Міністерство освіти і науки України
Рівненський державний гуманітарний університет
Кафедра інформаційних технологій та моделювання

КВАЛІФІКАЦІЙНА РОБОТА

за освітнім ступенем «бакалавр»

на тему:

«Візуалізація IoT даних засобами інструментарію Initial State»

Виконав:
студент IV-го курсу
групи КН-41
спеціальності 122 «Комп’ютерні науки»
Ярошик Вадим Юрійович
Керівник:
к.т.н., доцент Шинкарчук Н.В.

Рівне – 2025

АНОТАЦІЯ ДО КВАЛІФІКАЦІЙНОЇ РОБОТИ

Ярошик В.Ю. Візуалізація IoT даних засобами інструментарію Initial State. Кваліфікаційна робота на здобуття ступеня «бакалавр» за спеціальністю 122 «Комп'ютерні науки» – Рівненський державний гуманітарний університет. Рівне, 2025. 54 с.

Інтернет речей (Internet of things, IoT) – це концепція мережі, яка складається із взаємозв'язаних фізичних пристроїв, які мають вбудовані датчики, а також програмне забезпечення, яке дозволяє здійснювати передачу даних між навколишнім середовищем і комп'ютерними системами, за допомогою використання стандартних протоколів зв'язку. Основною концепцією Інтернету речей є можливість підключення всіляких об'єктів (речей), які людина може використовувати в повсякденному житті, наприклад: холодильник, кондиціонер.

Візуалізація – це процес побудови графічного образу даних, що допомагає у процесі загального аналізу даних вбачати аномалії, структури. Це також перетворення абстрактної або невидимої інформації у візуальні форми, що дозволяють краще зрозуміти і аналізувати цю інформацію.

Initial State – це IoT-платформа потокового передавання та візуалізації даних. Вона полегшує кожному, від початківців до підприємств, надсилати дані з підключених до Інтернету пристроїв, датчиків та програм у «хмару», де ці дані можна отримати в будь-який час і миттєво перетворити на інтерактивні панелі інструментів у реальному часі: діаграми, статистику, сповіщення тощо. Initial State був розроблений для забезпечення послідовного, зрозумілого і безпечного доступу користувачів у великих масштабах через протокол HTTPS/TLS, забезпечує простий спосіб для користувачів надсилати та отримувати дані з високонадійної бази (сховища) даних в режимі реального часу.

Ключові слова: візуалізація, Інтернет речей, дані, Initial State, Raspberry Pi, датчики.

ЗМІСТ

ВСТУП **4**

РОЗДІЛ 1. ТЕХНОЛОГІЯ ІНТЕРНЕТ РЕЧЕЙ. ОСНОВИ ВІЗУАЛІЗАЦІЇ ДАНИХ **6**

1.1.Поняття технології Інтернету речей **6**

1.2. Виникнення та становлення технологій Інтернет речей **7**

1.3 Архітектура Інтернету речей **10**

1.4. Основи візуалізації даних **13**

РОЗДІЛ 2 ОГЛЯД ІНСТРУМЕНТАЛЬНИХ ТА ФУНКЦІОНАЛЬНИХ ЗАСОБІВ ПЛАТФОРМИ INITIAL STATE ДЛЯ ВІЗУАЛІЗАЦІЇ ІОТ ДАНИХ **17**

2.1. Огляд платформи Initial State **17**

2.2. Інструментальні можливості Initial State **18**

2.3. Функціональні можливості Initial State **19**

2.4. Налаштування контейнера збору даних **23**

РОЗДІЛ 3. АПАРАТНО-ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ОДНОПЛАТНОГО КОМП'ЮТЕРА RASPBERRY PI 3 MODEL B. **30**

3.1. Поняття одноплатного комп'ютера **30**

3.2. Апаратне забезпечення Raspberry Pi 3 Model B **31**

3.3. Програмне забезпечення Raspberry Pi 3 Model B **33**

РОЗДІЛ 4. РОЗРОБКА СИСТЕМИ ЗБОРУ ТА ПЕРЕДАЧІ ІОТ ДАНИХ ДЛЯ ВІЗУАЛІЗАЦІЇ В INITIAL STATE **36**

4.1. Постановка задачі **36**

4.2. Вибір апаратних компонентів **37**

4.3. Налаштування і підключення датчиків **40**

4.4. Візуалізація даних у Initial State **47**

ВИСНОВКИ **51**

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ **52**

ДОДАТОК А **54**

ВСТУП

Актуальність роботи. У сучасному світі Інтернет речей (Internet of Things, IoT) стрімко перетворюється з концепції майбутнього на невід'ємну частину реального життя. Від носимих пристроїв, до розумних будинків до промислових сенсорних систем і логістичних платформ – IoT забезпечує безперервний потік даних між фізичними об'єктами та цифровими середовищами.

Візуалізація IoT-даних є важливою складовою технології, адже вона дозволяє не лише спостерігати за показниками в режимі реального часу, а й оперативно реагувати на відхилення, оптимізувати процеси та приймати обґрунтовані рішення. Саме тому платформи, що поєднують збір, зберігання та представлення інформації, стають важливими інструментами в екосистемі Інтернету речей.

Одним із таких рішень є Initial State – хмарна платформа, яка забезпечує інтуїтивну, гнучку та наочну візуалізацію даних, що надходять з пристроїв. Її простота інтеграції, підтримка API, інтерактивні дошки та можливість роботи в реальному часі роблять її актуальним інструментом для розробників прикладних систем Інтернету речей.

Мета роботи: дослідити інструментальні та функціональні можливості платформи Інтернету речей Initial State в процесі візуалізації IoT даних.

Об'єктом дослідження є процес збору, передачі та візуалізації даних в системах Інтернету речей.

Інструментом дослідження є хмарна платформа Initial State разом із засобами програмування Python і апаратним забезпеченням, що включає Raspberry Pi 3 Model B та підключені до нього датчики (температури, вологості, звуку, вібрації, RFID).

Предметом дослідження є інструментальні та функціональні можливості платформи Initial State в контексті реалізації візуалізації IoT даних, які зібрано з датчика температури і вологості, датчика виявлення звуку і вібрації, RFID-модуля.

Завдання дослідження:

1 Провести аналіз наукових і технічних джерел щодо сучасних технологій Інтернету речей та візуалізації даних.
2 Дослідити інструментальні та функціональні можливості Initial State для обробки й візуалізації даних.
3 Розробити апаратно-програмну систему збору даних на основі Raspberry Pi 3 Model B з підключеними датчиками.
4 Реалізувати програмне рішення для збору, обробки та передачі даних на платформу Інтернету речей Initial State.
5 Створити інтерактивну дошку у Initial State для візуалізації даних Інтернету речей в режимі реальному часі.

Апробація результатів кваліфікаційної роботи. Результати виконання кваліфікаційної роботи, окремі її аспекти та одержані узагальнення і висновки були оприлюднені на XVIII Всеукраїнській науково-практичній конференції здобувачів вищої освіти та молодих учених «Наука, освіта, суспільство очима молодих» (м. Рівне, 2025), звітній науковій конференції викладачів, співробітників і здобувачів вищої освіти Рівненського державного гуманітарного університету за 2024 рік (м. Рівне, 2025).

Публікації. Результати, які були отримані в ході кваліфікаційного дослідження частково опубліковані у вигляді тез XVIII Всеукраїнської науково-практичної конференції здобувачів вищої освіти та молодих учених «Наука, освіта, суспільство очима молодих» у Рівненському державному гуманітарному університеті (Додаток А).

Структура роботи. Робота складається зі вступу, чотирьох розділів, висновків, списку використаних джерел. У першому розділі описано основні концепції технології Інтернет речей. У другому – огляд інструментальних засобів платформи візуалізації даних Інтернету речей – Initial State. У третьому розділі описано апаратне та програмне забезпечення, яке буде використане в процесі реалізації проєкту, а у четвертому представлено демонстраційний проєкт візуалізації даних Інтернету речей, зокрема: використані датчики та програмний код. Список літератури містить вісімнадцять джерел.

РОЗДІЛ 1
ТЕХНОЛОГІЯ ІНТЕРНЕТ РЕЧЕЙ. ОСНОВИ ВІЗУАЛІЗАЦІЇ ДАНИХ

1.1. Поняття технології Інтернету речей

Інтернет речей (Internet of Things, скорочено IoT) – це глобальна мережа підключених до Інтернету речей пристроїв, оснащених сенсорами, датчиками, засобами передавання сигналів. Ці цифрові пристрої можуть сприймати датчиками різноманітні сигнали з навколишнього світу, вступати у взаємодію з іншими пристроями, обмінюватися даними з метою віддаленого моніторингу за станом об’єктів, аналізу зібраних даних і прийняття на їх основі рішень. Прикладом можуть бути гаражні двері, кавоварки, телевізори, мобільні телефони, відеокамери, датчики світла та температури тощо [1].

Основні компоненти Інтернету речей:

1. Фізичні пристрої та датчики. Це можуть бути термометри, камери, сенсори руху, датчики вологості тощо. Вони збирають дані з навколишнього середовища.

2. Підключення до мережі. Пристрої з’єднуються через Wi-Fi, Bluetooth, 5G або інші технології, щоб передавати дані.

3. Хмарні платформи. Дані, зібрані пристроями, надсилаються в хмару, де їх обробляють і аналізують.

4. Програмне забезпечення та інтерфейси. Програми на смартфонах або комп’ютерах дозволяють користувачам керувати пристроями або отримувати сповіщення. Наприклад, додаток може показати, скільки електроенергії споживає будинок.

Термін «розумний» у контексті Інтернету речей означає здатність пристроїв не просто виконувати команди, а й самостійно приймати рішення на основі даних. Наприклад, розумна лампочка не просто вмикається чи вимикається за вашою командою – вона може автоматично регулювати яскравість залежно від часу доби або вимикатися, коли ви залишаєте кімнату, завдяки датчику руху. Ця

«розумність» досягається завдяки поєднанню датчиків, алгоритмів і штучного інтелекту, який дозволяє системі вчитися та адаптуватися.

Інтернет речей являє собою не просто сукупність окремих пристроїв, а складну та взаємопов'язану екосистему (рис 1.1.). У цій системі різноманітні IoT-пристрої, такі як розумні годинники, камери безпеки та побутова техніка, функціонують у взаємодії один з одним. Вони обмінюються інформацією через центральну платформу, яка може бути представлена такими системами, як AWS IoT або Google Cloud. Ця інтеграція дозволяє створювати єдину мережу, де пристрої можуть ефективно співпрацювати, забезпечуючи користувачам зручність, автоматизацію та покращений контроль над їхнім оточенням.

Рисунок 1.1. Екосистема Інтернету речей

1.2. Виникнення та становлення технологій Інтернет речей

Перші ідеї, які лягли в основу Інтернету речей, почали формуватися в 1980-х роках. У цей час з'явилися технології, що дозволяли фізичним об'єктам передавати дані комп'ютерам. Одним із перших прикладів стала система радіочастотної ідентифікації (RFID), яка використовувалася для відстеження об'єктів. Наприклад, у 1980-х в магазинах почали застосовувати RFID-мітки для маркування товарів, щоб автоматизувати облік запасів. Це був перший крок до

того, що ми зараз називаємо IoT – об’єкти отримали можливість передавати дані про себе через радіосигнали.

У 1982 році було розроблено ранню концепцію мережевого інтелектуального пристрою як інтернет-інтерфейс для датчиків, встановлених у торговому автоматі Coca-Cola кафедри комп’ютерних наук Університету Карнегі-Меллона, наданому волонтерами-аспірантами. Він забезпечував температурну модель та стан запасів, натхненний комп’ютерним торговим автоматом в Лабораторії штучного інтелекту Стенфорда. Спочатку доступний лише в кампусі Університету Карнегі-Меллона, він став першим пристроєм, підключеним до ARPANET [2].

Термін «Інтернет речей» офіційно з’явився в 1999 році завдяки Кевіну Ештону, британському технологу, описуючи систему, де фізичні об’єкти можуть бути підключені до Інтернету через RFID (радіочастотну ідентифікацію). Тоді це здавалося футуристичною ідеєю, але з роками технології стали доступнішими, а Інтернет – швидшим і надійнішим.

Це було в 1999 році. Тоді Кевін Ештон працював в Procter & Gamble (P&G)**,** йому було 28 років, і він намагався вирішити одну проблему: найбільш популярні товари їх марки складно було купити в магазинах. Він з’ясував, що швидше розкуповували ті товари, які найбільше рекламувалися. Це була проблема інформації. Він вирішив цю проблему, встановивши на товари P & G датчики, підключені до мережі (щоб знати, коли їх уже немає на складі). Його ідея здавалася божевільною в 90-і роки. Люди підключалися до інтернету через модеми dial up, зв’язок був дуже повільним і займав телефонну лінію. Ештону довелося переконати виконавчих директорів P&G, які були набагато старші за нього, встановити датчики всюди. Він зрозумів, що, використовуючи слово «інтернет» в доводах, шансів привернути увагу до своєї ідеї у нього буде більше. Адже в 1998 всі менеджери розуміли, яку важливу роль відіграє інтернет. Слово «речі» прийшло завдяки ідеї про вбудовані в стіл комп’ютери або чогось подібного, так як гаджети з часом ставали все менше і доступніше. Спочатку ідея була розпливчатою і неясною, але досить привабливою, щоб почати дослідження

в Массачусетському технологічному інституті (MIT). Поки Ештон працював над своєю ідеєю, він провів тисячі презентацій по всьому світу [3].

На початку 2000-х років Інтернет речей почав виходити за межі промислових застосувань і з'являтися в побуті. Зростання швидкості Інтернету, поява Wi-Fi і розвиток мобільних технологій зробили можливим створення перших «розумних» пристроїв. Наприклад, у 2000 році компанія LG представила перший у світі Інтернет-холодильник, який міг відстежувати запаси продуктів і надсилати повідомлення власнику. Хоча цей пристрій був дорогим і не набув масової популярності, він став важливим кроком до концепції розумного дому.

2010-ті роки стали справжнім проривом для Інтернету речей. Завдяки кільком факторам – доступності смартфонів, розвитку хмарних технологій і появі нових стандартів зв'язку, таких як Bluetooth Low Energy (BLE) – Інтернет речей став частиною повсякденного життя. У цей період з'явилися перші масові продукти, які ми асоціюємо з IoT:

З початком 2020-х років Інтернет речей увійшов у нову фазу завдяки кільком ключовим технологіям:

1. *5G.* Нова мережа зв'язку, запущена комерційно в 2019–2020 роках, забезпечила швидший і надійніший обмін даними, що ідеально підходить для IoT.
2. *Штучний інтелект (ШІ).* Поєднання IoT із ШІ дозволило пристроям не лише збирати дані, а й аналізувати їх для прийняття розумних рішень. Наприклад, розумні камери безпеки можуть розпізнавати обличчя чи підозрілу поведінку.
3. *Edge Computing.* Ця технологія дозволяє обробляти дані безпосередньо на пристроях, а не в хмарі, що зменшує затримки та підвищує ефективність. Наприклад, розумні датчики на фермах можуть аналізувати вологість ґрунту без підключення до сервера.

Тенденції розвитку Інтернету речей в період з 2015 по 2025 рік подано на рисунку 1.2.

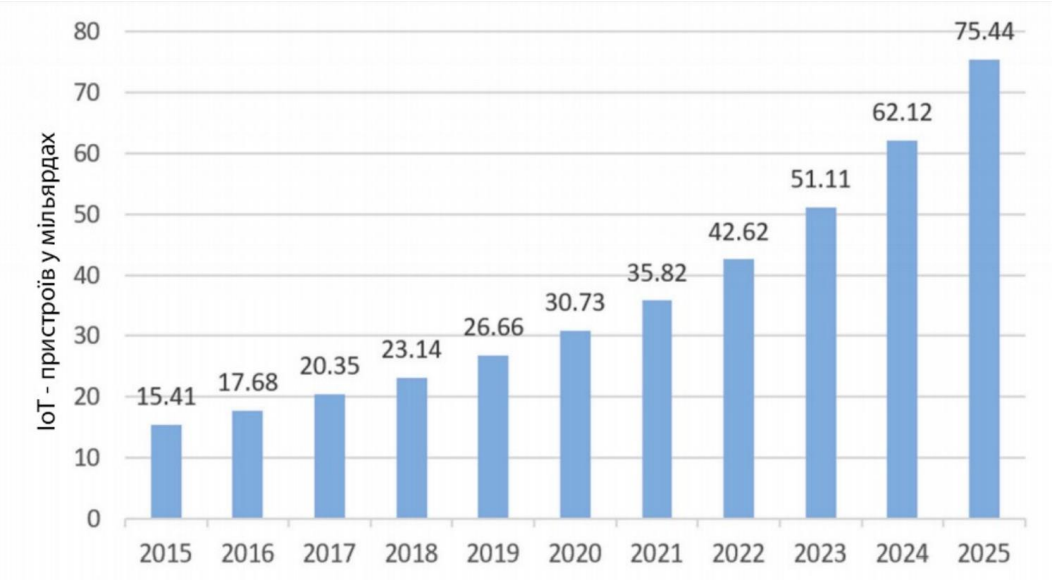

Рисунок 1.2. Тенденції розвитку Інтернету речей

1.3. Архітектура Інтернету речей

Архітектура Інтернету речей – це набір рівнів, які описують, як пристрої, мережі, дані та програми працюють разом, щоб забезпечити функціональність системи. Інтернет речей складається з кількох шарів, які забезпечують збір, передачу, обробку та використання даних. Зазвичай архітектуру Інтернету речей поділяють на три основні шари: шар сприйняття (або пристроїв), мережевий шар і прикладний шар. Деякі моделі додають додаткові шари, такі як обробка даних або безпека. Існує багато підходів до побудови мережі Інтернет речей найбільш базовий підхід – це тришарова модель яка включає:

1. Рівень сенсорів і сенсорних мереж – найнижчий рівень архітектури IoT складається з «розумних» (smart) об’єктів, інтегрованих з сенсорами (датчиками). Сенсори реалізують з’єднання фізичного і віртуального (цифрового) світів, забезпечуючи збір та обробку інформації в реальному масштабі часу [4].

