в якій система «Розумний дім» займає одне з визначних місць, оскільки реалізує потреби людини в спрощенні комфортного існування і покращення рівня життя. Незважаючи на те, що дана технологія, ще достатньо нова у розвитку інформаційної галузі, проте вона стрімко набирає популярність у суспільстві, завдяки своїй універсальності, а також має можливість адаптації під технологічні стандарти не тільки різних країн, а і конкретних груп людей.

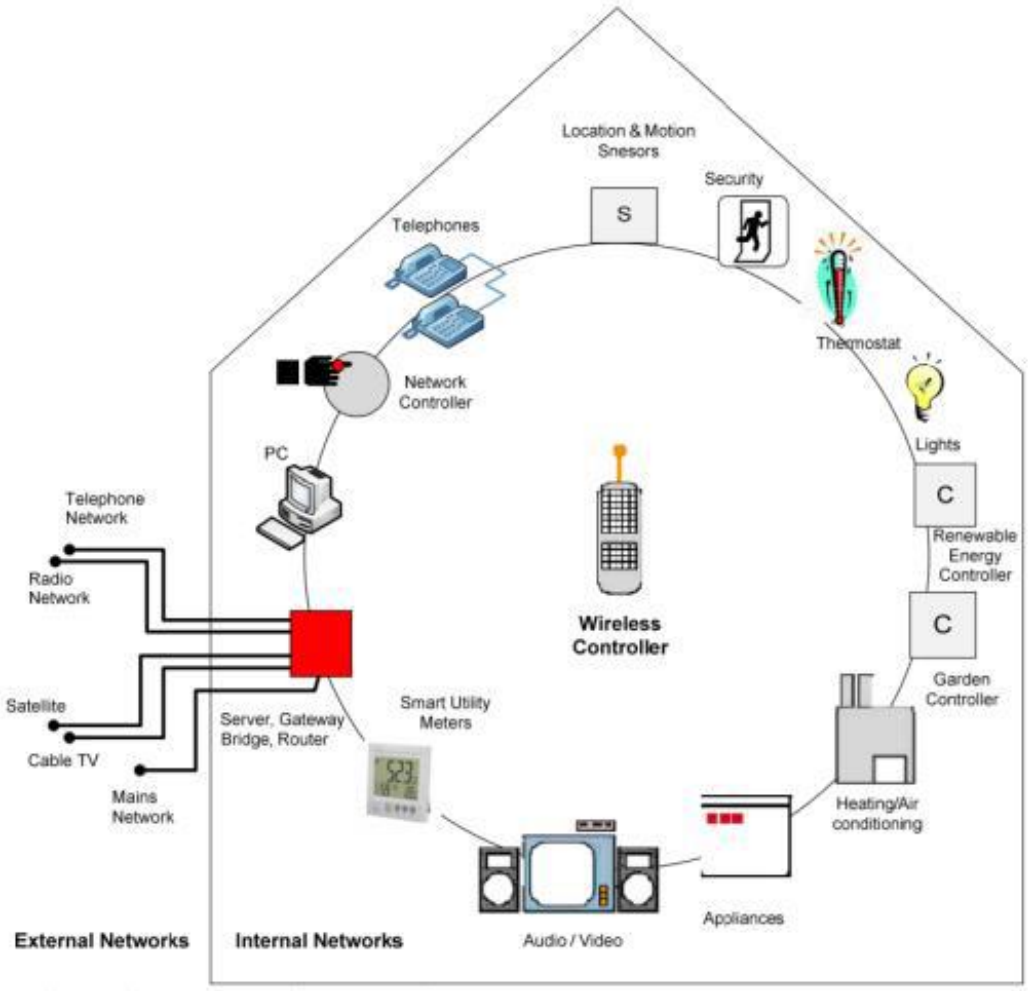

Рис. 1.3. Усі пристрої системи «Розумний дім» і передача інформації про них до бази даних для генерації статистики.

1.1.2 Операційна система Android

Android ОС – це операційна система для, що створена на базі ядра ОС Linux та власної реалізації Java від компанії Google. Використовується не тільки у смартфонах, а і у інших девайсах. Також дана операційна система вважається найбільш популярною у світі, оскільки 85% мобільних пристроїв працюють на ній. Щороку продається більше мільярда пристроїв на базі Android[5].

23 вересня 2008 р. офіційно вийшла перша версія операційної системи, а також і перший пакет розробки Android SDK 1.0. У 2014 р. була зафіксована

перша взаємодія операційної системи Android з пристроями системи «Розумний дім». В той же час вийшов анонс підсистеми Android Wear. У 2019 році на презентації Google I/O була представлена операційна система ОС Android Auto (для автомобілів) та Android TV (для телевізорів). Всі застосунки на даній ОС написані в байт-коді спеціального формату APK для віртуальної машини Dalvik. Наразі для загального користування є доступ в велику кількість бібліотек для роботи з такими програмами: Android Native, Bionic,а також мультимедійні бібліотеки на базі PacketVideo OpenCORE, WebKit та ін.

В даній роботі була вибрана саме ОС Android, оскільки вона має велике різноманіття середовищ (IDE), що дає можливість реалізувати рішення найпростішим шляхом у зв'язку з використанням додаткових засобів, які не тільки полегшують роботу, а і покращують продуктивність. Android Studio – є однією з найбільш популярних і функціональних на даний момент [13]. Даний застосунок було обрано в дипломному проекті, оскільки він має функцію редагування включає коду з можливістю автозаміни, ПЗ для редагування графічного інтерфейсу, емулятор для тестування застосунків та багато інших програм. Також в дане середовище складається з пакетів, які необхідні для написання коду, що адаптовано під різні версії ОС, і враховують властивості кожної з них.

1.2 Аналіз та порівняльна характеристика вже існуючих систем

Оскільки «Розумний дім» набирає широкої популярності у глобальному просторі, а також спостерігається швидкий розвиток в технологічній галузі, останнім часом була розроблена велика кількість застосунків для оптимізації та керування подібними технологіями.

В наступних розділах, увага буде привернута, на жаль, лише до чотирьох застосунків, що є найбільш популярними на даний момент. Також буде

проведено порівняльний аналіз їх основних характеристик – множину пристроїв, якими вони дозволяють керувати, протокол зв’язку смартфону з пристроями та простоту керування процесами.

1.2.1 Loxone App

Loxone – це відносно новий застосунок, що був розроблений на базі ОС Android командою програмістів Масачусетського Технологічного Інституту у 2016 році. Даний застосунок швидко набрав популярність в аудиторії завдяки простоті у використанні, надійності, довговічності та зручному інтерфейсу (Рис. 1.4).

Він дає можливість доволі легко керувати пристроями системи «Розумний дім» («Loxone Smart House») за допомогою адаптованого Wi-Fi протоколу («loxone») через зв'язок з закритою системою Loxone Miniserver (Рис. 1.5.).

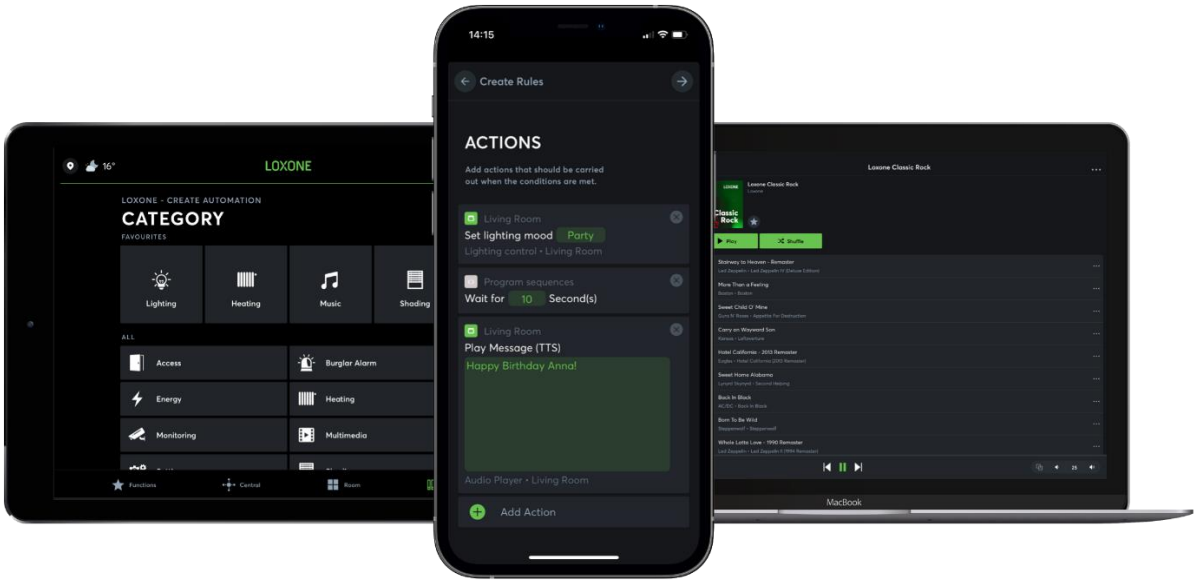

Рис. 1.4. Інтерфейс застосунку Loxone для користувачів

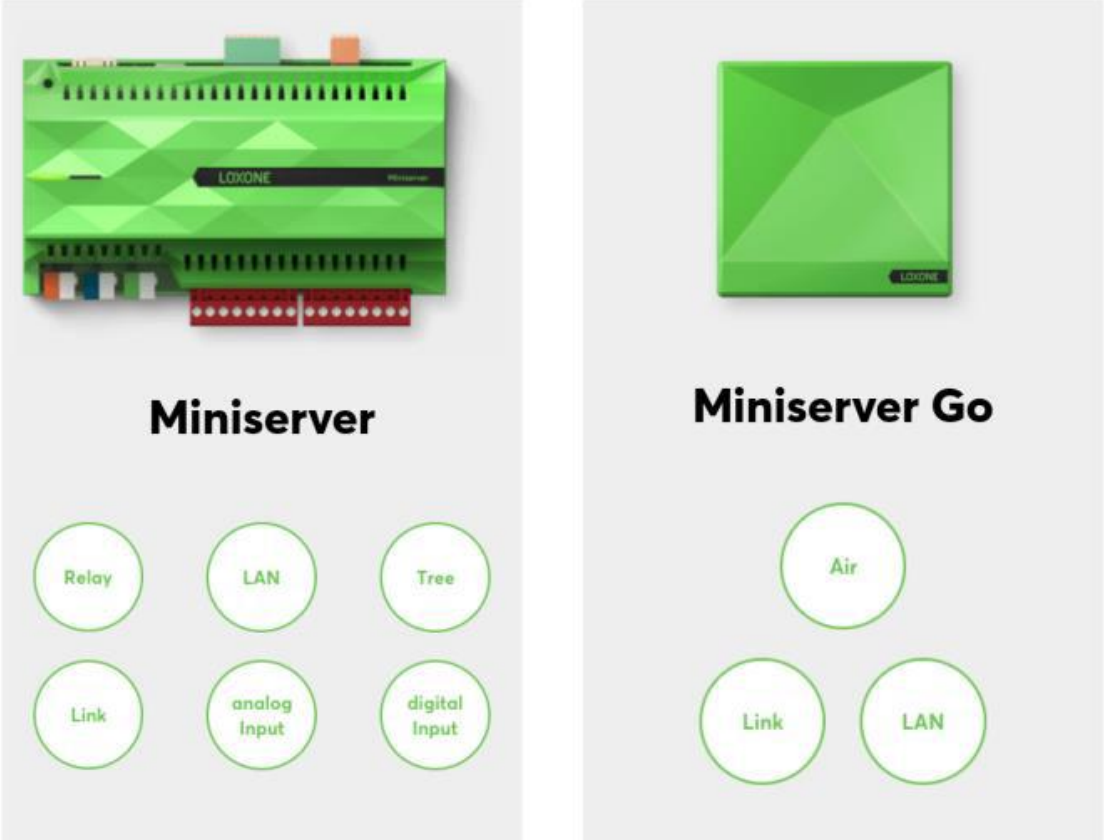

Рис. 1.5. Види пристроїв, до яких може підключатися гаджет – Loxone Miniserver і Loxone Miniserver Go.

