МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет імені Петра Могили

Факультет комп'ютерних наук

Кафедра комп'ютерної інженерії

ДОПУЩЕНО ДО ЗАХИСТУ

Завідувачка кафедри,
д-р техн. наук, проф.

_____Ірина ЖУРАВСЬКА

« __ » _____ 202__ р.

КВАЛІФІКАЦІЙНА РОБОТА

НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ БАКАЛАВРА

IoT-модуль для підрахування людей у бомбосховищі

Спеціальність 123 Комп'ютерна інженерія

Освітня програма «Комп'ютерна інженерія»

Здобувач _____Дмитро СЛІПЕНЬКИЙ
підпис

«__» _____ 202__ р.

Керівник завідувачка кафедри,
д-р техн. наук, проф. _____Ірина ЖУРАВСЬКА
підпис

«__» _____ 202__ р.

Миколаїв – 2025

Факультет	Комп’ютерних наук
Кафедра	Комп’ютерної інженерії
Рівень вищої освіти	Перший (бакалаврський)
Освітній ступень	Бакалавр
Спеціальність	123 Комп’ютерна інженерія
Освітня програма	Комп’ютерна інженерія

ЗАТВЕРДЖУЮ
Завідувач кафедри комп’ютерної інженерії
_____ Ірина ЖУРАВСЬКА
«_____» _____ 2024 р.

ЗАВДАННЯ
на кваліфікаційну роботу здобувача

_____Сліпенького Дмитра Віталійовича_____
(прізвище, ім’я, по батькові здобувача)

1. Тема кваліфікаційної роботи
_IoT-модуль для підрахування людей у бомбосховищі_____
_____

Затверджена наказом по ЧНУ ім. Петра Могили від 30.10.2024 № 297.

2. Строк представлення кваліфікаційної роботи «_20_» __червня__ 2025 р.

3. Очікуваний результат роботи та початкові дані, якщо такі потрібні
_Очікуваним результатом роботи є: апаратне та програмне забезпечення системи IoT-модуля для підрахування людей у бомбосховищі. Вхідними даними роботи є специфікація вимог, що описує характеристики зазначеного апаратного та програмного забезпечення_____

4. Перелік питань, що підлягають розробці:
1) аналітичний огляд існуючих систем підрахунку кількості людей у закритих приміщеннях з використанням технологій Bluetooth, Wi-Fi, інфрачервоних сенсорів та відеоаналітики;
2) аналіз переваг і недоліків різних методів виявлення присутності людей в умовах бомбосховищ;
3) розробка апаратної частини модуля підрахунку людей на базі одноплатного комп’ютера Raspberry Pi Zero 2 W;
4) розробка програмної частини комплексу, що включає Bluetooth-сканування, обробку сигналів та передачу даних на вебінтерфейс;
5) створення інтерфейсу виведення інформації та його адаптація для перегляду через локальну Wi-Fi-мережу;

6) моделювання та виготовлення корпусу пристрою на 3D-принтері, а також тестування роботи модуля в умовах реального укриття.

5. Перелік графічних матеріалів
_Презентація_____
_____
_____

6. Консультанти:

Консультант	Кафедра (організація)	Частина роботи

Керівник роботи _____ _Ірина ЖУРАВСЬКА_
Особистий підпис *Власне ім'я ПРІЗВИЩЕ*

Здобувач _____ _Дмитро СЛІПЕНЬКИЙ_
Особистий підпис *Власне ім'я ПРІЗВИЩЕ*

Дата видачі завдання «_____» _____ 2024 р.

КАЛЕНДАРНИЙ ПЛАН

виконання кваліфікаційної бакалаврської роботи

Тема: IoT-модуль для підрахування людей у бомбосховищі

№	Найменування роботи	Початок	Закінчення	Примітки
1	Розробка та затвердження завдання на виконання КБР	11.12.2024	12.12.2024	Виконано
2	Огляд літератури за темою роботи	15.01.2025	18.05.2025	Виконано
3	Складання календарного плану КБР	19.02.2025	06.03.2025	Виконано
4	Аналіз предметної області	26.02.2025	15.03.2025	Виконано
5	Розробка проєктних рішень	16.03.2025	17.04.2025	Виконано
6	Моделювання та конструювання АПЗ	18.04.2025	03.05.2025	Виконано
7	Перевірка працездатності, тестування та апробація розробленого АПЗ, аналіз результатів тестування	04.05.2025	15.05.2025	Виконано
8	Відгук керівника КБР	13.06.2025	14.06.2025	Виконано
9	Оформлення КБР та презентації	01.05.2025	03.06.2025	Виконано
10	Попередній захист	21.05.2025	04.06.2025	Виконано
11	Рецензування	09.06.2025	13.06.2025	Виконано
12	Завершення оформлення КБР та презентації	14.06.2025	20.06.2025	Виконано
13	Захист кваліфікаційної бакалаврської роботи	25.06.2025	25.06.2025	Виконано

