НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Навчально-науковий інститут прикладного системного аналізу
Кафедра системного проектування

До захисту допущено:
Завідувач кафедри
_____ Вадим МУХІН
«___»_____2023 р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою

“Інтелектуальні сервіс-орієнтовані розподілені обчислювання”

зі спеціальності 122 "Комп'ютерні науки"

на тему: «Порівняльний аналіз кастомізованих платформ для розумних речей з безкодовою та мало-кодовою реалізаціями»

Виконала:
студентка IV курсу, групи ДА-92
Скочко Єлизавета Дмитрівна _____

Керівник:
доцент, к.т.н.
Кирюша Б.А. _____

Консультант з нормконтролю:
доцент, к.т.н.
Кирюша Б.А. _____

Рецензент:
Посада, науковий ступінь, вчене звання,
Прізвище, ім’я, по батькові _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.

Студентка _____

Київ – 2023

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Навчально-науковий інститут прикладного системного аналізу

Кафедра системного проектування

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 122 "Комп'ютерні науки"

Освітньо-професійна програма – "Інтелектуальні сервіс-орієнтовані розподілені обчислювання"

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Вадим МУХІН

«___»_____2023 р.

ЗАВДАННЯ

на дипломну роботу студентці

Скочко Єлизаветі Дмитрівні

1. Тема роботи « Порівняльний аналіз кастомізованих платформ для розумних речей з безкодовою та мало-кодовою реалізаціями», керівник роботи Кирюша Богдан Анатолійович, доцент, к.т.н., затверджені наказом по університету від «___»_____ 20__ р. №_____

2. Термін подання студентом роботи – _ червня 2023 р.

3. Вихідні дані до роботи

Дослідження поняття розумні речі.

Порівняння кастомізованих платформ для розумних речей з безкодовою та мало-кодовою реалізаціями :

- Розробка системи метрик для порівняння платформ.
- Перевірка вживаності виконаного аналізу.

4. Зміст роботи

1. Вступ.

2. Постановка задачі проектування.
3. Визначення поняття «розумні речі».
4. Актуальність використання кастомізованих платформ з безкодовою та мало-кодовою реалізаціями.
5. Формування системи метрик для порівняльного аналізу.
6. Аналіз кастомізованих платформ за розробленою системою метрик.
7. Створення прототипу системи розумного будинку у симуляторі
8. Розробка системи взаємодії з розумними речами на обраній платформі.
9. Функціонально-вартісний аналіз програмного забезпечення.
10. Висновки.

5. Перелік ілюстративного матеріалу

1. Презентація до захисту роботи.

6. Консультанти розділів роботи:

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Рощина Н.В.		

7. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Отримання завдання	10.02.2022	
2	Дослідження поняття розумні речі	10.03.2022	
3	Проведення аналізу щодо актуальності та важливості кастомізованих платформ з безкодовою та мало-кодовою реалізаціями	30.03.2022	

4	Проведення порівняльного аналізу платформ для розумних речей на основі розробленої системи метрик	01.04.2022	
5	Створення прототипу системи розумного будинку у симуляторі	10.04.2022	
6	Розробка системи взаємодії з розумними речами на обраній платформі	30.04.2022	
7	Тестування системи	01.05.2022	
8	Порівняння розробленого рішення з доступними на ринку	15.05.2022	

Студент Скочко Є.Д.

Керівник Кирюша Б. А.

АНОТАЦІЯ

бакалаврської дипломної роботи Скочко Єлизавети Дмитрівни на тему «Порівняльний аналіз кастомізованих платформ для розумних речей з безкодовою та мало-кодовою реалізаціями»

Метою дипломної роботи є дослідження поняття «розумні речі», аналіз актуальності та важливості кастомізованих платформ з безкодовою та мало-кодовою реалізаціями для взаємодії з розумними речами. Розробка системи метрик для порівняльного аналізу кастомізованих платформ, а також перевірка вживаності виконаного аналізу на практичному прикладі.

У світі електроніки, що швидко розвивається, бізнесу стає дедалі складніше залишатися в курсі найновіших розробок. Вже кілька років, як Інтернет речей набирає обертів, і це принесло цілий ряд нових викликів у сфері технологій та бізнесу. Відповідаючи на ці вимоги та запити споживачів, деякі компанії почали використовувати платформи з безкодовою та мало-кодовою реалізаціями власних додатків без написання коду. Тобто актуальність даної теми є повністю виправданою.

Результатом дипломної роботи є дослідження поняття розумні речі, перевірка актуальності та впливу на процес взаємодії із розумними речами із застосуванням кастомізованих платформ, розробка системи метрик для порівняльного аналізу платформ та перевірка вживаності виконаного аналізу шляхом реалізації практичного прикладу їх використання.

Створену систему метрик можна буде використовувати при подальшому аналізі новостворених платформ, а практичний приклад продемонструє можливості ефективної взаємодії із розумними речами через кастомізовані платформ з безкодовою та мало-кодовою реалізаціями.

Загальний обсяг роботи: 78 сторінок, 23 рис., 13 таблиць, 1 додаток, 16 джерел.

Ключові слова: IoT, розумні речі, кастомізовані платформи, lo-code, no-code.

Annotation

bachelor's thesis of Skochko Yelyzaveta Dmytrivna on “ Comparison of customized platforms for smart things with no-code and low-code implementations”

The purpose of the thesis is to study the concept of "smart things", analyse the relevance and importance of customised platforms with no-code and low-code implementations for interacting with smart things. Developing a system of metrics for comparative analysis of customised platforms, as well as testing the applicability of the analysis on a practical example.

In the rapidly evolving world of electronics, it is becoming increasingly difficult for businesses to keep up with the latest developments. The Internet of Things has been gaining momentum for several years now, and it has brought a whole new set of technological and business challenges. In response to these demands and consumer requests, some companies have started using platforms with no-code and low-code implementations of their own applications without writing any code. Thus, the relevance of this topic is fully justified.

The result of the thesis is a study of the concept of smart things, verification of the relevance and impact on the process of interaction with smart things using customised platforms with no-code and low-code implementations, development of a system of metrics for comparative analysis of platforms and verification of the applicability of the analysis by implementing a practical example of their use.

The created metrics system can be used for further analysis of newly created platforms, and the practical example will demonstrate the possibilities of effective interaction with smart things through customised platforms with no-code and low-code implementations.

Total total volume of work is 78 pages, 23 figures, 13 tables, 1 appendice, 16 sourses

Keywords: IoT, smart things, customised platforms, lo-code, no-code.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

API (Application Programming Interface) –набір правил і протоколів, які дозволяють різним програмним додаткам взаємодіяти один з одним.

IoT (Internet of Things) – мережа фізичних об'єктів, оснащених датчиками, програмним забезпеченням і можливостями підключення, які дозволяють їм збирати та обмінюватися даними через Інтернет.

AWS(Amazon Web Services) – веб сервіси Амазон.

AWS IoT Simulator – інструмент від Amazon Web Services, який дозволяє користувачам моделювати поведінку пристроїв Інтернету речей у віртуальному середовищі.

Drag-and-drop – метод взаємодії з користувацьким інтерфейсом без необхідності складного програмування або точного ручного введення.

Lambda-функція – в контексті хмарних обчислень відноситься до безсерверного обчислювального сервісу AWS, дозволяє запускати ваш код без резервування або керування серверами.

MQTT(Message Queuing Telemetry Transport) – протокол обміну повідомленнями, розроблений для ефективного зв'язку між пристроями в обмеженому мережевому середовищі, такому як Інтернет речей (IoT).

HTTP(Hypertext Transfer Protocol) – прикладний протокол, який використовується для передачі гіпермедійних документів через Інтернет.

GET - метод HTTP, який використовується для отримання даних або ресурсів з сервера.

URL(Uniform Resource Locator) – адреса, яка використовується для пошуку ресурсу в Інтернеті

SQL(Structured Query Language) – мова програмування, яка використовується для управління та маніпулювання реляційними базами даних.

JSON(JavaScript Object Notation) – текстовий формат обміну даними.

ВСТУП

Кастомізовані платформи для розумних речей з з безкодовою та мало-кодовою реалізаціями - це програмні інструменти або платформи, які дозволяють користувачам створювати розумні додатки або пристрої без необхідності писати код з нуля. Ці платформи надають графічний інтерфейс користувача і компоненти перетягування, які користувачі можуть використовувати для створення власних рішень без глибоких знань програмування[1].

У контексті Інтернету речей ці платформи дозволяють користувачам створювати розумні пристрої, системи та додатки, які можуть взаємодіяти один з одним і управлятися дистанційно. Це може бути будь-що - від пристроїв для розумного дому до систем промислової автоматизації.

Підхід без коду і з низьким рівнем коду полегшує розробку і розгортання додатків Інтернету речей для підприємств і приватних осіб, оскільки він усуває необхідність в обширному досвіді кодування і може скоротити час і витрати на розробку. Ці платформи також сприяють швидшому впровадженню інновацій та створенню прототипів, оскільки користувачі можуть швидко експериментувати і повторювати свої ідеї без необхідності мати спеціалізовані технічні навички[1].

У швидкоплинній сфері електроніки, йти в ногу з останніми досягненнями стало складним завданням для бізнесу. Інтернет речей неухильно набирає обертів протягом кількох років, створюючи безліч технологічних і бізнес-викликів. Щоб відповісти на ці виклики та задовольнити очікування споживачів, деякі компанії впроваджують вище зазначені платформи, полегшуючи процес взаємодії зі складними технологіями[1].

Через наведені вище причини, було вирішено провести ретельний порівняльний аналіз кастомізованих платформ для розумних речей, а також підтвердити вживаність даного аналізу на практичному прикладі.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА МОЖЛИВИХ РІШЕНЬ

Метою дипломної роботи є дослідження поняття «розумні речі», аналіз актуальності та важливості кастомізованих платформ з безкодовою та малокодовою реалізаціями для взаємодії з розумними речами. Розробка системи метрик для порівняльного аналізу кастомізованих платформ, а також перевірка вживаності виконаного аналізу на практичному прикладі, що продемонструє ефективність нового рішення.

Для досягнення мети необхідно вирішити наступні задачі:

- Дослідити поняття розумні речі.
- Обгрунтувати цінність та актуальність використання кастомізованих малокодових та безкодових платформ.
- Сформувати систему метрик для порівняльного аналізу.
- Проаналізувати кастомізовані платформи за розробленою системою метрик.
- Створити прототип системи розумного будинку у симуляторі.
- Розробити систему взаємодії з розумними речами на обраній платформі.
- Протестувати рішення і довести його ефективність.

2 ВИЗНАЧЕННЯ ПОНЯТТЯ РОЗУМНІ РЕЧІ

Концепція Інтернету речей привернула значну увагу в останні роки, революціонізувавши спосіб нашої взаємодії з повсякденними предметами та цифровим світом. З розвитком технологій інтеграція розумних пристроїв та їх взаємозв'язок стали реальністю, трансформуючи різні аспекти нашого життя.

У цьому розділі необхідно заглибитись у визначення та розуміння "розумних речей" в контексті Інтернету речей, дослідити фундаментальні концепції, принципи та характеристики, які визначають розумні речі, надаючи всебічний огляд їхніх можливостей та застосувань.

2.1 Технології інтернету речей

Інтернет речей (IoT) - це сфера, що швидко розвивається і охоплює широкий спектр технологій. Цей підрозділ має на меті надати огляд ключових технологій, які забезпечують функціонування та експлуатацію систем IoT. Розуміння цих технологій має вирішальне значення для розуміння основних принципів і можливостей пристроїв і мереж IoT[2].

Технології зв'язку:

Бездротовий зв'язок: бездротові технології, такі як Wi-Fi, Bluetooth, і стільникові мережі забезпечують зв'язок між пристроями і хмарами.

Дротовий зв'язок: такі технології, як Ethernet, Powerline Communication і Industrial Ethernet, забезпечують надійне і безпечне дротове з'єднання.

Сенсорні технології:

Датчики навколишнього середовища: датчики температури, вологості, освітленості, тиску і газу збирають дані про навколишнє середовище в режимі реального часу.

Датчики руху та наближення: виявляють рух, присутність і близькість для запуску дій або виявлення змін у навколишньому середовищі.

Біометричні датчики: вимірюють фізіологічні параметри, такі як частота серцевих скорочень, кров'яний тиск і температура тіла, для медичних застосувань.

Датчики зображення та зору: камери та датчики зображення дозволяють збирати та аналізувати візуальні дані.

Вбудовані системи та мікроконтролери:

Мікроконтролери: малогабаритні обчислювальні пристрої, які керують і контролюють пристрої Інтернету речей, з інтегрованими процесорами, пам'яттю і периферійними пристроями.

Прошивка та розробка програмного забезпечення: програмування та розробка вбудованих систем для забезпечення функціональності пристроїв IoT.

Хмарні та периферійні обчислення:

Хмарні обчислення: інфраструктура і сервіси, розміщені в хмарі для зберігання, обробки та аналізу даних, що генеруються пристроями IoT.

Граничні обчислення: розподіл обчислювальних ресурсів і обробка даних ближче до межі мережі для зменшення затримок і підвищення конфіденційності.

Аналіз даних і машинне навчання:

Обробка та аналіз даних: методи і алгоритми для вилучення цінної інформації з великих обсягів даних, що генеруються IoT.

Машинне навчання та штучний інтелект: алгоритми, які дозволяють системам IoT навчатися і адаптуватися на основі шаблонів даних і приймати інтелектуальні рішення.

Безпека і конфіденційність:

Аутентифікація та авторизація пристроїв: безпечна ідентифікація та механізми контролю доступу для запобігання несанкціонованому доступу до пристроїв.

Шифрування та цілісність даних: методи захисту передачі та зберігання даних від несанкціонованого доступу та втручання.

Збереження конфіденційності: забезпечення конфіденційності та приватності даних користувачів, зібраних пристроями IoT.

Розуміння цих технологій забезпечує основу для розуміння складнощів і можливостей екосистеми Інтернету речей. Ефективно використовуючи ці технології, системи Інтернету речей можуть реалізувати свій потенціал у різних сферах, включаючи розумні будинки, охорону здоров'я, сільське господарство, транспорт і промислову автоматизацію[2-3].

2.2 Розумні речі у повсякденному житті

Розумні речі стали невід'ємною частиною нашого повсякденного життя, трансформуючи спосіб взаємодії з навколишнім світом. Ці інноваційні пристрої покликані підвищити зручність, ефективність та загальну якість життя. У цьому підрозділі досліджується значення та вплив "розумних" речей

на наше повсякденне життя, проливаючи світло на те, як вони революціонізували різні аспекти нашого життя.

Розумні речі зараз використовуються у різних сферах нашого життя, іноді ми навіть не помічаємо цього. Наприклад, для автоматизації розумного будинку: інтеграція розумних пристроїв застосовується для автоматизованого керування освітленням, температурою, системами безпеки та побутовими приладами. Голосові асистенти, що дозволяють керувати без допомоги рук і безперешкодно взаємодіяти з розумними пристроями[3].

Медицина та сфера охорони здоров’я використовують пристрої для відстеження занять фітнесом, моніторингу життєво важливих показників і пропаганди здорового способу життя. Розумні системи охорони здоров'я, що забезпечують віддалений моніторинг пацієнта, нагадують про прийом ліків та екстрену допомогу.

У логістичній сфері існують підключені транспортні засоби з вбудованою навігацією, оновленням інформації про дорожній рух у реальному часі та розумними рішеннями для паркування. Системи управління транспортом, що оптимізують маршрути, зменшують затори та підвищують безпеку[3].

Смарт-телевізори, потокові пристрої та розважальні системи з голосовим управлінням для персоналізованого споживання контенту, інтеграція розумних колонок і віртуальних помічників – все це розумні речі, створені для захоплюючого та інтерактивного медіа-досвіду.

Для покращення досвіду шоппінгу, створені покупки з підтримкою Інтернету речей, такі як "розумні" полиці, системи самообслуговування та персоналізовані рекомендації. Підключені прилади та пристрої для безперебійних онлайн-покупок, управління запасами та послуг доставки.

Навіть розумні лічильники та системи допомагають нам керувати енергоспоживанням для ефективного використання та збереження енергоресурсів. Активно впроваджується інтеграція відновлюваних джерел енергії та технологій "розумних мереж" для сталого енергоспоживання[3].

Впровадження інфраструктури Інтернету речей для ефективного міського планування, управління дорожнім рухом, утилізації відходів та надання комунальних послуг формують Розумні міста. Розумні системи освітлення, розумне паркування та інтелектуальне спостереження, що підвищують безпеку та якість життя в містах.