2. Мережевий рівень – зібрані сенсорами дані потрібно передати, тут вступають у роль мережі. Існуючі мережі, що підтримують різні протоколи, можуть використовуватися для міжмашинного зв’язку та додатків. Для надання різноманітних послуг IoT необхідна взаємодія мереж з різними технологіями та протоколами, що забезпечують якісну передачу даних з мінімальною затримкою,

достатньою пропускною здатністю та високим рівнем безпеки. Цей рівень забезпечує маршрут між пристроями і центральними системами, де відбувається обробка.

3. Прикладний рівень – це вже «мозок» системи, саме тут дані аналізуються, інтерпретуються й використовуються. Тут знаходяться додатки, аналітика, панелі керування – усе з чим взаємодіє користувач, зачасту управління даними і додатки з якими взаємодіє користувач виділяють в окремі рівні (рис 1.3.).

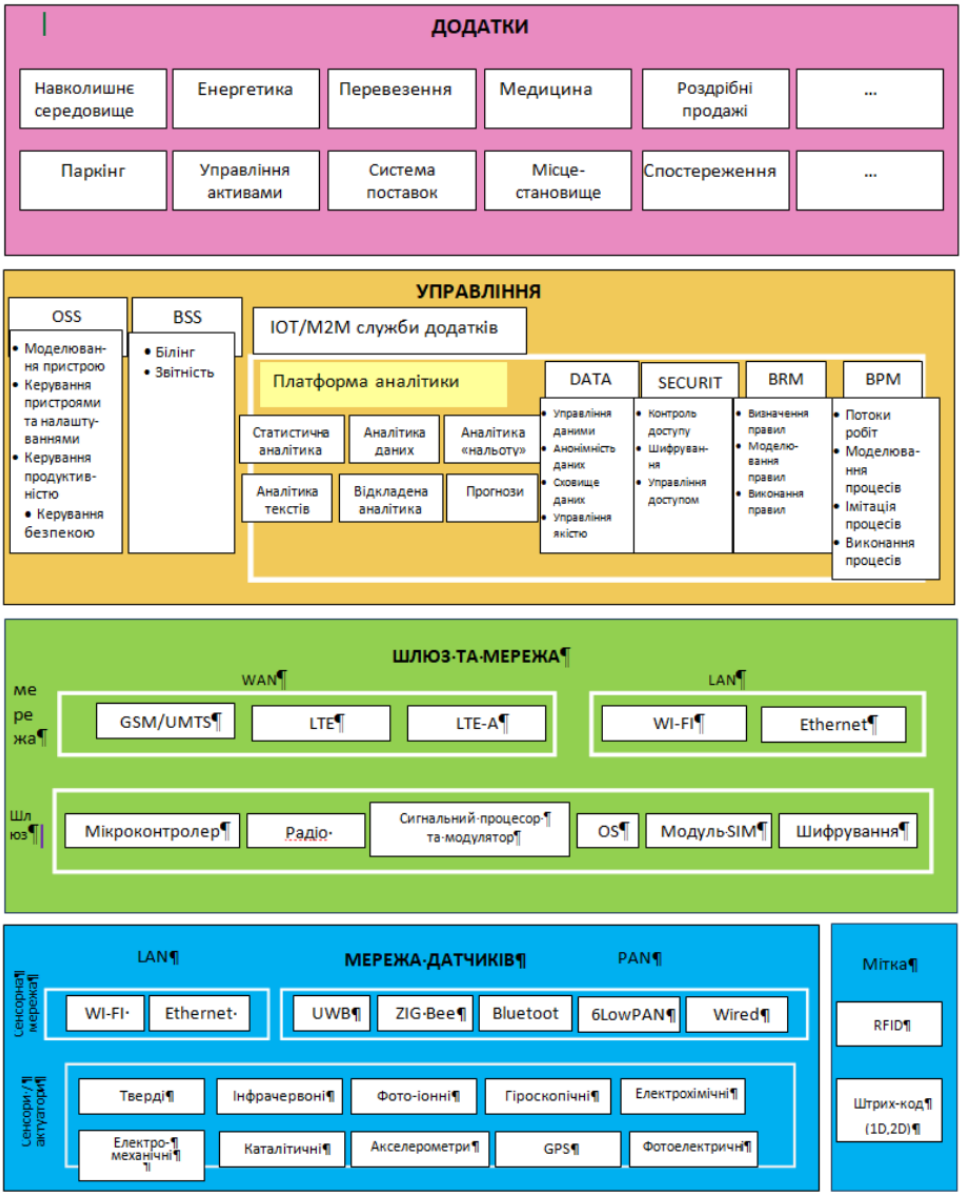

Рисунок 1.3. Архітектура IoT

На рис. 1.4 показана архітектура у вигляді сервісів. На ньому область Edge представлений у вигляді датчиків Device Hub/Gateway (збір та маршрутизація даних) та Device Management (керування пристроями). Останні частково

виконуються як хмарні обчислення так і на граничних пристроях. Усі функції збереження та первинної обробки подій (даних) зведені до Data Management. Усі інші функції обробки, в тому числі аналітичні показані як додатки PaaS, що взаємодіють з сервісами керування даних через API [1].

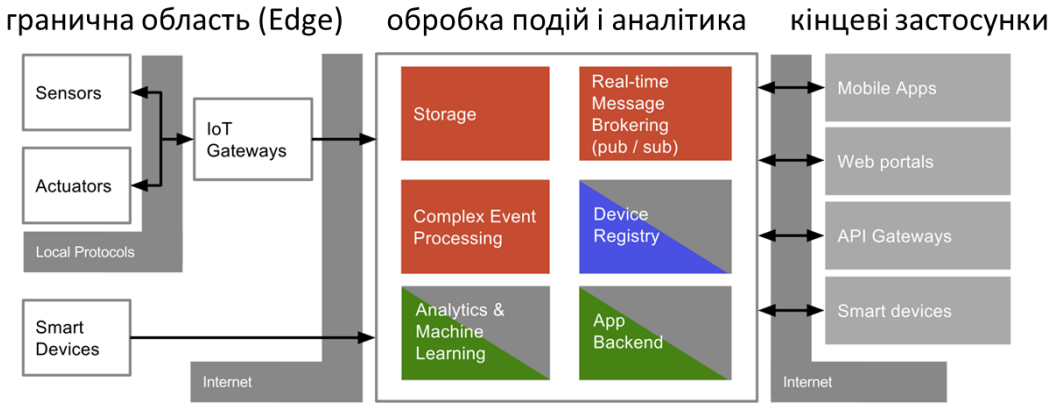

Рисунок 1.4. Архітектура Інтернету речей у вигляді сервісу

При побудові архітектури кожен пристрій, датчик чи програма відіграє свою роль, а їхня спільна робота базується на кількох ключових принципах. Ці принципи забезпечують, що система не просто збирає дані, а перетворює їх на корисні дії, роблячи наше життя зручнішим, ефективнішим і безпечнішим:

1. *Перший і основний принцип IoT – це підключеність*. Усі пристрої в системі IoT мають бути пов’язані між собою через мережу, щоб обмінюватися даними. Це може бути Wi-Fi, Bluetooth, 5G, Zigbee, LoRaWAN або навіть супутниковий зв’язок для віддалених регіонів. Без підключення IoT просто не працює (рис. 1.5.).

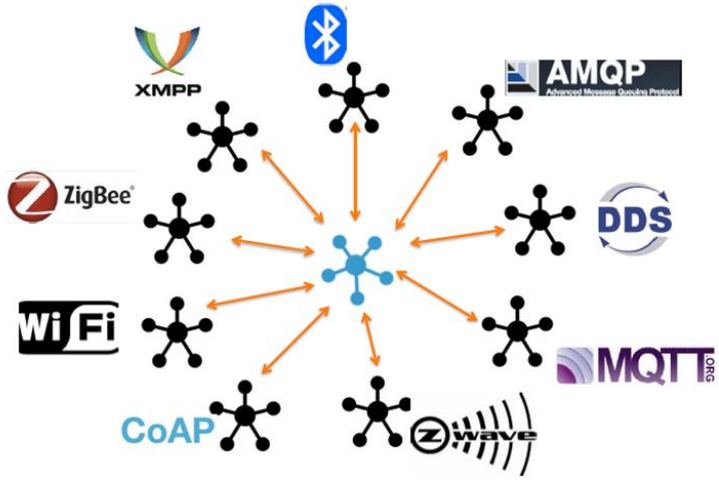

Рисунок 1.5. Найпопулярніші протоколи зв’язку

2. *Збір і обробка даних*. Інтернет речей починається з даних, без точного збору та обробки система не зможе виконувати свої функції. Кожен пристрій, оснащений датчиками, збирає інформацію про навколишнє середовище: температуру, вологість, рух, звук, світло чи навіть хімічний склад повітря. Ці дані передаються до хмарних серверів або обробляються локально (Edge Computing), щоб отримати корисну інформацію.

3. *Автоматизація*. Інтернет речей прагне мінімізувати втручання людини, дозволяючи пристроям самостійно виконувати завдання. Це принцип автоматизації, який робить IoT «розумним». Пристрої не просто збирають дані, а й реагують на них, виконуючи запрограмовані дії, або навіть адаптуючись до нових умов завдяки штучному інтелекту.

4. *Безпека*. Оскільки IoT-пристрої постійно передають і отримують дані, безпека є критично важливим принципом. Дані, які збираються, мають бути захищеними від хакерів, а пристрої – від несанкціонованого доступу.

5. *Масштабованість*. Системи Інтернету речей повинні працювати з великою кількістю пристроїв, від кількох датчиків у будинку до мільйонів у розумному місті. Масштабність означає, що система може зростати без втрати продуктивності.

6. *Адаптивність*. система має бути гнучкою, щоб адаптуватися до змін у середовищі чи потребах користувача. Це може включати навчання на основі даних за допомогою штучного інтелекту або можливість налаштування користувачем.

1.4. Основи візуалізації даних

Інтернет речей генерує величезну кількість даних, але самі по собі ці дані – лише набір чисел, які мало що говорять звичайній людині. Щоб вони стали корисними, потрібна візуалізація даних – процес перетворення сирої інформації в графіки, діаграми, карти чи інтерактивні дошки, які легко зрозуміти.

Аналіз та візуалізація даних є ключовими елементами в галузі цифрової трансформації як на рівні державного управління, так і підприємства. Ці навички дозволяють урядам збирати, аналізувати та представляти дані у зручній та зрозумілій формі. Це може бути корисним для прийняття рішень, розробки політики та вирішення важливих суспільних проблем [5].

Візуалізація даних – це метод, що полягає в конвертації сухих фактів та цифрових даних у зрозумілі, візуально привабливі зображення. Вона використовує різні графічні форми, такі як діаграми, графіки, та інформаційні панелі, для ілюстрації та аналізу даних, що допомагає сприймати та розуміти інформацію швидше і ефективніше. Вона передає чітку інформацію, використовуючи доступні дані. Зображуючи дані у зрозумілому форматі, візуалізація даних допомагає виявити закономірності та тенденції, які не завжди очевидні при перегляді необроблених даних [6].

Щоб візуалізація була ефективною, вона має відповідати кільком ключовим принципам. Ці принципи допомагають зробити дані зрозумілими, корисними та привабливими.

1. *Простота* – візуалізація має бути легкою для сприйняття. Занадто складні графіки чи перевантажені інформацією дашборди можуть заплутати користувача.

2. *Релевантність* – візуалізація має показувати лише ті дані, які потрібні для конкретного завдання.

3. *Точність* – дані, представлені у візуалізації, мають бути достовірними. Неправильне масштабуваня графіка чи некоректні дані можуть призвести до хибних висновків.

4. *Інтерактивність* – сучасні IoT-системи часто пропонують інтерактивні візуалізації, які дозволяють користувачам «інтерактив» з даними: змінювати період часу, вибирати конкретні датчики чи порівнювати показники. Це робить аналіз даних гнучким і зручним.

5. *Дизайн* – функціональний дизайн підвищує сприйняття інформації. Використання зрозумілих кольорів, наприклад, червоний для проблем, зелений

для норми, чітких шрифтів і логічного розташування елементів робить візуалізацію приємною для ока.

Також важливо обрати правильний метод візуалізації це дозволяє чітко передати ідею, швидко виявити закономірності, а також донести інформацію до інших у зручній та зрозумілій формі (рис. 1.6.)

Рисунок 1.6. Основні типи візуалізації даних

1. *Стовпчасті* – використовуються для порівняння величин між категоріями, наприклад продажі в різних регіонах.

2. *Гістограми* – різновид, що показує розподіл числових даних групуючи їх у діапазони, наприклад розподіл температурних значень за тиждень.

3. *Лінійні графіки* – цей метод часто застосовується для відстеження змін показників у часі. Добре підходить для відображення тенденцій і динаміки. Приклад: графік зміни рівня вологості в системі «розумного будинку».

4. *Кругові діаграми* – використовуються для демонстрації часток або співвідношень в межах одного цілого. Приклад: частка типів пристроїв підключених до мережі.

5. *Точкові діаграми* – цей метод дозволяє виявляти зв’язки між двома змінними. Добре підходить для дослідження кореляцій.

6. *Теплові карти* – використовуються для представлення значень у таблиці або на карті, де колір відображає інтенсивність або кількість.

7. *Картографічна візуалізація* – відображує координати з географічною прив’язкою на карті.

8. *Дашборди* – інтерактивні панелі, що об’єднують кілька типів візуалізації, найефективніший метод донести складну інформацію. Наприклад дашборд розумного будинку показує температуру, освітлення та споживання енергії.

Створення візуалізації даних для IoT – це структурований процес, який можна розбити на кілька етапів (рис 1.7).

1. *Визначення мети та аудиторії* – перш ніж створювати візуалізацію, потрібно зрозуміти, для чого вона потрібна і хто її використовуватиме, мета визначає, які дані відображати та в якому вигляді, а аудиторія – рівень складності.

2. *Збір і підготовка даних* – IoT-пристрої генерують величезну кількість даних, але не всі вони підходять для візуалізації. На цьому етапі потрібно зібрати дані та очистити їх.

3. *Вибір типу візуалізації.* Тип візуалізації залежить від мети та даних.

4. *Вибір інструменту для візуалізації.* Для створення візуалізацій у IoT використовуються різні інструменти, залежно від складності та потреб.

5. *Дизайн і створення візуалізації*. На цьому етапі створюється сама візуалізація, враховуючи принципи дизайну.

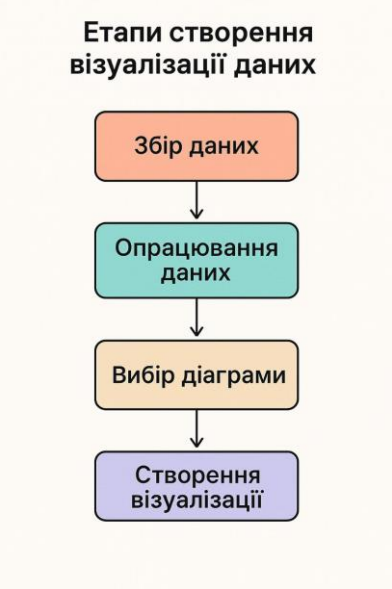

Рисунок 1.7. Етапи створення візуалізації даних

Створення візуалізації для IoT має свої труднощі, основні проблеми в процесі візуалізації даних полягають в великому об’єму генерованих даних, тому потрібно обирати найважливіші, також важлива візуалізація в реальному часі. Сумісність і доступність мають бути в приорітеті щоб легко інтегрувати будь які дані в малий проміжок часу. Також питання безпеки, конфіденційні дані потрібно захищати.