Приклади роутів для підключення:

- loxone://ms?mac=<MAC>
- loxone://ms?host=<IP>
- loxone://ms?host=<IP>&usr=<USER>&pwd=<PASSWORD>
- loxone://ms?host=<URL>&usr=<USER>
- loxone://ms?host=<IP>&usr=<USER>&pwd=<PASSWORD>&loc=favorites
- loxone://ms?mac=<MAC>&loc=home
- loxone://ms?mac=<MAC>&loc=weather
- loxone://ms?mac=<MAC>&loc=room%2F<roomUUID>
- loxone://ms?mac=<MAC>&loc=category%2F<categoryUUID>

- loxone://ms?mac=<MAC>&loc=control%2F<controlUUID>

Параметри запитів, що використовуються у Loxone:

- host – IP-адреса або URL системи Loxone Miniserver
- mac – MAC адреса системи Loxone Miniserver
- usr/pwd – Дані для авторизації користувача
- loc – локація, тип або категорія пристроїв, до яких є можливість підключення.

Архітектура, що використовується у застосунку:

- мова розробки – C.

В даному застосунку використовується сервер з Web-Socket протоколом, що відповідає стандарту RFC6455[6], та можливістю прийняття HTTP-запитів. Використовуючи в застосунку спеціальний token (рядок з унікальними символами) відбувається авторизація підключення до елементів системи.

Безпеку у даному застосунку частково забезпечує стандарт хешування HMAC-SHA1 [14] при роботі з Web-Socket та SSL-шифрування, проте з відключенням, коли запит відправляється через Web-Socket протокол. Основною проблемою такого шифрування є значні об’єми обчислень, що зростають при збільшені кількості повідомлень, що в той же час дає навантаження на CPU.

Даний застосунок ідентифікує повідомлення за його першими восьми байтами (табл 1). Потім зчитуються метадані про дію, що має бути виконана, ідентифікатори елементів системи, куди має бути відправлена команда, та ідентифікатори користувачів, які надіслали повідомлення про зміну стану. Повідомлення передається в шістнадцятковій системі.

Таблиця 1 - Структура повідомлення у системі Loxone

1 байт	Префікс – завжди однаковий (0x03)

2 байт	Ідентифікатор повідомлення, що визначає його тип (приклад – 0 – текстове повідомлення, 1 – файл, 2 – event-таблиця станів пристроїв і тд.)
3 байт	Додаткова інформація про повідомлення (метадані, параметри, дата і час, статус пристрою тощо)
4 байт	Довжина інформації, що передаються у решті повідомлення

Інтерфейс Loxone Miniserver розрахований на наступні форми запитів: Plain Text JSON і XML. Збереження даних відбувається тільки на серверах Loxone Miniserver в зашифрованому вигляді, таким чином застосунок не використовує для збереження інформації хмарних сховищ та баз даних.

У застосунку використовуються такі функції, як:

- System Schematics – графічно зображує всі пристрої в помешканні. Окрім того дає можливість графічно відобразити системи та плани поверхів у застосунку Loxone. Таким чином дана функція дає можливість у застосунку Loxone Config завантажувати зображення і накладати на них функціональні блоки для взаємного зв’язку;
- Remote Connect – End-To-End зв’язок між програмним забезпеченням та елементами системи з будь-якого місця, використовуючи хмарний сервіс Loxone Config (рис. 1.6).

Дана функція була додана нещодавно. Вона розширила можливості застосунку до керування пристроями на відстані, а також використання технології «End-To-End Encryption» підвищує безпеку через шифрування даних (Рис. 1.7.).

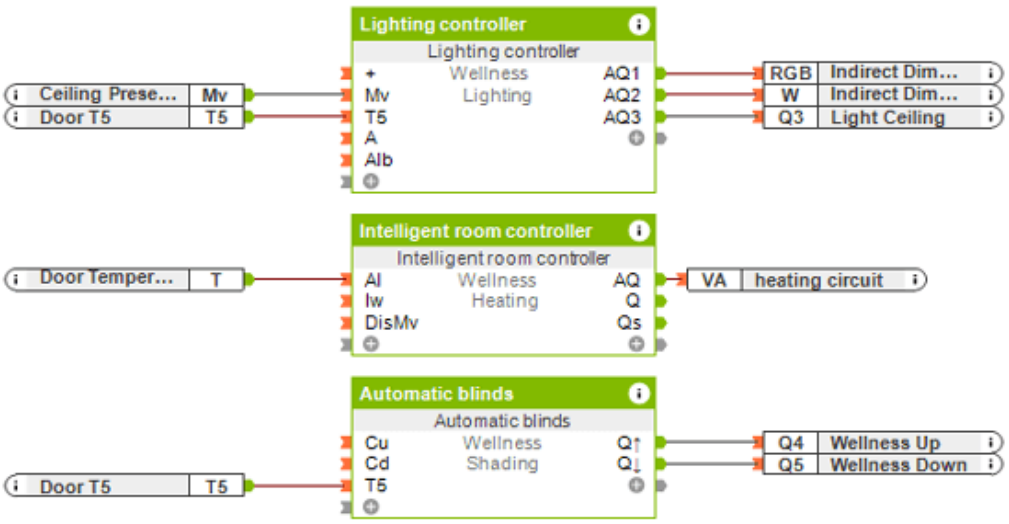

Рис. 1.6. Приклад конфігурації, що використовується у Loxone Config 8

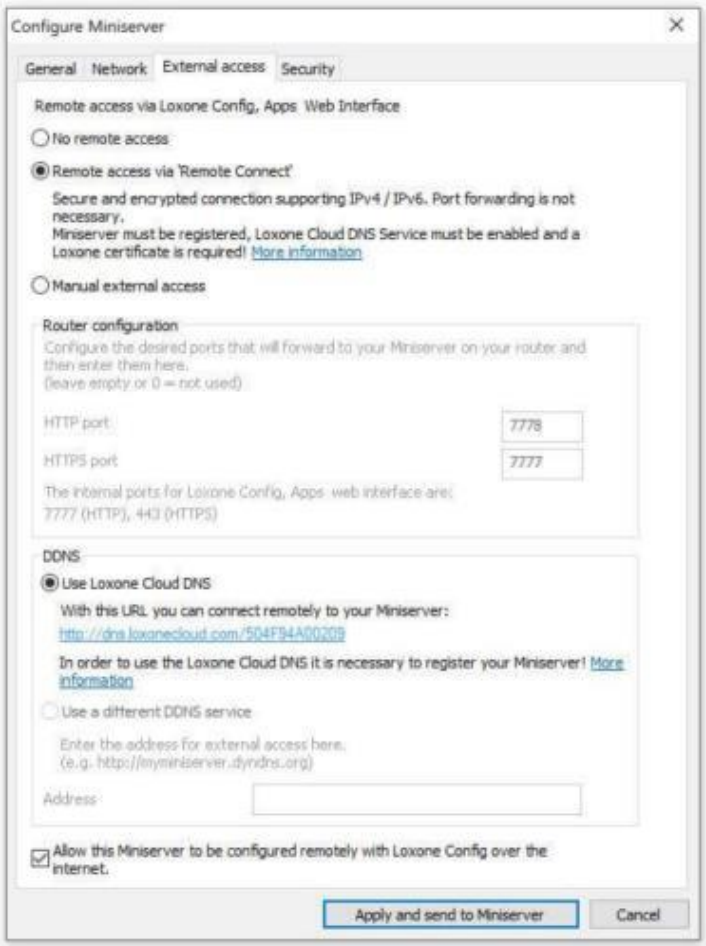

Рис. 1.7. Налаштування конфігурації підключення до Loxone Miniserver через зовнішній домен.

1.2.2 Houseinhand KNX

KNX – досить стара система «Розумний дім», що бере початок свого існування в дев'яностих роках минулого століття. Відмінністю даної системи є те, що вона децентралізована і використовує в своїй архітектурі «зіркову» топологію. Тобто кожен елемент системи має підключення напряму до плати живлення [16]. Дана система використовується як стандартна система автоматизації будинків, оскільки не пов'язана з конкретними виробниками чи брендами.

KNX Houseinhand – програмне забезпечення для контролю елементів системи віддалено, через підключення за допомогою протоколів Wi-Fi (Рис. 1.8.).

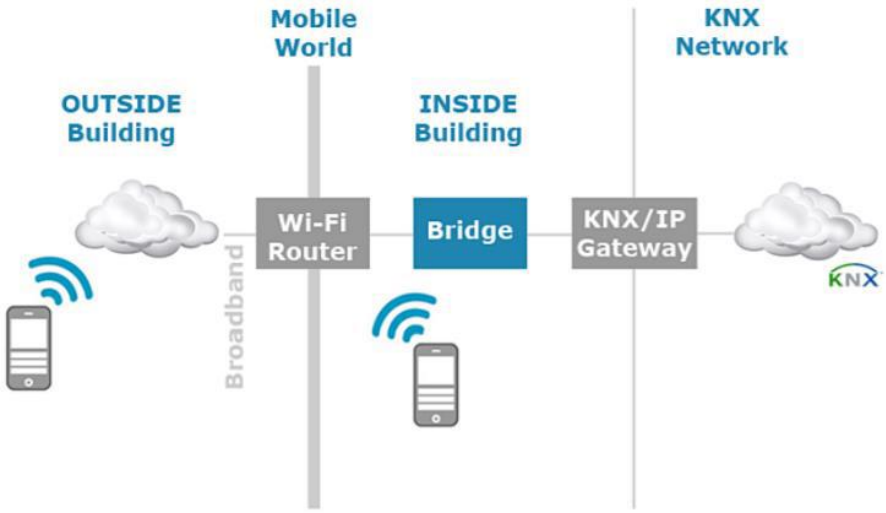

Рис. 1.8. Спрощена схема роботи системи KNX.

Щоб розпочати роботу з застосунком, користувач повинен створити обліковий запис на офіційній сторінці системи та отримати валідацію свого акаунту. Після цього, він зможе імпортувати файли конфігурації пристроїв системи (приклад profilename.hcf) до застосунку і подальшого їх відображення в

ньому. Сервіс BridgeCore (також Bridge) відповідає за комунікацію між елементами система та мобільним телефоном. BridgeCore – це внутрішній сервер KNX, який не тільки опрацьовує HTTP-запити, а також дешифрує інформацію в них [8] (Рис. 1.9.). Як приклад запиту до BridgeCore можна навести:

http://<BridgeCore IP Address or URL>/connection.jsp?

datawrapper=?&getData=1

Зробивши аналіз запиту, що представлений вище, можна отримати інформацію про адресу пристрою, а також необхідні дані для зворотного зв'язку.