Керівник роботи _____ Ірина ЖУРАВСЬКА

Особистий підпис *Власне ім'я ПРІЗВИЩЕ*

Здобувач _____ Дмитро СЛІПЕНЬКИЙ

Особистий підпис *Власне ім'я ПРІЗВИЩЕ*

«____» _____ 20__ р.

АНОТАЦІЯ

до кваліфікаційної бакалаврської роботи
«IoT-модуль для підрахування людей у бомбосховищі»
Студент 405 гр.: Сліпенький Дмитро Віталійович
Керівник: д-р техн. наук, проф. Журавська Ірина Миколаївна

У межах цієї роботи пропонується розробка апаратно-програмного модуля для автоматичного підрахунку людей у приміщеннях захисного типу (бомбосховищах) з використанням Raspberry Pi. Такий модуль має на меті забезпечити оперативний та конфіденційний моніторинг присутності людей під час надзвичайних ситуацій, коли важливо знати точну кількість осіб в укритті для ефективної організації допомоги, логістики або евакуації. Система базується на аналізі бездротових сигналів від мобільних пристроїв (Bluetooth), що дозволяє уникнути використання камер відеоспостереження та зберігати приватність користувачів.

Об’єктом дослідження є технології моніторингу присутності людей у закритих приміщеннях.

Предметом дослідження виступає апаратно-програмний IoT-модуль підрахунку кількості людей на базі одноплатного комп’ютера Raspberry Pi Zero 2 W з використанням бездротових інтерфейсів.

Мета роботи – розробка IoT-модуля на базі Raspberry Pi для автоматичного підрахунку людей у бомбосховищі на основі аналізу бездротових сигналів (Bluetooth/Wi-Fi).

Практична значимість роботи полягає в тому, що розроблений комплекс може бути застосований у будь-якому укритті, де є потреба в автоматизованому обліку людей без додаткової інфраструктури. Рішення не потребує використання відеоспостереження або біометрії, що важливо з міркувань етики та приватності. Крім того, система легко масштабується, є недорогою у виробництві та обслуговуванні, і може бути адаптована до різних умов використання.

У процесі виконання дипломної роботи було здійснено аналітичний огляд сучасних методів підрахунку людей: відеоаналітика, інфрачервоні сенсори, BLE/Wi-Fi-аналіз. Визначено переваги та недоліки кожного підходу в умовах бомбосховищ. Розроблено апаратну частину комплексу на основі Raspberry Pi Zero 2 W з використанням внутрішнього Bluetooth-модуля, що дозволяє ідентифікувати пристрої поблизу за їх MAC-адресами. Створено програмне забезпечення, що виконує сканування, обробку, підрахунок пристроїв та збереження даних. Для зручного відображення інформації реалізовано вебінтерфейс, доступний через локальну Wi-Fi-мережу. Було також змодельовано корпус пристрою у програмі 3D-моделювання та надруковано на 3D-принтері. Для стабільної роботи додано систему охолодження. Завантаження прошивки реалізовано так, щоб пристрій запускався автоматично після увімкнення живлення.

Пояснювальна записка містить 72 сторінки (без додатків), 27 рис., 21 джерело посилання та 2 додатки.

Ключові слова: Raspberry Pi Zero 2 W, підрахунок людей, бомбосховище, Bluetooth, IoT, вебінтерфейс, MAC-адреса, автономний модуль.

ABSTRACT

of the Bachelor's Thesis

"IoT module for counting people in a bomb shelter"

Student: Dmytro Slipenkyi

Supervisor: DrSc (Techn.), Prof. Iryna Zhuravska

Within the framework of this work, the development of a hardware and software module for automatic counting of people in protective-type premises (bomb shelters) using Raspberry Pi is proposed. Such a module is intended to provide operational and confidential monitoring of the presence of people during emergency situations, when it is important to know the exact number of people in the shelter for the effective organization of assistance, logistics or evacuation. The system is based on the analysis of wireless signals from mobile devices (Bluetooth), which allows avoiding the use of video surveillance cameras and preserving the privacy of users.