РОЗДІЛ 2
ОГЛЯД ІНСТРУМЕНТАЛЬНИХ ТА ФУНКЦІОНАЛЬНИХ ЗАСОБІВ ПЛАТФОРМИ INITIAL STATE ДЛЯ ВІЗУАЛІЗАЦІЇ IOT ДАНИХ

2.1. Огляд платформи Initial State

Initial State – це хмарна платформа, спеціально розроблена для потокової передачі, зберігання, аналізу та візуалізації даних із підключених до Інтернету пристроїв, сенсорів і додатків зокрема пристроїв Інтернету речей (IoT). Вона дозволяє користувачам створювати інтерактивні інформаційні панелі та аналітичні панелі, які відображають дані в режимі реального часу, забезпечуючи ефективний моніторинг та аналіз інформації [7].

Заснована в 2012 році в Нешвіллі, штат Теннессі, і нині є частиною компанії Tektronix, платформа орієнтована на користувачів від початківців до великих підприємств, які працюють із проєктами Інтернету речей. Її основна мета – спростити процес збору даних і перетворення їх на інтерактивні візуалізації, такі як дашборди, графіки та сповіщення, для полегшення моніторингу та прийняття рішень (рис 2.1).

Рисунок 2.1. Приклади візуалізації за допомогою Initial State

Платформа дозволяє підключення багатьох наборів інструментів для створення інтеграцій з різними платформами, наприклад є доступними: REST

API, Python Data Streamer and Reader, NodeJS, Weatherstack, Google Sheets, LabVIEW, Keithley Data Logger, CoinCap, Electric Imp, PubNub, Arduino, Visual Basic, Node-RED.

2.2. Інструментальні можливості Initial State

Інструментальні можливості стосуються технічної інфраструктури, API, інтеграцій і масштабованості платформи, які дозволяють їй працювати з даними IoT-пристроїв. Initial State використовує хмарну архітектуру з високопродуктивним бекендом, оптимізованим для обробки великих обсягів даних у реальному часі. Дані передаються через REST API (за протоколом HTTPS/TLS), що забезпечує безпечне та надійне з'єднання. Платформа підтримує потокову передачу даних із пристроїв, таких як Raspberry Pi, Arduino, ESP32, або програмних додатків.

Initial State надає простий і гнучкий REST API, який дозволяє відправляти та отримувати дані з будь-якого пристрою чи програми, що підтримує HTTP-запити. API підтримує JSON-формати для структурованих даних. Інтеграція з хмарними сервісами, такими як AWS IoT або Google Cloud, через API дозволяє використовувати Initial State як інструмент візуалізації та моніторингу даних в хмарі.

Також сумісність із мовами програмування: Python, Node.js, C++, що полегшує інтеграцію для розробників. Наприклад Використання ISStreamer SDK для Python дозволяє легко інтегрувати Initial State з IoT-сенсорами (рис 2.2).

```
1  from ISStreamer.Streamer import Streamer
2  streamer = Streamer(bucket_name="CarTelemetry", bucket_key="car123", access_key="xyz")
3  streamer.log("Speed", 65)
4  streamer.log("Fuel Level", 43.2)
5
```

Рисунок 2.2. Приклад передачі даних через потік ISStreamer

Initial State забезпечує хмарне зберігання даних із гнучкими опціями доступу (публічний доступ за посиланням, приватний доступ, або за електронною поштою). Дані організовано у вигляді «потоків» (streams), кожен із яких

відповідає певному джерелу. Масштабованість дозволяє створювати кілька потоків даних, наприклад окремо для кожного пристрою, окремо для кожного типу даних (температура, швидкість, напруга тощо). Підтримується одночасна передача тисяч подій за секунду. Дані відображаються на дашборді одразу після надсилання. Затримка візуалізації складає менше 1 секунди за умови стабільного з'єднання. Обсяг передачі даних залежить від тарифного плану (студентський, індивідуальний, корпоративний). Студентський план обмежений за кількістю потоків і обсягом даних. Дані зберігаються від 30 днів (студентський план), 1 року (індивідуальний) і до необмеженого часу (корпоративний). Кількість API викликів також обмежена, 60 викликів в хвилину (студентський), та 180 викликів в хвилину (індивідуальний).

Сповіщення та автоматизація також доступна на платформі у вигляді тригерів. Тригери дають можливість встановити дію, яка відбудеться за певної умови для потоку даних у режимі реального часу. Наприклад, надсилати лист на пошту щоразу, коли температура перевищує 40 градусів. Щойно ця умова стає істинною, дія виконується без пакетної обробки, без затримок (рис. 2.3).

Рисунок 2.3. Приклад сповіщення

2.3. Функціональні можливості Initial State

Функціональні можливості стосуються того, як платформа допомагає користувачам аналізувати, візуалізувати та використовувати дані через

інтуїтивний інтерфейс і аналітичні інструменти. Initial State спеціалізується на створенні інтерактивних дашбордів, які дозволяють користувачам у реальному часі відображати дані у вигляді графіків, діаграм і карт (рис 2.4).

Типи візуалізацій:

1 *Лінійні графіки*. Для відображення змін даних у часі, наприклад, температура.
2 *Стовпчасті діаграми*. Для порівняння значень, наприклад енергоспоживання різних пристроїв.
3 *Кругові діаграми.* Відображають відсотковий розподіл, наприклад розподіл енергоспоживання між кімнатами або пристроями.
4 *Вимірювальні діаграми*. Існує кілька стилів: дугова, стовпчаста, температурна, рідинна, що найкраще підходять для відображення вологості, температури, швидкості.
5 *Карта.* Може показувати місцезнаходження.
6 *Гістограма*. Для візуалізації частоти появи значень у певних межах.
7 *Текстові логи*. Відображають події, повідомлення, емодзі.

Рисунок 2.4. Доступні типи дошок Initial State

На платформі доступна можливість налаштовувати кольори, масштаби та розташування елементів, користувачі можуть змінювати період часу, масштабувати графіки, або фільтрувати дані (рис. 2.5).

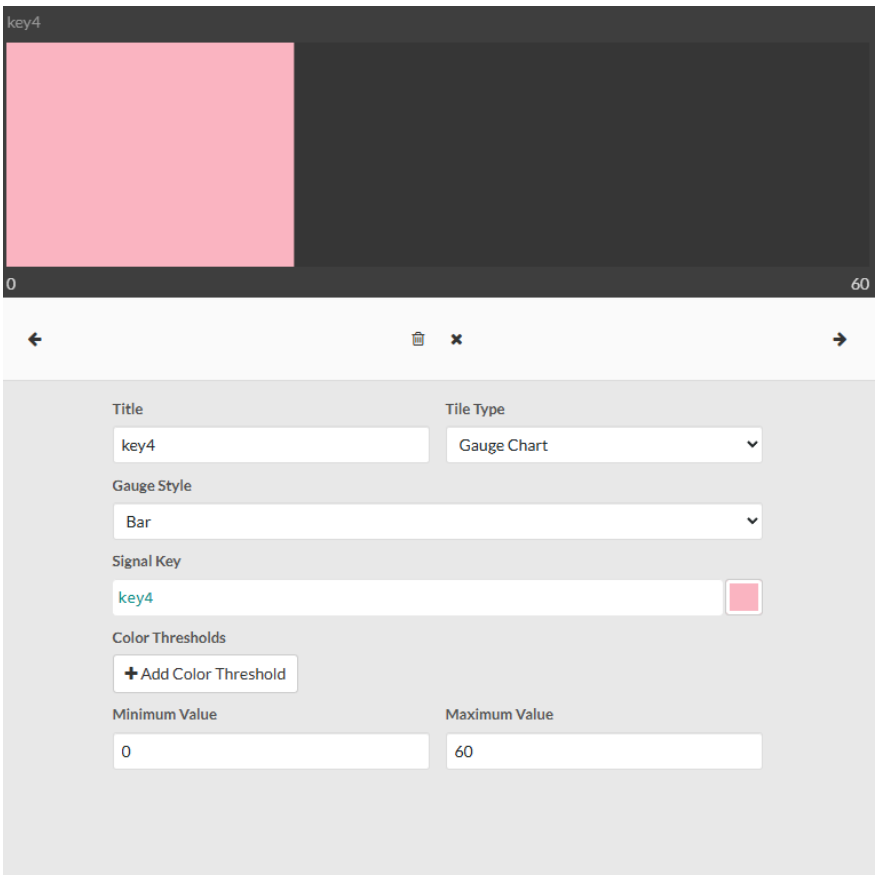

Рисунок 2.5. Налаштування віджету

Натиснувши на кнопку Edit Tiles вмикається режим редагування, візуальне середовище drag & drop дозволяє перетягувати кожну окрему плитку в нове положення або ж змінювати розмір (рис 2.6).

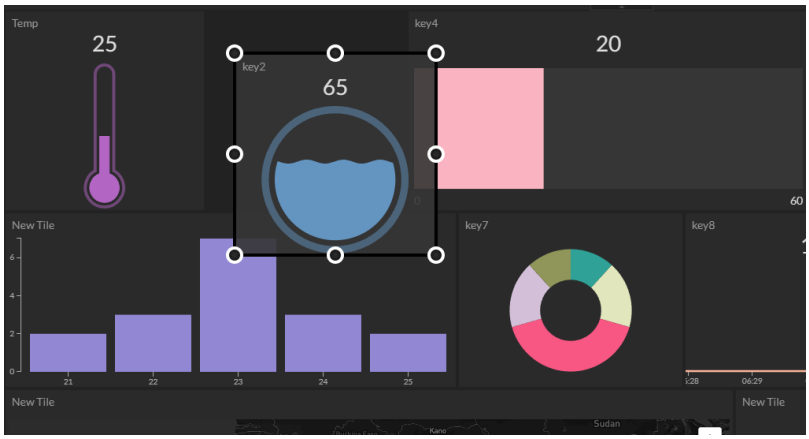

Рисунок 2.6. Drag & drop середовище

Платформа дозволяє налаштувати аналітику та тригери для автоматичних дій на основі даних. Окрім основного режиму з дашбордами також є режим interactive waveform, він дозволяє аналізувати дані шляхом збору цільової

статистики. Тут можна фільтрувати конкретні дані, групувати за часовим інтервалом, що дозволяє виявляти аномалії (рис. 2.7).

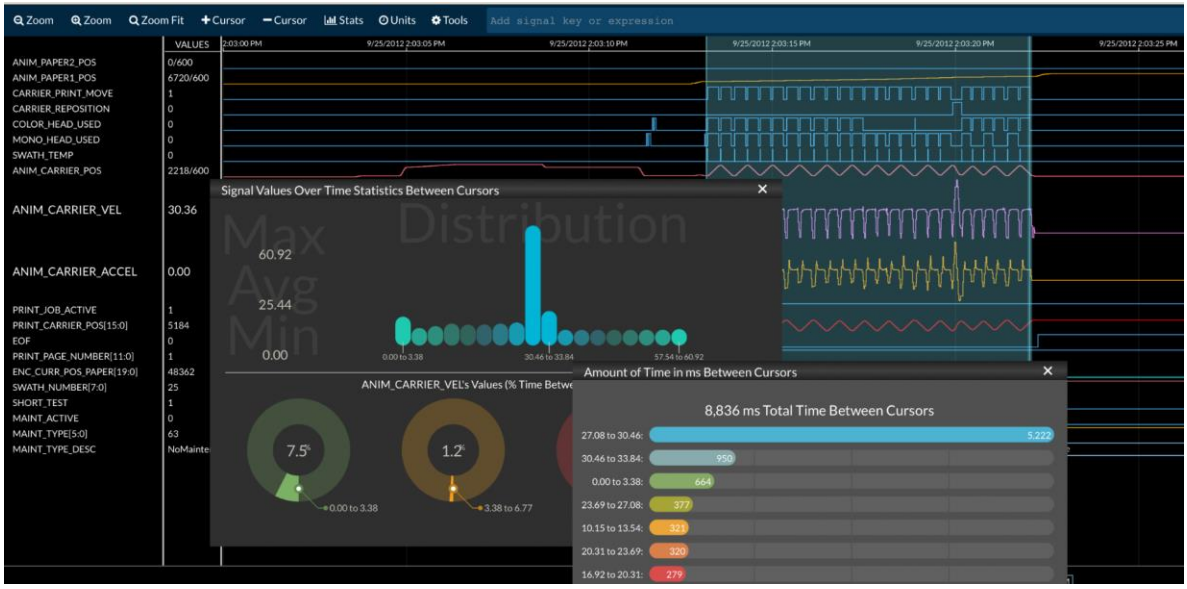

Рисунок 2.7. Interactive waveform

Також є можливість змінити задній фон серед пропонованих варіантів або ж власного фото, і є можливість редагування колірної гами віджетів та фону (рис. 2.8).

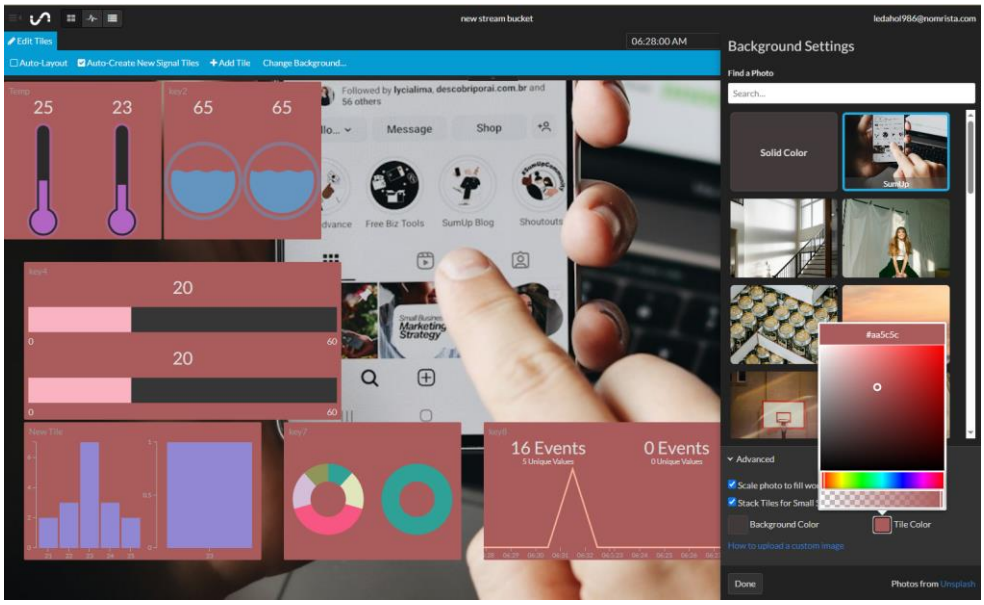

Рис 2.8. Редагування фону

Крім режиму дашбордів і interactive waveform є режим логів де видно кожен потік який надсилався і дані які були надіслані, тут можна зберегти дані в CSV форматі (рис 2.9).

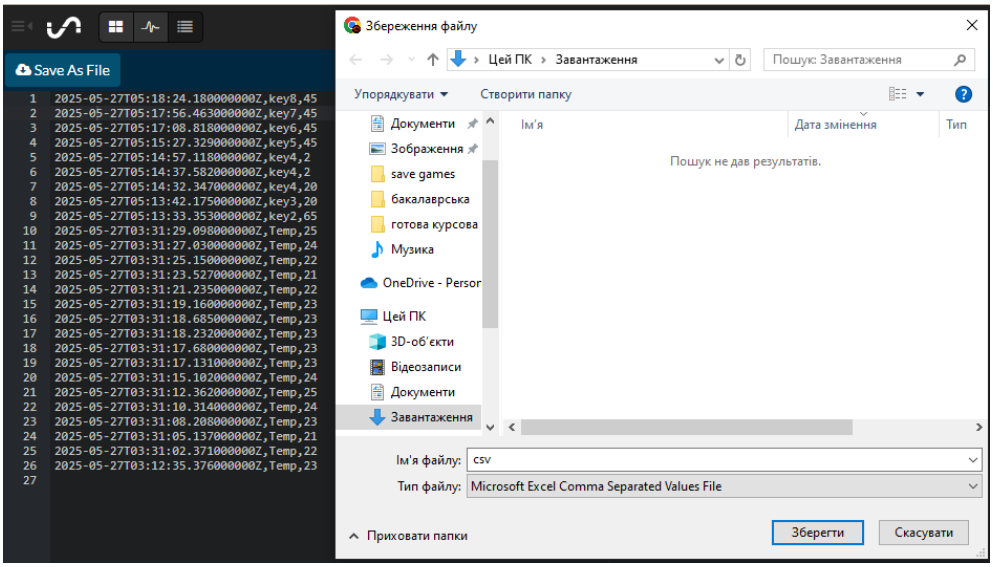

Рисунок 2.9. Імпорт даних про потоки

2.4. Налаштування контейнера збору даних

Щоб потоки даних і візуалізація співвідносилися між собою, існує контейнер даних. Контейнер даних – це набір потоків даних. Кожна доступна візуалізація відображатиме всі потоки даних для певного сегмента даних.