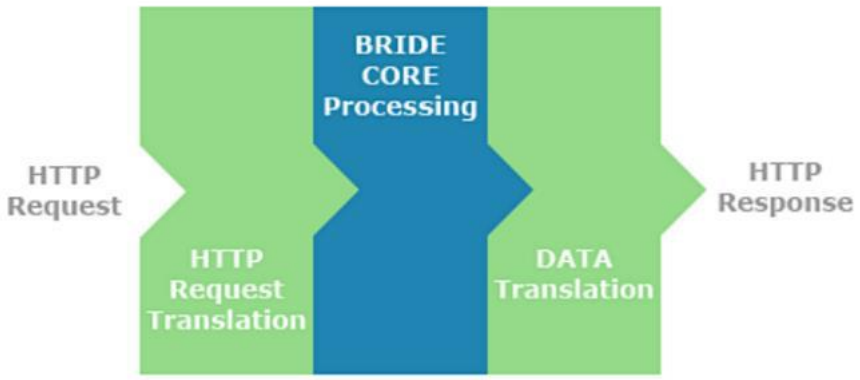

Рис. 1.9. Схема роботи BridgeCore

В даному програмному забезпеченні мобільний застосунок виконує роль графічного інтерфейсу, а сам сервіс BridgeCore виступає «ядром» системи.

Мови програмування, що були використані у даному застосунку, - Java, HTML, CSS. В той час для розробки сервісу BridgeCore використали тільки мову Java. За безпеку передачі інформації відповідає SSL-шифрування HTTP-запитів, що поєднано з внутрішнім шифруванням BridgeCore.

Позитивними функціоналом системи KNX можна вважати її універсальність. Тобто до системи можна підключити будь-який пристрій, що підключений до електромережі, це дає можливість розрити можливості

програмного забезпечення не тільки для одного помешкання, а й для використання у більш широкому спектрі.

1.2.3 Calaos for Android

Calaos – є системою з відкритим кодом, що на даний момент є одним з найбільш цілісних рішень в системі «Розумний дім». Даний застосунок був створений у 2013 р. однойменною французькою компанією. База коду випущена у форматі GPL та відкрита, таким чином даний проект отримує підтримку в спільності open-source розробників.

Calaos працює на окремій операційній ситемі Calaos OS, що створена на базі ядра Linux. Таким чином застосунок для мобільних пристроїв Calaos for Android представляє собою тільки графічний інтерфейс, що відкривають користувачу доступ до трьох наступних сервісів:

- calaos-server – «ядро» системи, відповідає за обчислення даних та збереження інформації;
- calaos-home – сервіс-посередник між calaos-server та елементами системи;
- calaos-webapp - веб-інтерфейс, дає можливість користувачу працювати з системою через веб-браузер.

У даному застосунку для запитів використовуються такі протоколи, як HTTP(S) та WS(S), а посилання виглядає наступним чином:

https://calaos_server_ip/api.php

Передача інформації відбувається у форматі JSON. Також calaos-webapp дає можливість проводити тестування запитів (Рис. 1.10.), після активації його наступною командою:

calaos_config set debug_enabled true

Далі користувач отримує доступ на сторінку для тестування:

https://calaos_server_ip/debug або ж http://calaos_server_ip:5454/debug

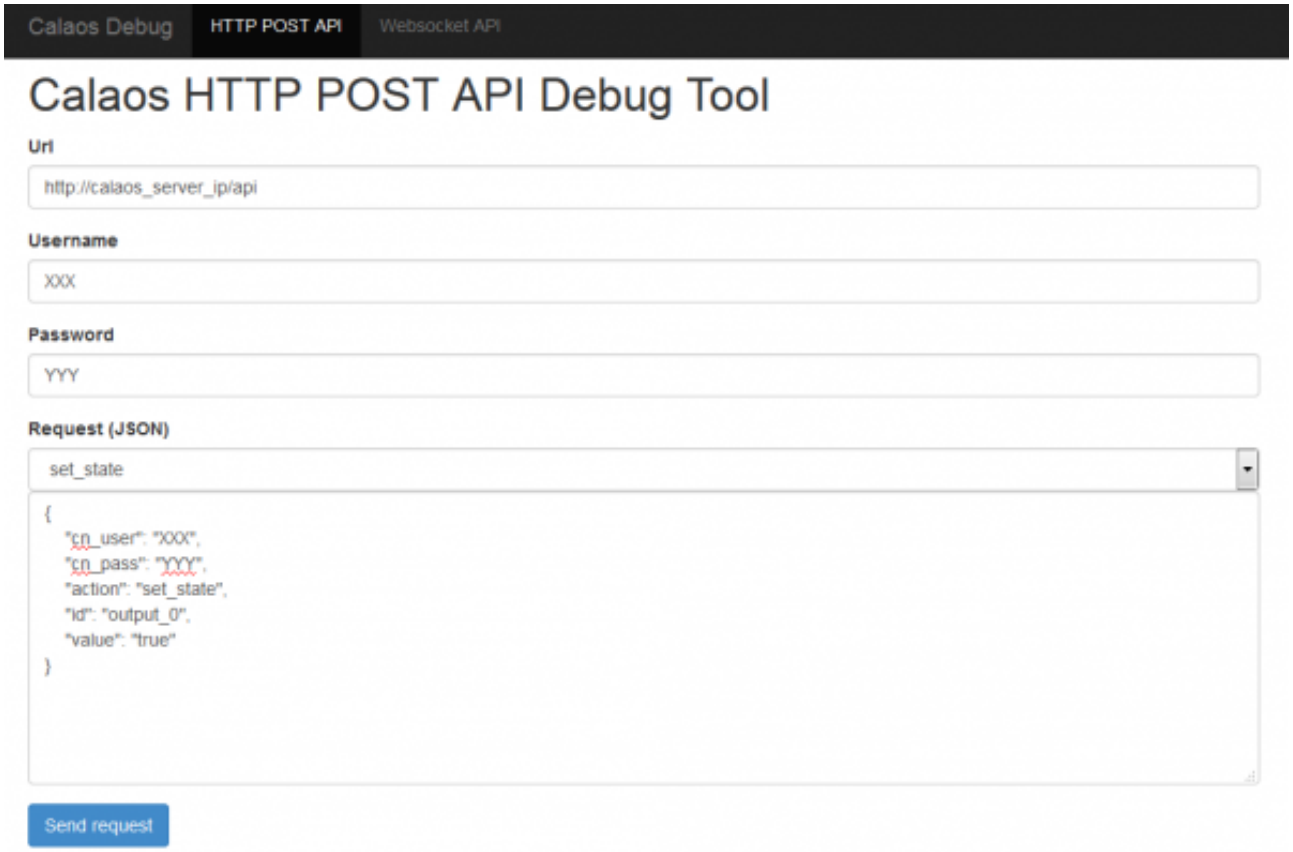

Рис. 1.10. Один з прикладів інтерфейсу сервісу для тесту запитів.

В даному програмному забезпечені використовуються бібліотеки (SDK), що розроблені на мові програмування Python.

Якщо розглядати основні алгоритми роботи, то для роботи з застосунком потрібно пройти реєстрацію аккаунту, а потім налаштувати розташування та вимоги по роботі пристроїв через веб-інтерфейс.

В системі Calaos кожний елемент системи виглядає як набір певних метаданих у JSON-форматі, де вказуються такі дані як назва пристрою, тривалість та циклічність роботи.

Архітектура, що використовується у застосунку:

- мова програмування - C/C++/Python;
- база даних – SQLite.

В даному застосунку використовується принцип Modbus для здійснення комунікації між пристроями в системі. Принцип роботи Modbus можна описати наступним чином: головний керуючий сервіс (master) надсилає інформацію у вигляді фреймів підсервісам (slaves) [15]. Дані підсервіси виконують дії напряму з елементами системи та відправляють відповідь, що містить дані про статус виконаної операції (Рис. 1.11.).

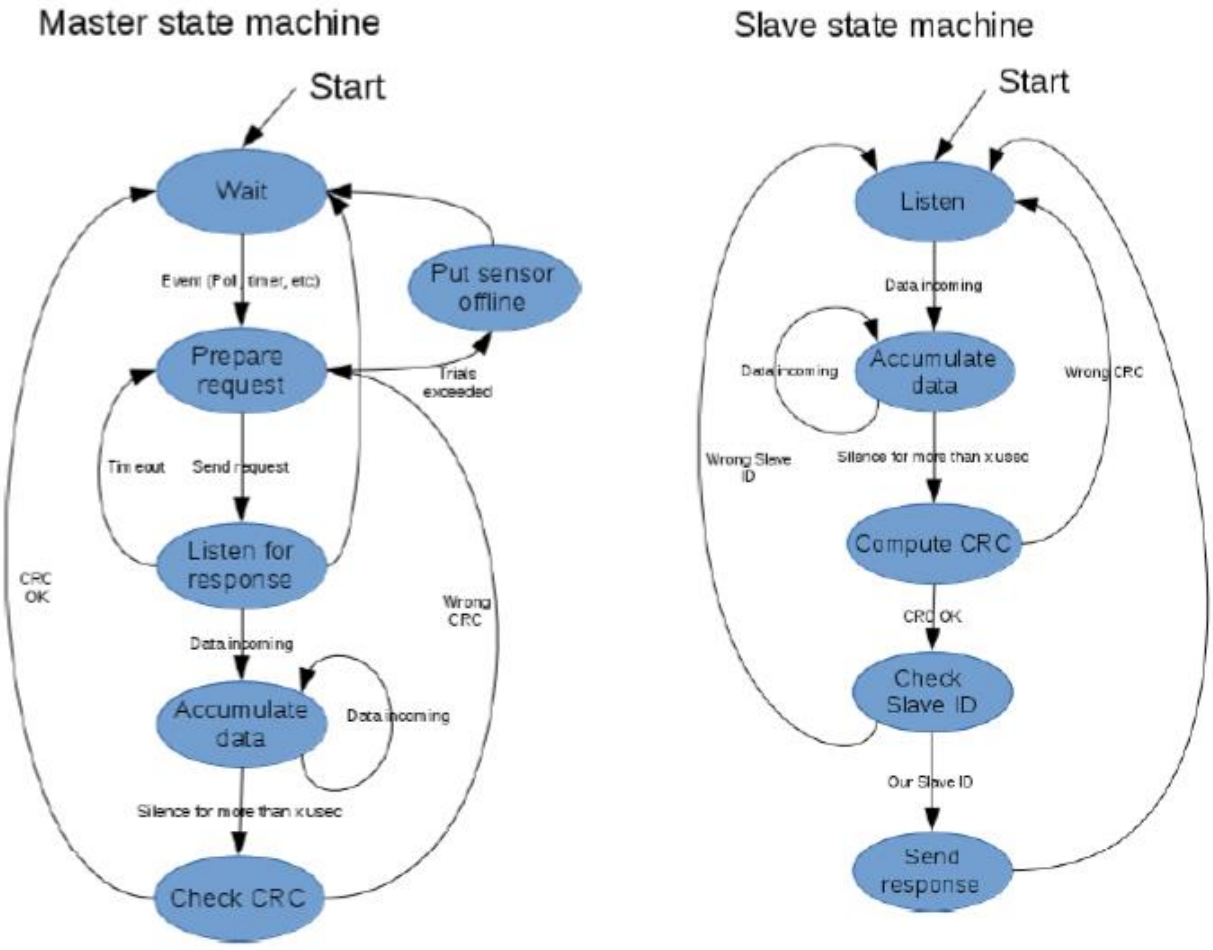

Рис. 1.11. Життєвий цикл сервісів “Master” та “Slave”

Фрейм – це набір байтів в спеціальному форматі, за допомогою якого визначається ідентифікатор сервісу, тип запиту і дані в ньому (Рис. 1.12.).