The object of the study is the technology of monitoring the presence of people in closed rooms.

The subject of the study is a hardware and software IoT-module for counting the number of people based on the Raspberry Pi Zero 2 W single-board computer using wireless interfaces.

The purpose of the work is the development of an IoT module based on Raspberry Pi for automatic counting of people in a bomb shelter based on analysis of wireless signals (Bluetooth/Wi-Fi).

The practical significance of the work lies in the fact that the developed complex can be used in any shelter where there is a need for automated counting of people without additional infrastructure. The solution does not require the use of video surveillance or biometrics, which is important for ethical and privacy reasons. In addition, the system is easily scalable, inexpensive to manufacture and maintain, and can be adapted to various conditions of use.

In the process of completing the thesis, an analytical review of modern methods of counting people was carried out: video analytics, infrared sensors, BLE/Wi-Fi analysis. The advantages and disadvantages of each approach in bomb shelter conditions were determined. The hardware part of the complex was developed based on the Raspberry Pi Zero 2 W using an internal Bluetooth module, which allows identifying nearby devices by their MAC addresses. Software was created that performs scanning, processing, counting devices, and saving data. For convenient display of information, a web interface was implemented, accessible via a local Wi-Fi network. The device body was also modeled in a 3D modeling program and printed on a 3D printer. A cooling system was added for stable operation. The firmware download was implemented so that the device started automatically after powering on.

The explanatory note contains 72 pages (without appendices), 27 figures, 21 references and 2 appendices.

Keywords: Raspberry Pi Zero 2 W, people counting, bomb shelter, Bluetooth, IoT, web interface, MAC address, autonomous module.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

АПЗ – апаратно-програмне забезпечення

ІЧ – інфрачервоний

КБР – кваліфікаційна бакалаврська робота

ОС – операційна система

ПЗ – програмне забезпечення

IoT – Internet of Things (укр. Інтернет речей)

MAC – Media Access Control (укр. контроль доступу до медіа)

MQTT – Message Queue Telemetry Transport (укр. спрощений мережевий протокол)

PIR – Passive Infrared Sensor (укр. пасивний інфрачервоний датчик)

QoS – Quality of Service (укр. якість обслуговування)

SBC – Single Board Computer (укр. одноплатний комп’ютер)

TCP – Transmission Control Protocol (укр. протокол управління передачею)

ВСТУП

У сучасних умовах, коли безпекова ситуація в Україні залишається нестабільною через воєнну агресію російської федерації, значну увагу приділяють організації та удосконаленню систем цивільного захисту населення. Одним з ключових елементів такої системи є бомбосховища – захисні споруди, призначені для укриття людей під час повітряної тривоги, артобстрілів чи інших надзвичайних ситуацій. Проте ефективне функціонування таких укриттів потребує не лише фізичної готовності приміщення, а й наявності технічних рішень, що дозволяють забезпечити облік та контроль за кількістю людей у реальному часі.

В умовах обмеженого простору, складних евакуаційних маршрутів і потреби в раціональному розподілі ресурсів (запасів води, їжі, медикаментів) знання кількості людей, що перебувають у бомбосховищі, набуває вирішального значення. Така інформація дозволяє приймати зважені управлінські рішення у критичних ситуаціях, своєчасно залучати додаткові сили й засоби для допомоги, а також зменшує ризики дезорганізації.

Традиційні методи обліку, наприклад ручний підрахунок або списки зареєстрованих осіб, є неефективними в умовах стресу та швидкого реагування. Крім того, вони не дозволяють оперативно оновлювати інформацію у разі динамічної зміни кількості присутніх. У цьому контексті на допомогу приходять сучасні цифрові технології, зокрема концепція Інтернету речей (IoT), яка передбачає використання мікрокомп’ютерів та бездротових комунікацій для автоматичного збору й аналізу даних.