У цьому підрозділі ми заглибились в кожну з цих сфер, досліджуючи, як "розумні речі" проникли в нашу повсякденну діяльність, забезпечуючи зручність, персоналізацію та підвищення ефективності. Дослідження їх впливу у різних сферах допоможе нам зрозуміти значення та потенційні можливості у створенні розумнішого та ефективнішого світу[3].

2.3 Перспективи розвитку сфери розумних речей

Заглиблюючись у сферу розумних речей, важливо дослідити потенційні перспективи їхнього розвитку. Хоча визначення "розумних речей" дає фундаментальне розуміння, не менш важливо розглянути майбутню траєкторію розвитку цієї технології. Цей підрозділ має на меті пролити світло на потенційні досягнення та можливості, які чекають на нас у розвитку розумних речей.

Очікується, що "розумні" речі стануть все більш взаємопов'язаними, використовуючи такі технології, як 5G, мережі Інтернету речей і периферійні обчислення. Ці досягнення уможливлять безперебійний зв'язок, обмін даними в режимі реального часу та розширену функціональність між різними пристроями[4].

Інтеграція технологій штучного інтелекту, таких як машинне навчання і обробка природної мови, має великі перспективи для розумних речей. ШІ може розширити можливості цих пристроїв завдяки розширеній аналітиці, алгоритмам прогнозування та інтелектуальній автоматизації. Така інтеграція може призвести до персоналізованого досвіду, ефективного прийняття рішень та проактивної поведінки пристроїв.

Розвиток передових сенсорних технологій відкриє нові можливості для розумних речей. Мініатюрні датчики з підвищеною точністю, чутливістю і

функціональністю дозволять пристроям збирати більш точні дані, що призведе до поліпшення моніторингу, аналізу і контролю. Це, в свою чергу, може призвести до покращення користувацького досвіду та оптимізації продуктивності[4].

Майбутнє розумних речей залежить від досягнення інтероперабельності та стандартизації різних платформ, пристроїв та екосистем. Зусилля зі створення спільних протоколів, стандартів зв'язку та форматів даних сприятимуть безперешкодній інтеграції та співпраці між різними "розумними" пристроями, що дозволить користувачам змішувати та поєднувати пристрої різних виробників без проблем сумісності.

Розвиток розумних речей буде зосереджений на енергоефективності та сталості. Оптимізоване управління енергоспоживанням, технології збору енергії та екологічний дизайн стануть ключовими факторами при розробці цих пристроїв. Такий підхід не лише мінімізує вплив на навколишнє середовище, але й збільшить довговічність пристроїв та зменшить експлуатаційні витрати.

Майбутні "розумні" речі надаватимуть пріоритет користувацькому досвіду, орієнтуючись на інтуїтивно зрозумілі інтерфейси, персоналізовану взаємодію та безперешкодну інтеграцію в повсякденне життя. Голосові помічники, розпізнавання жестів, технології доповненої реальності і віртуальної реальності можуть відігравати значну роль у створенні інтуїтивно зрозумілого користувацького досвіду[4].

2.4 Висновки до розділу

Визначення "розумні речі" охоплює технології, які забезпечують їхню функціональність, інтеграцію в повсякденне життя, а також багатообіцяючі перспективи їхнього майбутнього розвитку. Оскільки ми продовжуємо входити в еру взаємопов'язаних пристроїв, вкрай важливо розвивати цілісне розуміння розумних речей та їхніх наслідків, щоб скористатися їхніми перевагами та подолати пов'язані з ними виклики.

3 АКТУАЛЬНІСТЬ ВИКОРИСТАННЯ КАСТОМІЗОВАНИХ ПЛАТФОРМ З БЕЗКОДОВОЮ ТА МАЛО-КОДОВОЮ РЕАЛІЗАЦІЯМИ

Підхід без коду і з низьким рівнем коду полегшує розробку і розгортання додатків Інтернету речей для підприємств і приватних осіб, оскільки він усуває необхідність в обширному досвіді кодування і може скоротити час і витрати на розробку. Ці платформи також сприяють швидшому впровадженню інновацій та створенню прототипів, оскільки користувачі можуть швидко експериментувати і повторювати свої ідеї без необхідності мати спеціалізовані технічні навички[5].

3.1 Основні переваги та недоліки платформ

3.1.1 Переваги

1. Легкість використання

Платформи, як правило, мають невеликий набір інструментів, що допомагає розробляти різноманітні додатки без особливих зусиль. Розробка додатків стає набагато більш керованою для нетехнічного персоналу, який не володіє навичками в цій галузі. Основною перевагою розробки без коду, особливо для досвідчених розробників, є те, що вона є швидкою. Готові модулі скорочують час, необхідний для реалізації функціональності програми, тому розробники можуть витрачати час на завдання, які потребують більшої оригінальності або які мають більший пріоритет для бізнесу. Малокодова розробка також може допомогти розробникам інтегрувати функцію із зовнішньою платформою, не вивчаючи всі тонкощі цієї зовнішньої платформи[5].

2. Вартість

Платформи малокодової/безкодової розробки допомагають компаніям розробляти додатки для негайного використання швидко і з меншими витратами. Традиційна розробка вимагає наявності власної команди або залучення сторонніх розробників. При low-code розробці команда не потрібна: організація може найняти розробників на неповний робочий день для роботи

над відповідним проектом. Це також залежить від вартості підписки на платформи, що використовуються. Зазвичай це набагато дешевше, ніж наймати професійних розробників, які вимагають вищої оплати відповідно до своїх навичок і знань.

3. Обслуговування

Завдяки платформам без коду та їхнім заздалегідь створеним і протестованим модулям, впровадження зміни є більш простим та ефективним. Крім того, ризики несумісності низькі. Платформи з низьким рівнем коду, які є прикладними платформами як послугами, зазвичай керуються і управляються організацією, яка обслуговує платформу.

4. Гнучкість

Ці платформи набагато простіші у використанні завдяки інтуїтивно зрозумілому інтерфейсу, а також завдяки функції перетягування. На відміну від індивідуальної розробки, тут немає потреби в кодуванні для розробки додатків/програмного забезпечення, оскільки для цього достатньо використовувати техніки малювання[5].

3.1.2 Недоліки

1. Залежність від третьої сторони

При використанні платформи з низьким рівнем коду або без коду, зменшення ризиків та захист від вразливостей здебільшого залежать від продавця, а також у зміні графіку оновлень для узгодження з постачальником.

2. Обмежена кастомізація

Платформи розробки з низьким рівнем коду/без коду зазвичай пропонують компаніям напрочуд обмежені можливості для розробки кастомізованих або створених на замовлення програмних додатків.

3. Обмежені можливості інтеграції

Можливості створення додатків на платформі розробки з низьким рівнем коду/без коду обмежують можливості інтеграції для розробників. Це може бути значною проблемою для компаній із застарілими системами, які є життєво важливими для їхнього бізнесу.

4. Нестача розробників

Оскільки розробка з низьким рівнем коду/без коду не є популярною сферою діяльності, компаніям складно знайти розробників, які володіють необхідними знаннями. Компаніям складно знайти розробників, які мають навички малокодової/безкодової розробки[5].

3.2 Огляд популярних платформ з безкодовою реалізацією

Платформи інтернету речей без коду є найважливішою частиною ринку IoT. Цей тип платформ дозволяє розробляти і розгортати багато додатків Інтернету речей без необхідності будь-якого досвіду програмування. Її легко розгортати і вона здатна створювати додатки для найрізноманітніших сценаріїв використання. Деякі з переваг платформ IoT без коду - це можливість швидкого процесу розробки, швидке розгортання і можливість розробляти додатки в галузях, які ніколи раніше не мали додатків IOT[6].

1. Losant

Losant Enterprise IoT Platform - це платформа для розробки додатків, яка дозволяє підприємствам створювати продукти, послуги та додатки Інтернету речей, які безпечно масштабуються до мільйонів пристроїв. Losant дозволяє створювати рішення Інтернету речей для клієнтів, які працюють у різних галузях, включаючи телекомунікації, виробництво та промисловий Інтернет речей.Шаблони додатків Losant надають набір реальних еталонних реалізацій, які ви можете використовувати для орієнтації в архітектурі та найкращих практик впровадження[7].

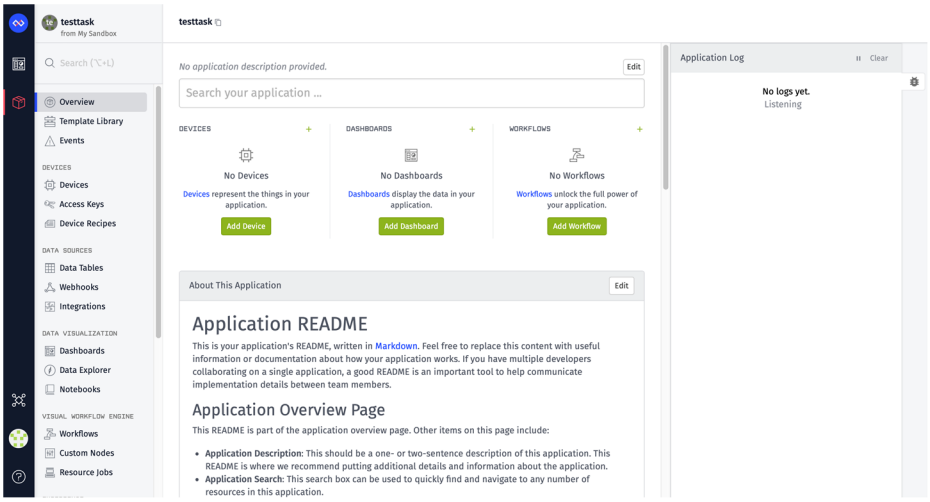

Рисунок 3.1– Інтерфейс платформи Losant

Дана демо-версія платформи дозволяє

– Розгортати логіку на периферійних пристроях, використовуючи функціонал периферійних обчислень Losant. Зменшувати залежність від підключення і попередньо відфільтрованих даних, які будуть відправлені в хмару.

– Організувати дані і пристрої за допомогою функцій тегування і моделювання пристроїв від Losant. Забезпечити безпеку за допомогою протоколів шифрування TLS і повністю відкличних ключів доступу для кожного пристрою.

Рисунок 3.2– Можливості платформи Losant

– Збирати, порівнювати, звітувати та реагувати на дані в режимі реального часу або інтегрувати Jupyter Notebooks для пакетної обробки історичних даних та глибокої аналітики. Візуалізувати інформацію за допомогою

графіків, карт і журналів Losant, щоб впливати на прийняття негайних бізнес-рішень.

– Прискорити розробку навіть найскладніших IoT-рішень за допомогою перетягування Visual Workflow Engine від Losant.
– Перетворити необроблені дані в досвід для внутрішніх користувачів або клієнтів за допомогою Losant Experience Views. Створюйте багатокористувацькі додатки для клієнтів ваших клієнтів. Швидко створюйте унікальні брендовані домени, додатки або звіти[7].

2. Akenza

Akenza - це платформа для розробки додатків Інтернету речей, що дозволяє створювати чудові продукти та послуги Інтернету речей з високою вартістю. Вона з'єднує, контролює та керує пристроями Інтернету речей - і все це в одному місці. Пряма інтеграція зі сторонніми аналітичними системами, вдосконалені тригери правил, функції з низьким рівнем коду та підтримка численних технологій підключення[8].

Рисунок 3.3– Девайси, якими можна керувати через платформу Akenza

Дана платформа дозволяє використовувати кодери та декодери корисного навантаження для каналів зв'язку найпопулярніших готових пристроїв Інтернету речей, вже інтегрованих у нашу бібліотеку типів пристроїв з відкритим кодом. Ви також можете створити власний тип пристрою відповідно до ваших потреб у нормалізації даних і поділитися ним зі своєю командою

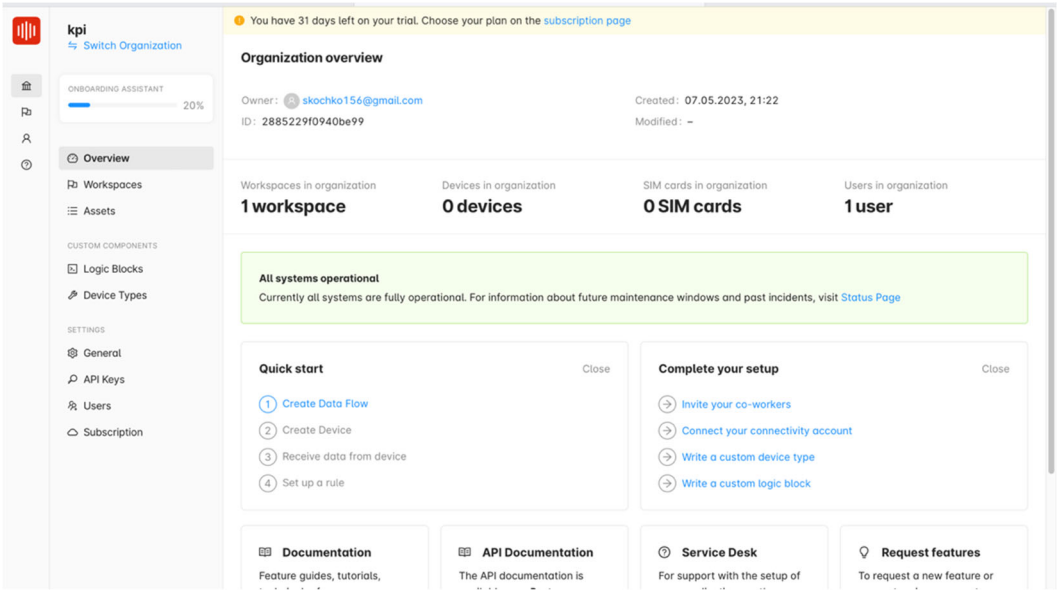

Рисунок 3.4– Інтерфейс платформи Akenza

Проста візуалізація даних дозволить отримати перше уявлення про ваше рішення IoT. Використовуйте вбудовану візуалізацію даних для безпосереднього огляду даних з вашого пристрою, легко підключіться до конструктора дашбордів, такого як Grafana, або скористайтеся однією з наших інтеграцій з Azure IoT Hub та іншими розширеними інструментами візуалізації.

Пряма інтеграція зі сторонніми аналітичними системами, вдосконалені правила запуску, функції з низьким рівнем коду та підтримка численних технологій підключення гарантують, що ви отримаєте в свої руки найкращу технологію Інтернету речей.

Потоки даних дозволяють миттєво і без кодування створювати шлях даних. Заощаджуйте дорогоцінний час, визначивши загальний потік обробки даних від пристроїв до з'єднання, через нормалізацію даних і до бажаної програми виводу[8].

3. Blynk

Blynk став піонером у створенні додатків Інтернету речей без коду і здобув світову популярність завдяки своєму редактору мобільних додатків. Сьогодні компанії будь-якого розміру використовують нашу програмну платформу для створення та управління підключеними продуктами[9].

Платформа Blynk складається з чотирьох основних компонентів, які злагоджено працюють разом:

1. Blynk.Edgent: програмне забезпечення, яке працює на вашому пристрої та взаємодіє з Blynk.Cloud.
2. Blynk.Console: веб-додаток, за допомогою якого ви можете налаштовувати, підключати, контролювати свої пристрої, аналізувати дані датчиків, оновлювати прошивку OTA та керувати доступом інших користувачів та організацій до своїх пристроїв.
3. Blynk.Apps: мобільні додатки для iOS та Android, де ви можете створювати інтерфейс для своїх пристроїв без кодування та ділитися ним з іншими користувачами.
4. Blynk.Cloud: сервер, який безпечно надсилає дані між вашими пристроями та додатками

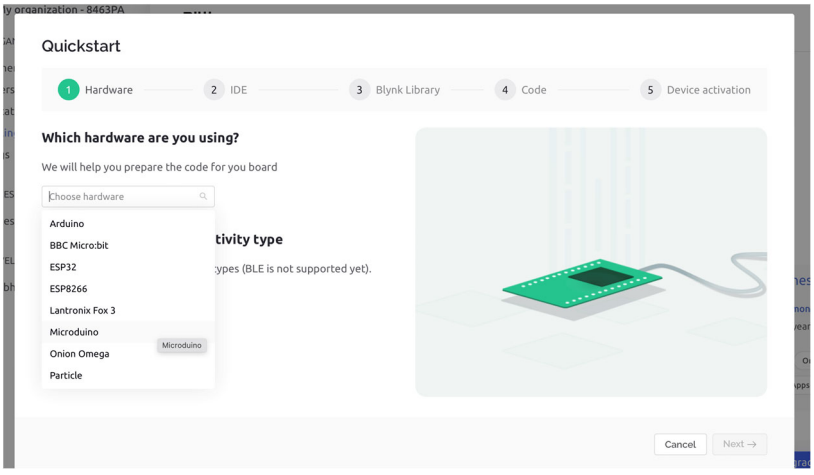

Рисунок 3.4– Онбордінг на платформі Blynk

Конфігурації пристроїв зберігаються у так званому шаблоні пристрою. Кожен пристрій починається з шаблону, що полегшує роботу з кількома пристроями, які виконують схожі функції.