Slave id	Payload length	Request type	Payload	CRC16
2 bytes	2 bytes	2 bytes	1 .. 65535 bytes	2 bytes

Рис. 1.12. Розшифрування даних у фреймі

1.2.4 Home Assistant Companion for Android

Home Assistant – даний сервіс має підтримку всесвітньої спільноти розробників open-source. Характеризується відкритим кодом та ставить на перше місце контроль та конфіденційність. Найчастіше для запуску сервіс використовує Raspberry Pi або ж локальний сервер.

Мова програмування даного застосунку - Kotlin. До переваг даного програмного забезпеченняможна віднести різноманітні віджети, що дають можливість слідкувати за системою без відкриття самого застосунку.

Якщо розглядати загальний алгоритм роботи застосунку, то для початку роботи мобільний телефон та сервер мають бути підключені до однієї і тієї ж мережі для налагодження взаємозв’язку. Після реєстрації користувача у системі, вся його інформація автоматично відсилається на Home Assistant Cloud (хмарне сховище) і він отримає можливість віддаленого керуванння пристроями.

Стандартний хост і порт сервісу, до якого звертається застосунок – це 192.168.1.4:8123 (протокол HTTP(S)). А також Home Assistant стає доступний за наступним посиланням: http://hassio.local:8123

Комунікація між пристроями в системі і головним сервером відбувається за допомогою HTTP-запитів з інформацією, що структурована в YML форматі. Наприклад:

```
automation:
  - alias: "Action Turn Lights Off"
  initial_state: true
```

```
    trigger:
        - platform: event
        event_type: ios.action_fired
        event_data:
        actionName: 'Bed Time'
action:
    service: light.turn_off
    entity_id: group.all_lights
```

Де:

- alias - ім'я повідомлення, що налаштовується користувачем
- initial_state - параметр, що визначає, чи повертається пристрій до початкового стану (виключений)
- trigger - інформація про те, звідки і коли прийшло повідомлення
- action - інформація про дію, яку потрібно зробити

Повідомлення приймаються сервісом на сервері за допомогою сенсорів (Sensors).

Вся інформація записується у log-файли і зберігається у хмарному сховищі, після реакції сенсора на отриману команду. Тож з’являється можливість проводити аналіз даних на основі реалізованої схеми рекомендацій для користувачів, яка заснована на основі найчастіших та найперіодичніших повідомленнях.

1.2.5 Порівняльна характеристика існуючих систем

Щоб максимально об’єктивно оцінити кожну з систем, потрібно врахувати не лише технічні характеристики, а й доступний функціонал, перспективи розвитку у майбутньому і можливість розширення системи до нового рівня (scalability).

В чотирьох системах, що були розглянуті у попердньому розділі, центральне ядро знаходиться не в смартфоні. Це є позитивним аспектом у розробці подібного програмного забезпечення, оскільки різні мобільні пристрої мають різні характеристики і не завжди мають відповідні технічні характеристики для виконання задач з обробки та передач пакетів даних, що є необхідними для нормального функціонування системи.

Найбільшу перевагу надаємо програмному забезпеченню, що мають хмарні сховища для зберігання баз даних, оскільки дана технологія підтримує можливість бекапу у разі настання екстремальної ситуації. Також дана технологія зарекомендувала себе як надійна, тому що сервіс, що надає ці бази даних, надає певні гарантії. Архітектура Loxone не виправдовує себе з точки зору зберігання даних, оскільки всі дані зберігаються на одному пристрої, і можлива їх повна втрата у разі виникнення проблем з девайсом.

Щодо шифрування даних, кожна з систем зарекомендувала себе достатньо надійною, оскільки безпека – це один з основних пріоритетів при виборі клієнтом продукту. Тож даний параметр підтверджується досвідом компаній та розробників на світовому ринку інформаційних технологій.

Негативним аспектом компанії Loxone є необхідність повної заміни системи у випадку припинення діяльності компанії, оскільки буде втрачена підтримка програмного забезпечення і пристроїв в даному середовищі. На відміну від Loxone, що має можливість працювати тільки з продукцією, що напряму підтримується компанією, системи KNX, Calaos чи Home Assistant Companion не мають прив’язки до конкретних виробників. Таким чином, можна прийти висновку, що при розробці застосунку одним з пріоритетних аспектів має бути адаптивність до будь-яких компонентів системи.

Застосунок KNX дає можливість користувачу можливість вибору не тільки програмного забезпечення, а і виробників. При роботі з Calaos та Home Assistant у користувача з’являються варіанти використання великої кількості інтеграцій

(Amazon Alexa, IKEA, тощо), що в свою чергу відкриває спектр до розширення можливостей застосунку в коротший термін, ніж при роботі з Loxone. Також Calaos та Home Assistant потребують мінімальних фінансових витрат для закупки серверів, де вони будуть працювати, в той же час вони потребують володіння необхідними технічними навичками та знанням в ОС Linux, також хоча б базовими знаннями в програмуванні і роботі з алгоритмами для коректної інсталяції. З цієї причини вони є найменш привабливими для середньостатистичних користувачів. KNX дає найбільші можливості для реалізації різних ідей і підключення в систему саме тих пристроїв, які можуть задовольнити потреби користувача. Loxone найменш вибагливий до початкових інвестицій.

Не можна, не відзначити таку важливу характеристику застосунку як адаптивність (можливість налаштування під різні характеристики різноманітних приладів). Вона є однією з ключових проблем, яку має вирішувати застосунок. Оскільки на даний момент на ринку є величезна кількість процесорів з різною частотою та кількістю ядер, монітори з різним співвідношенням сторін і роздільною здатністю.

Тож пріоритетною задачею є створення такого програмного забезпечення, що буде актуальним хоча б певний час без періодичних оновлень і буде задовольняти потреби користувачів на даний момент. Слід зауважити, що прогрес розвитку операційної системи Android зростає дуже високими темпами, а отже тягне за собою постійну зміну функціональної частини, додавання нових фрагментів та можливостей і видалення застарілого контенту. Тож очікувана розробка дипломного проекту – створення застосунку, що буде мати можливість підтримки у майбутньому.

Висновки до розділу

У першому розділі дипломного проекту було охарактеризовано систему «Розумний дім» та загальний принцип роботи. Також була проведена основна характеристика та аналіз найбільш популярних застосунків для операційної системи Android, а саме: Loxone App, Houseinhand KNX, Calaos for Android, Home Assistant Companion for Android та були визначені їх переваги та недоліки.

У всіх розглянутих застосунках центральне ядро знаходиться не в самому смартфоні, що є безумовною перевагою, так як різні мобільні пристрої мають різні характеристики, що можуть не відповідати необхідним технічним характеристикам для нормальної роботи. Тож таке рішення мінімізує виникнення можливості некоректної роботи через відсутність певних технічних характеристик та робить застосунок більш універсальним.

Серед явних мінусів можна виокремити відсутність хмарного середовища, як, наприклад, у застосунку Loxone App, оскільки всі дані зберігаються на пристрої і можуть бути повністю втрачені і разі проблем з самим девайсом.

Також до мінусів Loxone App можна віднести необхідність повної заміни системи у випадку припинення діяльності компанії через відсутність підтримки ПЗ та пристроїв у середовищі. Інші розглянуті аналоги зарекомендували себе краще через відсутність прив’язки до конкретних виробників. Отже, з цього випливає, що розроблений застосунок має бути адаптивним.

До плюсів можна також віднести можливість інтеграції з іншими системами, як наприклад Amazon Alexa, що значно розширює функціонал в більш короткий термін. Таку можливість дає Calaos та Home Assistant. Також плюсом цього застосунку є більш низька ціна при закупці серверів, та при цьому потребують мінімальних навичок у програмуванні та знання ОС Linux. Завдяки

цим нюансам, цей застосунок є менш привабливим для середньостатистичного користувача, незважаючи на всі переваги. Тож можна зробити висновок, що розроблений застосунок має бути простим і зрозумілим для користувачів, що не мають більш глибоких навичок володіння комп’ютером.

Дослідивши більш детально існуючі аналоги, можна сказати, що у застосунку має бути можливість налаштування під різні види характеристик приладів.

В результаті аналізу випливає необхідність у двох типів користувачів для управління системою:

- адміністратор (дає можливості повного контролю системи, а також налаштування конфігурацій вручну);
- користувач (не повний контроль системи, обробка та аналіз інформації про роботу системи).

2. ПРОЕКТУВАННЯ СИСТЕМИ

На основі детального аналізу вже існуючих застосунків, можна зробити висновок, які основні елементи мають бути присутні у системі:

- датчики (сенсори) – компоненти системи, які будуть вести управління;

- ядро (центральний процесор) – сервер, що буде обробляти бази даних та передевати необхідні пакети даних шляхом зашифрованих повідомлень до датчиків;

- інтерфейс – мобільний застосунок на базі операційної системи Android для комфортної взаємодії з сервером.

Датчики та сервер будуть розташовані в помешканні. Програма для центрального процесора буде працювати на Raspberry Pi з підключенням до хмарної бази даних.

У проектуванні системи був використаний низхідний метод проектування. Переваги такого методу в тому, що все програмне забезпечення можна розглядати як багатомодульну систему, де кожен з модулів є імплементацією однієї з підсистем. Таким чином, ми маємо систему, що складається з менших підсистем, а отже кожен з модулів можна розробляти окремо від інших, що дає можливість розбивати складну задачу на декілька простих. У роботі був використаний такий шаблон проектування, як «Фасад». Суть даного шаблону полягає в тому, що окремі інтерфейси надають доступ до основних методів модуля, і в той же час закривають підмодулі, що необхідні для внутрішніх обчислень.

2.1 Розробка архітектурної схеми системи

В роботі буде представлена багаторівнева архітектура системи, оскільки її перевага в тому, що є можливість розподілити за допомогою розмежування

залежностей її окремі частини. На кожному з «шарів» системи буде знаходитись відповідний компонент архітектури.

Застосунок, представлений в даній роботі, на базі ОС Android буде мати наступну архітектуру (Рис. 2.1.) (Додаток Б):

- рівень представлення (інтерфейс користувача);

- рівень бізнес-логіки (робота з базами даних);

- рівень доступу до даних.

Багаторівнева архітектура має такі основні принципи, як чітке розмежування методів і класів у рівнях розробки, а також взаємодію модулів лише на сусідніх рівнях ієрархії. Таким чином верхні модулі можуть користуватися інтерфейсами нижніх модулів, але не можуть напряму взаємодіяти з ними.

Що до переваг багаторівневої архітектури, то можна виділити наступні параметри:

- ізоляція – розробка та оновлення програмного забезпечення можуть бути тільки в рамках одного рівня;

- продуктивність – підвищена продуктивність і відмовостійкість завдяки розподілу рівнів на різні фізичні комп’ютери.