Завдяки розвитку одноплатних комп’ютерів, таких як Raspberry Pi, відкриваються нові можливості для створення компактних, недорогих та автономних рішень для моніторингу. Одним із підходів є виявлення присутності людей за допомогою фіксації сигналів їхніх мобільних пристроїв (смартфонів, планшетів, ноутбуків), які періодично передають ідентифікаційні сигнали через Bluetooth. Це дає змогу без вторгнення в особистий простір

користувача отримувати інформацію про присутність у приміщенні без встановлення камер або інших систем відеоспостереження.

У межах цієї роботи пропонується розробка апаратно-програмного ІоТ-модуля для автоматичного підрахунку людей у приміщеннях захисного типу (бомбосховищах) з використанням Raspberry Pi. Модуль дозволяє здійснювати моніторинг кількості пристроїв, що випромінюють бездротові сигнали, фіксувати та обробляти ці дані в реальному часі, а також надавати доступ до результатів через інтерфейс користувача.

Мета роботи: розробка ІоТ-модуля на базі Raspberry Pi для автоматичного підрахунку людей у бомбосховищі на основі аналізу бездротових сигналів (Bluetooth/Wi-Fi).

Об'єкт дослідження: процес моніторингу присутності людей у приміщеннях.

Предмет дослідження: апаратно-програмний ІоТ-модуль виявлення присутності людей у бомбосховищі на основі Raspberry Pi та бездротових технологій.

Завдання дослідження:

– проаналізувати існуючі методи підрахунку людей у приміщеннях;

– обрати апаратну платформу та засоби збору даних;

– реалізувати програмну частину для виявлення та підрахунку пристроїв;

– забезпечити відображення зібраної інформації у зручному вигляді.

Практичне значення роботи: Розроблений ІоТ-модуль може бути використаний у будь-якому укритті або закритому приміщенні для оперативного підрахунку кількості людей без встановлення складних або дорогих систем відеоспостереження. Це сприятиме кращій організації цивільного захисту, покращенню логістики в надзвичайних ситуаціях.

1 АНАЛІТИЧНИЙ ОГЛЯД ПРОБЛЕМИ ТА РІШЕНЬ

Російсько-українська війна, що розпочалась у лютому 2022 року, висунула перед українським суспільством низку нових викликів, зокрема щодо забезпечення ефективного захисту цивільного населення в умовах бойових дій. У зв’язку з цим на законодавчому рівні було посилено вимоги до облаштування захисних споруд, зокрема бомбосховищ, які відтепер повинні бути обов’язковим елементом при проєктуванні нових об’єктів та реконструкції існуючих будівель.

Однак аналіз поточного стану укриттів в Україні свідчить про те, що значна частина з них не відповідає сучасним вимогам. Часто бомбосховища не мають евакуаційних виходів, системи водопостачання та електропостачання, зон для тривалого перебування людей чи навіть базових умов для збереження фізичного та психічного здоров’я у разі довготривалого укриття.

На тлі цих проблем особливо актуальним стає питання обліку кількості осіб, які перебувають у захисних спорудах. Відсутність такого контролю може ускладнити евакуаційні заходи, розподіл ресурсів, а також роботу служб цивільного захисту. Саме тому необхідною є розробка сучасних технологічних засобів, які дозволять автоматизовано здійснювати підрахунок присутніх осіб без порушення їх приватності.

Одним із перспективних рішень є створення IoT-модуля на базі одноплатного комп’ютера, зокрема Raspberry Pi, який дозволить здійснювати облік людей у бомбосховищі шляхом фіксації сигналів від їхніх мобільних пристроїв. Такий підхід сприятиме підвищенню рівня безпеки та ефективності організації укриттів у воєнний час.

Перед створенням власного IoT-модуля варто дослідити існуючі рішення для поставленої мети.

1.1 Огляд сучасних рішень для відстеження кількості людей у бомбосховищі

Сучасні технології моніторингу присутності людей активно застосовуються у сферах безпеки, обліку персоналу, логістики та організації заходів. У контексті цивільного захисту, зокрема в умовах війни, ці рішення набувають нового змісту – вони мають забезпечити швидкий, точний та автономний облік людей, що перебувають у захисних спорудах.

Системи відеоспостереження, що працюють на основі комп’ютерного зору та алгоритмів штучного інтелекту, на сьогодні є одними з найточніших засобів підрахунку людей у закритих приміщеннях. Їх принцип роботи полягає в аналізі відеопотоку з камер спостереження у режимі реального часу. Завдяки спеціальному програмному забезпеченню або вбудованим алгоритмам розпізнавання, система здатна виявляти фігури людей, розпізнавати їх напрямок руху (вхід чи вихід), формувати лічильники та статистику щодо присутності.