Кожен контейнер потокових даних має два пов’язані з ним ключі: ключ контейнера – bucket_key (згенерований випадковим чином, але його можна зробити його будь-яким) і приватний ключ доступу до потокового передавання – access_key. За допомогою цих двох ключів будь-який пристрій або програма може передавати дані в цей контейнер даних (тільки для запису, ці ключі не надають права на читання). Розглянемо існуючий контейнер даних.

В налаштуваннях контейнера даних існує три вкладки: дані, тригери та спільний доступ, розглянемо детально кожну з них.

Вкладку налаштування даних (рис. 2.10). Відображає назву контейнера, ключ контейнера та ключ доступу, є URL-адреса для передачі потоку даних, поля для відправки даних та імпорт іншого контейнера даних, копіювання поточного або ж видалення. Щоб передати дані в контейнер даних, потрібна URL-адреса кінцевої точки API – API Endpoint.

https://groker.init.st/api/events?accessKey=ist_0F9g_gH1Yw00wQ15EMobrYJjpjavv2dk&bucketKey=9JRDN5X8CVPA

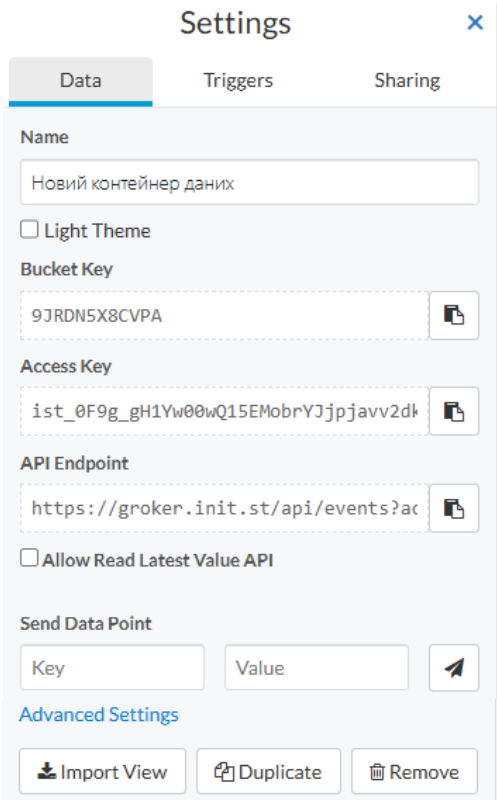

Рисунок 2.10. Налаштування контейнера даних

Розглянемо, що міститься в URL-адресі API Endpiont. Посилання складається з першої частини – потоку API Endpiont – https://groker.init.st/api/events, решта – параметри URL-адреси. Є два параметри, accessKey і bucketKey:

- accessKey=ist_0F9g_gH1Yw00wQ15EMobrYJjpjavv2dk;
- bucketKey=9JRDN5X8CVPA.

Щоб передати дані в цей контейнер даних, потрібно додати додаткові параметри URL-адреси, щоб вказати назву потоку даних і відповідне значення. Додамо "&myNumber=5" до URL-адреси. Вставивши готову URL-адресу в адресний рядок браузера була надіслана подія "5" у потік "myNumber" (рис. 2.11.). З’явилася нова плитка з назвою myNumber зі значенням 5, лінійний графік будується автоматично, оновлюючись в реальному часі.

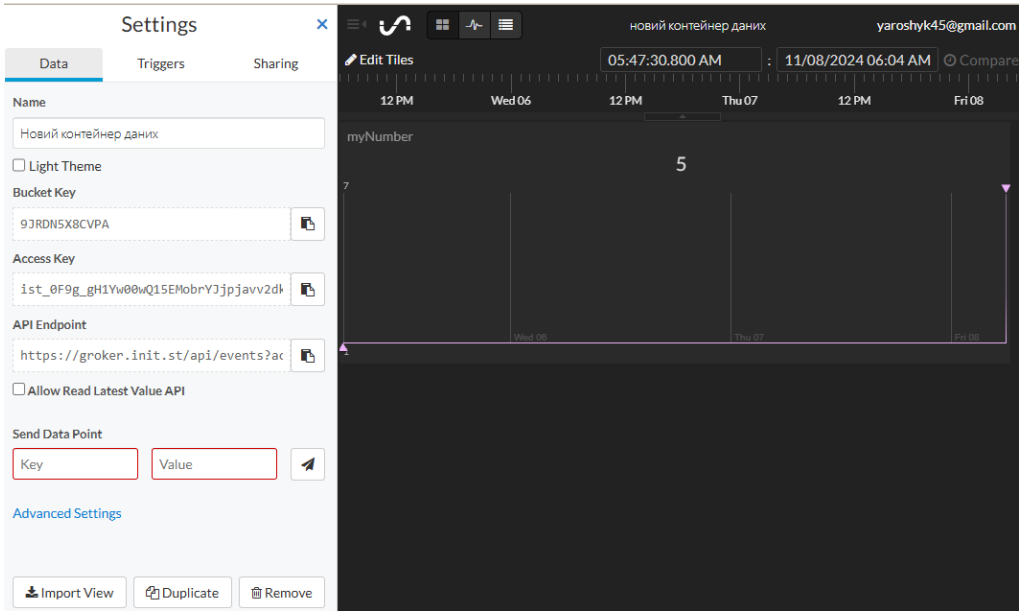

Рисунок 2.11. Приклад події у потоці даних

Також замість вписування в URL-адресу можна просто вказати назву потоку даних і значення в самій вкладці налаштувань даних і відправити ці дані (рис. 2.12.).

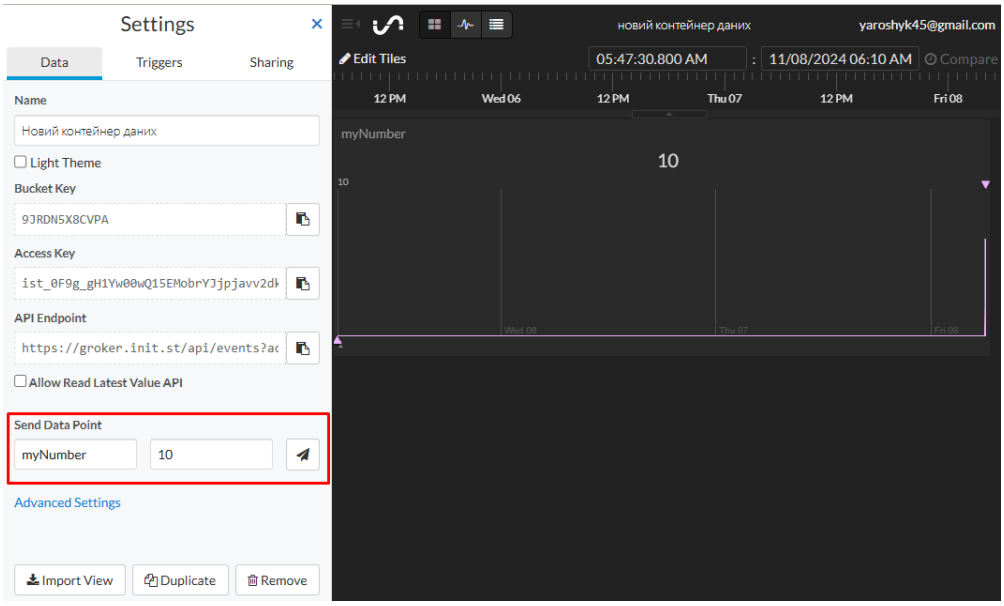

Рисунок 2.12. Відправка даних через налаштування

Спробуємо додати ще одну плитку але замість числа використаємо рядки. Створимо новий потік даних «myMessage» з текстом «Привіт» (рис. 2.13.).

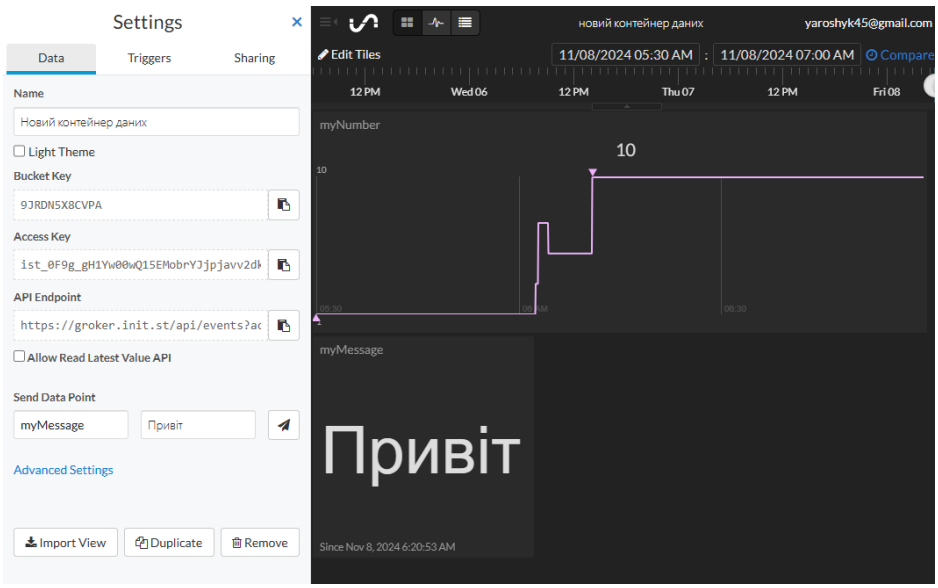

Рисунок 2.13. Додавання текстової панелі

Плитку було автоматично створено для нового сегмента даних, проте його можна налаштувати. Редактор інформаційної панелі вбудовано безпосередньо у веб-програму віджетів. Натиснувши на кнопку Edit Tiles. Переводимо панель в режим редагування, кожну окрему плитку можна перетягувати в нове положення, або ж змінювати розмір (рис. 2.14.).

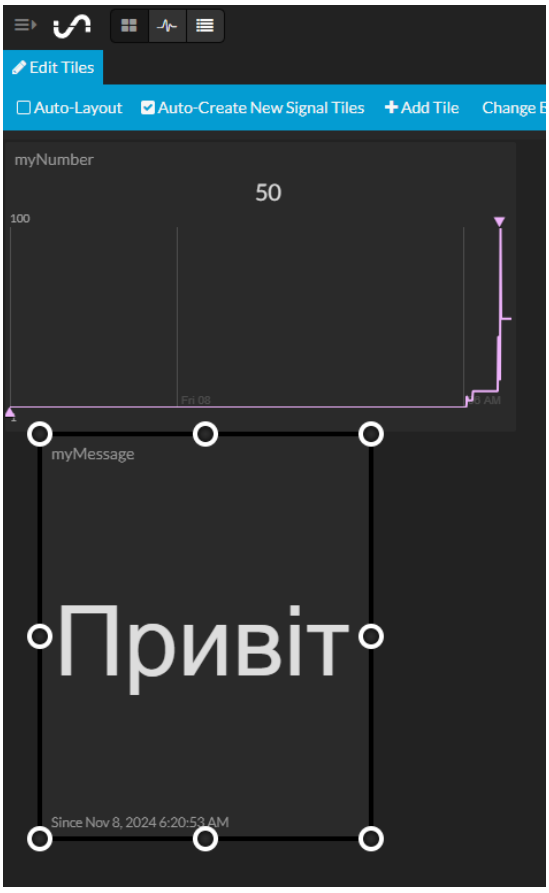

Рисунок 2.14. Режим редагування

Також можна додати на інформаційну панель скільки завгодно плиток, одного потоку даних. Для цього потрібно натиснути «+Додати плитку» вгорі, щоб додати нову плитку. Встановити назву плитки в полі Title, тип плитки в полі Tile Type і джерело даних у полі Signal Key. Наприклад, додамо потік даних myNumber як ключ сигналу та Map зі списку Tile Type (рис. 2.15.). На інформаційну панель буде додано нову плитку, карту.

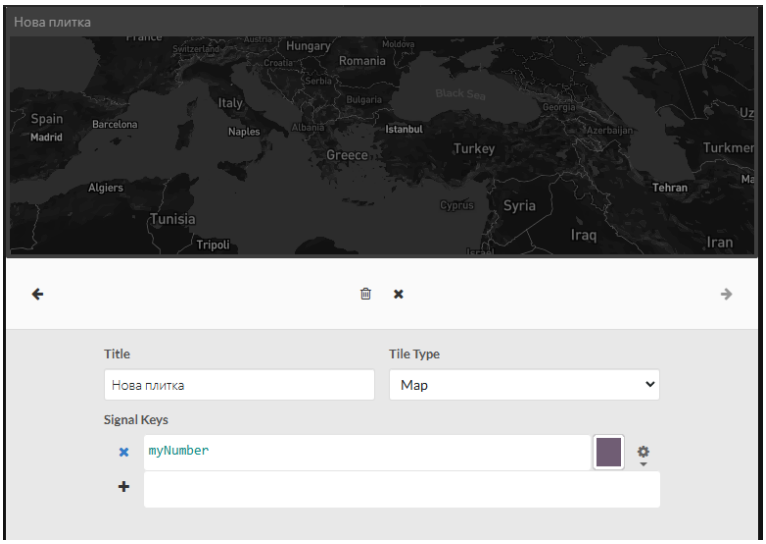

Рисунок 2.15. Додавання нової плитки

Також в додаткових налаштуваннях можна відфільтрувати показувати всі дані, або впродовж якогось часу (рис. 2.16).

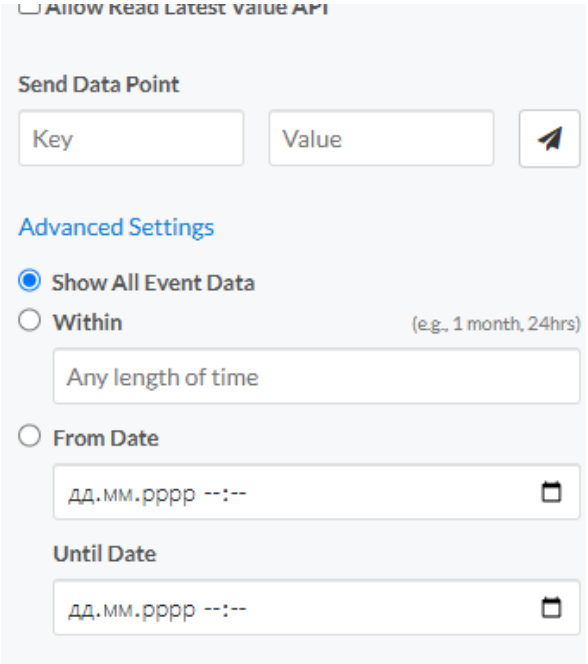

Рисунок 2.16. Додаткові налаштування

Вкладка тригерів. Тригери дають можливість отримувати сповіщення про досягнення умови, яку вказано у даних. Наприклад, «надішліть мені електронний лист, якщо температура вище 15°C». Спробуємо налаштувати сповіщення електронною поштою щоразу, коли число myNumber дорівнює 50 (рис. 2.17).

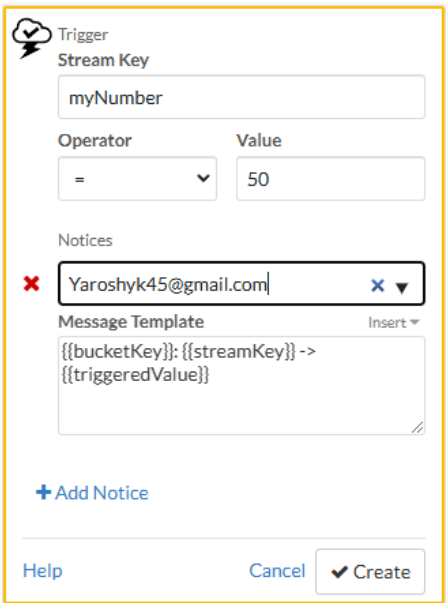

Рисунок 2.17. Налаштування сповіщення

І при надсиланні «50» приходить сповіщення на пошту (рис. 2.18).

Рисунок 2.18. Приклад сповіщення електронною поштою

Це можна використовувати для автоматичного логування помилок в системі, увімкнення пристроїв, зміну стану.

Вкладка спільний доступ. За замовчуванням ніхто інший не може бачити наявні дані, але можна надати комусь приватний доступ до даних і переглядів або надати публічний доступ.