- тестування – можливість незалежного тестування різних рівнів.

Верхній рівень архітектури даного застосунку – це рівень представлення. Він буде представлений у вигляді інтерфейсу користувача, без можливості доступу до рівня даних в цілях забезпечення безпеки. На цьому рівні буде відбуватися перетворення отриманих даних від пристроїв або сервера на такі, що будуть зрозумілі пересічному користувачу.

Рівень бізнес-логіки – основний рівень даної системи, що має доступ до усіх інших рівнів системи. Саме на ньому будуть проходити основні процеси, що пов’язані з взаємодією сигналів від користувача і пристроїв у системі, а саме обробка, перетворення і керування вхідними даними.

Найнижчим, але найважливішим, рівнем архітектури застосунку буде рівень доступу до даних. На цьому рівні будуть представлені імплементовані бази даних, структури та алгоритми зберігання, оновлення та зберігання даних. Він буде найбільш захищений.

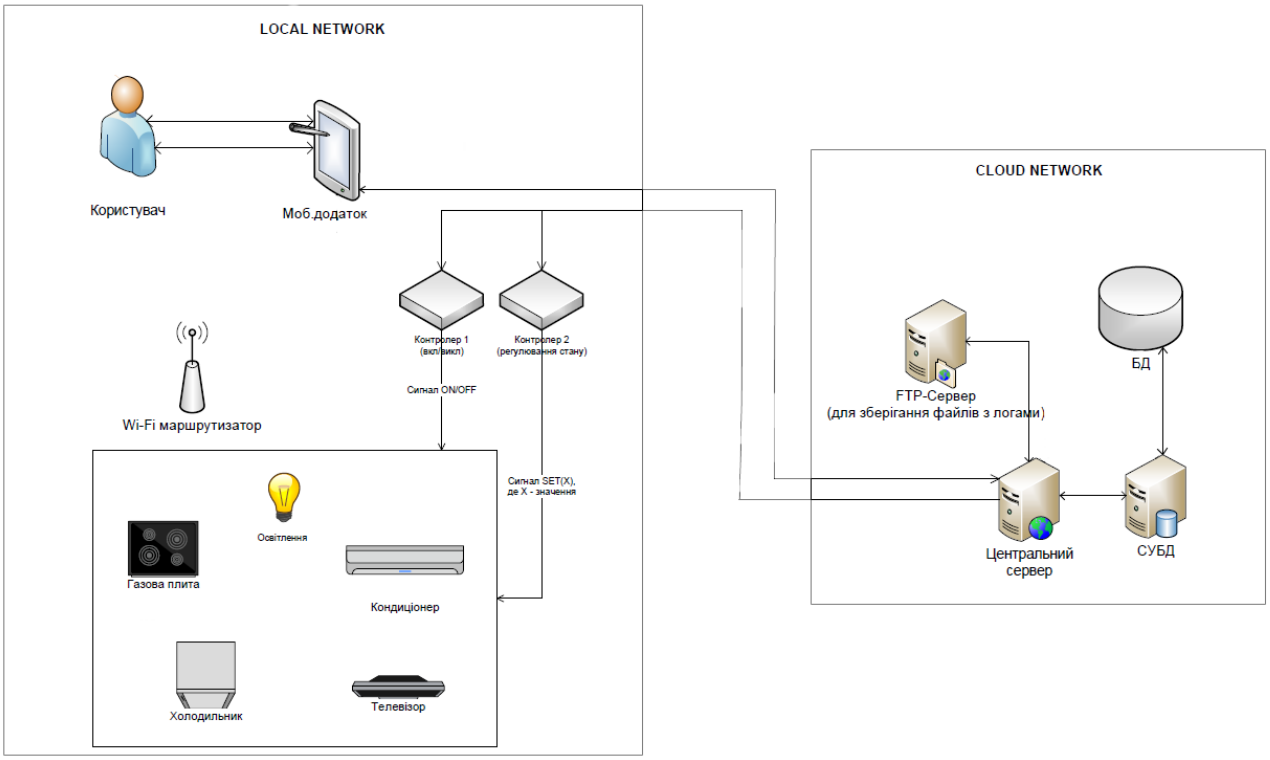

Рис. 2.1. Схема архітектури системи.

2.2 Розробка структурної схеми системи

Розширена імплементація архітектури системи можна розподілити на три основні складові, що представляє її структуру.

Представлення даних - Android застосунок, що дає можливість користувачу керувати пристроями в системі.

Логіка системи представлена у вигляді сукупності всіх елементів системи, що відповідають за обробку, зберігання і передачу даних. Компоненти логіки системи в даному випадку – це (Рис. 2.2.) (Додаток В):

- ядро системи (центральний сервер) – основний елемент системи, функція якого полягає у взаємодії з БД, інтерфейсом і контролерами;

- головний комп'ютер – елемент системи, що відповідає за зміну стану компонентів системи та надсилає сигнали про це (відкриття дверей, ввімкнення системи опалення і т.д.)

- контролер клімат-контролю – комбінація датчиків, що відповідають за підтримання і зміну температури у будинку;

- контролер присутності – визначають місцезнаходження людини в приміщенні за допомогою датчиків руху;

- контролер пожежної безпеки – сенсори, що реагують на зміну (підвищення) рівня вуглекислогог газу в приміщенні;

- контролер комунальних лічильників – система датчиків, що передають показники лічильників (витрати електроенергії, газу, води);

Власне базу даних та додаткові модулі, що необхідні для розширення її можливостей, представлено на рівні даних системи:

- FTP-сервер – збереження файлів і пакетів даних (потрібен для логування інформації системи);

- модуль бекапів – збереження актуального стану баз даних з заданою періодичністю для забезпеченння відновлення системи у разі втрати інформації.

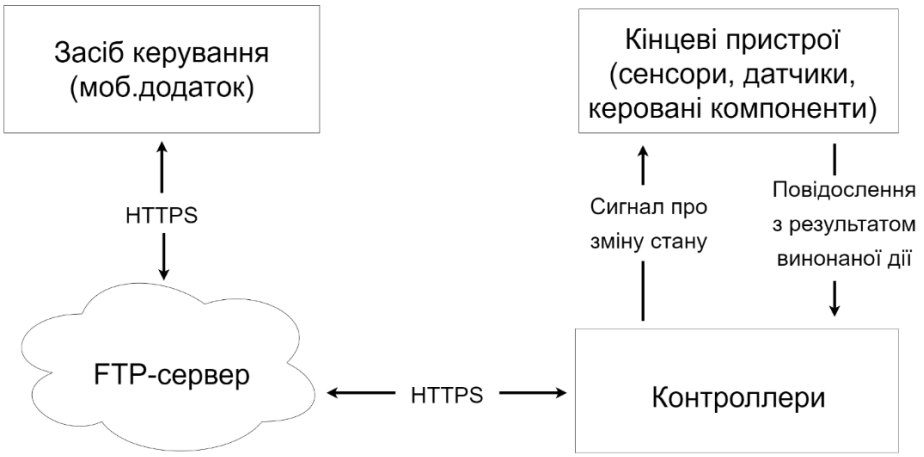

Рис. 2.2. Взаємодія між компонентами системи

2.3Розробка алгоритму роботи системи

Дана система має виконувати наступні функції:

- керування пристроями;
- генерація статистики;
- система сповіщень.

Основна задача кожної з функцій надавати завжди однаковий результат при однакових вхідних даних, оскільки в них запрограмована низка певних сценаріїв взаємодії з користувачем.

2.3.1 Керування пристроями

В даному застосунку використаний наступний алгоритм керування девайсами: користувач, за допомогою інтерфейсу, відправляє сигнал про внесення змін до стану певного пристрою. В даному випадку термін «стан» характеризує набір параметрів, що відповідають за властивості пристрою та вказані в базах даних. Далі ядро (центральний сервер) отримує сигнал, перетворює його (дешифрування, десеріалізація, перевірка на коректність) і відправляє відповідь зі статусом отримання. У випадку, якщо отримано позитивний статус, сервер надсилає інформацію до датчика ідентифікованого пристрою. Датчик, отримавши повідомлення, спочатку дає відповідь у вигляді зміни стану девайсу, а потім відправляє звіт про статус операції. Задача сервера в подальшому перетворити отриману відповідь у зручний формат, який буде надіслано до інтерфейсу користувача. Зазвичай на подібні операції будуть встановлені обмеження у часі. Ця функція має бути запроваджена для того, щоб у разі випадкового відключення девайсів від системи, очікування не тривало протягом нескінченного часу. Отже ми маємо наступну систему, в якій після

відправки сигналу буде проведена перевірка на наявність відповіді на нього, в заданих часових конфігураціях.

Таким чином в роботі даної системи можна виділити три основні стани:

- режим користувача – в даному періоді користувач взаємодіє з системою та задає задачі на зміну стану її компонентів (включення/виключення, зміна температури);

- автоматичний режим;

- режим очікування.

Отже розглянути даний процес можна на простому прикладі, який буде універсальним алгоритмом для опису взаємодії користувача і системи (Рис. 2.3.) (Додаток Г).

Приклад: відправка сигналу про відкриття дверей.

Початок: користувач активує функцію «відкрити двері» у інтерфейсі застосунку. Далі програма відправляє HTTP запит у вигляді пакетів даних про вказану дію на ядро (центральний сервер).

Пре-процесинг: отримана інформація десеріалізується, визначається її тип та визначається контролер (за його унікальним ідентифікатором), якому потрібно надіслати сигнал про зміну стану.

Основний процес: відправка зашифрованої інформації з сигналом «відкрити двері» до контролера; Сенсор замка ловить сигнал і переводить сам замок в інше положення, після чого надсилає відповідь про зміну статусу у вигляді завершення операції;

Пост-процесинг: перевірка результату виконаної операції і якщо статус позитивний, то будуть додані логи з інформацією про зміну стану системи, а також надсиланння повідомлення зі статусом «успіх» до інтерфейсу користувача. Якщо статус негативний, то будуть додані логи з інформацією про неуспішне завершення процесу в системі і відповідне повідомлення в подальшому буде відправлено до інтерфейсу користувача.

Рис. 2.3. Схема алгоритму, за якою система реагую на вхідний сингал від користувача у користувацькому режимі.

2.3.2 Генерація статистики

Якщо розглядати низку функцій, що пов’язана з аналізом і проведенням обробки інформації для отримання статистики по девайсам, що знаходяться в діючій системі, то даний функціонал повинен працювати незалежно від маніпуляцій користувача, при цьому у будь-який період часу потрібно, щоб користувач мав можливість скористатися ініціалізацією процесу «тут і зараз».

Суть даної задачі в тому, щоб виконувати дану низку функцій при вказаних часових рамках з певною періодичністю (наприклад, щоранку о 6:00). Даний процес виглядає наступним чином: сервер запускає crawling процес. Це означає, що проходить збір інформації з усіх девайсів, що знаходяться на даний момент в системі, зчитується їх стан, перевіряється інформація про зміну станів за вказаний період часу, наприклад, від останнього запуску до «зараз». Як тільки вся необхідна інформація зібрана, вона буде відправлена до рівня «Інтерфейс користувача», який проводить її обробку в зрозумілий вигляд: діаграми, графіки та таблиці, що мають всі необхідні дані про стан системи і генерує статистику (Рис. 2.4.).