Наприклад, ви можете створити шаблон датчика температури і використовувати його для всіх подібних датчиків у вашому домі.

– Кожен шаблон складається з:
– Потоки даних - канали для передачі даних з/на пристрій
– Інтерфейс мобільного додатку
– Інтерфейс веб-панелі управління
– Сповіщення

Коли ви оновлюєте шаблон, зміни будуть застосовані до всіх пристроїв, створених на основі цього шаблону[9].

3.3 Огляд популярних платформ з малокодовою реалізацією

Платформи Інтернету речей з низьким рівнем коду, які дозволяють швидко і легко створювати додатки та інформаційні панелі для Інтернету речей. Платформи з низьким рівнем коду дозволяють непрограмістам розробляти і розгортати додатки та інформаційні панелі без необхідності володіти поглибленими навичками програмування.

Платформи з низьким рівнем коду для IoT готові змінити спосіб розробки та розгортання Інтернету речей. Платформи з низьким рівнем коду призначені для непрограмістів, які можуть легко створювати додатки та інформаційні панелі без поглиблених навичок програмування. Платформи з низьким рівнем коду для IoT ідеально підходять для підприємств, які шукають недорогий і якісний спосіб створення та управління своїм Інтернетом речей.

1. Any2Info

Any2Info – це провідна хмарна платформа, яка дозволяє клієнтам створювати складні бізнес-додатки швидше, якісніше і дешевше, ніж традиційне програмування. Будь-хто може використовувати Any2Info для створення високоякісного додатку за меншу вартість, ніж традиційне програмування. За допомогою Any2Info користувачі можуть створювати практично все, що завгодно, включаючи мобільні додатки для збору даних, інформаційні панелі KPI та цифрові замовлення на виконання робіт, а також

клієнтські портали та рішення для бек-офісу[10].

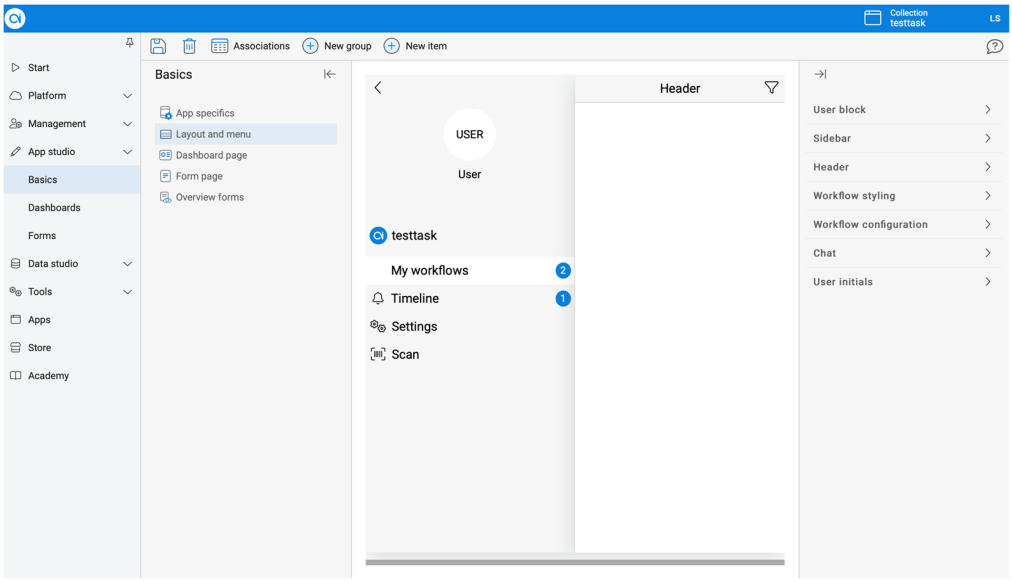

Рисунок 3.5– Інтерфейс платформи Any2Info

Використовуючи платформу Any2info, компанії можуть автоматизувати процеси вилучення та аналізу даних, зменшити кількість ручної роботи та підвищити точність і надійність своїх даних. Академія Any2info надає навчання та підтримку, щоб допомогти компаніям розпочати роботу з платформою та оптимізувати використання її функцій і можливостей[10].

2. Fogwing

Fogwing – промислова IoT-платформа нової ери, розроблена з функціями для побудови IoT-рішень. Це комплексне рішення для всіх потреб бізнесу, організацій та приватних осіб. Це перша промислова IoT-платформа, яка є на 100% безпечною, економічно ефективною та масштабованою. Fogwing дозволяє користувачам створювати надійні, масштабовані та оптимізовані рішення за менший час, ніж будь-коли раніше[11].

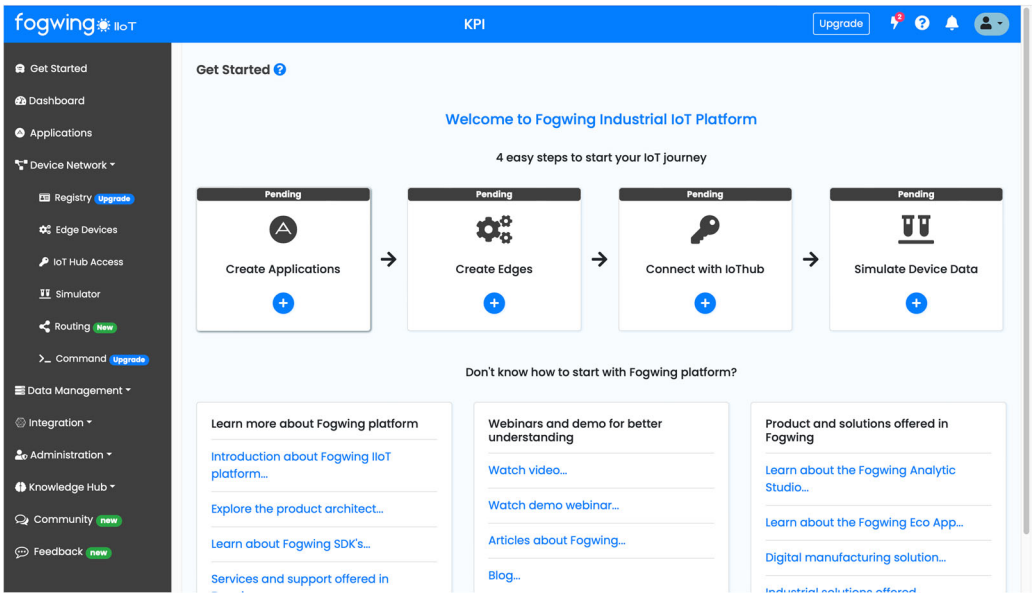

Рисунок 3.6– Інтерфейс платформи Fogwing

Архітектура платформи Fogwing IIoT складається з чотирьох рівнів:

1. Рівень пристроїв: Цей рівень передбачає підключення промислових активів, пристроїв і датчиків до платформи за допомогою різних варіантів підключення, таких як Bluetooth, Zigbee, Wi-Fi і стільникові мережі.
2. Шлюзовий рівень: Шлюзовий рівень виступає посередником між рівнем пристроїв і хмарним рівнем. Він збирає дані з пристроїв, обробляє їх локально і надсилає на хмарний рівень для зберігання та аналізу.
3. Хмарний рівень: Цей рівень зберігає дані, зібрані з пристроїв, і виконує різні аналізи та обчислення для отримання цінної інформації. Платформа Fogwing використовує різні хмарні сервіси, такі як AWS, Azure та Google Cloud Platform для зберігання та обробки даних.
4. Прикладний рівень: Прикладний рівень надає різні програми та інструменти для візуалізації, аналізу та управління промисловими процесами на основі даних, отриманих з хмарного рівня[11].

3. Make

Make – це платформа низькокодової розробки, яка дозволяє користувачам створювати кастомні веб- та мобільні додатки без написання коду. Платформа має інтерфейс drag-and-drop, що дозволяє користувачам створювати користувацькі інтерфейси, підключатися до джерел даних та визначати бізнес-

логіку. Платформа Make також включає функції для управління робочим процесом, звітності та аналітики[12].

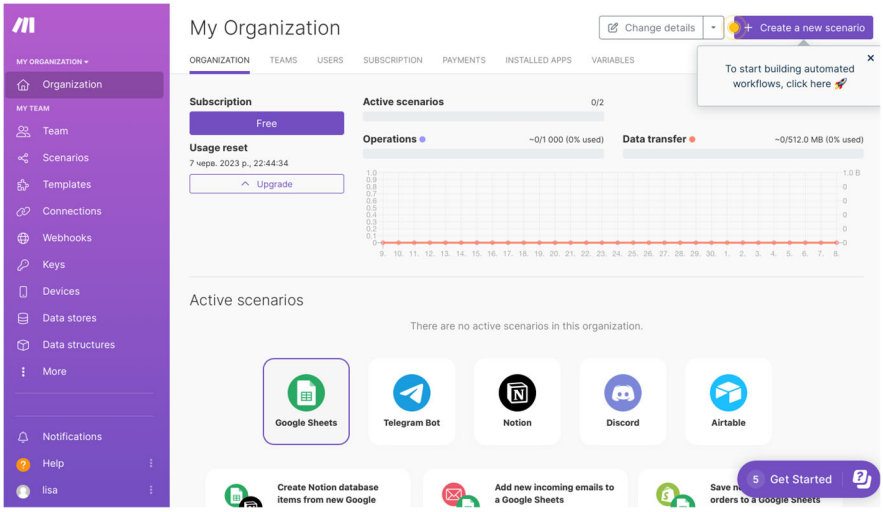

Рисунок 3.7– Інтерфейс платформи Make

За допомогою платформи Make можна:

– Інтегрувати всі свої інструменти та керувати процесами в одній платформі. Вибирати з тисяч готових інтеграцій з додатками або підключитись до будь-якого онлайн-додатку за допомогою наших потужних інструментів без коду.
– Використоввати зручний конструктор перетягування, щоб підключати програми в кілька кліків і створювати робочі процеси, які називаються сценаріями.
– Створити сценарії з такою кількістю кроків або програм, яка вам потрібна.
– Створити сценарій і спостерігати за його виконанням у режимі реального часу. Заплануйте запуск сценарію миттєво або коли вам потрібно.
– Вибирати з тисяч готових додатків або підключитись до будь-якого загальнодоступного API за допомогою HTTP-додатку Make.

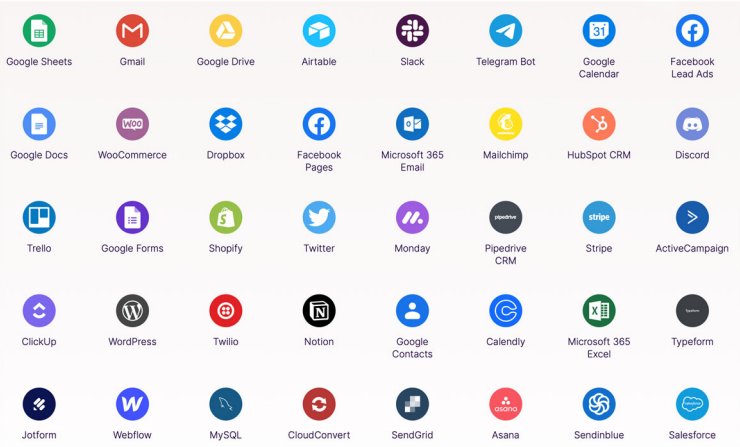

Рисунок 3.7– Додатки, що можна інтегрувати у Make

Make робить акцент на візуальному, спільному підході до розробки додатків, що дозволяє користувачам працювати разом над одним проектом в режимі реального часу. Платформа також надає шаблони та готові компоненти, які допоможуть користувачам швидко розпочати роботу, а також інтеграцію з популярними інструментами, такими як Slack та Trello[12].

3.4 Висновки до розділу

З наведеного вище аналізу різноманітних публікацій за темою можна зробити висновок, що як платформи No code та Low Code Development мають свої переваги та недоліки. Залежно від бізнес-вимог, можна обрати відповідний стек технологій. При цьому важливо пам'ятати, що платформи не призначені для повного усунення необхідності в традиційних/кастомних методологіях розробки. Вони здебільшого призначені для того, щоб забезпечити кожного члена команди, що співпрацює, необхідними інструментами та ресурсами, які вони можуть використовувати.

4 ФОРМУВАННЯ СИСТЕМИ МЕТРИК ДЛЯ ПОРІВНЯЛЬНОГО АНАЛІЗУ

4.1 Методи аналізу

Платформи розробки з низьким рівнем коду/без коду забезпечують знайоме і влучне рішення для бізнес-специфікацій за меншу частину вартості на відміну від традиційних еквівалентів розробки, а також вдвічі, якщо не більше, швидше. За допомогою готових модулів та інтуїтивно зрозумілого інтерфейсу, який зазвичай включає в себе функцію перетягування для налаштування моделей процесів та фреймворку додатку, вони роблять розробку додатків простішою, швидшою та розширюваною. Це також зменшує складність вивчення різних мов програмування і допомагає людям з меншим досвідом у сфері технологій розробляти та створювати додатки чи продукти для свого бізнесу без особливих труднощів. Однак вони не призначені для заміни традиційних практик чи методологій розробки, а радше слугують альтернативою для конкретних випадків використання та бізнес-вимог організацій.

Як можна аналізувати такі платформи? Існує кілька способів аналізу платформ з низьким рівнем коду та без коду, залежно від цілей дослідження та конкретних аспектів платформ, які ви хочете оцінити. Ось кілька можливих методів:

- Функціональний аналіз оцінка функціональності кожної платформи, беручи до уваги функції та можливості, що надаються платформою. Це може включати порівняння платформ на основі їхньої здатності обробляти різні типи даних, інтегруватися з іншими системами, надавати аналітику та звітність, а також підтримувати різні типи робочих процесів.
- Аналіз користувацького досвіду: аналіз користувацького досвіду (UX) кожної платформи, враховуючи простоту використання, дизайн інтерфейсу та загальну зручність. Це може включати проведення користувацького тестування, збір відгуків від користувачів і порівняння

платформ на основі таких показників, як залученість і задоволеність користувачів.

- Технічний аналіз: оцінка базових технологій та архітектури кожної платформи, враховуючи такі фактори, як масштабованість, безпека та продуктивність. Це може включати перегляд документації та технічних специфікацій платформ, проведення оцінки безпеки та порівняльного аналізу продуктивності платформ.
- Аналіз ринку: перевірка ринкового ландшафту для платформ, враховуючи такі фактори, як розмір ринку, ключові гравці, ринкові тенденції та потенціал зростання. Це може включати проведення маркетингових досліджень, аналіз галузевих звітів та оцінку конкурентного середовища.
- Тематичні дослідження та кейси використання: аналіз тематичних досліджень та реальних випадків використання платформ, враховуючи такі фактори, як галузі та випадки використання платформ, переваги та виклики, з якими стикаються користувачі, а також потенціал для майбутніх

4.2 Спільні характеристики платформ

Для початку, необхідно обрати базові характеристики, які наявні в усіх платформах, щоб зробити первинний огляд. На цьому етапі клієнту буде зрозуміло, яка саме платформа йому точно не підходить. Обрані метрики:

- Варіанти розміщення
- Модель ціноутворення
- Ключові особливості
- Інтеграції
- Цільова аудиторія.

Після цього перевіримо кожну з обраних для дослідження платформ за цими показниками[7-12].

Losant:

- Варіанти розгортання: Локальне, хмарне (AWS, Azure, Google Cloud)

- Модель ціноутворення: На основі підписки (на основі використання)
- Ключові особливості: Механізм робочого процесу, візуалізація даних у реальному часі, управління пристроями, периферійні обчислення, розробка користувацьких API
- Інтеграції: AWS IoT, Microsoft Azure IoT, Google Cloud IoT, Twilio, Slack, Salesforce
- Цільова аудиторія Losant - це підприємства та розробники корпоративного рівня, які потребують передової платформи Інтернету речей з потужними можливостями робочого процесу, можливостями машинного навчання та широкими інструментами аналітики та візуалізації даних.