Щоб відкрити доступ до даних, потрібно натиснути прапорець «Поділитися публічно». URL-адреса буде розташована в полі «Поділитися за URL-адресою». У наведеному прикладі (рис. 2.19.) URL-адреса спільного доступу є

https://go.init.st/k658p67. Щоб вимкнути публічний доступ потрібно зняти прапорець.

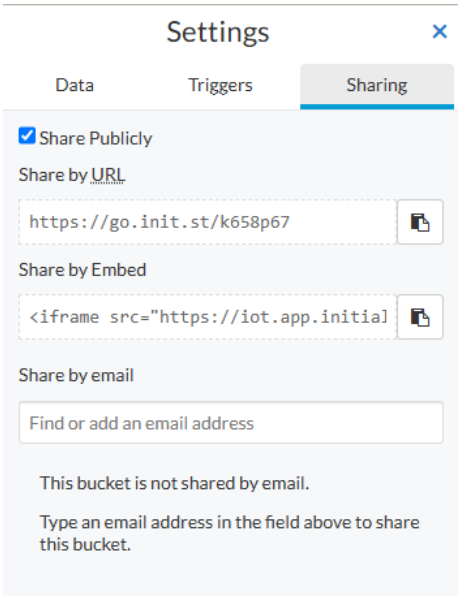

Рисунок 2.19. Публічний доступ

Щоб відкрити доступ до даних, потрібно натиснути прапорець «Поділитися публічно». URL-адреса буде розташована в полі «Поділитися за URL-адресою». У наведеному вище прикладі (рис. 2.19.) URL-адреса спільного доступу є https://go.init.st/k658p67. Щоб вимкнути публічний доступ потрібно зняти прапорець.

Якщо потрібно поділитися приватно можна добавити доступ по вказаній поштовій адресі. Лише користувачі, зазначені в розділі «Надіслати електронною поштою», зможуть переглядати дані та пов’язані з ними перегляди. можна легко видалити приватний доступ від певної особи, натиснувши на синій знак «-» біля її електронної адреси.

В загальному Initial State – це ефективна, масштабована та зручна у використанні платформа для обробки та візуалізації даних IoT. Її інструментальні та функціональні можливості дозволяють швидко створити наочні дашборди для моніторингу фізичних параметрів у режимі реального часу. Вона є особливо корисною для прототипування, навчальних проєктів, а також для невеликих комерційних систем.

РОЗДІЛ 3
АПАРАТНО-ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ОДНОПЛАТНОГО КОМП'ЮТЕРА RASPBERRY PI 3 MODEL B

3.1. Поняття одноплатного комп'ютера

Одноплатний комп'ютер (англ. single-board computer**)** – це повноцінний комп'ютер, побудований на одній друкованій платі з мікропроцесором, пам'яттю, входом/виходом та іншими функціями, необхідними для функціонального комп'ютера. Одноплатні комп'ютери зазвичай виготовляються як демонстраційні або розробницькі системи, для освітніх систем або для використання як вбудовані комп'ютерні контролери. Багато типів домашніх або портативних комп'ютерів інтегрують усі свої функції на одній друкованій платі [8].

Історія одноплатних комп'ютерів починається не з великих промислових розробок чи дорогих обчислювальних машин, а з простої ідеї: зробити комп'ютер доступним, компактним і таким, що поміщається в одну руку. Ідея об'єднати процесор, пам'ять і всі необхідні компоненти на одній платі здалася спочатку майже фантастичною, однак саме вона стала основою вбудованих систем.

Справжній прорив відбувся у 2012 році, коли на ринку з'явився Raspberry Pi (рис. 3.1) – невелика плата, яка запропонувала повноцінне середовище Linux, відеовихід HDMI, USB, GPIO для сенсорів, Wi-Fi, Bluetooth. І найголовніше – вона була орієнтована на дітей, вчителів, винахідників, ентузіастів, а не лише інженерів з промисловості. Raspberry Pi змінив уявлення про те, що таке комп'ютер. Це стало початком нової епохи одноплатних комп'ютерів – доступних, відкритих і призначених для творчості. Після цього на ринку з'явилися десятки інших моделей і брендів – Orange Pi, Banana Pi, BeagleBone, Odroid кожен зі своєю специфікою, потужністю й ціною. Одноплатні комп'ютери стали основою не лише IoT-рішень, а й персональних серверів, домашніх медіацентрів, систем відеоспостереження, навчальних стендів, роботів і навіть повноцінних настільних ПК.

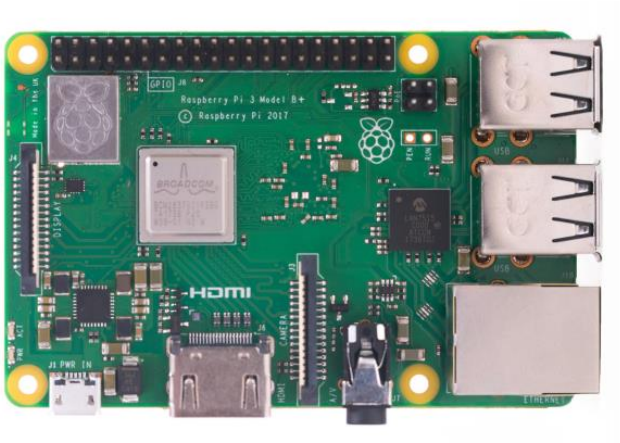

Рисунок 3.1 Зовнішній вигляд плати Raspberry Pi

3.2. Апаратне забезпечення Raspberry Pi 3 Model B

Одноплатний комп’ютер Raspberry Pi 3 Model B є оптимальним вибором для створення компактних, енергоефективних і функціональних вбудованих систем. У контексті даного проєкту він виконує роль центрального вузла, що забезпечує взаємодію між датчиками, локальним програмним забезпеченням і хмарною інфраструктурою для візуалізації та зберігання даних.

Raspberry Pi 3 Model B – найдавніша модель Raspberry Pi третього покоління. Вона замінила Raspberry Pi 2 Model B у лютому 2016 року [9]. Продуктивність в порівнянні з колишньою моделлю виросла на 50%. Форм фактор, розташування роз’ємів, отвори кріплення залишилися такими ж як і у Raspberry Pi Model B+ і Raspberry Pi 2 Model B, тобто не потрібно купувати новий корпус, чи якісь інші аксесуари, при переході на нову модель Raspberry Pi 3 B. Єдине, що збільшився струм споживання, тому рекомендується блок живлення з вихідним струмом не менше 2.5A [10].

Технічні характеристики Raspberry Pi 3 Model B:

- SoC Broadcom BCM2837 – Raspberry Pi 3 Model B оснащений чотириядерним 64-бітним ARM Cortex-A53 процесором з тактовою частотою 1,2 ГГц [11].
- Графічний процесор – VideoCore IV: підтримка OpenGL ES 2.0, H.264 (1080p30) [11].

- Оперативна пам’ять 1 ГБ LPDDR2 SDRAM.
- Сховище – слот microSD (рекомендується 8–32 ГБ).

Мережеві модулі. В третій версії комп’ютер має вбудовані модулі Wi-Fi 802.11n та Bluetooth 4.1, що дозволяє легко інтегрувати його в локальну мережу або з’єднуватися з іншими пристроями. Тому її можна повноцінно використовувати для роботи без додаткових модулей.

Ще однією цікавою особливістю, якою володіє Raspberry Pi 3 в порівнянні зі своїми попередниками – наявність підтримки 64-бітних інструкцій це впливає позитивно на швидкість роботи

Інтерфейси та периферія (рис 3.2):

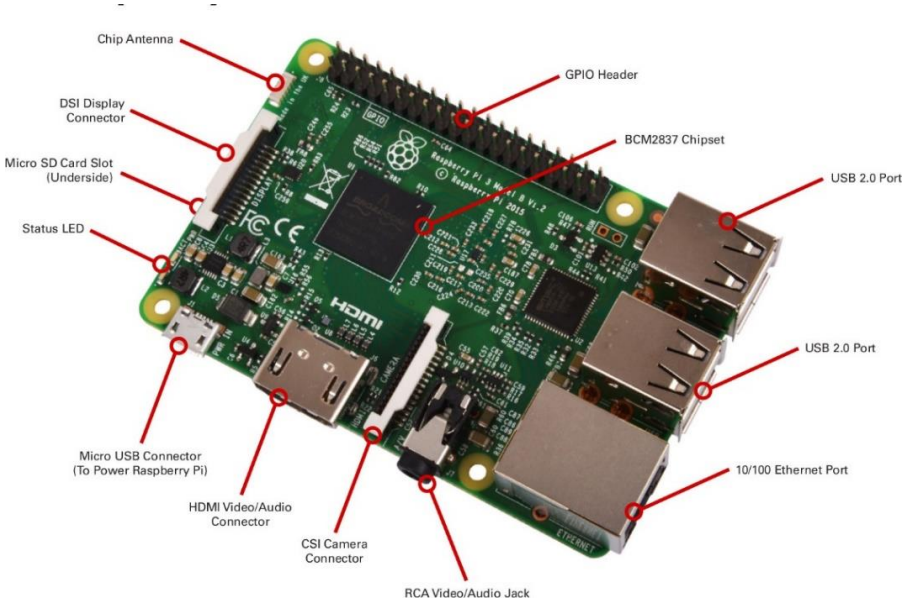

Рисунок 3.2. Інтерфейси Raspberry Pi 3 Model B

- GPIO (40 пінів): підключення сенсорів, реле, модулів (рис. 3.3).

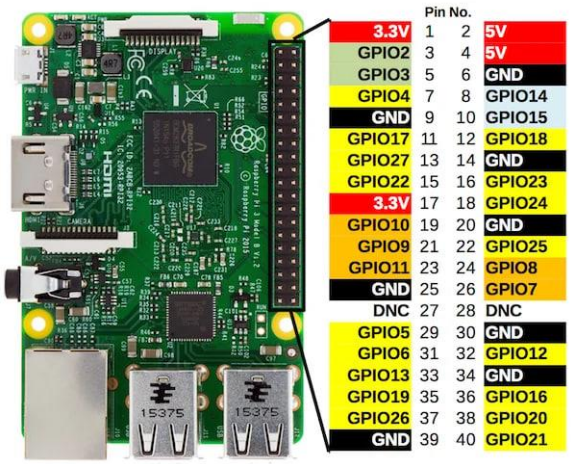

Рисунок 3.3. Розпіновка GPIO

- USB: 4 порти USB 2.0 для клавіатури, миші, накопичувачів.
- HDMI, CSI, DSI (дисплей), аудіо/відео (3,5 мм).
- Живлення: Raspberry Pi 3 Model B живиться напругою на рівні (5В, 2.5А) через роз'єм micro-USB, але його логіка працює на рівні 3.3 В, тому випадкове або навмисне підключення більш високої напруги 5 В, безпосередньо до штирів роз'єму введення / виведення GPIO може пошкодити плату [12]. Необхідно використовувати схеми узгодження рівнів сигналу навіть якщо периферія має сигнали рівня 5В.
- Охолодження: для підвищення стабільності роботи рекомендується використовувати алюмінієві радіатори (радіатори 9 × 9 × 5 мм), для охолодження процесора.
- Габаритні розміри 86x56x17 мм (рис. 3.4).
- Вага 50 грам.

3.3. Програмне забезпечення Raspberry Pi 3 Model B

Raspberry Pi може працювати під багатьма системами наприклад: Windows CE, Debian, Fedora, Gentoo, Arch Linux, RISC OS, AROS або FreeBSD, навіть існує Android для Raspberry Pi. Також розроблені ОС які базуються на Debian (Raspbian) і Fedora (FedoraRemix, Pidora) оптимізовані під Raspberry Pi [13].

Найчастіше для роботи Raspberry Pi 3 Model B використовується спеціально адаптована операційна система Raspberry Pi OS (раніше Raspbian) (рис 3.4), яка базується на Debian Linux. Система підтримує велику кількість мов програмування, зокрема Python, C/C++, Java, а також має повну сумісність із більшістю бібліотек і фреймворків, що застосовуються у розробці для IoT. Оскільки Raspberry Pi OS є похідною від Debian, вона дотримується поетапного циклу випусків Debian. Випуски відбуваються приблизно кожні 2 роки [14]. Raspberry Pi OS спеціально оптимізована для роботи на малопотужному апаратному забезпеченні.

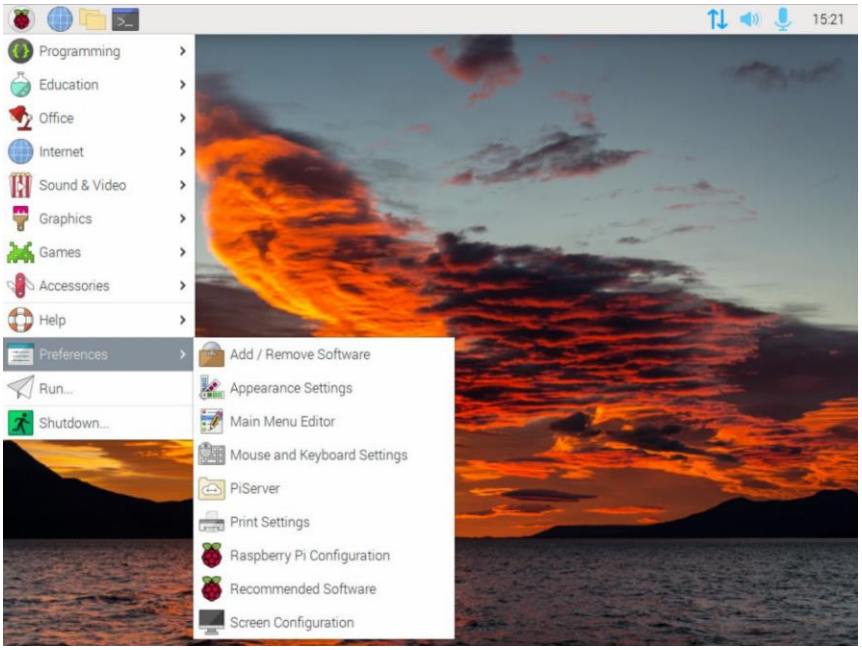

Рисунок 3.4. Вигляд графічної оболонки Raspberry Pi OS

Raspberry Pi OS випускається в трьох варіантах:

- Full – з графічною оболонкою, офісним пакетом і додатками;
- Lite – тільки термінал, без GUI (зручно для серверів, headless-проєктів);
- Legacy – стара стабільна версія (підтримка специфічних моделей або застосунків).

Операційна система Raspberry Pi OS надає користувачеві повноцінне середовище для розробки програмного забезпечення, особливо у сфері вбудованих систем, автоматизації та Інтернету речей. Завдяки відкритій архітектурі, підтримці популярних мов програмування та зручним інструментам зокрема для систем, де важлива швидкість обробки сигналів або прямий доступ до апаратури, доцільно використовувати мови C або C++. Для складних логічних систем, або коли потрібна кросплатформеність можна використовувати Java з бібліотеками доступу до GPIO через Pi4J. Node.js Підходить для створення веб-інтерфейсів, REST API або вебсервісів, що працюють на самому Raspberry Pi.

Проте Python є де-факто стандартом для програмування на Raspberry Pi завдяки своїй простоті, читабельності та великій кількості бібліотек.

Ключові бібліотеки:

- gpiozero високорівнева бібліотека для керування GPIO (включення світлодіодів, зчитування даних із сенсорів тощо) (рис. 3.5);
- RPi.GPIO низькорівнева бібліотека для точного контролю пінів GPIO;
- Adafruit_DHT, adafruit_bme280 для роботи з датчиками температури, вологості, тиску;
- requests, urllib, ISStreamer для надсилання HTTP-запитів, інтеграції з хмарними API (Initial State, Firebase, Telegram).

```
1   from gpiozero import LED
2   from time import sleep
3
4   led = LED(17)
5
6   while True:
7       led.on()
8       sleep(1)
9       led.off()
10      sleep(1)
```

Рисунок 3.5. Python код для керування світлодіодом через gpiozero

Розробники Raspberry Pi OS рекомендують використовувати IDE Thonny для редагування коду Python на Raspberry Pi [14] рис(3.6).