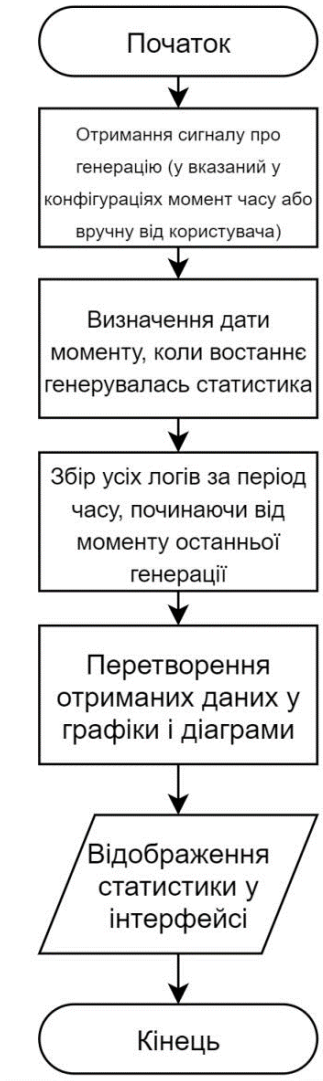

Рис. 2.4. Схема алгоритму, за яким генерується статистика

2.3.3 Система повідомлень та сповіщення

Система сповіщень в даному застосунку буде використовувати алгоритми, які побудовані на основі логів. Таким чином представлена сутність баз даних має в своїй основі все необхідне для збереження інформації про зміну станів девайсів, повідомлення з поточною інформацією та попередженнями, а також необхідну додаткову інформацію про пристрої. Кожному логу буде надана булева властивість. Це означає, що він сам буде визначити чи потрібно надсилати сповіщення про лог користувачу.

Система сповіщень буде працювати наступним чином. З вказаною потрібною періодичністю буде запускатися процес, що збирає всю необхідну інформацію з усіх логів від моменту останнього запуску до моменту «зараз». Далі з усіх логів будуть обрані лише ті, інформацію з яких потрібно додати у сповіщення. Масив з визначених логів серіалізується у повідомлення, далі ядро системи надсилає дану інформацію до інтерфейсу користувача, який вже отримує його у вигляді нотифікації на смартфоні (Рис. 2.5.).

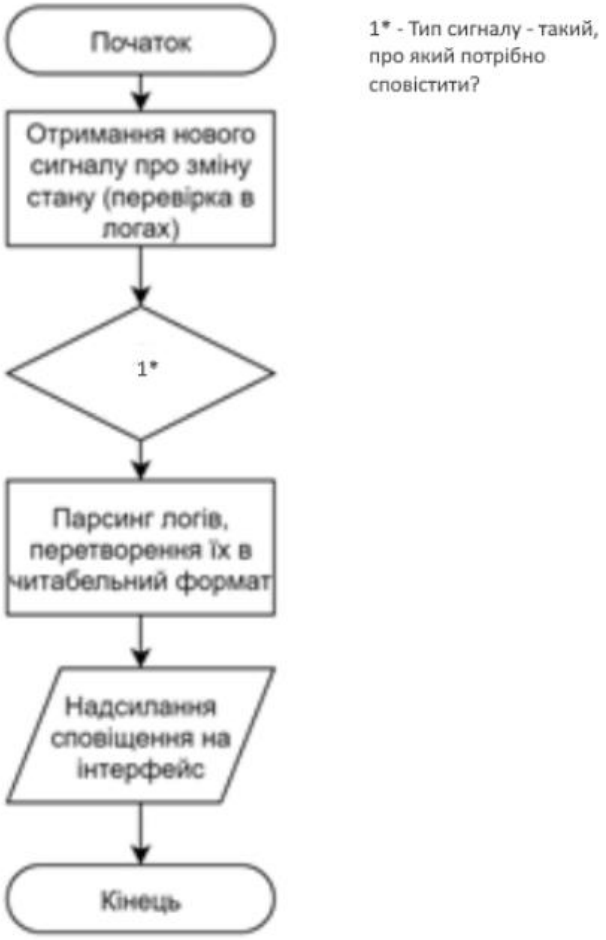

Рис. 2.5. Схема алгоритму, за якою генеруються сповіщення

2.4 Безпека системи

Одним з найважливіших аспектів життя людини є параметр безпеки. Таким чином неможливо оминути такі вагомі фактори як надійність і відмовостійкість при виборі системи «Розумний дім».

У даному випадку аспект безпеки потрібно розглядати з двох сторін, що мають бути реалізовані у даному застосунку.

Фізична безпека – даний параметр має автоматично визначати умови для тривоги за допомогою датчиків. Даний параметр має реалізацію у низці програмного забезпечення, що вже доступні на ринку продуктів типу «Розумний дім». Основний принцип його роботи включає в себе систему різноманітних датчиків і сенсорів, що контролюють конкретні умови і ситуації, передають їх на локальний сервер, а у разі змін станів та критичних обставин негайно передають сповіщення користувачу через інтерфейс застосунку. Як приклад, можна навести наступні датчики і сенсори, що мають відношення до фізичної безпеки: детектори диму, датчик газу, детектори на розбиття вікон, сповіщення про відключення і т.д.

Технологічна безпека – це безпека цілісності даних та забезпечення їх надійної передачі. Тобто основними функціями даного типу безпеки будуть забезпечення безпечної передачі інформації, обрання надійного способу шифрування, SSL сертифікат, тощо. Найбільш вразливою ланкою в системі передачі інформації буде безпосередньо зв'язок між центральним процесором і підключеними девайсами. Недостатня надійність Wi-Fi з’єднання пов’язана з тим, що не зважаючи на те, що постійно з’являються нові алгоритми шифрування, паралельно проводиться і нові шляхи їх зламу. На разі, єдиними виходами забезпечення безпечного Wi-Fi з’єднання можна визначити наступні дії: використання надійного паролю в мережі, регулярне оновлення програмного забезпечення точок доступу з додаванням нових алгоритмів шифрування.

Висновки до розділу

В даному розділі була проведена формалізація завдань, та на основі визначення основних компонентів системи був сформований загальний вигляд системи, розроблена її архітектура та алгоритми роботи системи.

Для того, щоб спростити процес розробки даної системи, було прийнято рішення ізолювати розробку всіх модулів один від одного. Таким чином це дає можливість розбити основну задачу на необхідну кількість більш простіших елементів системи.

Також в даному розділі були сформовані основні групи алгоритмів для коректної роботи системи і застосунку:

- керування;
- статистика;
- сповіщення.

В даному застосунку будуть використані модулі Raspberry Pi, як в одному з розглянутих додадків Home Assistant Companion for Android, через такі переваги, як:

- універсальність;
- відкритість;
- доступність у ціні.

Система зв’язку буде використовувати Wi-Fi протоколи, а всі запити будуть захищені SSL-шифруванням.

3. РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Конфігурація сенсорів

Датчик (сенсор) – це фізичний компонент системи, який приймає команди від ядра системи (центрального процесора) і включає в собі функції перетворення фізичної або біологічної величини у фізичну. Калібрування даних величин у цифровому форматі необхідно для подальшої обробки їх контролером. Для коректного функціонування системи «Розумний дім» більшість сенсорів повинні знаходитись у ввімкненому стані постійно для подальшої комунікації та регулярного підключення до системи.

Основною перевагою даних сенсорів вважається можливість встановлення програмного забезпечення. В даному проекті були використані такі датчики, як Particle Computer (Decker et al., 2005), Berkeley MOTES (Estrin, 2002)… тощо. Таким чином була вирішена одна з проблем в реалізації системи, що пов’язана з автономністю роботи. Оскільки даний тип сенсорів, завдяки наявності в них ПЗ, дає можливість отримувати та обробляти інформацію про їх роботу, то дана характеристика дає змогу їм працювати і в критичних умовах, наприклад без постійно ввімкнених зовнішніх контролерів і девайсів.

3.2 Розробка центрального контролера

Контролер (центральний процесор) – виконує функцію взаємодії користувача та системи у всіх можливих варіантах. В свою структуру він включає сервер, де відбувається реалізація алгоритмів роботи та бази даних, де зберігається вся інформація, що має відношення до системи і її коректної роботи.

3.2.1 Розробка серверної частини застосунку

Для коректного опису сервера потрібно зауважити, що він складається з двох частин, а саме менеджера девайсів та основного ядра. Функцією ядра є робота з базами даних, що виконується на базі операцій Read та Write. Функція менеджера девайсів полягає в роботі з повідомленнями. Таким чином все, що стосується отримання і відправлення, дешифрування та перевірки на наявність помилок, виконує саме менеджер пристроїв для подальшої передачі в роботу контролерам.

Контролером в даній системі виступають підсервери, що є сполучною ланкою з датчиками і девайсами. Принцип роботи контролерів можна описати наступним чином:

- надходження інформації від головного контролера про будь-які дії з пристроєм в системі;
- передача сигналу до сенсорів з необхідною командою. Дана дія може отримати як статус «успішно», так і статус «не успішно»;
- Якщо дана дія отримала статус «успішно», то контролер отримує зворотній зв'язок у вигляді завершення операції, який передає до центрального процесора, в якому вже створюється сповіщення для користувача системи (якщо в даному випадку є необхідність цього). Якщо дана дія отримала статус «не успішно», тобто девайс відправив негативну відповідь або ж вийшли часові рамки зворотного зв’язку після початку процесу, тоді контролер передає до центрального процесора про помилку операції. Далі центральний процес відправляє повідомлення на сервер, де вже буде сформовано сповіщення про негативний статус завершення операції.

3.2.2 Розробка бази даних

База даних – це компонент системи, що відповідає за збереження інформації як про всю систему, так і про її складові. БД представляє собою сховище файлів з чіткою структурою, і щоб полегшити взаємодію з нею, переважно використовуються такі вбудовані мови як DDL/DCL.

Варто звернути увагу, що одним з важливих завдань є також забезпечення безпеки і збереження інформації, що зберігається в БД. Даний аспект пов'язаний з тим, що бази даних володіють не тільки інформацією про пристрої і їх роботу в системі, а і мають в своєму розпорядженні всю інформацію про користувача, в тому числі і конфіденційну.

Дану проблему можна вирішити наступними шляхами:

- використання вивірених паролів для доступу в бази даних;
- використаннння власних протоколів для шифрування даних;
- регулярні бекапи для збереження інформації на хмарному сховищі, що підтримують можливість видобути її для відновлення баз даних.

В даному застосунку будуть використані наступні сутності БД:

- Account – користувач. Дана сутність буде відповідати за збереження інформації про користувачів системи (авторизація, роль та права);
- Group – група девайсів. Дана сутність буде відповідати за поєднаних між собою пристроїв за принципом їх роботи. Керувати кожною групою пристроїв буде відповідний контролер;
- Sensor – сенсор, датчик. Дана сутність відповідає за роботу сенсорів та їх характеристики;
- Device – пристрій. Дана сутність відповідає безпосередньо за конкретний пристрій у системі, зміни його стану та ідентифікає групу, до якої він має відношення.

Також в даному застосунку буде сформована таблиця SensorToDevice. Функція даної таблиці у збереженні додаткової інформації про налаштування кожного конкретного девайсу та ID користувача, що створив даний зв'язок (Рис. 3.1.) (Додаток Ґ).