Akenza:

- Варіанти розгортання: Хмарне
- Модель ціноутворення: На основі підписки (на основі використання)
- Ключові особливості: Платформа IoT з низьким рівнем коду, візуалізація даних, управління пристроями, периферійні обчислення, аналітика IoT, прогнозоване обслуговування
- Інтеграції: AWS IoT, Microsoft Azure IoT, Google Cloud IoT, Sigfox, LoRaWAN
- Основна цільова аудиторія Akenza - це підприємства та розробники, яким потрібна платформа з сильним акцентом на можливості периферійних обчислень, предиктивного обслуговування та аналітики Інтернету речей.

Make:

- Варіанти розгортання: Хмарне
- Модель ціноутворення: На основі підписки (на основі використання)
- Ключові особливості: Платформа IoT з низьким рівнем коду, візуалізація даних, управління пристроями, периферійні обчислення, аналітика IoT, прогнозоване обслуговування

- Інтеграції: AWS IoT, Microsoft Azure IoT, Google Cloud IoT, Sigfox, LoRaWAN, десятки різних додатків
- Основна цільова аудиторія Make - це компанії та розробники, яким потрібна платформа з широкими можливостями інтеграції, аналітикою Інтернету речей та гнучкими функціями трансформації даних.

Any2info:

- Варіанти розгортання: Хмарне
- Модель ціноутворення: На основі підписки (на основі використання)
- Ключові можливості: Витяг даних, перетворення, аналіз, візуалізація
- Інтеграції: можливість інтегрувати дані із різних джерел
- Основна цільова аудиторія Any2info - це компанії та розробники, яким потрібна платформа з широкою підтримкою вилучення та перетворення даних, а також сильним фокусом на аналізі та візуалізації даних.

Fogwing:

- Варіанти розгортання: Хмарне
- Модель ціноутворення: На основі підписки (на основі використання)
- Ключові особливості: Платформа IoT, управління пристроями, аналіз даних, прогнозне обслуговування, моніторинг в реальному часі, дистанційне керування
- Інтеграції: AWS, Azure, Google Cloud Platform
- Основна цільова аудиторія Fogwing - це компанії та розробники, яким потрібна платформа з сильним акцентом на можливості периферійних обчислень, надійні функції управління пристроями та моніторинг в реальному часі.

Blynk:

- Варіанти розгортання: Хмарне
- Модель ціноутворення: Модель Freemium (базові функції безкоштовні, розширені функції на основі підписки)
- Ключові особливості: Платформа IoT, інтерфейс drag-and-drop,

візуалізація даних, управління пристроями, периферійні обчислення, розробка власних API

- Інтеграції: Arduino, Raspberry Pi, ESP32, Particle
- Основна цільова аудиторія Blynk - це компанії та розробники, яким потрібна платформа з широкими можливостями кастомізації, сильним акцентом на розробку мобільних додатків та комплексними функціями безпеки. Blynk особливо популярний серед любителів та ентузіастів, які хочуть створювати власні IoT-проекти.

Ці показники можна використовувати для порівняння варіантів розгортання, цінових моделей, ключових функцій та інтеграцій кожної платформи. Інші фактори, які слід враховувати, можуть включати простоту використання, масштабованість, безпеку та підтримку[7-12].

4.3 Детальне дослідження характеристик

Кожна платформа має свої особливості, тож можна детальніше заглибитись в аналіз, що допоможе у порівнянні.

Losant:

- Можливості робочого процесу: Чи надає платформа можливості для розробки та автоматизації робочих процесів за допомогою перетягування?
- Управління пристроями: Який рівень функцій управління пристроями, таких як реєстрація, конфігурація та оновлення прошивки, забезпечує платформа?
- Аналітика: Який рівень аналізу даних і можливостей машинного навчання забезпечує платформа, наприклад, моделювання даних і предиктивну аналітику?

Akenza:

- Граничні обчислення: Який рівень можливостей периферійних обчислень забезпечує платформа, таких як локальна обробка даних і прийняття рішень?
- Прогнозоване обслуговування: Який рівень функцій предиктивного

обслуговування забезпечує платформа, наприклад, прогнозування і виявлення несправностей?

- Аналітика IoT: Який рівень аналітичних можливостей IoT забезпечує платформа, наприклад, моделювання даних і виявлення аномалій?

Make:

- Інтеграція: Який рівень інтеграційних можливостей надає платформа, таких як управління API та інтеграція даних із зовнішніми системами?
- Аналітика Інтернету речей: Який рівень аналітичних можливостей IoT забезпечує платформа, наприклад, моделювання даних та виявлення аномалій?
- Трансформація даних: Який рівень можливостей трансформації даних забезпечує платформа, наприклад, нормалізація та фільтрація даних?

Any2info:

- Видобування даних: Який рівень можливостей вилучення даних забезпечує платформа, наприклад, веб-скрепінг та синтаксичний аналіз PDF-файлів?
- Трансформація даних: Який рівень можливостей трансформації даних забезпечує платформа, наприклад, нормалізацію та фільтрацію даних?
- Аналіз даних: Який рівень можливостей аналізу даних забезпечує платформа, наприклад, візуалізація даних та інформаційні панелі?

Fogwing:

- Граничні обчислення: Який рівень можливостей периферійних обчислень забезпечує платформа, наприклад, локальну обробку даних і прийняття рішень?
- Управління пристроями: Який рівень функцій управління пристроями, таких як реєстрація, конфігурація і оновлення прошивки, забезпечує платформа?
- Моніторинг в режимі реального часу: Який рівень можливостей моніторингу в реальному часі забезпечує платформа, наприклад, попередження та сповіщення в реальному часі?

Blynk:

- Кастомізація: Який рівень можливостей кастомізації надає платформа, наприклад, кастомні віджети та теми?
- Розробка мобільних додатків: Який рівень можливостей для розробки мобільних додатків надає платформа, наприклад, публікація та управління додатками?
- Безпека: Який рівень безпеки забезпечує платформа, наприклад, безпечну передачу даних та автентифікацію користувачів?

Ці додаткові показники можуть забезпечити більш детальне порівняння функцій і можливостей кожної платформи і допомогти бізнесу прийняти обґрунтоване рішення при виборі платформи IoT або аналітики даних[7-12].

4.4 Формування списку метрик для подальшого аналізу

Враховуючи особливості, спільні характеристики, переваги та недоліки кожної з платформ, я прийшла до такого списку метрик, за яким можна провести детальне порівняння систем[7-12].

- Керування пристроями: Ця метрика вимірює рівень функцій управління пристроями, що надаються платформою, таких як реєстрація, конфігурація та оновлення прошивки.
- Граничні обчислення: Ця метрика вимірює рівень можливостей периферійних обчислень, що надаються платформою, таких як локальна обробка даних і прийняття рішень.
- Аналітика: Цей показник вимірює рівень можливостей аналізу даних і машинного навчання, що надаються платформою, таких як моделювання даних і предиктивна аналітика.
- Інтеграція: Ця метрика вимірює рівень інтеграційних можливостей, що надаються платформою, таких як управління API та інтеграція даних із зовнішніми системами.
- Видобування даних: Ця метрика вимірює рівень можливостей вилучення даних, що надаються платформою, таких як веб-скрепінг та парсинг

PDF.

- Кастомізація: Ця метрика вимірює рівень можливостей кастомізації, що надаються платформою, наприклад, кастомні віджети та теми.
- Розробка мобільних додатків: Цей показник вимірює рівень можливостей розробки мобільних додатків, що надаються платформою, таких як публікація та управління додатками.
- Безпека: Ця метрика вимірює рівень функцій безпеки, що надаються платформою, таких як безпечна передача даних та автентифікація користувачів.
- Моніторинг у реальному часі: Ця метрика вимірює рівень можливостей моніторингу в реальному часі, що надаються платформою, наприклад, попередження та сповіщення в реальному часі.
- Профілактичне обслуговування: Цей показник вимірює рівень функцій превентивного обслуговування, що надаються платформою, таких як прогнозування та виявлення несправностей.
- Перетворення даних: Ця метрика вимірює рівень можливостей трансформації даних, що надаються платформою, таких як нормалізація та фільтрація даних.
- Аналіз даних: Ця метрика вимірює рівень можливостей аналізу даних, що надаються платформою, таких як візуалізація даних та інформаційні панелі.
- Цільова аудиторія
- Можна додати пункти головна перевага та головний недолік.

4.5 Переваги та недоліки використання кастомізованих платформ

Додаткова інформація про платформи, що допоможе при порівнянні

Losant:

+ Плюси: Надійні можливості робочого процесу, розширені функції керування пристроями, а також широкі можливості аналітики та машинного навчання.

– Мінуси: дорогі тарифні плани, крута крива навчання для початківців та обмежена підтримка периферійних обчислень.

Akenza:

+ Плюси: Велика увага до можливостей периферійних обчислень, широка підтримка предиктивного обслуговування та надійні можливості аналітики Інтернету речей.
– Мінуси: обмежені можливості налаштування, менш комплексні функції перетворення даних і відносно нова платформа на ринку порівняно з іншими.

Make:

+ Плюси: Широкі можливості інтеграції, сильний акцент на аналітиці Інтернету речей та гнучкі функції перетворення даних.
– Недоліки: обмежені можливості керування пристроями, менш надійні можливості периферійних обчислень і відносно новий продукт на ринку порівняно з іншими платформами.

Any2info:

+ Плюси: Широка підтримка вилучення та перетворення даних, сильний фокус на аналізі та візуалізації даних.
– Мінуси: обмежені функції управління пристроями, менш комплексні можливості аналітики Інтернету речей та обмежені можливості кастомізації.

Fogwing:

+ Плюси: Сильний акцент на можливості периферійних обчислень, надійні функції управління пристроями та широка підтримка моніторингу в режимі реального часу.
– Недоліки: обмежена підтримка трансформації даних, менш комплексні можливості аналітики Інтернету речей та обмежені можливості налаштування.

Blynk:

+ Плюси: Широкі можливості кастомізації, сильна орієнтація на

розробку мобільних додатків і комплексні функції безпеки.

– Мінуси: обмежена підтримка периферійних обчислень, менш широкі можливості аналітики Інтернету речей та обмежені функції трансформації даних.

4.6 Висновки до розділу

Платформи з безкодовою та малокодовою реалізацією дуже популярні у сучасному світі. Кожна платформа пропонує унікальні рішення та спрощує керування бізнес-процесами. Проаналізувавши основні характеристики, кожен користувач може знайти найбільш вигідну для себе пропозицію[7-12].

5 АНАЛІЗ КАСТОМІЗОВАНИХ ПЛАТФОРМ ЗА РОЗРОБЛЕНОЮ СИСТЕМОЮ МЕТРИК

Мета даного розділу - протестувати обрані метрики на прикладах, тобто на досліджених раніше безкодових та малокодових платформах. Розробити систему оцінки, що остаточно виділить найкращу систему для подальшої розробки і інтеграції розумної речі.

Щоб зробити процес оцінки зрозумілим та прозорим, а також обов'язково об'єктивним – застосуємо комплексну систему оцінки, де для кожної метрики існує оцінка від 1-5.

– Керування пристроями:

1: Лише базові функції керування пристроєм

2: Надано деякі функції керування пристроями, але вони не є вичерпними

3: Надано хороший рівень функцій керування пристроєм

4: Надано розширені функції керування пристроями

5: Надано комплексні та налаштовувані функції керування пристроями

– Граничні обчислення:

1: Граничні обчислення не підтримуються

2: Надано базові можливості периферійних обчислень

3: Надано хороший рівень можливостей периферійних обчислень

4: Надано розширені можливості периферійних обчислень

5: Надано комплексні та налаштовувані можливості периферійних обчислень

– Аналітика:

1: Можливості аналізу даних або машинного навчання не надаються

2: Надаються базові можливості аналізу даних або машинного навчання

3: Надається хороший рівень аналізу даних та можливостей машинного навчання

4: Надаються розширені можливості аналізу даних та машинного навчання

5: Надано комплексні та налаштовувані можливості аналізу даних і машинного навчання

– Інтеграція:

1: Надано обмежені можливості інтеграції

2: Надано деякі можливості інтеграції, але вони не є всеохоплюючими

3: Надано хороший рівень інтеграційних можливостей

4: Надано розширені можливості інтеграції

5: Надано комплексні та налаштовувані можливості інтеграції

- Отримання даних:

1: Можливості отримання даних не надано

2: Надано базові можливості отримання даних

3: Надано хороший рівень можливостей отримання даних

4: Надано розширені можливості отримання даних

5: Надано комплексні та налаштовувані можливості отримання даних

- Налаштування:

1: Надано обмежені можливості налаштування

2: Надано деякі можливості налаштування, але вони не є вичерпними

3: Надано хороший рівень можливостей налаштування

4: Надано розширені можливості налаштування

5: Надано всебічні та індивідуальні можливості кастомізації

- Розробка мобільних додатків:

1: Можливості для розробки мобільних додатків не надаються

2: Надано базові можливості для розробки мобільних додатків

3: Надано хороший рівень можливостей для розробки мобільних додатків

4: Надано розширені можливості для розробки мобільних додатків

5: Надано комплексні та налаштовувані можливості для розробки мобільних додатків

- Безпека:

1: Надано обмежені функції безпеки

2: Деякі функції безпеки надано, але вони не є всеохоплюючими

3: Надано хороший рівень функцій безпеки

4: Надано розширені функції безпеки

5: Надано комплексні та налаштовувані засоби захисту

– Моніторинг у режимі реального часу:

1: Можливості моніторингу в режимі реального часу не передбачено

2: Надано базові можливості моніторингу в режимі реального часу

3: Надано хороший рівень можливостей моніторингу в режимі реального часу

4: Надано розширені можливості моніторингу в режимі реального часу

5: Надано комплексні та налаштовувані можливості моніторингу в режимі реального часу

– Профілактичне обслуговування:

1: Функції профілактичного обслуговування не передбачені

2: Надаються базові функції профілактичного обслуговування

3: Надано хороший рівень функцій профілактичного обслуговування

4: Надано розширені функції профілактичного обслуговування

5: Надано комплексні та кастомізовані функції профілактичного обслуговування

– Перетворення даних:

1: Можливості трансформації даних не передбачено

2: Надано базові можливості трансформації даних

3: Надано хороший рівень можливостей трансформації даних

4: Надано розширені можливості трансформації даних

5: Надано комплексні можливості трансформації даних

– Аналіз даних:

1: Надано обмежені можливості аналізу даних

2: Деякі можливості аналізу даних надаються, але не є всеохоплюючими

3: Надано хороший рівень можливостей аналізу даних

4: Надано розширені можливості аналізу даних

5: Надано комплексні та кастомізовані можливості аналізу даних

– Цільова аудиторія:

Описує основну цільову аудиторію платформи, таку як розробники, підприємства або приватні особи.

– Основний недолік:

Основний недолік: Визначає основний недолік платформи, наприклад, обмежена підтримка певної мови програмування, висока ціна або відсутність певних функцій.

– Головна перевага:

Вказує на головну перевагу платформи, наприклад, простоту використання, широкі можливості або потужну підтримку спільноти.

5.1 Аналіз безкодових платформ

У цьому розділі оцінки та пояснення до них стосовно кожної з досліджуваних платформ будуть сформовані у таблиці для кращого розуміння.