Подібні системи забезпечують високу точність результатів, особливо в умовах доброго освітлення. Крім того, вони дозволяють не лише рахувати кількість осіб, а й забезпечують візуалізацію – можливість перегляду відеозаписів, а також інтеграцію з іншими системами безпеки, наприклад, для сповіщення про перевищення максимально допустимої кількості людей у приміщенні. Проте відеоаналітика має й свої недоліки. По-перше, вона порушує конфіденційність – у бомбосховищах, де перебувають цивільні особи, далеко не кожен погодиться бути під відеонаглядом. По-друге, вартість обладнання є доволі високою: якісні камери, програмне забезпечення та мережеве устаткування вимагають значних фінансових витрат. Також варто враховувати залежність роботи таких систем від умов освітлення та стабільного електро- і мережевого живлення, що не завжди є доступним у бомбосховищах.

Серед популярних рішень, що реалізують підрахунок людей за допомогою відеоаналізу, варто виділити кілька моделей. Наприклад,

Hikvision DS - 2CD6825G0/C-IVS – це спеціалізована IP-камера із двома об’єктивами та вбудованим процесором, яка дозволяє точно рахувати кількість осіб при вході та виході з приміщення (рис. 1.1). Камера може вести статистику та передавати дані до центрів обробки інформації.

Рисунок 1.1 – Камера Hikvision для підрахунку людей у приміщенні [20]

У контексті використання таких систем у бомбосховищах слід підкреслити, що хоча технологія й забезпечує високу точність, у більшості випадків вона може бути непридатною або надмірною. Відсутність постійного живлення, обмеження в інтернет-доступі, а також необхідність збереження приватності громадян змушують шукати більш автономні та конфіденційні рішення для підрахунку людей. Утім, для модернізованих укриттів із належним рівнем інфраструктури подібні камери можуть розглядатися як частина комплексної системи моніторингу.

Ще одним напрямом, що набуває популярності для визначення кількості людей у приміщенні, є використання бездротових технологій, зокрема Wi-Fi та Bluetooth. Суть таких систем полягає в аналізі радіосигналів, які випромінюють мобільні пристрої: смартфони, планшети, розумні годинники тощо. Кожен із цих пристроїв періодично передає сигнали, які можуть бути виявлені спеціальними скануючими модулями.

У ролі такого модуля може виступати одноплатний комп’ютер (наприклад, Raspberry Pi), оснащений відповідним програмним забезпеченням

та адаптерами бездротового зв’язку. Сканер реєструє MAC-адреси пристроїв поблизу, після чого програмно відфільтровує дублікати, виконує агрегацію даних та виводить оцінку кількості унікальних пристроїв, що перебувають у зоні дії.

Цей підхід має низку переваг. По-перше, він не потребує встановлення камер або інших засобів візуального спостереження, що дозволяє зберегти приватність людей. По-друге, апаратна реалізація є відносно дешевою – для створення подібного лічильника достатньо модуля на зразок Raspberry Pi Zero 2 W з активованим Bluetooth/WiFi-моніторингом. Додатковим плюсом є автономність таких пристроїв: вони можуть працювати від акумуляторів або PowerBank, що особливо важливо в умовах відсутності постійного електропостачання в укриттях.

Серед апаратних рішень, які вже реалізують подібний підрахунок, можна згадати ESP32-Paxcounter – мініатюрний модуль на базі ESP32, який може рахувати пристрої з активним Bluetooth або Wi-Fi (рис. 1.2).

Рисунок 1.2 – ESP32-Paxcounter [3]

Однак, технологія має й обмеження. Оцінка кількості людей базується на наявності в них увімкнених гаджетів із активованими модулями передачі. Відтак, користувач без телефона або з відключеним Bluetooth залишиться непоміченим. До того ж, деякі сучасні мобільні ОС маскують MAC-адреси або змінюють їх із часом, що може ускладнити точність підрахунку.

Попри це, використання бездротових технологій для обрахунку людей у бомбосховищах є практичним і технічно досяжним рішенням. Особливо в умовах, коли важливо зберегти конфіденційність, забезпечити автономність та адаптувати пристрій до будь-якого укриття без складного монтажу. Надалі такі системи можуть масштабуватись або інтегруватись у більші інформаційні системи цивільного захисту.