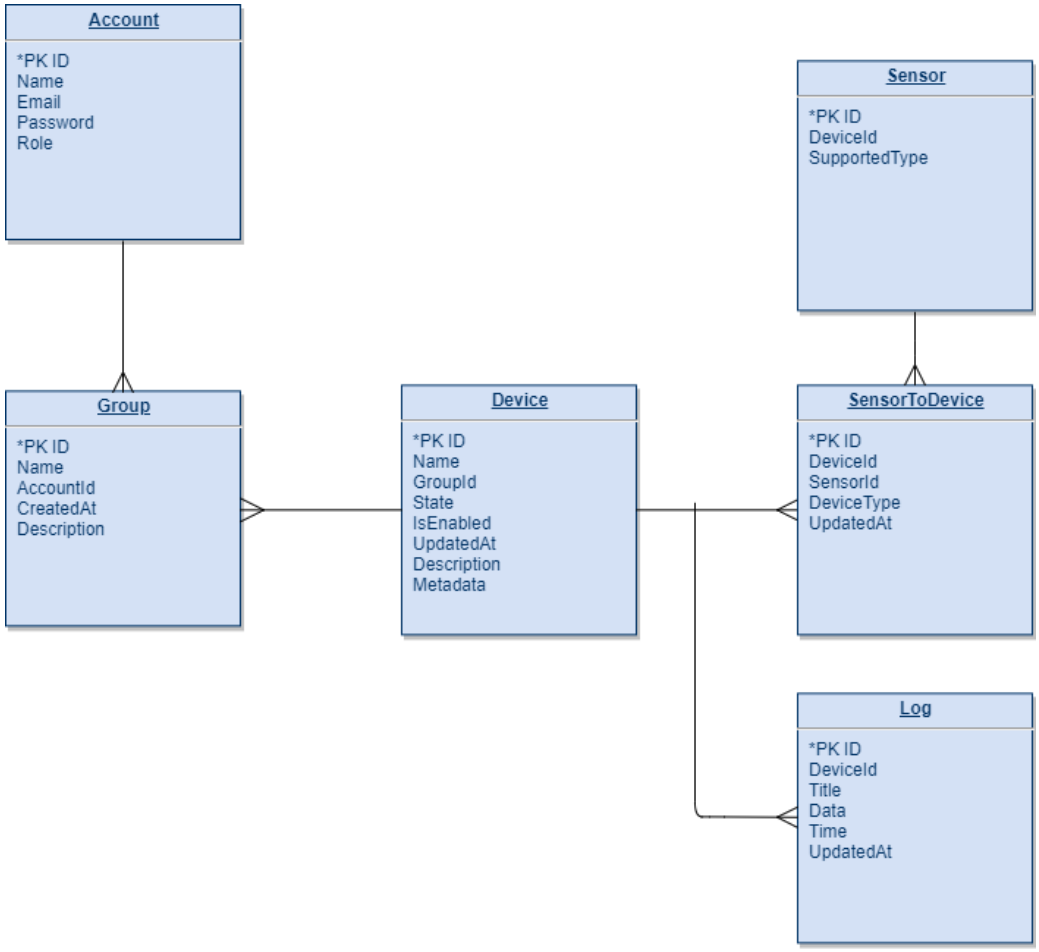

Рис. 3.1. ER-діаграма сутностей і відношень у базі даних

3.3 Інтерфейс

Неможливо уявити застосунок, який створений для підвищення комфорту пересічної людини, та не має інтерфейсу користувача, що має значно спростити взаємозв’язок між системою і людиною. В той же час користувач повинен мати достатню кількість можливостей і опцій для керування. Зв'язок між центральним

контролером і користувачем буде проходити за допомогою передачі протоколів через з’єднання Wi-Fi, а двостороння комунікація через HTTP-запити.

Для зручної та коректної роботи застосунку буде використано протокол періодичного оновлення даних в системі (2с). При такий реалізації задачі інформація буде регулярно оновлюватися в інтерфейсі користувача, без запиту про актуалізацію даних.

Наше програмне забезпечення буде виконувати наступні функції, які будуть розглянуті в наступнгих підрозділах:

- авторизація;
- керування пристроями;
- статистика;
- панель адміністратора.

3.3.1 Реєстрація та авторизація

Функція авторизації в програмному забезпеченні буде складатися з двох взаємопов’язаних між собою частин – реєстрація в застосунку та подальша авторизація у системі.

При відкриті застосунку в перший раз, користувач бачить безпосередньо вікно авторизації (Рис. 3.2.) (Додаток Д). Щоб зареєструватися в застосунку, користувачу необхідно пройти наступні кроки:

- заповнити поля з особистими даними (електрона пошта та пароль);
- пройти валідацію (спеціальний лист з посиланням, що відправлений на вказану електрону пошту, для підтвердження особистості;
- відправити запит адміністратору, щоб отримати доступ до системи;
- налаштувати конфігурації девайсів, використовуючи інтерфейс користувача (Рис. 3.3.);

При подальших відкриттях застосунку потрібно лише заповнити поля з особистими даними та натиснути на кнопку авторизації. І в залежності від отриманої раніше ролі будуть автоматично доступні функції та можливості у застосунку.

Права адміністратора у розробленому застосунку отримує перший зареєстрований користувач у системі. Він отримує доступ до панелі адміністратора і зможе налаштовувати права і ролі наступних користувачів, що зареєстровані в системі.

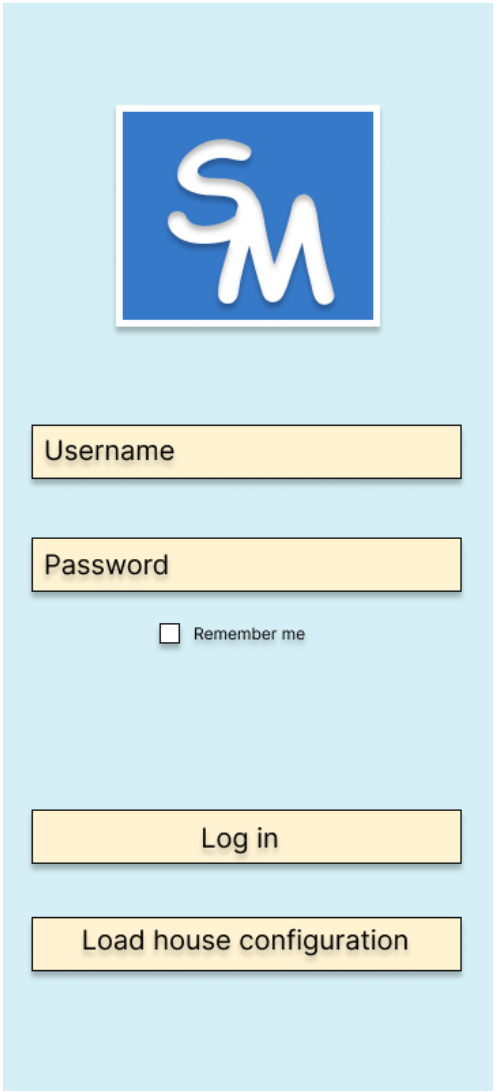

Рис. 3.2. Вікно авторизації

Рис. 3.3. Схема алгоритму, за якою авторизується користувач

3.3.2 Керування

Однією з найскладніших і найважливіших функцій програмного забезпечення є функція керування девайсами, оскільки потрібно якомога детальніше описати різні типи датчиків та варіанти взаємодії з ними в залежності від виду. При цьому інтерфейс повинен бути простим у використанні та інтуітивно зрозумілим для користувача (Рис. 3.4.) (Додаток Д).

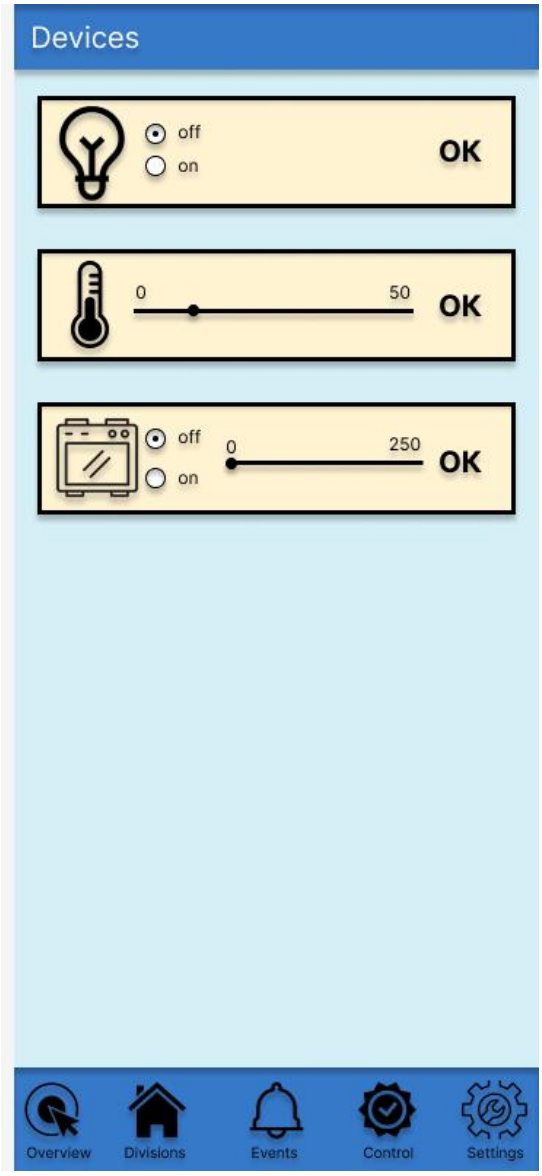

Рис. 3.4. Вікно регуляції параметрів пристроїв

Датчики в системі мають можливість як зберігати інформацію про девайси, та і виконувати роботу без збереження цієї інформації, відправляючи з певною періодичністю лише запити про зміну станів пристроїв.

Основою функцією датчиків є збереження підключення до девайсів та підтримувати зв'язок протягом заданого часу.

Таблиця 2 детально відображає як типи датчиків, що будуть виконувати керування девайсами, так і відповідні девайси, якими буде вестись керування.

Таблиця 2 - Види сенсорів і пристроїв у системі

Назва	Приклад девайсів	Взаємодія
ВКЛ/ВКЛ	Світло, вимикачі	Зміна статусу увімкнено-вимкнено
Вимірюючі	Термометр, лічильники	Отримання числових даних за період часу
З’являються Зникають	Машина, мотоцикл, людина в приміщені	Зміна статусу на основі геолокації
Відкриті Закриті	Двері, вікна	Зміна статусу відкрито-закрито

3.3.3 Статистика

Також складно оминути таку можливість даного застосунку, як статистика. Дана функція буде мати підтримку завдяки логуванню, тобто внесення інформації про усі зміни в діючій системі. Таким чином, протягом обраного періоду часу у користувача буде можливість переглянути у зручному вигляді всю необхідну інформацію про дані, що його цікавлять (Рис. 3.5.) (Додаток Д). До даних статистики в даному випадку будуть відноситися не тільки показники лічильників, зміна температури протягом доби, а і, наприклад, час увімкненого світла.

За допомогою реалізації даної функції користувач матиме можливість постійно бути в курсі подій в помешканні і мати можливість економії коштів, що пов’язані з його утриманням. Також, при внесені даних про комунальні тарифи та інші послуги, можна буде увімкнути функцію автоматичного обчислення витрат.