– Losant

Таблиця 5.1 - Оцінка характеристик платформи Losant за розробленою системою метрик

Метрика	Оцінка	Опис
Керування пристроями	4	Losant надає комплексні функції управління пристроями, включаючи реєстрацію, конфігурацію та оновлення прошивки.
Граничні обчислення	3	підтримує деякі можливості периферійних обчислень, включаючи локальну обробку даних і прийняття рішень.
Аналітика	5	пропонує потужні можливості аналізу даних і машинного навчання, такі як моделювання даних і предиктивна аналітика.
Інтеграція	5	надає надійні інтеграційні можливості, включаючи управління API та інтеграцію даних із зовнішніми системами.
Отримання даних	3	пропонує деякі можливості для видобування даних, такі як веб-скрепінг та парсинг PDF-файлів.
Розробка мобільних додатків	3	підтримує деякі можливості розробки мобільних додатків, такі як публікація та управління додатками.
Кастомізація	5	пропонує широкі можливості кастомізації, такі

		як кастомні віджети та теми.
Безпека	4	забезпечує надійні функції безпеки, такі як безпечна передача даних та аутентифікація користувачів.
Моніторинг у режимі реального часу	4	надає можливості моніторингу в реальному часі, включаючи оповіщення та сповіщення в реальному часі.
Профілактичне обслуговування	4	пропонує деякі функції прогнозованого обслуговування, такі як прогнозування та виявлення несправностей.
Перетворення даних	5	надає надійні можливості перетворення даних, такі як нормалізація та фільтрація даних.
Аналіз даних	5	пропонує комплексні можливості аналізу даних, такі як візуалізація даних та дашборди
Цільова аудиторія	Промислові компанії, постачальники IoT-рішень та розробники	
Основний недолік	Крута крива навчання	
Головна перевага	Надійна інтеграція та можливості кастомізації	

Akenza

Таблиця 5.2 - Оцінка характеристик платформи Akenza за розробленою системою метрик

Метрика	Оцінка	Опис
Керування пристроями	4	надає комплексні функції управління пристроями, включаючи реєстрацію, конфігурацію та оновлення прошивки.
Граничні обчислення	3	підтримує деякі можливості периферійних обчислень, включаючи локальну обробку даних і прийняття рішень.
Аналітика	4	пропонує надійні можливості аналізу даних, включаючи моделювання даних і базову предиктивну аналітику
Інтеграція	5	надає потужні інтеграційні можливості, включаючи управління API та інтеграцію

		даних із зовнішніми системами.
Отримання даних	3	пропонує деякі можливості для видобування даних, такі як базовий веб-скрепінг та парсинг даних.
Розробка мобільних додатків	3	підтримує деякі можливості розробки мобільних додатків, що дозволяє користувачам розробляти та керувати власними мобільними додатками.
Кастомізація	4	забезпечує хороший рівень кастомізації, надаючи можливості для налаштування користувацьких інтерфейсів та робочих процесів.
Безпека	4	забезпечує надійні функції безпеки, включаючи безпечну передачу даних та автентифікацію користувачів.
Моніторинг у режимі реального часу	4	надає можливості моніторингу в режимі реального часу, дозволяючи користувачам контролювати пристрої та отримувати сповіщення та повідомлення в режимі реального часу.
Профілактичне обслуговування	3	пропонує деякі базові функції прогнозованого обслуговування, що дозволяють користувачам виявляти та вирішувати потенційні проблеми.
Перетворення даних	4	надає гнучкі можливості перетворення даних, включаючи нормалізацію даних і базові опції фільтрації.
Аналіз даних	4	пропонує комплексні можливості аналізу даних, включаючи візуалізацію даних і функції інформаційних панелей.
Цільова аудиторія	постачальники рішень IoT, підприємства та розробники	
Основний недолік	Обмежена інтеграція зі сторонніми розробниками	
Головна перевага	Зручний інтерфейс і простота використання	

– Blynk

Таблиця 5.3 - Оцінка характеристик платформи Blynk за розробленою системою метрик

Метрика	Оцінка	Опис
Керування пристроями	4	надає функції управління пристроями, дозволяючи користувачам реєструвати, налаштовувати та керувати своїми пристроями на платформі.
Граничні обчислення	2	не пропонує вбудованих можливостей граничних обчислень. Він в першу чергу зосереджений на підключенні та управлінні пристроями Інтернету речей.
Аналітика	2	пропонує базові можливості аналізу даних, що дозволяє користувачам візуалізувати та аналізувати дані, згенеровані їхніми пристроями.
Інтеграція	3	надає надійні можливості інтеграції, підтримуючи різні сторонні інтеграції та API для підключення до зовнішніх систем.
Отримання даних	2	не має спеціальних можливостей для видобування даних,
Розробка мобільних додатків	4	підтримує розробку мобільних додатків, дозволяючи користувачам створювати та керувати власними мобільними додатками для взаємодії зі своїми пристроями.
Кастомізація	5	дозволяє здійснювати широкі можливості кастомізації, надаючи опції для створення користувацьких інтерфейсів та віджетів для управління пристроями IoT.
Безпека	4	забезпечує безпечну передачу даних та автентифікацію користувачів, надаючи функції шифрування та контролю доступу для зв'язку між пристроями IoT.
Моніторинг у режимі	5	перевершує можливості моніторингу в

реального часу		реальному часі, надаючи користувачам оновлення даних, попередження та сповіщення з їхніх пристроїв в режимі реального часу.
Профілактичне обслуговування	2	не має вбудованих функцій прогнозованого обслуговування.
Перетворення даних	3	пропонує базові можливості перетворення даних, дозволяючи користувачам форматувати та маніпулювати даними, згенерованими їхніми пристроями.
Аналіз даних	3	надає базові функції аналізу даних, дозволяючи користувачам візуалізувати та аналізувати дані з підключених пристроїв.
Цільова аудиторія	ентузіасти, виробники та розробники Інтернету речей, які шукають зручну платформу для контролю та моніторингу пристроїв.	
Основний недолік	Blynk може не задовольнити потреби користувачів, які потребують розширеної аналітики або функцій периферійних обчислень, оскільки він більше зосереджений на контролі та моніторингу пристроїв.	
Головна перевага	зручнbq інтерфейс і простота, що робить його доступним для початківців і DIY IoT-проектів.	

5.2 Аналіз малокодових платформ

– Any2Info

Таблиця 5.4 - Оцінка характеристик платформи Any2Info за розробленою системою метрик

Метрика	Оцінка	Опис
Керування пристроями	4	надає функції управління пристроями для реєстрації, конфігурації та управління пристроями в межах платформи.
Граничні обчислення	3	підтримує деякі можливості периферійних обчислень, що дозволяє локально обробляти дані та приймати рішення на пристроях.

Аналітика	4	пропонує надійні можливості аналізу даних, включаючи моделювання даних і базову предиктивну аналітику для аналізу та прийняття рішень.
Інтеграція	5	надає потужні інтеграційні можливості, забезпечуючи безперебійне управління API та інтеграцію даних із зовнішніми системами.
Отримання даних	4	пропонує розширені можливості видобування даних, включаючи веб-скрейпінг, парсинг документів та інтеграцію з джерелами даних.
Розробка мобільних додатків	4	забезпечує хороший рівень кастомізації, надаючи опції для кастомних віджетів, тем і персоналізованих інформаційних панелей.
Кастомізація	4	ідтримує розробку мобільних додатків, дозволяючи користувачам створювати та керувати власними мобільними додатками для доступу до даних та контролю.
Безпека	4	забезпечує надійні функції безпеки, включаючи безпечну передачу даних, аутентифікацію користувачів і механізми контролю доступу.
Моніторинг у режимі реального часу	4	надає можливості моніторингу в режимі реального часу, дозволяючи користувачам контролювати пристрої та отримувати сповіщення та повідомлення в режимі реального часу.
Профілактичне обслуговування	4	надає можливості моніторингу в режимі реального часу, дозволяючи користувачам контролювати пристрої та отримувати сповіщення та повідомлення в режимі реального часу.
Перетворення даних	4	надає гнучкі можливості трансформації даних, включаючи процеси нормалізації, фільтрації та збагачення даних.

Аналіз даних	4	пропонує комплексні можливості аналізу даних, включаючи візуалізацію даних, аналіз трендів та інтерактивні дашборди.
Цільова аудиторія	Підприємства, постачальники рішень IoT та розробники	
Основний недолік	Обмежена підтримка певного обладнання або протоколів	
Головна перевага	Розширені можливості отримання даних та профілактичного обслуговування	

– Fogwing

Таблиця 5.5 - Оцінка характеристик платформи Fogwing за розробленою системою метрик

Метрика	**Оцінка**	**Опис**
Керування пристроями	4	надає комплексні функції управління пристроями, включаючи реєстрацію, конфігурацію та оновлення прошивки.
Граничні обчислення	5	перевершує можливості периферійних обчислень, забезпечуючи локальну обробку даних, прийняття рішень і аналітику в реальному часі на периферії.
Аналітика	4	пропонує надійні можливості аналізу даних, включаючи моделювання даних, аналітику в реальному часі та функції предиктивної аналітики.
Інтеграція	5	надає потужні інтеграційні можливості, підтримуючи різні протоколи та інтеграцію з хмарними платформами і системами сторонніх виробників.
Отримання даних	3	пропонує базові можливості вилучення даних, дозволяючи збирати дані з різних джерел, таких як API та бази даних.
Розробка мобільних додатків	3	ідтримує деякі можливості розробки мобільних додатків, що дозволяє користувачам створювати власні мобільні додатки та керувати ними.
Кастомізація	4	забезпечує хороший рівень кастомізації, надаючи можливості для створення власних

		правил, робочих процесів і конвеєрів обробки даних.
Безпека	5	забезпечує надійні функції безпеки, включаючи безпечну передачу даних, автентифікацію користувачів та механізми контролю доступу.
Моніторинг у режимі реального часу	5	вирізняється можливостями моніторингу в режимі реального часу, що дозволяє користувачам контролювати пристрої та отримувати сповіщення та повідомлення в режимі реального часу.
Профілактичне обслуговування	4	пропонує розширені функції предиктивного технічного обслуговування, що дозволяють прогнозувати несправності, виявляти аномалії та здійснювати проактивне технічне обслуговування.
Перетворення даних	4	надає гнучкі можливості трансформації даних, включаючи процеси нормалізації, фільтрації та збагачення даних.
Аналіз даних	4	пропонує комплексні можливості аналізу даних, включаючи візуалізацію даних, аналіз тенденцій та інтерактивні інформаційні панелі.
Цільова аудиторія	постачальники рішень для Інтернету речей, підприємства та розробники	
Основний недолік	Обмежені можливості розробки мобільних додатків	
Головна перевага	Потужні периферійні обчислення і моніторинг в реальному часі	

– Make

Таблиця 5.6 - Оцінка характеристик платформи Any2Info за розробленою системою метрик

Метрика	**Оцінка**	**Опис**
Керування пристроями	5	надає комплексну та настроювану платформу управління пристроями, яка дозволяє користувачам легко керувати підключеними

		пристроями.
Граничні обчислення	4	надає розширені можливості граничних обчислень, які дозволяють користувачам обробляти дані ближче до джерела та оптимізувати розгортання IoT.
Аналітика	5	надає комплексну і настроювану платформу для аналізу даних і машинного навчання, яка дозволяє користувачам отримувати інформацію з даних IoT.
Інтеграція	5	надає комплексну та настроювану інтеграційну платформу, яка дозволяє користувачам легко інтегрувати свої IoT-пристрої з іншими системами та додатками.
Отримання даних	5	надає комплексну платформу для вилучення даних, яка дозволяє користувачам витягувати дані з підключених пристроїв і надсилати їх в інші системи для обробки та аналізу.
Розробка мобільних додатків	5	надає комплексну платформу для розробки мобільних додатків, яка дозволяє користувачам легко розробляти та розгортати мобільні додатки для своїх підключених пристроїв.
Кастомізація	5	надає комплексну платформу, яка дозволяє користувачам легко налаштовувати свої розгортання IoT відповідно до їхніх конкретних потреб і вимог.
Безпека	4	адає розширені функції безпеки, які дозволяють користувачам убезпечити свої розгортання IoT та захистити свої дані від несанкціонованого доступу та кіберзагроз.
Моніторинг у режимі реального часу	5	надає комплексну платформу моніторингу в режимі реального часу, яка дозволяє користувачам контролювати свої розгортання IoT в режимі реального часу і швидко реагувати на будь-які проблеми або аномалії.

Профілактичне обслуговування	5	надає комплексну та настроювану платформу для прогнозованого обслуговування, яка дозволяє користувачам прогнозувати збої в роботі обладнання та запобігати простоям.
Перетворення даних	5	надає комплексну платформу для перетворення даних, яка дозволяє користувачам перетворювати дані Інтернету речей у формат, необхідний для інших систем і додатків.
Аналіз даних	5	надає комплексну платформу для аналізу даних, яка дозволяє користувачам аналізувати дані IoT та отримувати інформацію про їхню діяльність та продуктивність.
Цільова аудиторія	Make в першу чергу орієнтований на підприємства, які потребують комплексної платформи Інтернету речей для управління підключеними пристроями та отримання інформації з їхніх даних.	
Основний недолік	Висока ціна	
Головна перевага	Простота використання	

5.3 Результати аналізу

Таблиця 5.7 - Узагальнені результати аналізу платформ за розробленою системою метрик.

Метрика	Losant	Akenza	Blynk	Any2Info	Fogwing	Make
Керування пристроями	4	4	4	4	4	5
Граничні обчислення	3	3	2	3	5	4
Аналітика	5	4	2	4	4	5
Інтеграція	5	5	3	5	5	5
Отримання даних	3	3	2	4	3	5
Розробка мобільних додатків	3	3	4	4	3	5
Кастомізація	5	4	5	4	4	5
Безпека	4	4	4	4	5	4
Моніторинг у режимі реального часу	4	4	5	4	5	5

Профілактичне обслуговування	4	3	2	4	4	5
Перетворення даних	5	4	3	4	4	5
Аналіз даних	5	4	3	4	4	5
Всього	**50**	**45**	**39**	**48**	**50**	**58**

5.4 Висновки до розділу

У результаті, найбільше балів, 58 з 60-ти, набрала платформа Make - платформа низькокодової розробки, яка дозволяє користувачам створювати кастомні веб- та мобільні додатки без написання коду. Платформа має інтерфейс drag-and-drop, що дозволяє користувачам створювати користувацькі сценарії, підключатися до джерел даних та визначати бізнес-логіку. Платформа Make також включає функції для управління робочим процесом, звітності та аналітики.

В подальшій роботі розробка системи для розумних речей буде проводитись саме у цій платформі, базуючись на об’єктивних оцінках.

6 ПЕРЕВІРКА ВЖИВАНОСТІ ВИКОНАНОГО АНАЛІЗУ

У попередніх розділах було розглянуто популярні малокодові та безкодові платформи, проаналізовано їх особливості. Після цього створено систему метрик для порівняльного аналізу, за яким об'єктивно найвищу оцінку отримала платформа з низьким рівнем коду Make.

Наступним етапом буде перевірка вживаності проведеного аналізу, тобто розробка сценарію на платформі Make, що дозволяє отримувати дані від розумних девайсів та передавати їх для подальшої обробки.

За відсутності фізичних девайсів, будуть спроєктовані три прототипи датчиків розумного дому у симуляторі AWS IoT Simulator. Датчик вимірювання температури, вологості та рівня вмісту вуглекислого газу є основними девайсами при проєктуванні розумних будинків, тому саме їх використано в моїй роботі[13].

6.1 Створення прототипу системи розумного будинку у симуляторі

Як було сказано раніше, через відсутність у виконавця фізичних розумних девайсів, було прийняте рішення симулювати їх діяльність у симуляторі. AWS IoT Simulator надає можливість створити власні типи девайсів, параметри, та тип повідомлення, що вони будуть надсилати[14].

Мною було розроблено три пристрої:

- Датчик температури – Temperature_sensor(Whp s),
- Датчик вологості повітря – Humidity_sensor(kxp lgl|bydoxh„
- Датчик контролю вмісту вуглекислого газу в повітрі – CO2_sensor(fr5).

Датчик температури симулює вимірювання температури повітря в квартирі, генерує дані в діапазоні від 15 до 35 градусів по Цельсію.

```
-#
##qdp h%=#Whp s%/#
##ghidxo%=#53/#
##p lq%=#18/#
##w|sh%=#iordw%/#
##p d{%=#58/#
##suhflvlrq%=#5#
$#
```

Та видає повідомлення типу:

```
~#
##Whp s%=#53#
c#
```

Датчик вологості симулює вимірювання вологості повітря у квартирі, генерує дані в діапазоні від 20 до 60 відсотків.

```
~#
##qdp h%=#kxp lglw|bydoxh%#
##ghidxo%=#53/#
##p lq%=#53/#
##w|sh%=#kordw%/#
##p d{%=#93/#
##suhflvlrq%=#5#
c#
```

Видає повідомлення у форматі:

```
~#
##kxp lglw|bydoxh%=#53#
c#
```

І третій девайс, симулює вимірювання рівня вуглекислого газу в повітрі квартири, генерує дані в діапазоні від 200 до 400 ppm(часток на мільйон).

```
~#
##qdp h%=#fr5%/#
##ghidxo%=#533/#
##p lq%=#533/#
##w|sh%=#kordw%/#
##p d{%=#733/#
##suhflvlrq%=#5#
c#
```

Видає повідомлення у форматі:

```
#
~#
##fr5%=#533#
c#
#
```

Після проектування девайсів, треба розробити та запустити симуляцію. Створено три симуляції окремо для кожного пристрою: Temperature_simulation, Humidity_simulation, C02_simulation. Після їх запуску деваси видають свої повідомлення у заданому форматі раз на 15 секунд[14].