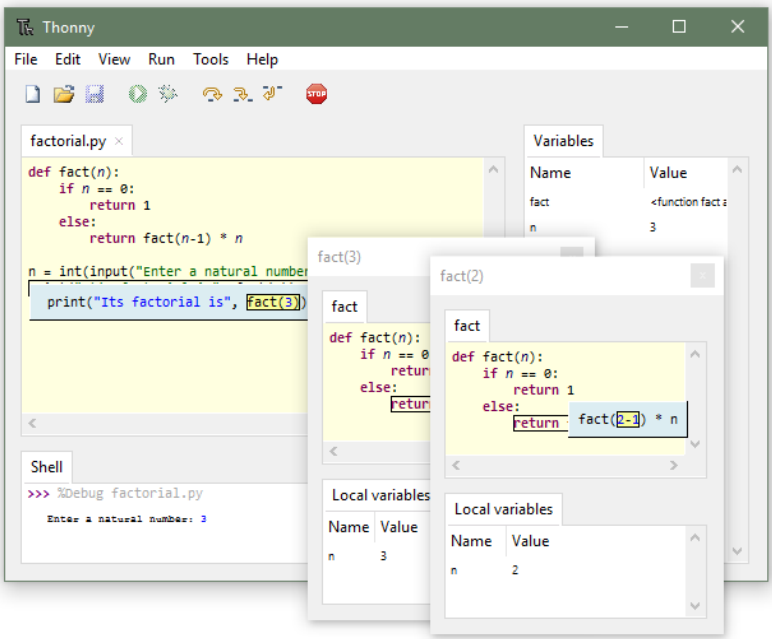

Рисунок 3.6. Інтерфейс IDE Thonny

РОЗДІЛ 4
РОЗРОБКА СИСТЕМИ ЗБОРУ ТА ПЕРЕДАЧІ IOT ДАНИХ ДЛЯ ВІЗУАЛІЗАЦІЇ В INITIAL STATE

4.1. Постановка задачі

Потрібно створити простоу, зрозумілоу, але водночас функціональноу IoT-систему, яка буде дозволяти в режимі реального часу збирати, обробляти та візуалізувати важливі дані з навколишнього середовища. Система має збирати кілька типів даних, які відображають як стан навколишнього середовища, так і взаємодію з пристроєм. Ось які саме параметри відстежуються (рис. 4.1):

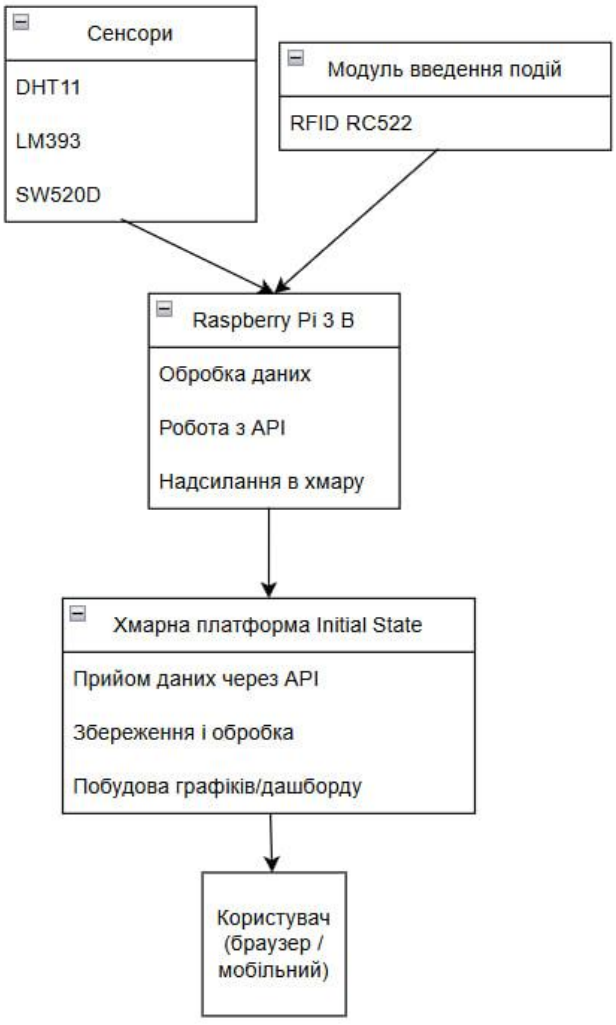

Рисунок 4.1. Блок-схема загальної структури системи

- *Температура і вологість повітря*. За допомогою сенсора DHT11 система постійно вимірює поточну температуру та рівень вологості. Це базові параметри, які дозволяють стежити за комфортом або змінами клімату.

- *Вібрація та нахил*. Датчик SW-520D реагує на струс або зміну положення. Це дозволяє зафіксувати ситуації, коли пристрій було зрушено, впав або його торкнулися, або ж стабільність системи.

- *Звук*. Використовується простий мікрофонний модуль для фіксації наявності або відсутності звукових коливань. Наприклад, система може “почути”, що щось відбулося поруч, і передати цю подію.

- *Місцезнаходження*. Визначається широта та довгота пристрою. Це дозволяє стежити за тим, де саме перебуває система, або відстежувати переміщення.

- *Доступ через RFID*. RFID-модуль дозволяє ідентифікувати користувача за допомогою карти. Таким чином фіксується, хто саме отримав доступ до пристрою або активував систему.

Дата і час події. До кожного вимірювання додається позначка часу, щоб у візуалізації було зрозуміло, коли саме відбулась подія або було зняте значення.

Оскільки система орієнтована на реальний моніторинг подій, вона повинна працювати в режимі реального часу, тобто дані мають надсилатися одразу після отримання, без суттєвих затримок.

4.2. Вибір апаратних компонентів

В якості основного обчислювального вузла було обрано Raspberry Pi 3 Model B. Цей одноплатний комп’ютер поєднує високу продуктивність, малі розміри, вбудовану підтримку Wi-Fi та Bluetooth, а також широке програмне забезпечення.

Сенсори та модулі.

Датчик температури та вологості DHT11. Сенсор має пластиковий корпус із вентиляційними отворами, всередині якого знаходяться: NTC-термістор для вимірювання температури; ємнісний сенсор вологості; мікросхема, що обробляє

аналогові сигнали та передає їх у цифровому вигляді через один пін. Передача даних відбувається через спеціальний однопровідний протокол. Для роботи з датчиком потрібна бібліотека Adafruit_DHT, яка забезпечує простий інтерфейс для зчитування температури та вологості (рис 4.2).

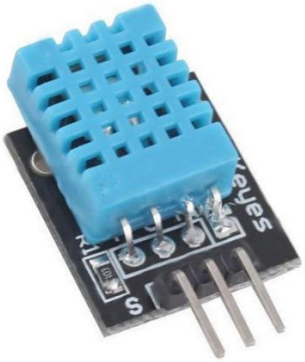

Рисунок 4.2. Датчик температури і вологості

Також тут є 8-бітний мікроконтролер для виведення значень температури та вологості у вигляді послідовних даних. Датчик має заводське калібрування, тому його легко взаємодіяти з іншими мікроконтролерами. Датчик може вимірювати температуру від 0°C до 50°C та вологість від 20% до 90% з точністю ±1°C та ±1% [15].

Датчик звуку LM393. Цей модуль виявлення звуку складається з мікрофона, резисторів, конденсатора, потенціометра, компаратора LM393, індикатора живлення та світлодіода стану в інтегральній схемі. Датчик виявлення звуку визначає інтенсивність звуку, коли звук виявляється за допомогою мікрофона та подається на операційний підсилювач LM393. Він містить вбудований потенціометр для регулювання заданого рівня звуку. Чутливість мікрофона (1 кГц): від 52 до 48 дБ. Мікросхема LM393 використовується як напруговий компаратор у цьому модулі виявлення звуку. Другий пін LM393 підключено до попередньо встановленого резистора (потенціометра 10 кОм), тоді як третій підключено до мікрофона. Компаратор порівнює порогову напругу, встановлену за допомогою потенціометра (другий пін), із сигналом з мікрофона (третій пін) Для зміни чутливості потрібно розкрутити гвинтик [16].

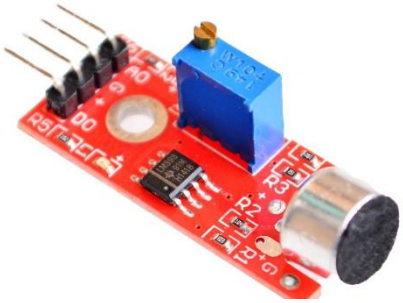

Рисунок 4.3. Датчик звуку

Радіочастотний модуль RFID RC522. Це радіочастотний модуль, що складається з RFID-зчитувача, RFID-картки та брелока для ключів. Модуль працює на частоті 13,56 МГц, що є промисловим діапазоном (ISM), тому його можна використовувати без проблем з ліцензуванням. Модуль зазвичай працює на напрузі 3,3 В і тому широко використовується в конструкціях з напругою 3,3 В. Зазвичай він використовується в тих випадках, коли певну особу/об’єкт необхідно ідентифікувати за допомогою унікального ідентифікатора. Брелок має 1 КБ пам’яті, яку можна використовувати для зберігання унікальних даних. Модуль зчитування RC522 може як зчитувати, так і записувати дані в ці елементи пам’яті. Зчитувач може зчитувати дані лише з пасивних міток, що працюють на частоті 13,56 МГц. RC522 має робочу напругу від 2,5 В до 3,3 В, тому зазвичай живиться від напруги 3,3 В і повинен використовуватися з лініями зв’язку 3,3 В. Однак, комунікаційні контакти цього модуля стійкі до 5 В [17].

Рисунок 4.4. RFID-модуль

Датчик вібрації і нахилу SW-520D. Це високочутливий датчик у корпусі конденсатора, призначений для визначення кута нахилу та механічних вібрацій. Він використовується у різних системах безпеки, розумних пристроях та проєктах

для визначення змін положення або вібраційних впливів. Датчик побудований на механічному кульковому контакті, який спрацьовує при зміні кута нахилу або впливі вібрації. Усередині знаходиться металеві кульки, які при нахилі замикають контакт, що дозволяє фіксувати зміну положення (рис. 4.5.). Цей датчик можна використовувати для визначення аварійних ситуацій (наприклад, падінь або сильних вібрацій), а також у розумних системах для реагування на зміну положення об’єкта. Чутливість датчика висока, він реагує на найменші зміни кута нахилу або вібрації [18].

Рисунок 4.5. Датчик вібрації і нахилу SW-520D

4.3. Налаштування і підключення датчиків

На цьому етапі здійснюється фізичне підключення сенсорів до одноплатного комп’ютера Raspberry Pi 3 Model B, а також їхнє програмне налаштування для подальшого зчитування даних. Для передачі сигналів від сенсорів використовуються GPIO-піни.

Підключимо датчик DHT11 як на рис. 4.6.

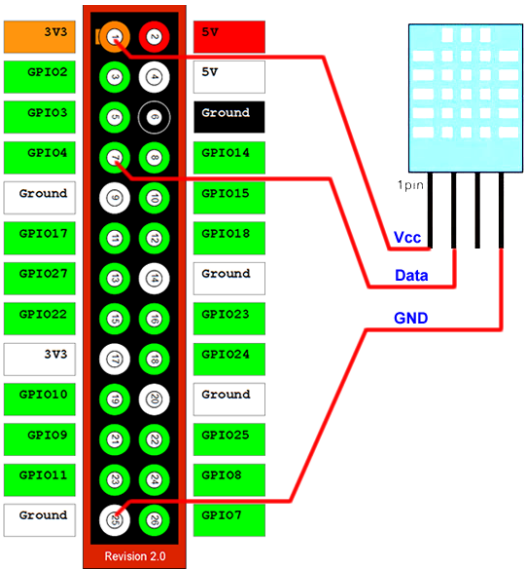

Рисунок 4.6. Схема підключення датчика температури і вологості

Щоб датчик передав дані використовується наступний програмний код.

```
import Adafruit_DHT
import time

# Вибір сенсора DHT11
DHT_SENSOR = Adafruit_DHT.DHT11

# GPIO пін, до якого підключено сигнал DHT11
DHT_PIN = 14

print("Запуск зчитування з датчика DHT11...")

try:
    while True:
        # Зчитування з сенсора
        humidity, temperature = Adafruit_DHT.read_retry(DHT_SENSOR, DHT_PIN)

        if humidity is not None and temperature is not None:
            print(f"Температура: {temperature:.1f}°C  |  Вологість: {humidity:.1f}%")
        else:
            print("❌ Не вдалося зчитати дані з датчика.")

        time.sleep(2)  # Затримка 2 секунди між зчитуваннями

except KeyboardInterrupt:
    print("⛔ Зчитування перервано користувачем.")
```

Тепер підключимо датчик звуку LM393 як показано на рис. 4.7.

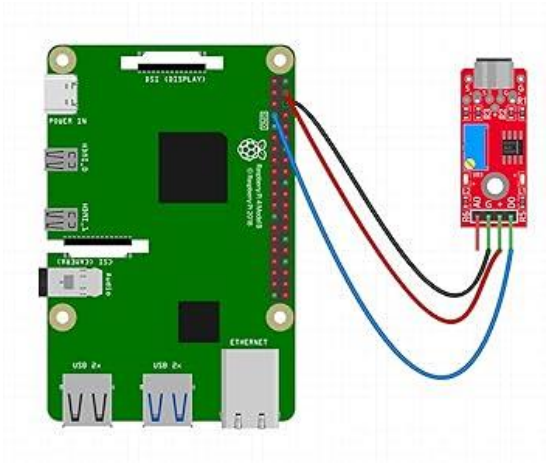

Рисунок 4.7. Схема підключення LM393

Для запуску датчика використовується цей код, він визначає гучні звуки з цифровим виходом, наприклад плескання в долоні і підраховує кількість виявлень:

```
from RPi import GPIO
from time import sleep
```

```
Detect = 2
GPIO.setmode(GPIO.BCM)
GPIO.setup(Detect, GPIO.IN)

print('І ТЕПЕР...ГУЧНО БЄМО В ДОЛОНІ')
print('-----------------------------\n')

i=0

while True:
    if GPIO.input(Detect) == GPIO.LOW: # Звуковий поріг по LOW
        i=i+1
        print('Я чую тебе %s раз\n' % i)
        sleep(1)
```

Підключимо датчик вібрації/нахилу скориставшись схемою на рис. 4.8.

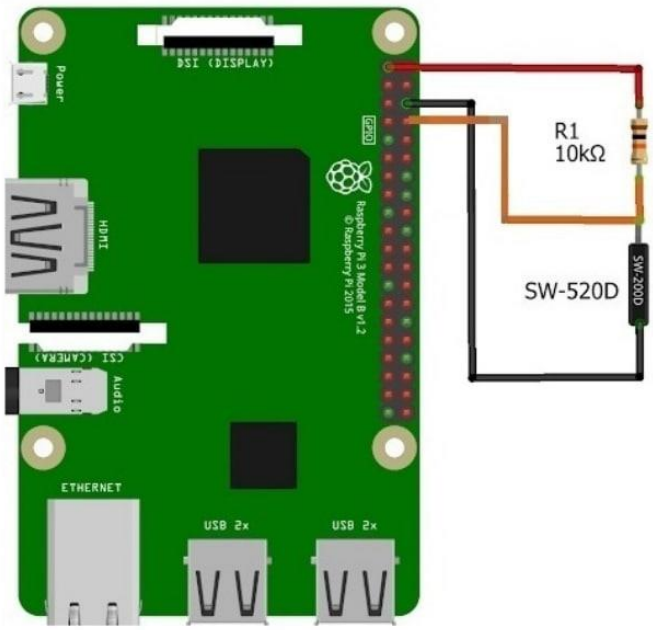

Рисунок 4.8. Схема підключення датчика вібрації/нахилу

Написаний програмний код для передачі даних з датчика, яка реалізує повідомлення в разі порушення спокою:

```
from RPi import GPIO

SW=14

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(SW, GPIO.IN)

print('Завмираєм в очікуванні...\n')
```

```
try:
    while True:
        if GPIO.wait_for_edge(SW, GPIO.BOTH):
            print('A-A-A...Нахил і Відбрація')

except KeyboardInterrupt:
    print('На все добре')
    GPIO.cleanup()
```

Далі потрібно підключити радіочастотний модуль за схемою на рис. 4.9.