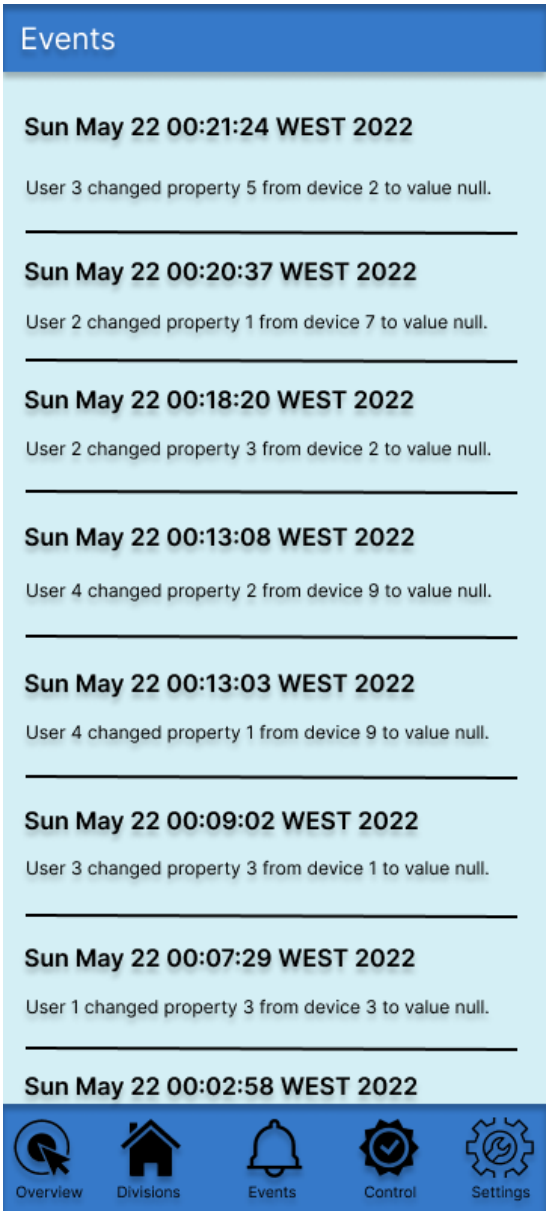

Рис. 3.5. Режим перегляду статистики у вигляді log-ів

3.3.4 Панель адміністратора

Доступ до даної функції буде лише у користувачі, яким присуджена роль «Адміністратор» (Рис. 3.6.) (Додаток Д), вона відкриває наступні можливості у застосунку (Додаток Є):

- керування користувачами (додавання, видалення, зміна прав) (Рис. 3.7.);

- керування пристроями (додавання, видалення, налаштування тарифів та інших конфігурацій);
- налаштування сповіщень (тригерів на конкретні зміни в системі);
- налаштування періодів генерації статистики;
- налаштування підключення (вибір Wi-Fi мережі, пошук пристроїв, підключення, перевірка статусу підключення);
- керування цілими групами.

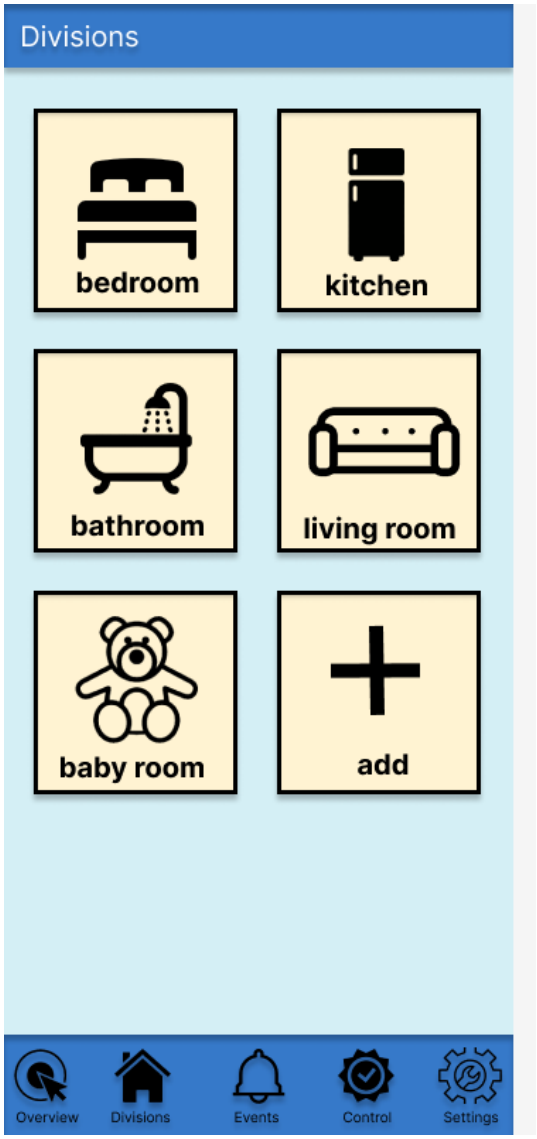

Рис. 3.6. Вікно для керування групами (приміщеннями) у панелі адміністратора.

Рис. 3.7. Приклад схеми алгоритму, за якою адміністратор додає нових користувачів.

3.4 Налаштування зв’язку

Взаємозв’язок усіх елементів системи буде реалізовано за допомогою протоколів Wi-Fi, що включає в себе два типи пристроїв: точка доступу (AP) та бездротовий пристрій, яким може виступати будь-який елемент системи.

Точка доступу (AP) – це з’єднання між стаціонарною провідною та безпровідною мережею. Його основною властивістю є підтримка до 30

бездротових пристроїв у радіусі до 50 метрів у приміщенні та до 100 метрів на відкритому просторі.

Таким чином інфраструктура нашою системи буде складатися з наступного набору компонентів: точка доступу та бездротових пристроїв, що підключені до системи. Така топологія є доволі зручною та практично, оскільки у системі є виокремлений центр, до якого під'єднуються вже інші пристрої, де кожна з локальних зон керується своєю точкою доступу. Для того, щоб збільшити зону покриття можливо використовувати декілька основних зон.

Висновки до розділу

Даний розділ докладно описує структуру та процес розробки програмного забезпечення для системи «Розумний дім». Також були детально розглянуті основні складові частини системи, призначення кожної з них та алгоритми дії. Архітектура клієнт-сервер та шаблон проектування «Фасад» були застосовані про розробці даного програмного забезпечення. Також застосунок був розподілений на окремі модулі, для того, щоб полегшити підтримку і оновлення в майбутньому.

У даному розділі було представлено ER-діаграму, з описом усіх сутностей та взаємозв'язків у базі даних. Були представлені такі сутності, як Account, Group, Sensor та Device. Також наявна така таблиця, як SensorToDevice, функцією якої полягає у збереженні додаткової інформації про налаштування конкретного пристрою та інформації про користувача (ID), що створив зв'язок.

В даному застосунку були використані механізми авторизації користувача, підтримки взаємозв'язку з зовнішніми пристроями за допомогою датчиків, робота з даними статистики. Інтерфейс користувача був створений на базі операційної системи Android, що дало можливість керувати системою

дистанційно. Також завдяки вказаному періоду оновлення даних, вони будуть оновлюватись і відображатися самостійно без необхідності надсилати запит.

За допомогою попереднього аналізу була виявлена потреба у двох користувачах, а саме: адміністратор та звичайний користувач. Перший зареєстрований користувач автоматично отримує статус адміністратора, а всі подальші – не мають прав адміністратора та настільки ж розширені функції.

Важливою є додана функція для застосунку у вигляді статистики, яка підтримується завдяки логам. Завдяки цьому користувач має можливість переглядати необхідну йому інформацію за певний проміжок часу. Завдяки зберіганню інформації, власник знає про все, що відбувається в приміщенні та має можливість аналізувати та заощаджувати свої кошти в результаті.

ВИСНОВКИ

В даному дипломному проекті було створено застосунок для керування інтелектуальними пристроями розумного дому на базі операційної системи Android.

Для того, щоб розробити найбільш ефективний застосунок, була проведена робота з розбору та аналізу вже існуючих рішень на ринку технологій. Завдяки проведеному аналізу були виявлені певні недоліки, які необхідно було виправити. В цілому було проаналізовано чотири найпопулярніших на даний час існуючих аналогів: Loxone App, Houseinhand KNX, Calaos for Android, Home Assistant Companion for Android.

Також були розроблені і спроектовані наступні схеми, як: архітектурна схема системи, алгоритм роботи системи, блок-схема алгоритмів програми. Також було представлено ER-діаграму сутностей та відношень БД.

Про розробці програмного забезпечення, на основі попереднього аналізу, були виправлені наступні недоліки таким чином: підвищена безпека даних, можливість бекапів у разі втраті даних, підвищена відвовостійкійсть та надійність у роботі системи. Всі ці недоліки були виправлені через інтеграцію додаткових бібліотек та використання нових способів проектування. Також в ході розробки застосунку була спроектована база даних, ER-діаграма сутностей і зв’язків між ними.

Цілісність системи складається з декількох основних модулів. Мета даного рішення була розділити одну складну задачу на декілька простіших, для того, щоб полегшити розробку і тестування кожного з модулів в ізоляції від інших. Система використовує Wi-Fi протокол для зв’язку між усіма компонентами системи. Інтерфейс користувача був розроблений на базі операційної системи Android та представлений у вигляді застосунку для смартфону та інших девайсів на даній ОС. Також розробка даного програмного забезпечення має можливість підтримки та розширення в майбутньому.

ПЕРЕЛІК ПОСИЛАНЬ

1. Sydney Morning Herald. 2016. How a high-tech home can make life easies for people with disabilities.
2. McKinsey. 2016. Internet of Things: Opportunities and challenges for semiconductor companies.
3. Jacob Kastrenakes. 2019. Apple, Google, and Amazon are teaming up to develop a smart home standart.
4. Mikhail T. Galeev. 2006. Catching the Z-Wave.
5. G. Blake Meike, Lawrence Schiefer. 2021. Inside the Android OS: Building, Customizing, Managing and Operating Android System Services (Android Deep Dive).
6. Шестаков В.С., Сагидуллин А.С. 2015. Применение технологии WEBSOCKET в WEB-приложениях технологического назначения.
7. International Journal of Sustainable. 2016. Assessing the number of users who are excluded by domestic heating controls.
8. Пасеков В. 2010. Возможности использования платформы KNX в системах безопасности, 42-44
9. Dan Goodin. 2013. Guerilla researcher created epic botnet to scan billions of IP addresses.
10. Carrie Mihalcik. 2019. Apple, Amazon, Google, and other want to create a new standard for smart home tech.
11. Heidi Monson. 1999. Bluetooth Technology and Implications.
12. Steve Gartner. 2015. Wireless LANs.
13. https://developer.android.com/studio/intro
14. Mihir Bellare, Ran Canetti, Hugo Krawczyk. 1996. Keying hash functions for message authentication.

15. Charles Palmer, Sujeet Shenoi. 2009. Critical Infrastructure Protection III: Third IFIP WG 11. 10 International Conference, Hanover, New Hampshire, USA, 87
16. Lawrence G. Roberts, Barry D. Wessler. 1970. Computer network development to achieve resource sharing