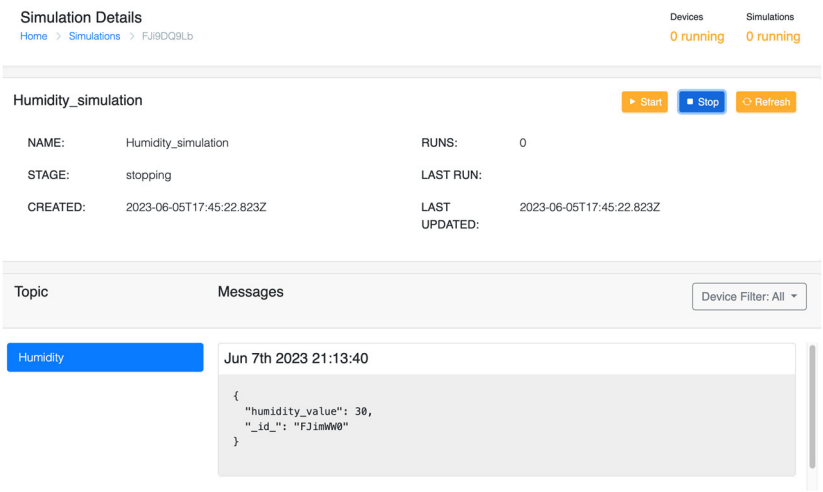

Рис. 6.1 – Демонстрація роботи симуляції датчика вологості.

Дані симуляції передаються за протоколом MQTT. Щоб перевірити справність програми, на платформі AWS є розроблений спеціально MQTT test client. Ви можете використовувати тестовий клієнт MQTT для моніторингу MQTT-повідомлень, що передаються у вашому обліковому записі AWS. Пристрої публікують MQTT-повідомлення, які ідентифікуються за темами, щоб повідомити AWS IoT про свій стан. AWS IoT також публікує MQTT-повідомлення, щоб інформувати пристрої та додатки про зміни та події. Ви можете підписатися на теми MQTT-повідомлень і публікувати MQTT-повідомлення в темах за допомогою тестового клієнта MQTT[14].

Перевіримо справність продемонстрованої раніше симуляції датчика вологості. Тема, за якою розташований цей девайс називається Humidity. Підпишемось на дану тему, щоб отримувати від симуляції повідомлення.

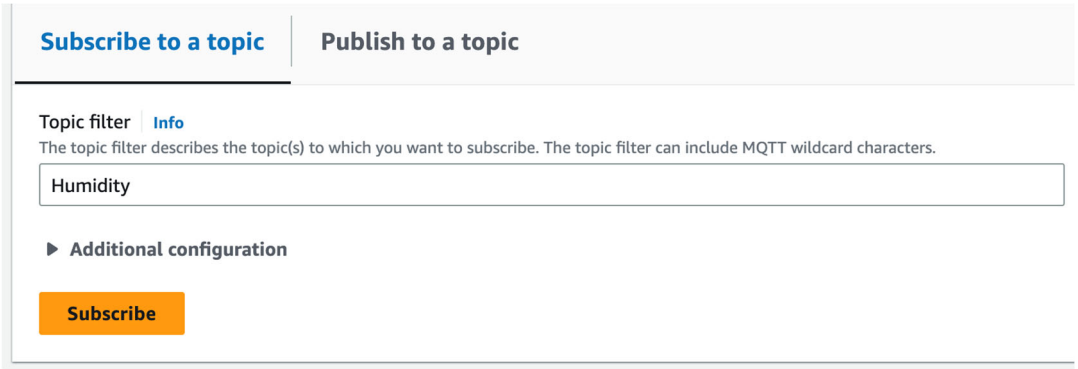

Рис. 6.2 – Підписка на датчик вологості.

Після цього, запустимо симуляцію вимірювання вологості знову і побачимо, що девайс справно відправляє дані на сервер Amazon за допомогою протоколу MQTT.

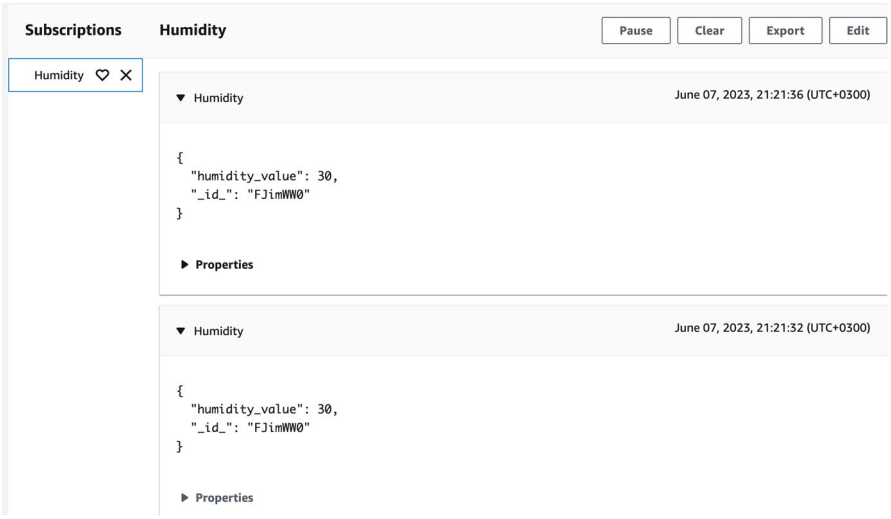

Рис. 6.3 – Тестування протоколу MQTT

Так як нам необхідно виводити дані з датчиків із системи AWS для подальшої обробки, наступним кроком буде створення Lambda-функції, що надсилатиме дані у форматі JSON через HTTP-протокол.

Перед цим створимо правила для кожного девайсу, що за допомогою SQL-скрипта передає до Lambda-функції лише ті дані, що нам потрібні, а саме – показники температури, вологості та рівня CO2.

SELECT Temp FROM Temperature - виділяє змінну Temp з повідомлення, що надсилає датчик температури та надсилає до функції.

SELECT humidity_value FROM Humidity - виділяє змінну humidity_value з повідомлення, що надсилає датчик вологості та надсилає до функції.

SELECT co2 FROM Oxygen - виділяє змінну co2 з повідомлення, що надсилає датчик вуглекислого газу та надсилає до функції.

Остаточним кроком є створення вище згаданих Lambda-функцій, що будуть пов’язувати платформу Make та дані, отримувані з симуляцій. Було створено три прості функції: Temperature_data, Humidity_data та Co2_data.

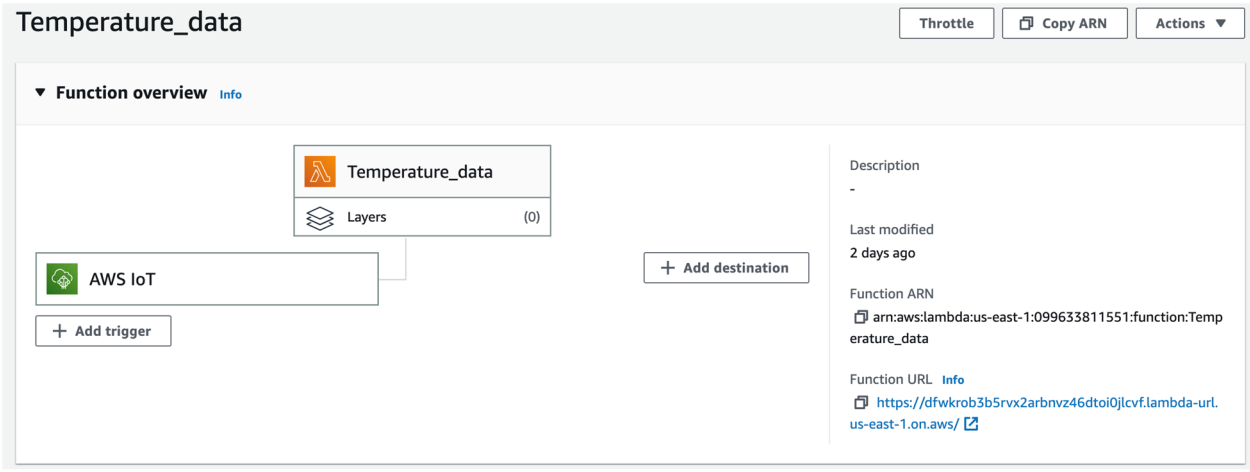

Рис. 6.4 – Структура lambda-функції Temperature_data.

Функція напряму пов’язана із розробленим у симуляторі девайсом через привило з SQL-скриптом і має Url адресу, на яку відправляються дані у форматі JSON. Код функції Temperature_data на мові програмування Python:

```
lp sru#mrq#
lp sru#dqgrp #
gh#dp egdbkdqgdhuhyhqw#frqwh{w,=#
###&#I{wdfw#wkh#hp shudwuh#ydoxh#urp #wkh#qfrp lqj#p hvvdjh#
whp shudwuh#@#-whp s*#sd|ordg^whp s'#
###&#Fuhdwh#d#MVRQ#hvsrqvh#z lwk#wkh#whp shudwuh#
###hvsrqvh#@##
######vwdwxvFrgh*#533/#
#######erg|*#mrq1gxp sv+-whp shudwuh*#whp shudwuh¢,#
###'#
###hwxq#hvsrqvh#
```

Функція надсилає дані у форматі JSON, переглянути їх можна за посиланням.

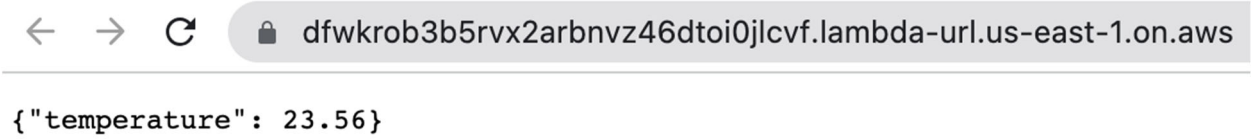

Рис. 6.5 – Вивід повідомлення функцією Temperature_data

Інші функції, а саме Humidity_data та Co2_data працюють за аналогічним алгоритмом.

6.2 Розробка системи взаємодії користувача з розумними речами на обраній платформі

Наступним етапом створимо сценарій на платформі Make. Додамо роутер, що буде об’єднувати три елементи, і запускати їх одночасно. Ці елементи – GET запити HTTP протоколу. Вони запитують дані із раніше створених функцій на сервері Amazon.

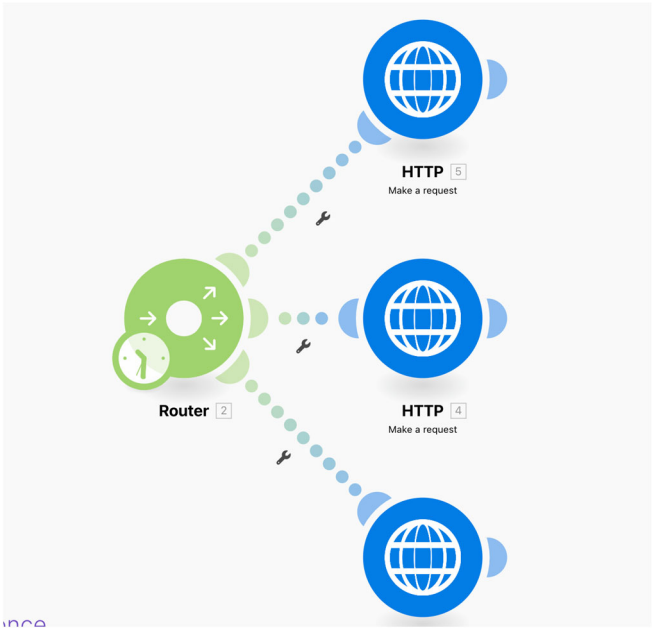

Рис. 6.5 - Структура сценарію

Запустивши сценарій, ми переконаємось у його коректній роботі. Вихідні дані методу мають показники температури, вологості та вуглекислого газу.

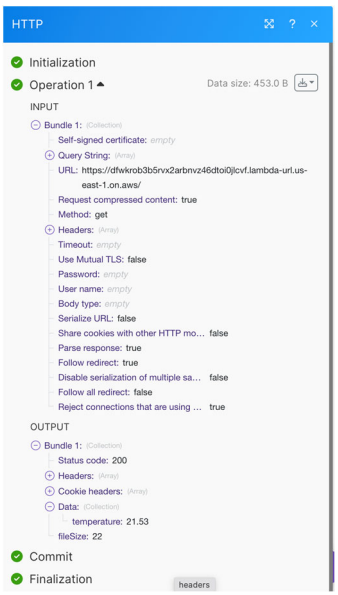

Рис. 6.6 – Результати роботи запиту до функції Temperature_data

Після того, як дані зібрані, можна відправляти їх до локальної бази даних. Для цього створимо декілька аркушів у GoogleSheets, окремо для збору інформації з кожного датчика, назвемо її Smart Home Data. Дозволимо збір даних з програми Make через API платформи, і створимо три модуля для кожного запиту, вони будуть додавати новий рядок із показником температури та маркером дати та часу.

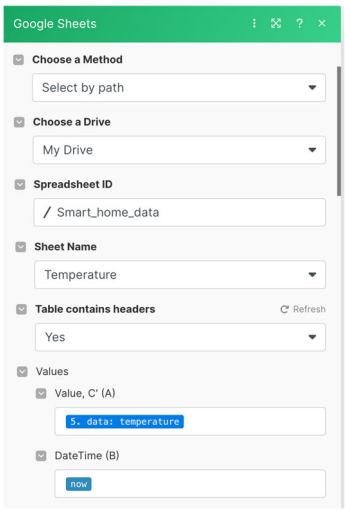

Рис. 6.7 – Налаштування модулю GoogleSheets

Тепер сценарій виглядає так:

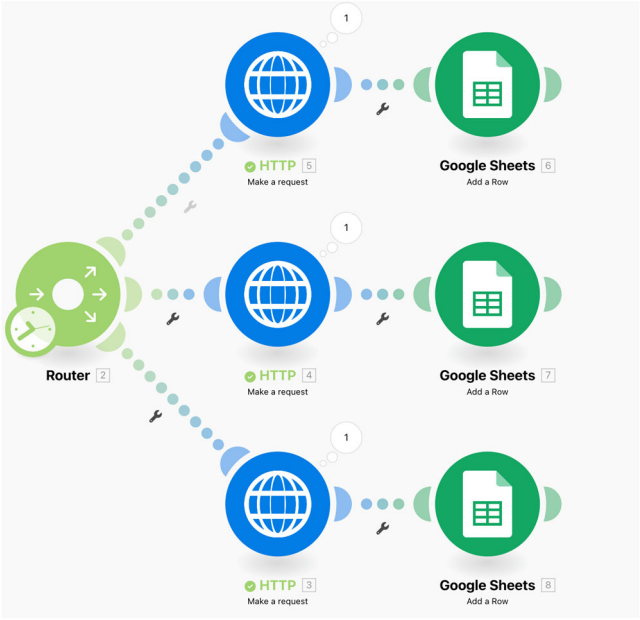

Рис. 6.8 – Новий сценарій роботи

Тепер, при кожному запуску сценарію, здійснюються запити до функцій на сервер Amazon AWS, і відповідні дані з датчиків записуються до таблиці. Запуск сценарію можна налаштувати так, щоб він виконувався раз на день, таким чином слідкувати за змінами показників в будинку протягом місяця.

Для подальшого виконання роботи було згенеровано данні з усіх датчиків для тридцяти денного періоду часу.

6.4 Огляд роботи системи

Щоб слідкувати за змінами показників в реальному часі, та у зручному форматі, було вирішено з’єднати базу даних із системою візуалізації даних Google Data Studio. Кожного разу, коли запускається сценарій, показники з датчиків записуються до відповідних таблиць та підвантажуться до дашборду[15].

Мною був створений візуальний звіт з назвою Smart Home Dashboard, з яким легко і зрозуміло аналізувати дані протягом місяця. До ного входять:

- Графіки динамічної зміни показників датчиків
- Максимальні, мінімальні та середні значення кожного показника
- Фільтр за датою
- Лінії тренду

Необов’язково запускати сценарій раз на день, можна отримувати більше інформацій, запускаючи сценарій декілька разів на добу, слідкуючи за змінами показників впродовж дня. Тоді дашборди можна будувати для кожного тижня, чи будь якого іншого проміжку часу.