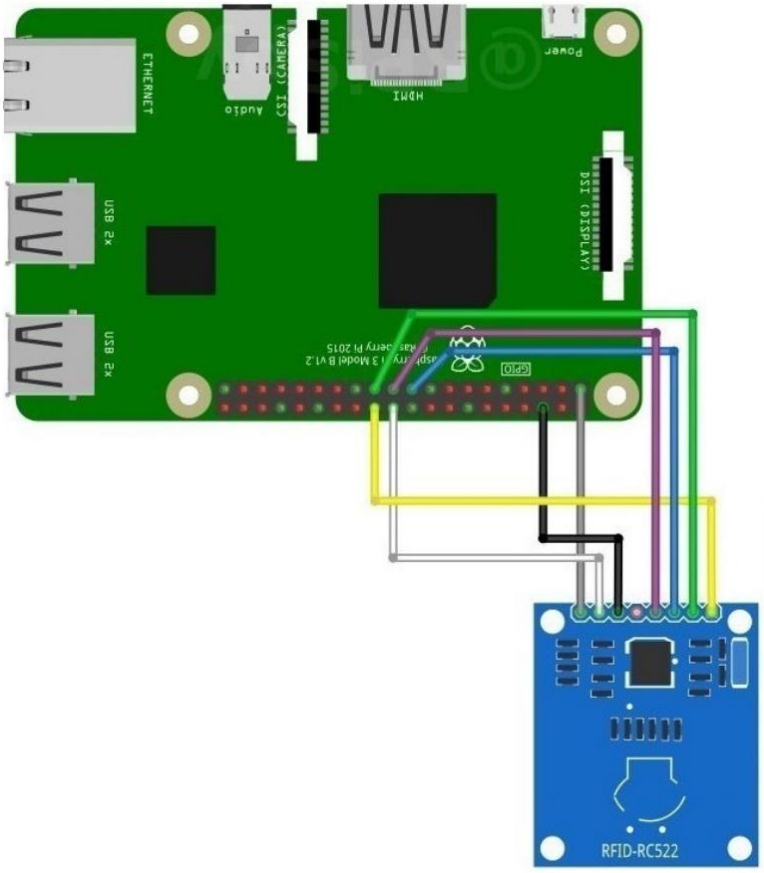

Рисунок 4.9. Схема підключення радіочастотного модуля

Для роботи ідентефікатора потрібна записати користувача на брелок, для цього використаний цей код:

```
from RPi import GPIO
from mfrc522 import SimpleMFRC522

reader = SimpleMFRC522()

try:
        text = input('Введіть імя: ')
        print('Тепер піднесіть картку')
```

```
        reader.write(text)
        print('OK! ЗаписанО')
finally:
        GPIO.cleanup()
```

Скориставшись записом був записаний користувач Вадим з власним ідентефікатором. За допомогою цього програмного коду можна зчитувати дані:

```
from RPi import GPIO
from mfrc522 import SimpleMFRC522

reader = SimpleMFRC522()

print('Зчитуємо дані...')
try:
        id, text = reader.read()
        print(id)
        print(text)
finally:
        GPIO.cleanup()
```

Тепер всі датчики під'єднані як вказано на монтажній схемі (рис 4.10.).

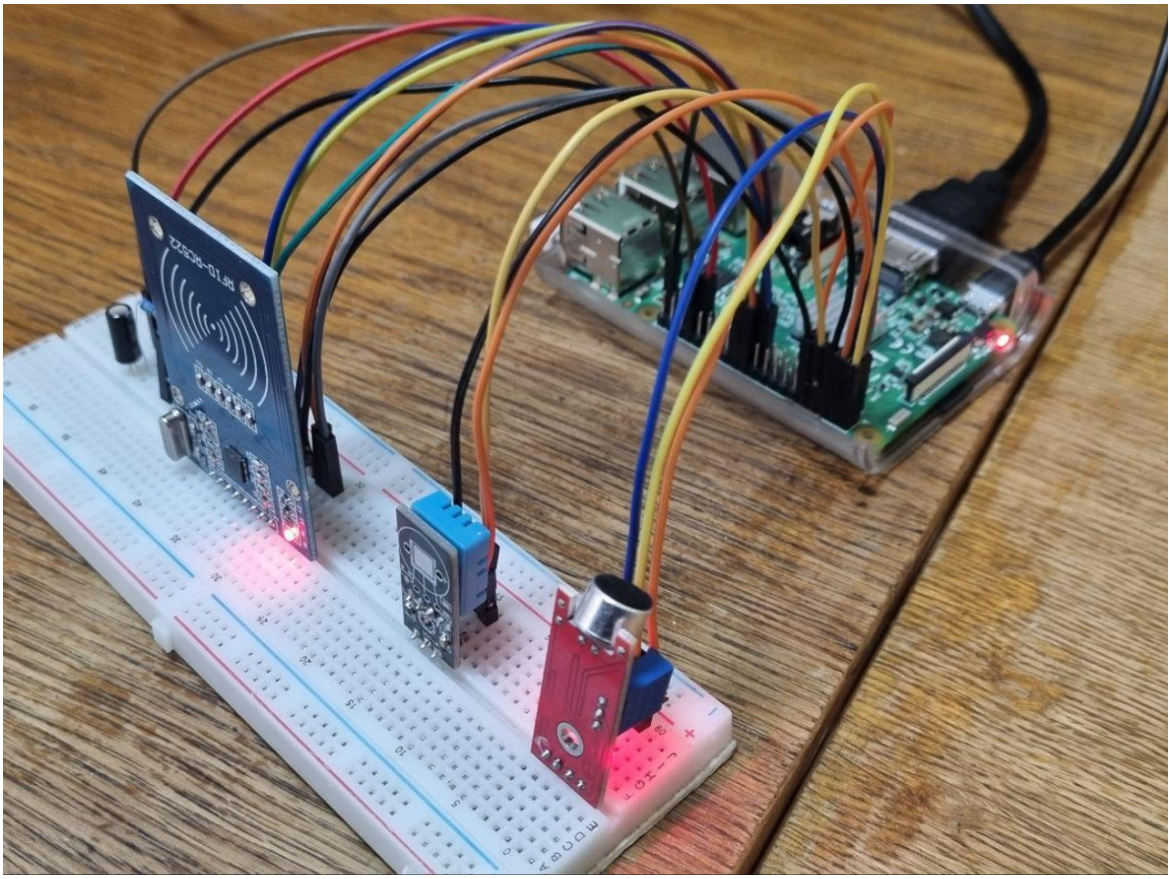

Рисунок 4.10. Монтажна схема підключення датчиків

Тепер коли всі датчики підключено об’єднаємо програмний код для реалізації роботи датчиків паралельно в потоках (для кожного з датчиків створюється окремий потік), і передачі даних в Initial State:

```
import Adafruit_DHT
import threading
import time
from RPi import GPIO
from mfrc522 import SimpleMFRC522
from ISStreamer.Streamer import Streamer

ACCESS_KEY = "ist_NmtG_x-u0IZozSnV_ApZBOcZ7VFzQS8Атут_твій_ACCESS_KEY"
BUCKET_NAME = "Raspberry Pi Live Sensors"
BUCKET_KEY = "FCBMRV5WJJTL"

streamer = Streamer(bucket_name=BUCKET_NAME,
                    bucket_key=BUCKET_KEY,
                    access_key=ACCESS_KEY)

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)

DHT_SENSOR = Adafruit_DHT.DHT11
DHT_PIN = 14

SW_PIN = 15
GPIO.setup(SW_PIN, GPIO.IN)

SOUND_PIN = 2
GPIO.setup(SOUND_PIN, GPIO.IN)

reader = SimpleMFRC522()

def read_dht11():
    print("[DHT11] Запуск зчитування з датчика температури і вологості...")
    while True:
        humidity, temperature = Adafruit_DHT.read_retry(DHT_SENSOR, DHT_PIN)
        if humidity is not None and temperature is not None:
            print(f"[DHT11] Температура: {temperature:.1f}°C | Вологість:
{humidity:.1f}%")
            streamer.log("Temperature", temperature)
            streamer.log("Humidity", humidity)
        else:
            print("[DHT11] Не вдалося зчитати дані.")
        time.sleep(2)

def read_sw520d():
    print("[SW520D] Очікуємо нахил/вібрацію...")
    prev_state = GPIO.input(SW_PIN)
    while True:
```

```
        curr_state = GPIO.input(SW_PIN)
        if curr_state != prev_state:
            vibration_state = "Виявлено" if curr_state == GPIO.LOW else "Немає"
            print(f"[SW520D] {vibration_state} вібрації/нахилу")
            streamer.log("Vibration", vibration_state)
            time.sleep(0.5)
        prev_state = curr_state
        time.sleep(0.05)

def read_lm393():
    print("[LM393] Очікуємо звук (плеск)...")
    while True:
        if GPIO.input(SOUND_PIN) == GPIO.LOW:
            print("[LM393] Виявлено звук!")
            streamer.log("Sound", "Виявлено звук")
            time.sleep(1)

def read_rfid():
    print("[RC522] Система контролю доступу активна...")
    ALLOWED_ID = 535451640595

    while True:
        print("[RC522] Очікую на картку доступу...")
        try:
            id, text = reader.read()
            name = text.strip()

            if id == ALLOWED_ID:
                print(f"[RC522] ДОПУСК ДОЗВОЛЕНО — {name}")
                streamer.log("Access", f"ДОПУСК ДОЗВОЛЕНО — {name}")
                time.sleep(10)
            else:
                print(f"[RC522] ДОПУСК ЗАБОРОНЕНО — ID: {id}")
                streamer.log("Access", "ДОПУСК ЗАБОРОНЕНО")
                time.sleep(5)

        except Exception as e:
            print(f"[RC522] Помилка зчитування: {e}")
            time.sleep(1)


threads = [
    threading.Thread(target=read_dht11),
    threading.Thread(target=read_sw520d),
    threading.Thread(target=read_lm393),
    threading.Thread(target=read_rfid)
]

for thread in threads:
    thread.daemon = True
    thread.start()
```

```
try:
    while True:
        time.sleep(1)
except KeyboardInterrupt:
    print("\n[СИСТЕМА] Завершення програми...")
    GPIO.cleanup()
```

4.4. Візуалізація даних у Initial State

Всі дані отримані з датчиків були відправлені в сховище даних Initial State і це можна побачити на рис. 4.11.

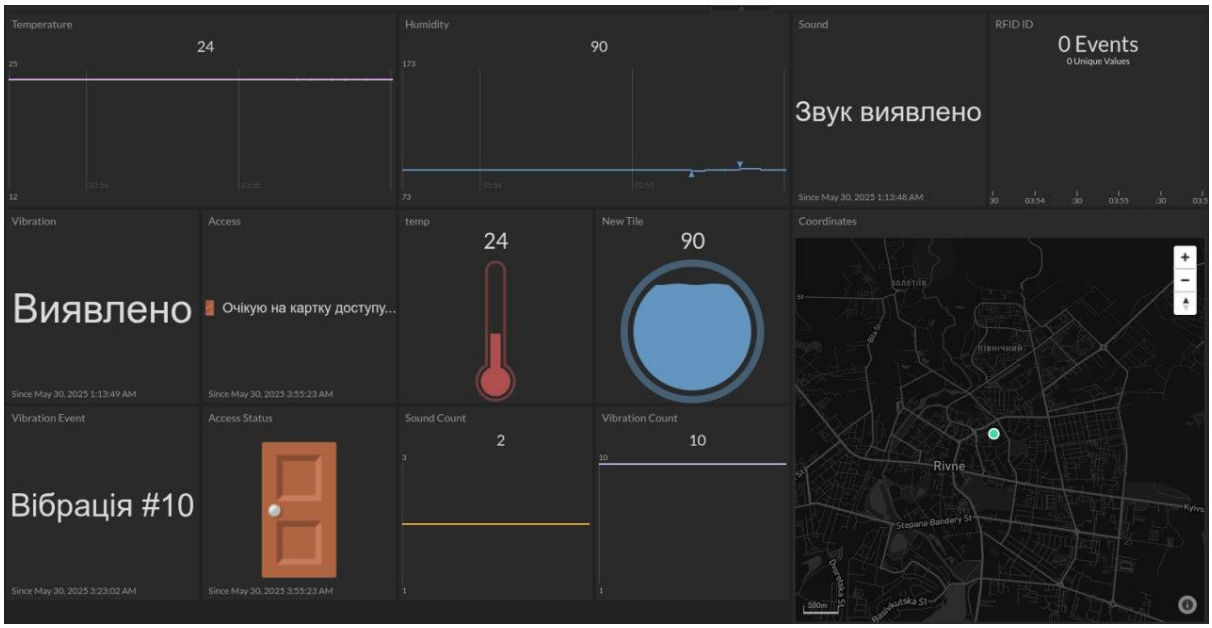

Рисунок 4.11. Вивід даних на дашбордах

Всі дані відображаються правильно, але для гарної візуалізації потрібно відредагувати плитки (рис. 4.12).

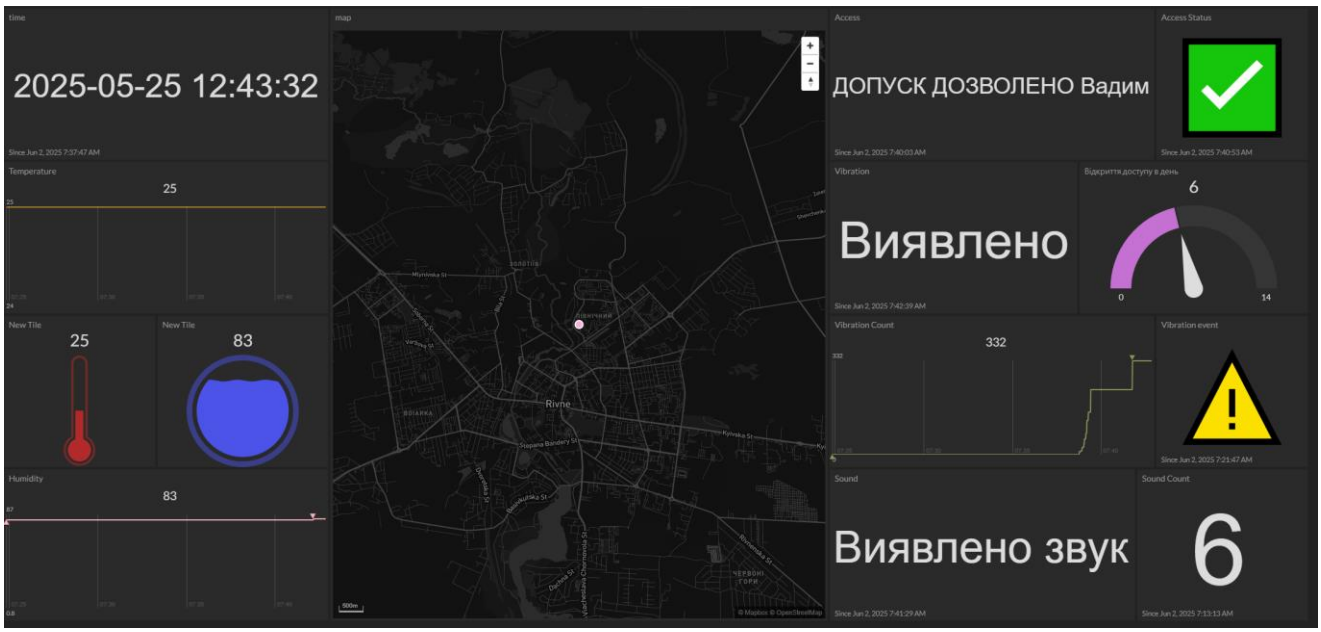

Рисунок 4.12. Відредаговані плитки

Тепер додамо заднє тло і спробуємо інший дизайн (рис. 4.13).

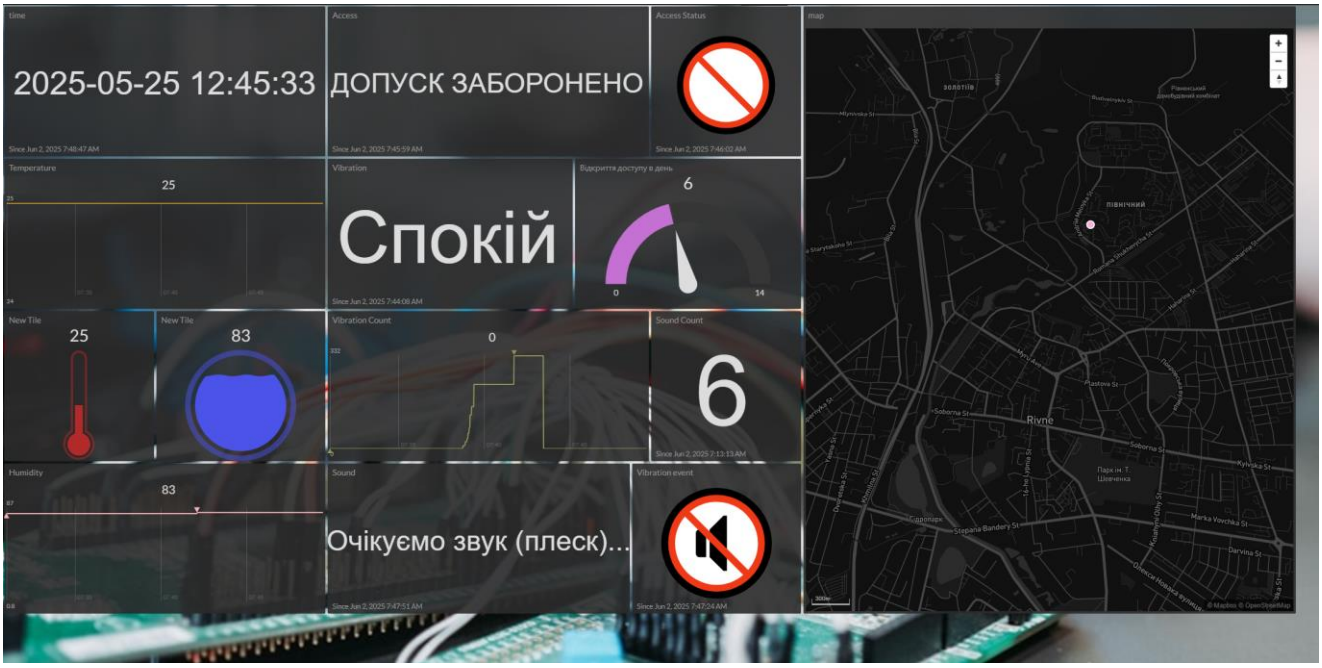

Рисунок 4.13. Змінене тло та інший дизайн

Тепер додаємо публічний доступ і поширимо за посиланням, відкривши з мобільного пристрою, в цьому варіанті інтерфейс буде виглядати так (рис. 4.14).