Зручними функціями є лінії трендів, що допомагають відслідкувати залежності та незвичну поведінку в даних.

Розроблений візуальний звіт виглядає так:

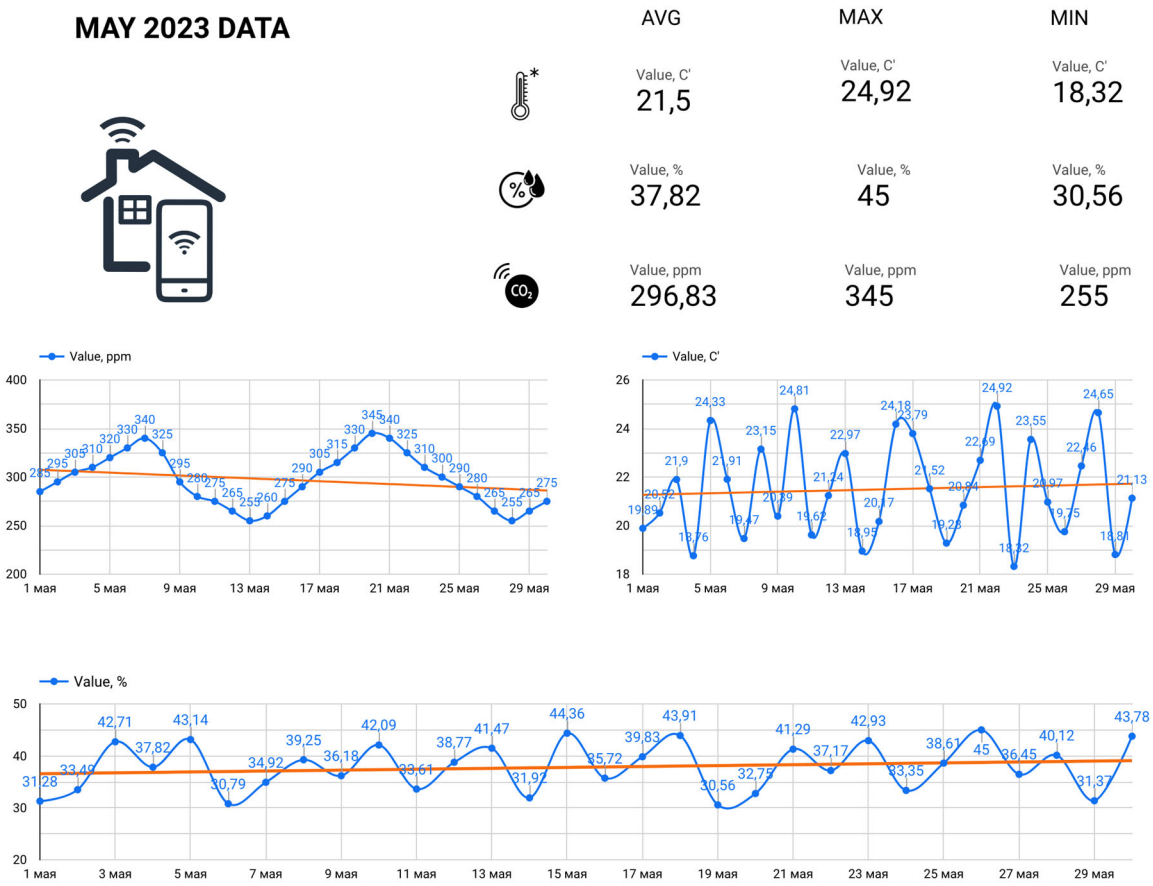

Рис. 6.9 – Дашборд

Моніторинг показників датчиків стає зручним, адаптивним і привабливим. А головне – оновлення даних відбувається у реальному часі.

6.5 Висновки до розділу

Платформа Make виявилась дуже зручним рішенням для отримання даних з девайсів, та подальшим їх керуванням. Завдяки багатьом вбудованим

модулям та легкому налаштуванню, процес розробки стає цікавим та захоплюючим.

7 ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

У даному розділі ий порівняльний функціональний та вартісний аналіз розробленого рішення, по відношенню до існуючих пропозицій у традиційних платформах як ThingSpeak. Враховується оцінка часу, необхідного для проектування системи за допомогою безкодового та традиційного програмування.

3.1 Постановка задачі техніко-економічного аналізу

Основною задачею проектування є створення програмного продукту для взаємодії з розумними пристроями.

Вимоги до продукту:

а) ПП має бути простий у користуванні;

б) ПП повинен функціонувати на різних ПК;

в) ПП повинен мати високу швидкість роботи;

г) ПП має коректно працювати з різними вибірками;

7.2 Обґрунтування функцій програмного продукту

Головна функція F_0 – розробка програмного продукту, який вирішує задачу отримання та передачу даних з розумного пристрою. Виходячи з конкретної мети, можна виділити наступні основні функції програмного продукту:

F_1 – вибір середовища для симуляції роботи розумного пристрою;

F_2 – вибір платформи для отримання та розподілу даних з пристрою;

F_3 – вибір платформи для візуалізації даних.

Кожна з основних функцій може мати декілька варіантів реалізації.

Функція F_1:

1) AWS IoT Simulator

Функція F_2:

1) Make
2) ThingSpeak[16]

Функція F_3:

1) Google Data Studio
2) ThingSpeak

Варіанти реалізації основних функцій наведені у морфологічній карті системи на рисунку 7.1. Морфологічна карта відображує всі можливі комбінації варіантів реалізації функцій, які складають повну множину варіантів ПП.

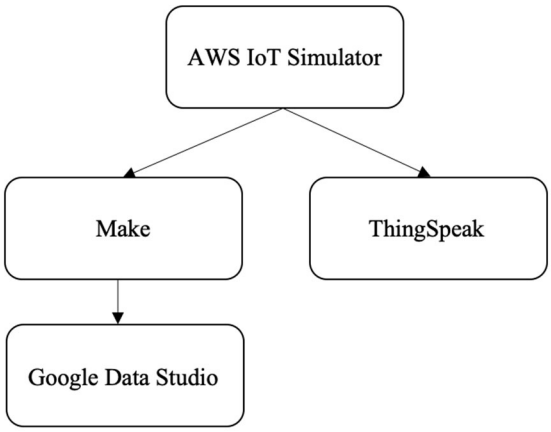

Рисунок 7.1 – Морфологічна карта варіантів реалізації функцій

На основі цієї карти побудовано позитивно-негативну матрицю варіантів основних функцій (таб. 7.1).

Таблиця 7.1 – Позитивно-негативна матриця варіантів основних функцій

Функція	Варіант реалізації	Переваги	Недоліки
F_1	А	Реалістичне середовище моделювання, яке точно імітує поведінку реальних пристроїв IoT.	Обмежений набір моделей пристроїв і протоколів
F_2	А	Пропонує широкі можливості кастомізації, дозволяючи користувачам створювати власні робочі процеси, скрипти та користувацькі інтерфейси.	Додатково необхідна інтеграція з іншими платформами для візуалізації та аналізу даних.
	Б	Надає вбудовану аналітику MATLAB для обробки даних і підтримує візуалізацію даних за допомогою настроюваних діаграм і графіків	Використання повного потенціалу ThingSpeak вимагає знайомства з MATLAB та його мовою сценаріїв[16]
F_3	А	Легко інтегрується з будь якими джерелами даних, платформами тощо.	Поглиблений статистичний аналіз, складні перетворення даних та можливості моделювання даних обмежені.

На основі аналізу позитивно-негативної матриці робимо висновок, що при розробці програмного продукту деякі варіанти реалізації функцій варто відкинути, тому, що вони не відповідають поставленим перед програмним продуктом задачам. Ці варіанти відзначені у морфологічній карті.

Таким чином, будемо розглядати такі варіанти реалізації програмного продукту:

– $F_1(А) – F_2(А) – F_3(А)$
– $F_1(А) – F_2(Б)$

На підставі даних про основні функції, що повинен реалізувати програмний продукт, вимог до нього, визначаються основні параметри виробу, що будуть використані для розрахунку коефіцієнта технічного рівня. Для того, щоб охарактеризувати програмний продукт, будемо використовувати наступні параметри:

– X1 – час, витрачений на створення симуляції девайсів та решти програмного продукту.
– X2 – швидкість запиту, обробки та аналізу даних;
– X4 – потенційний об’єм програмного коду, який необхідно створити безпосередньо розробнику.

Гірші, середні і кращі значення параметрів вибираються на основі вимог замовника й умов, що характеризують експлуатацію програмного продукту як показано у таб 7.2.

Таблиця 7.2 – Основні параметри програмного продукту

Назва Параметра	Умовні позначення	Одиниці виміру	Значення параметра		
			гірші	середні	кращі
Час, витрачений на розробку продукту	X1	год	35	20	10
Швидкість обробки та аналізу даних	X2	мс	5	3	2
Потенційний об’єм програмного коду	X3	кількість рядків коду	150	100	50

За даними таблиці 7.2 будуються графічні характеристики параметрів

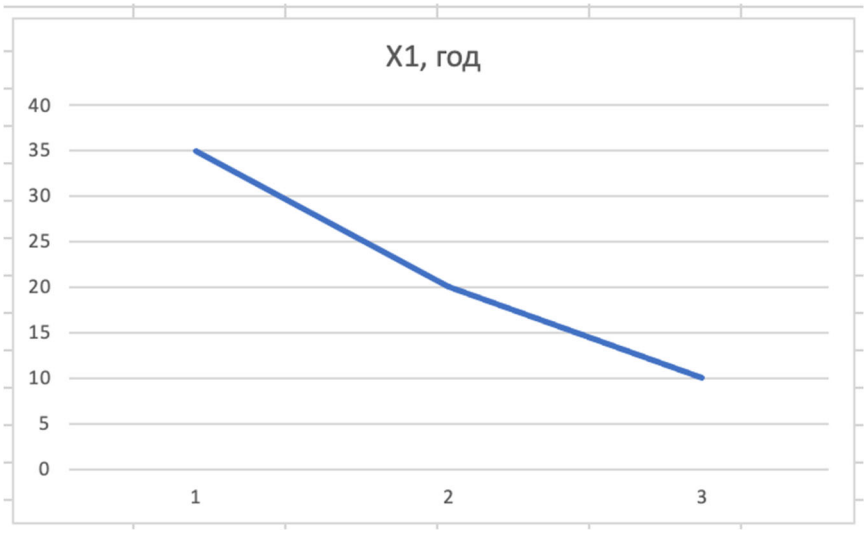

Рис. 7.2 - Час, витрачений на розробку продукту

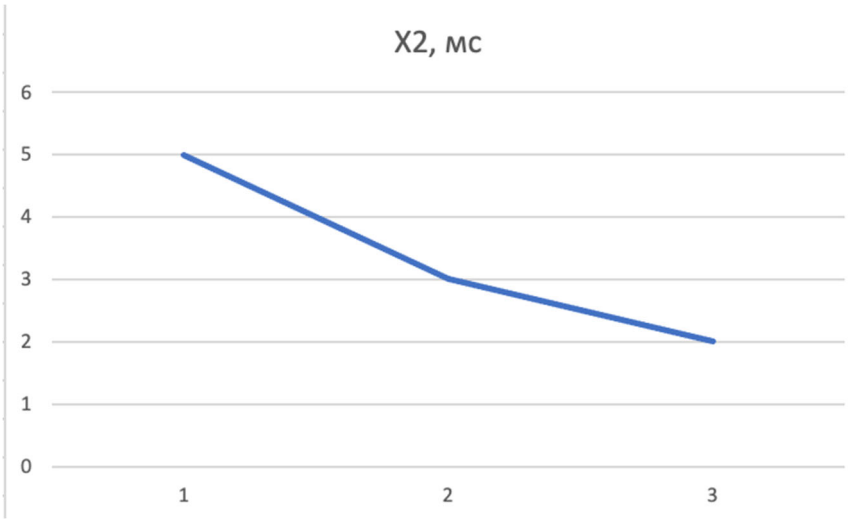

Рис. 7.3 - Швидкість обробки та аналізу даних

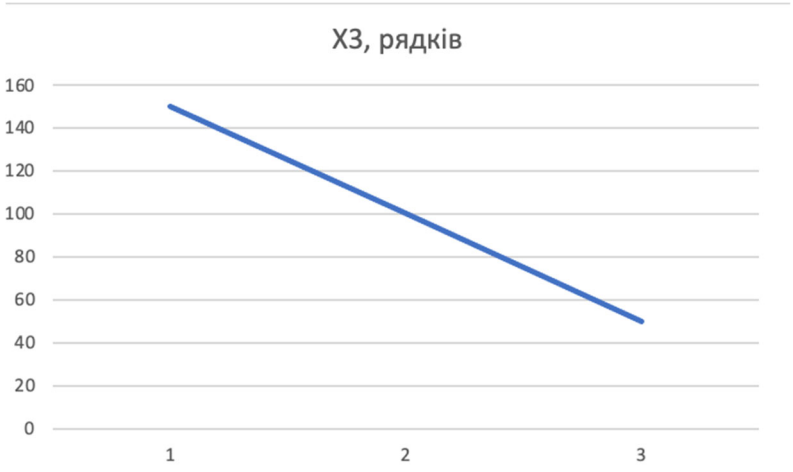

Рис. 7.4 - Потенційний об’єм програмного коду

7.3 Аналіз експертного оцінювання параметрів

Вагомість параметрів будемо оцінювати за допомогою методів попарного зрівняння. Так як параметра усього три, то ранги варіюються від 1 до 3. Результати продемонстровано у таблицях 7.3-7.4

Табл. 7.3 Результат оцінки параметрів

Параметр	Ранг параметру по оцінці експерта							Сума рангів, R_i	Відхилення Δ_i	Квадрат відхилення, $(\Delta_i)^2$
	1	2	3	4	5	6	7			
X3	3	3	2	3	2	3	2	18	4	16
X2	2	2	3	2	3	2	3	17	3	9
X1	1	1	1	1	1	1	1	7	-7	42
Разом	6	6	6	6	6	6	6	42	0	67

Табл. 7.4 Попарне зрівняння параметрів

Параметри	**Експерти**							**Кінцева оцінка**	**Числове значення**
	1	**2**	**3**	**4**	**5**	**6**	**7**		
X1 та X2	>	>	<	>	<	>	<	>	1.5
X1 та X3	>	>	>	>	>	>	>	>	1.5
X2 та X3	>	>	>	>	>	>	>	>	1.5

Розрахуємо коефіцієнт узгодженості:

$$W = \frac{12S}{N^2(n^3 - n)} = \frac{12 \cdot 211}{7^2(4^3 - 4)} = 0{,}86 > W_k = 0{,}67$$

Оскільки коефіцієнт узгодженості більше нормативного, результати можемо вважати достовірними. Використаємо результати попарного порівняння та обчислимо вагомість кожного з параметрів.

Табл. 7.5 Розрахунок вагомості параметрів

Xi	Параметри Xj,			Перший крок		Другий крок	
	X3	X2	X1	$в_i$	$K_{вi}$	$в'_i$	$K'_{вi}$
X1	1.0	1.5	1.5	4	0.445	11.5	0.46
X2	0.5	1.0	1.5	3	0.333	8	0.32
X3	0.5	0.5	1.0	2	0.222	5.5	0.22
Усього:				9	1	25	1

Табл. 7.6 Розрахунок вагомості параметрів

Основна функція	Варіант реалізації	Параметри	Абсолютне значення параметра	Бальна оцінка параметра	Коефіцієнт вагомості параметра	Коефіцієнт якості
F1	А	X1	5	4	0.46	1,84
		X3	150	5	0.22	1,1
F2	А	X2	2	4	0.32	1,28
		X1	3	5	0.46	2,3
	Б	X2	3	3	0.32	0,96
		X1	4	3	0.46	1,38
F3	А	X1	1	5	0.317	1,585

K1 = 1,84+1,1 + 1,28 + 2,3+1,56 = 8,08

K2 = 1,84+1,1+0,96+1,38 = 5,28

Так як варіант №1 має більший коефіцієнт якості, то він є кращим.

7.4 Економічний аналіз варіантів розробки ПП

Для визначення вартості розробки ПП спочатку проведемо розрахунок трудомісткості.