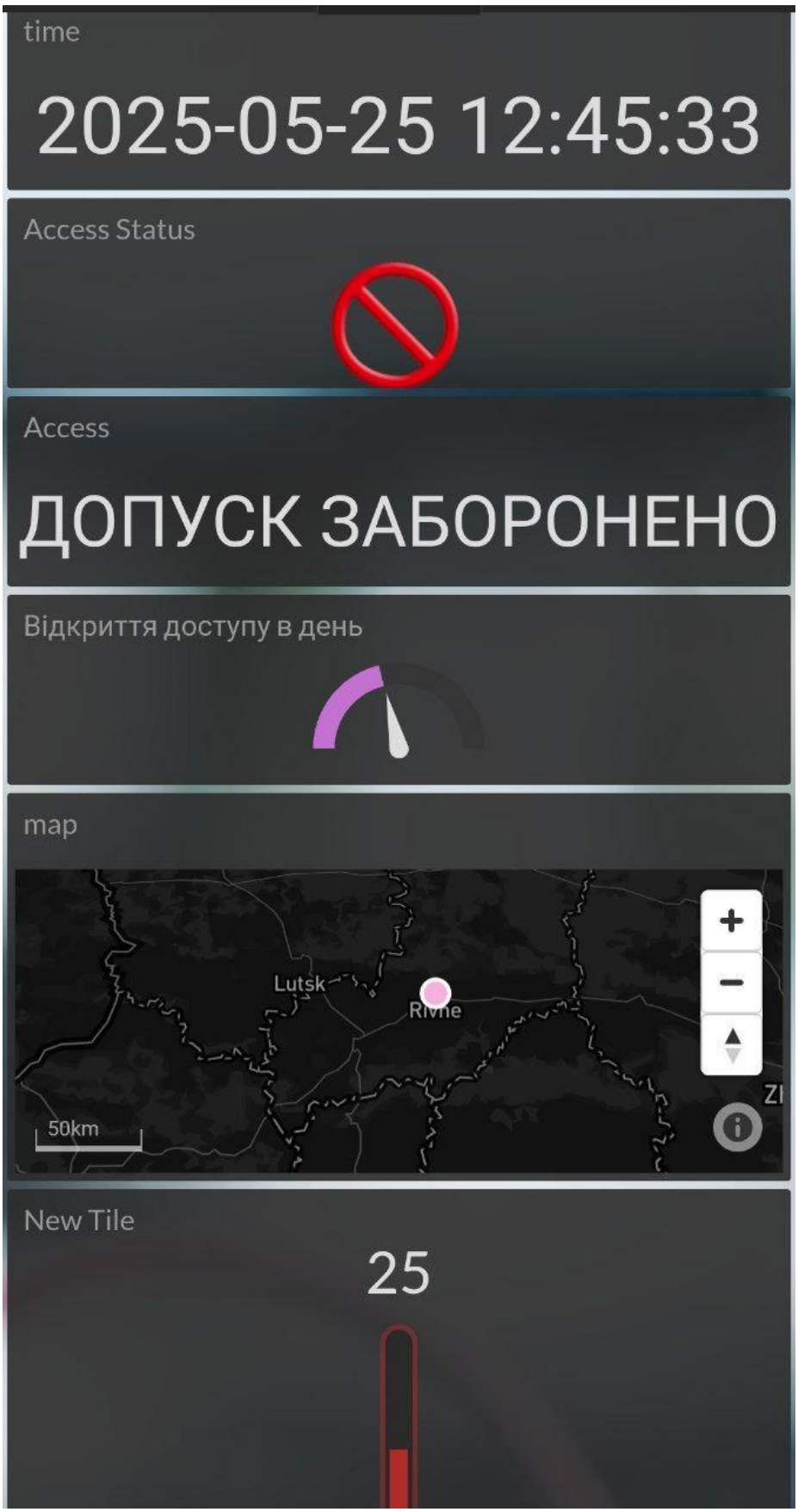

Рисунок 4.14. Вигляд з мобільного пристрою

Скористаємося тригерами і задамо відправлення попереджень коли хтось отримує допуск (рис. 4.15).

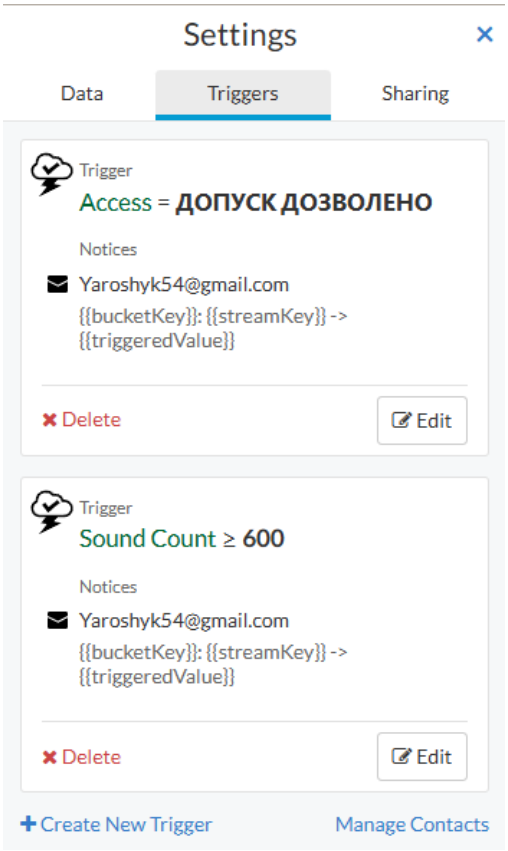

Рисунок 4.15. Задання налаштувань тригерів

Надається повідомлення коли хтось отримав доступ (рис. 4.16).

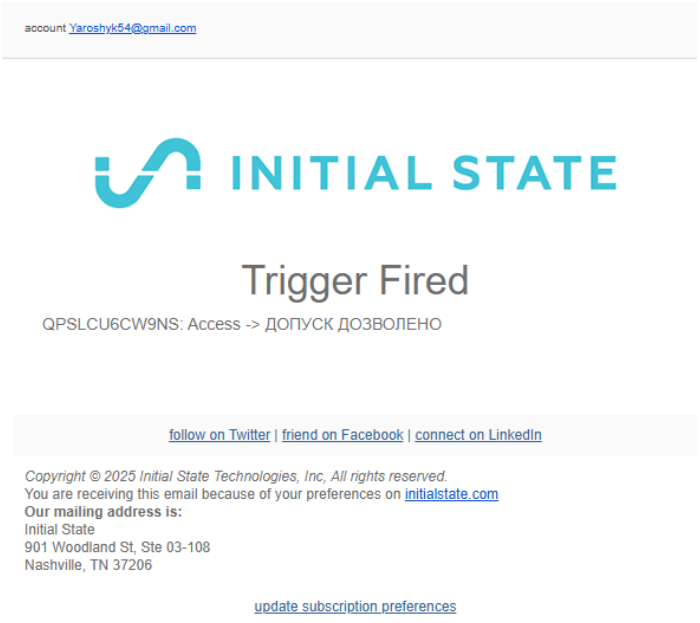

Рисунок 4.16. Сповіщення про отримання доступу

ВИСНОВКИ

Сьогодні роль IoT-технологій стрімко зростає, а потреба у швидкому доступі до даних та їх візуальному представленні стає ключовим фактором ефективної взаємодії між цифровими пристроями, користувачами та бізнес-середовищем. Візуалізація даних є важливим етапом у побудові будь-якої IoT-системи, оскільки вона дозволяє оперативно відслідковувати параметри середовища, виявляти події та приймати рішення на основі зрозумілої, структурованої інформації. У межах цієї бакалаврської роботи було розглянуто платформу Initial State, яка надає інструментальні засоби для збору, обробки та візуалізації IoT-даних у реальному часі. Її переваги полягають у простоті інтеграції, наочності відображення результатів та підтримці REST API. Для реалізації проєкту було використано одноплатний комп’ютер Raspberry Pi 3 Model B разом із сенсорами температури, вологості, вібрації, звуку і RFID-модулем, що дозволило охопити широкий спектр фізичних і логічних подій у навколишньому середовищі. Програмна частина системи реалізована на мові Python, із використанням бібліотек для роботи з сенсорами та API платформи Initial State, що забезпечило надійне та швидке передавання інформації з пристроїв у хмару. Досвід реалізації та тестування системи підтвердив доцільність використання Initial State як інструменту для швидкого прототипування, моніторингу та візуалізації даних у навчальних, дослідницьких або прикладних умовах. У контексті загального розвитку цифрових технологій, хмарних сервісів та автоматизованого моніторингу, подібні рішення є актуальними та можуть бути основою для побудови складніших систем: зворотного зв’язку, інтелектуального аналізу даних або дистанційного управління.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Жураковський Б.Ю., Зенів І.О. Технології інтернету речей. Навчальний посібник: навч. посіб. Для студ. спеціальності 126 «Інформаційні системи та технології», спеціалізація «Інформаційне забезпечення робототехнічних систем» Київ: КПІ ім. Ігоря Сікорського, 2021. 271 с.

2. Contributors to Wikimedia projects. Internet of things - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Internet_of_things (дата звернення: 27.11.2024).

3. Про історію створення інтернету речей. Searchip. URL: https://searchip.in.ua/articles/38-pro-stor-yu-stvorennja-nternetu-rechei.html (дата звернення: 28.11.2024).

4. Пулеко І. В. Єфіменко А. А. Архітектура та технології Інтернету речей: навч. посіб. / І.В. Пулеко, А.А. Єфіменко. Електронні дані. Житомир: Державний університет «Житомирська політехніка», 2022. 234 с.

5. Фатенок-Ткачук, Алла Олександрівна; ЯНУШ, Роман Ігорович; ГУЗЬО, Максим Степанович. Візуалізація облікових даних для прийняття ефективних управлінських рішень. Економіка та суспільство, 2024, 62 с.

6. How to Use Data Visualization for Effective Decision Making. True Chart. URL: https://www.truechart.com/ data-visualization-for-decisions/ (дата звернення: 05.12.2024).

7. Initial State – IoT Platform for Data Visualizations. URL: https://www.initialstate.com (дата звернення: 16.12.2025).

8. Одноплатний комп’ютер – Вікіпедія. URL: https://en.wikipedia.org/wiki/Single-board_computer (дата звернення: 04.03.2025).

9. Raspberry Pi 3 Model B [Електронний ресурс] / Raspberry Pi Foundation. – 2022. – URL: https://www.raspberrypi.com/products/raspberry-pi-3-model-b/ (дата звернення: 10.03.2025).

10. Огляд Raspberry Pi 3 Model B [Електронний ресурс] / DiyLab. – 2024. – Режим доступу: https://diylab.com.ua/p260057854-raspberry-model-ghz.html (дата звернення: 05.04.2025).

11. Технічні характеристики Raspberry Pi 3 Model B [Електронний ресурс] / Mini-Tech. – 2023. – Режим доступу: https://www.mini-tech.com.ua/raspberry-pi-3 (дата звернення: 07.04.2025).

12. Мікрокомп’ютер Raspberry Pi 3 Model B+ [Електронний ресурс] / evo.net.ua – 2023. – URL: https://evo.net.ua/raspberry-pi-3-model-b/?srsltid=AfmBOoqjBKogvikjf6tblOwjHRjtO18TO4GckmQAKfOiB-83QxSRybvS (дата звернення: 10.04.2025).

13. Raspberry Pi – Що це таке? URL: https://blog.avislab.com/raspberry-pi-install/ (дата звернення: 01.06.2025).

14. GPIO Raspberry Pi: посібник [Електронний ресурс] / Raspberry Pi Documentation. – 2025. – Режим доступу: https://www.raspberrypi.com/documentation/computers/os.html (дата звернення: 12.04.2025).

15. DHT11 Sensor Pinout, Features, Equivalents & Datasheet – Components101. – 2021. URL: https://components101.com/sensors/dht11-temperature-sensor (дата звернення: 15.04.2025).

16. LM393 Sound Detection Sensor Module Pinout, Features, Circuit & Datasheet - Components101. – 2020. URL: https://components101.com/modules/lm393-sound-detection-sensor-module (дата звернення: 16.04.2025).

17. RC522 RFID Module Pinout, Features, Specs & How to Use – C omponents101. – 2019. URL: https://components101.com/wireless/rc522-rfid-module (дата звернення: 17.04.2025).

18. Датчик нахилу і вібрації SW-520D. URL: https://arduinokit.com.ua/ua/p1592103645-datchik-naklona-vibratsii.html?srsltid=AfmBOorw44R30VjVLpFrvGxqA7pNkvRm0FtPuLpXtGxCb-KO4HNyn5oh (дата звернення: 018.04.2025).

ДОДАТОК А

ВИКОРИСТАННЯ ПЛАТФОРМИ INITIAL STATE ДЛЯ ВІЗУАЛІЗАЦІЇ ДАНИХ ІНТЕРНЕТУ РЕЧЕЙ У РЕАЛЬНОМУ ЧАСІ

Ярошик В.Ю., здобувач ступеня вищої освіти «бакалавр»
Шинкарчук Н.В., кандидат технічних наук, доцент кафедри інформаційних технологій та моделювання
Рівненський державний гуманітарний університет

Інтернет речей (IoT) – це сучасна технологія, яка дозволяє фізичним пристроям автоматично обмінюватися даними через мережу. Основні компоненти IoT включають фізичні пристрої з датчиками, програмне забезпечення для обробки даних та мережеві технології, що забезпечують зв'язок між пристроями [1]. Візуалізація таких даних у режимі реального часу має важливе значення для моніторингу, аналізу та прийняття рішень.

Initial State – це хмарна платформа, спеціально розроблена для збору, обробки та візуалізації даних з різноманітних джерел, зокрема пристроїв Інтернету речей. Initial State дозволяє будувати візуальні інформаційні панелі, які динамічно відображають показники з датчиків, наприклад: температуру, вологість, освітленість, місцезнаходження тощо (рис. 1). Переваги платформи включають простоту інтеграції, доступність, підтримку зміни показників в реальному часі та наочність інтерфейсу [2].

Рис.1 Демонстраційний проєкт використання платформи Initial State для відстеження датчиків автомобіля

Ключові можливості Initial State:

- *інформаційна панель та візуалізація.* Initial State надає широкий спектр інструментів для створення інтерактивних інформаційних панелей, що візуалізують зібрані дані. Користувач може налаштувати вигляд графіків, вибирати з різних типів відображення, таких як лінійні графіки, гістограми, теплові карти тощо;
- *налаштування тригерів*. Initial State підтримує налаштування тригерів та автоматичних сповіщень, наприклад, відправку SMS або електронних листів, коли параметри даних досягають певних значень;
- *налаштовування доступу.* Initial State дозволяє надавати доступ до інформаційних панелей по посиланню, публічно або для окремих акаунтів;
- *збір даних та інтеграція з іншими платформами та сервісами.* Initial State може збирати дані з різних джерел, включаючи датчики, інші IoT-пристрої, а також API й сервери. Дані можуть передаватися у реальному часі, що робить платформу зручною для моніторингу динамічних процесів. Initial State інтегрується з такими платформами як AWS IoT і Raspberry Pi, що дозволяє використовувати її у комплексі з іншими сучасними технологіями;
- *підтримка різних протоколів зв'язку.* Такі протоколи як MQTT та HTTPS, широко використовуються в IoT. Це дозволяє легко інтегрувати Initial State з іншими пристроями.

Отже, Initial State демонструє значний потенціал як ефективний засіб для організації IoT-візуалізації в реальному часі. На цій платформі можна змоделювати передачу параметрів транспортного засобу та створити наочну інформаційну панель. Це показує цінність платформи як надійного інструмента збору, обробки, аналізу і візуалізації даних Інтернету речей у режимі реального часу.

Список використаних джерел:

1. Інтернет речей – Вікіпедія. URL: https://uk.wikipedia.org/wiki/Інтернет_речей (дата звернення: 22.04.2025).
2. Initial State – IoT Platform for Data Visualizations. URL: https://www.initialstate.com (дата звернення: 24.04.2025).