Всі варіанти включають в себе два окремих завдання:

1. Розробка розумних девайсів у симуляторі;
2. Розробка системи отримання та керування даних з девайсів;
3. Розробка дашборду для візуалізації даних

Завдання 1 за ступенем новизни відноситься до групи А, завдання 2 – до групи Б. За складністю алгоритми, які використовуються в завданні 1 належать до групи 1; а в завданні 2 – до групи 3.

Для реалізації завдання 1 використовується довідкова інформація, а завдання 2 використовує інформацію у вигляді даних.

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань.

Загальна трудомісткість обчислюється як

$$T_О = T_Р \cdot К_П \cdot К_{СК} \cdot К_М \cdot К_{СТ} \cdot К_{СТ.М}, \qquad (7.3.1)$$

де $T_Р$ – трудомісткість розробки ПП;

$К_П$ – поправочний коефіцієнт;

$К_{СК}$ – коефіцієнт на складність вхідної інформації;

$К_М$ – коефіцієнт рівня мови програмування;

$К_{СТ}$ – коефіцієнт використання стандартних модулів і прикладних програм;

$К_{СТ.М}$ – коефіцієнт стандартного математичного забезпечення

Для першого завдання маємо: алгоритм групи складності 1, ступінь новизни Г, вид використаної інформації БД

$T_1 = 27 \cdot 0.4 \cdot 1,07 \cdot 1,5 = 17.334$ людино-днів.

Для другого завдання (використовується алгоритм третьої групи складності, степінь новизни Б), тобто $T_Р$ =30 людино-днів, $К_П$ =0.9,$К_{СК}$ = 1,$К_{СТ}$ =0.8:

$T_2 = 27 \cdot 0.9 \cdot 0.8 = 19.44$ людино-днів.

Для третього завдання маємо: алгоритм групи складності 3, ступінь новизни Г, вид використаної інформації БД.

$T_2 = 8 \cdot 0.3 \cdot 1.07 = 3.6$ людино-днів.

Складаємо трудомісткість відповідних завдань для кожного з обраних варіантів реалізації програми, щоб отримати їх трудомісткість:

$T_I = (19.44 + 3.6+4.8) \cdot 8 = 222.7$ людино-годин.

$T_{II} = (19.44 + 3.6+6.91) \cdot 8 = 240$ людино-годин.

Найбільш високу трудомісткість має варіант II.

В розробці бере участь один програміст з окладом 21000 грн та аналітик з окладом 2000 грн.

Розрахуємо середню заробітну плату за годину:

$$С_ч = \frac{М}{T_m \cdot t} \text{грн.}, \quad (7.3.2)$$

де М – місячний оклад працівників;

T_m – кількість робочих днів тиждень;

t – кількість робочих годин в день.

$$С_ч = \frac{41000}{3 \cdot 21 \cdot 8} = 112,02 \text{ грн.} \quad (7.3.3)$$

Тоді, розрахуємо заробітну плату за формулою:

$$С_{ЗП} = С_ч \cdot Т_i \cdot К_Д , \quad (7.3.4)$$

де $С_ч$– величина погодинної оплати праці програміста;

$Т_i$ – трудомісткість відповідного завдання;

$К_Д$ – норматив, який враховує додаткову заробітну плату.

Зарплата розробників за варіантами становить:

I. $С_{ЗП}$ = 112,02 · 222.7 · 1.2 =29 984 грн.

II. $С_{ЗП}$ = 112,02 · 240 · 1.2 =32 314 грн.

Відрахування на єдиний соціальний внесок становить 22%:

I. $С_{ВІД} = С_{ЗП} \cdot 0.22$ = 29 984· 0.22 = 6596.5 грн.

II. $С_{ВІД} = С_{ЗП} \cdot 0.22$ = 32 314 · 0.22 = 7109 грн.

Тепер визначимо витрати на оплату однієї машино-години. ($С_М$)

Так як одна ЕОМ обслуговує одного програміста з окладом 21000 грн., з коефіцієнтом зайнятості 0,2 то для однієї машини отримаємо:

$$С_Г = 12 \cdot М \cdot К_З = 12 \cdot 21000 \cdot 0,2 = 50400 \text{ грн.}$$

З урахуванням додаткової заробітної плати:

$$С_{ЗП} = С_Г \cdot (1 + К_З) = 50400 \cdot (1 + 0.2) = 60480 \text{ грн.}$$

Відрахування на соціальний внесок:

$$С_{ВІД} = С_{ЗП} \cdot 0.22 = 60480 \cdot 0,22 = 13305 \text{ грн.}$$

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 50000 грн.

$$С_А = К_{ТМ} \cdot К_А \cdot Ц_{ПР} = 1.15 \cdot 0.25 \cdot 50000 = 14\ 375 \text{ грн.},$$

де $К_{ТМ}$– коефіцієнт, який враховує витрати на транспортування та монтаж приладу у користувача;

$К_А$– річна норма амортизації;

$Ц_{ПР}$– договірна ціна приладу.

Витрати на ремонт та профілактику розраховуємо як:

$$С_Р = К_{ТМ} \cdot Ц_{ПР} \cdot К_Р = 1.15 \cdot 50000 \cdot 0.05 = 2875 \text{ грн.},$$

де $К_Р$ – відсоток витрат на поточні ремонти.

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$$Т_{ЕФ} = (Д_К – Д_В – Д_С – Д_Р) \cdot t_З \cdot К_В = (365 – 104 – 12 – 16) \cdot 8 \cdot 0.9 = 1677.6 \text{ годин},$$

де $Д_К$ – календарна кількість днів у році;

$Д_В$, $Д_С$ – відповідно кількість вихідних та святкових днів;

$Д_Р$ – кількість днів планових ремонтів устаткування;

t –кількість робочих годин в день;

$К_В$– коефіцієнт використання приладу у часі протягом зміни.

Витрати на оплату електроенергії розраховуємо за формулою:

$$С_{ЕЛ} = Т_{ЕФ} \cdot N_С \cdot К_З \cdot Ц_{ЕН} = 1677.6 \cdot 0.6 \cdot 0.3 \cdot 4.79 = 1446.43 \text{ грн.},$$

де N_C – середньо-споживча потужність приладу;

$K_З$– коефіцієнтом зайнятості приладу;

$Ц_{ЕН}$ – тариф за 1 КВт-годин електроенергії.

Накладні витрати розраховуємо за формулою:

$$С_Н = Ц_{ПР} \cdot 0.67 = 50000 \cdot 0,67 = 33\ 500 \text{ грн.}$$

Тоді, річні експлуатаційні витрати будуть:

$$С_{ЕКС} = С_{ЗП} + С_{ВІД} + С_А + С_Р + С_{ЕЛ} + С_Н, \qquad (7.3.5)$$

$$С_{ЕКС} = 50\ 400 + 13\ 305 + 14\ 375 + 2875 + 1446.43 + 33\ 500 = 115\ 871 \text{ грн.}$$

Собівартість однієї машино-години ЕОМ дорівнюватиме:

$$С_{М\text{-}Г} = С_{ЕКС} / Т_{ЕФ} = 115\ 871 / 1677.6 = 69.07 \text{ грн/год.}$$

Оскільки в даному випадку всі роботи, які пов‘язані з розробкою програмного продукту ведуться на ЕОМ, витрати на оплату машинного часу, в залежності від обраного варіанта реалізації, складає:

$$С_М = С_{М-Г} \cdot Т, \qquad (7.3.6)$$

I. $С_М = 69.07 \cdot 222.7 = 15\ 382$ грн.

II. $С_М = 69.07 \cdot 240 = 16\ 577$ грн.

Накладні витрати складають 67% від заробітної плати:

$$С_Н = С_{ЗП} \cdot 0,67, \qquad (7.3.7)$$

I. $С_Н = 29\ 984 \cdot 0{,}67 = 20\ 089$ грн.

II. $С_Н = 32\ 314 \cdot 0{,}67 = 21\ 650$ грн.

Отже, вартість розробки ПП за варіантами становить:

$$С_{ПП} = С_{ЗП} + С_{ВІД} + С_М + С_Н, \quad (7.3.8)$$

I. $С_{ПП} = 29\ 984 + 13\ 305 + 15\ 382 + 20\ 089 = 78\ 761$ грн.

II. $С_{ПП} = 32\ 314 + 13\ 305 + 16\ 577 + 21\ 650 = 83\ 846$ грн.

7.5 Вибір кращого варіанту ПП техніко-економічного рівня

Розрахуємо коефіцієнт техніко-економічного рівня за формулою:

$$К_{ТЕР\,j} = К_{Кj} / С_{Фj}, \quad (7.4.1)$$

$К_{ТЕР1} = 8.08/ 78\ 761 = 10.3 \cdot 10^{-5}$,
$К_{ТЕР2} = 5.28/ 83\ 846 = 6.3 \cdot 10^{-5}$.

Як бачимо, найбільш ефективним є перший варіант реалізації програми з коефіцієнтом техніко-економічного рівня $К_{ТЕР1} = 10.3 \cdot 10^{-5}$.

Після виконання функціонально-вартісного аналізу програмного комплексу що розроблюється, можна зробити висновок, що з альтернатив, що залишились після першого відбору двох варіантів виконання програмного комплексу оптимальним є перший варіант реалізації програмного продукту. У нього виявився найкращий показник техніко-економічного рівня якості $К_{ТЕР} = 1{,}13 \cdot 10^{-5}$.

Цей варіант реалізації програмного продукту має такі параметри:

- мову програмування Python;
- Симулятор AWS IoT Simulator
- Використання платформи Make;
- Візуалізація даних у GoogleDataStudio.
- найвищу точність результатів обробки даних.
- Швидкий запит та обробка даних.

Даний варіант виконання програмного комплексу дає користувачу зручний інтерфейс, легкий у використанні та швидкість обробки. .

7.6 Висновки до розділу

Проведено повний функціонально-вартісний аналіз програмного продукту. Визначено та проведено оцінку основних функцій програмного продукту. Визначено параметри, які характеризують програмний продукт. Проведено експертне оцінювання параметрів та аналіз якості варіантів реалізації функцій.

Проведено економічний аналіз варіантів розробки – трудомісткість, витрати на заробітну плату та інші витрати.

Проведений аналіз показав що використання платформи з низьким рівнем коду Make для взаємодії з розумними речами є вигідним з економічної точки зору.

ВИСНОВКИ

В ході виконання дипломної роботи було детально досліджено та визначено поняття розумних речей в контексті інтернету речей(IoT). Було визначено вплив та актуальність використання розумних девайсів, їх вплив на життя людей та перспективи подальшого розвитку області.

В дипломній роботі було розглянуто декілька популярних платформ з малокодовою та безкодовою реалізацією, досліджено їх особливості, переваги та недоліки, можливості використання, та способи інтеграції із розумними речами. На основі вивчення публікацій за темою та використовуючи відомі методи аналізу, було розроблено систему метрик для подальшого порівняльного аналізу платформ.

За власною системою метрик та сформованою об’єктивною градацією оцінок, було порівняно шість відомих платформ для взаємодії з розумними девайсами. У систему оцінки включено основний недолік, первагу та цільову аудиторію продукту. Після проведених порівнянь обрано платформу Make для подальшої роботи.

На обраній платформі вирішено розробити кастомізований сценарій для отримання даних з розумних пристроїв та передачі їх для подальшого аналізу та візуалізації. Перед цим робота девайсів була симульована у середовищі AWS IoT Simulator. Система є надзвичайно корисною, адже показує, як можна використовувати нові та маловідомі drag-and-drop платформи з низьким рівнем коду для взаємодії зі складними інструментами. Легкий і привабливий інтерфейс продукту робить процес розробки значно приємнішим. А функціонально-економічний аналіз та порівняння розробленого у дипломній роботі рішення із добре дослідженими аналогами, такими як ThingSpeak довів його ефективність.

В якості перспективи розвитку системи передбачається створення більш складних сценаріїв та інтеграція більшої кількості розумних девайсів, переважно фізично існуючих, а не симуляцій.

ПЕРЕЛІК ПОСИЛАНЬ

1. No-code and low-code IoT platforms speed up app development [Електронний ресурс]. — Режим доступу до ресурсу: https://www.techtarget.com/iotagenda/tip/No-code-and-low-code-IoT-platforms-speed-up-app-development

2. What is IoT[Електронний ресурс]. — Режим доступу до ресурсу: https://www.cisco.com/c/en/us/solutions/internet-of-things/what-is-iot.html

3. 7 Examples of IoT in Everyday Life [Електронний ресурс]. — Режим доступу до ресурсу: https://www.cbtnuggets.com/blog/technology/networking/seven-examples-of-iot-in-everyday-life

4. The Future of IoT: What Should We Expect? [Електронний ресурс]. — Режим доступу до ресурсу: https://www.iotforall.com/what-can-we-expect-for-the-future-of-iot

5. Analysis of Low Code-No Code Development Platforms in comparison with Traditional Development Methodologies[Електронний ресурс]. — Режим доступу до ресурсу: https://www.academia.edu/67855214/Analysis_of_Low_Code_No_Code_Development_Platforms_in_comparison_with_Traditional_Development_Methodologies

6. Top 10 low-code IoT platforms in the market [Електронний ресурс]. — Режим доступу до ресурсу: https://www.coderesist.com/top-10-low-code-iot-platforms-in-the-market/

7. Losant [Електронний ресурс]. — Режим доступу до ресурсу: https://www.losant.com

8. Akenza [Електронний ресурс]. — Режим доступу до ресурсу: https://akenza.io

9. Blynk [Електронний ресурс]. — Режим доступу до ресурсу: https://blynk.io

10. Any2Info [Електронний ресурс]. — Режим доступу до ресурсу: https://www.any2info.com

11. Fogwing [Електронний ресурс]. — Режим доступу до ресурсу: https://www.fogwing.io

12. Make [Електронний ресурс]. — Режим доступу до ресурсу: https://www.make.com

13. AWS IoT Device Simulator [Електронний ресурс]. — Режим доступу до ресурсу: https://medium.com/@mhatrep/aws-iot-device-simulator-3abf919a8675

14. Tutorial: Formatting a notification by using an AWS Lambda function[Електронний ресурс]. — Режим доступу до ресурсу: https://docs.aws.amazon.com/iot/latest/developerguide/iot-lambda-rule.html

15. Google Data Studio[Електронний ресурс]. — https://datastudio.withgoogle.com/

16. ThingSpeak [Електронний ресурс]. — Режим доступу до ресурсу: https://thingspeak.com/

ДОДАТОК А

Коди Lambda-функцій девайсів

```
#

lp sru#mvrq#

lp sru#dqgrp #

gh i#dp egdbkdqgdnu#hyhqw/#frqwh{w,=#

###&#I{wdfw#wkh#mp shudwxh#ydoxh#urp #wkh#qfrp lqj#p hvvdjh#

whp shudwxh#@#-*mp s*#sd|ordg^mp s\#

###&#Fuhdwh#d#MVRQ#ihvsrqvh#z lwk#wkh#mp shudwxh#

###ihvsrqvh#@##

######vwdwxvFrgh*#533/#

#######erg|*#mvrq1gxp sv+~*mp shudwxh*#mp shudwxh¢,#

###\#

###ihwxuq#ihvsrqvh#

#

#

lp sru#mvrq#

lp sru#dqgrp #

#

gh i#dp egdbkdqgdnu#hyhqw/#frqwh{w,=#

###&#I{wdfw#wkh#mp shudwxh#ydoxh#urp #wkh#qfrp lqj#p hvvdjh#

###r{|jhq#@#-*fr5*#sd|ordg^fr5\#
###&#Fuhdwh#d#MVRQ#ihvsrqvh#z lwk#wkh#mp shudwxh#

###ihvsrqvh#@##

######vwdwxvFrgh*#533/#

#######erg|*#mvrq1gxp sv+~*F35#byho*#r{|jhq4¢,#

###\#

###ihwxuq#ihvsrqvh#

#

lp sru#mvrq#

lp sru#dqgrp #

#

gh i#dp egdbkdqgdnu#hyhqw/#frqwh{w,=#

###&#I{wdfw#wkh#mp shudwxh#ydoxh#urp #wkh#qfrp lqj#p hvvdjh#

###kxp lglw|#@#-*kxp lglw|bydoxh*#sd|ordg^kxp lglw|bydoxh\#
###&#Fuhdwh#d#MVRQ#ihvsrqvh#z lwk#wkh#mp shudwxh#
```

```
###hvsrqvh#@##
######vwdwxvFrgh*#533/#
######erg|*#mvrq1gxp sv+-*kxp lglw|*#kxp lglw|4¢,#
###¢#
###hwxuq#hvsrqvh#
```