Іншим поширеним методом визначення кількості людей у приміщенні є використання інфрачервоних (ІЧ) сенсорів і датчиків руху. Ці пристрої працюють за принципом виявлення теплового випромінювання, яке генерується людським тілом. Коли людина проходить повз датчик або перетинає певну зону, сенсор реєструє зміну температури у полі зору та фіксує подію входу чи виходу.

Найчастіше для таких задач використовують пасивні інфрачервоні сенсори (PIR-сенсори), які мають низьке енергоспоживання, просту конструкцію та високий рівень надійності. Для точного підрахунку людей зазвичай встановлюють два або більше сенсори в коридорі або при вході до приміщення – це дозволяє не лише виявити рух, але й визначити напрямок (вхід чи вихід). На основі цього ведеться балансування кількості людей у приміщенні.

До прикладу, популярний лічильник відвідувачів ТК-02 (рис. 1.3). Цей пристрій оснащений інфрачервоними сенсорами для точного підрахунку людей, що входять та виходять з приміщення. Його компактний корпус дозволяє легко інтегрувати його в різні середовища.

Такий підхід також має свої переваги: він не залежить від наявності мобільних пристроїв у користувачів, не потребує підключення до інтернету, а також може функціонувати в умовах поганого зв’язку чи повної автономності. Це особливо ціно для бомбосховищ, де іноді немає стабільного зв’язку або електропостачання.

Рисунок 1.3 – Лічильник відвідувачів ТК-02, інфрачервоний сенсорний пристрій для підрахунку людей у приміщенні [19]

Однак у порівнянні з бездротовими методами цей підхід має обмежену зону дії й може бути менш точним у випадках, коли кілька людей входять чи виходять одночасно. Також PIR-сенсори потребують правильного монтажу та калібрування для мінімізації помилкових спрацювань.

Незважаючи на це, ІЧ-технології залишаються ефективним, простим і доступним способом підрахунку людей у приміщеннях, особливо в умовах обмежених ресурсів, як це часто буває в укриттях.

Отже, провівши аналітичний огляд наявних рішень для підрахунку кількості людей у приміщеннях, зокрема у бомбосховищах, було встановлено, що на ринку представлено низку пристроїв, здатних виявляти присутність людей за допомогою відеоспостереження, інфрачервоних сенсорів або бездротових технологій. Проте більшість із них вирішують лише окремі завдання, не забезпечуючи повноцінного, автономного та конфіденційного підрахунку у реальному часі. Частина рішень потребує постійного підключення до електромережі або має обмежені можливості в умовах нестабільного зв’язку. Інші – надмірно дорогі чи складні у впровадженні. Таким чином, не було виявлено комплексного пристрою, який би одночасно задовольняв вимоги щодо автономності, конфіденційності, точності підрахунку та можливості інтеграції у системи цивільного захисту. Це підтверджує актуальність розробки власного апаратно-програмного рішення у вигляді ІоТ-модуля на базі одноплатного комп’ютера Raspberry Pi.

1.2 Вибір апаратних компонентів для розроблення IoT-модуля

Для реалізації системи автоматичного підрахунку кількості людей у приміщенні бомбосховища необхідно підібрати апаратну платформу, яка буде відповідати ряду вимог: мала енергоспоживання, підтримка бездротових інтерфейсів Bluetooth, можливість автономної роботи, компактність, стабільність у тривалому використанні та доступність в умовах українського ринку.

Центральним елементом системи виступає одноплатний комп’ютер (англ. Single Board Computer, SBC). Саме він відповідає за виявлення сигналів мобільних пристроїв, їхню обробку, підрахунок, а також передачу інформації або її збереження. Серед доступних і доцільних варіантів для такого завдання було проаналізовано кілька моделей, зокрема:

Raspberry Pi Zero 2 W (рис. 1.4) – один із найбільш популярних мініатюрних комп’ютерів у світі. Він має вбудовані модулі Wi-Fi 802.11n та Bluetooth 4.1, що дозволяє йому виявляти як WiFi-пакети (probe request), так і Bluetooth-випромінювання від смартфонів. Пристрій характеризується дуже низьким енергоспоживанням – менш як 0,5 Вт у середньому режимі роботи. До його переваг також відноситься велика спільнота, широка підтримка бібліотек, офіційна ОС Raspberry Pi OS та численні приклади реалізації подібних проєктів.

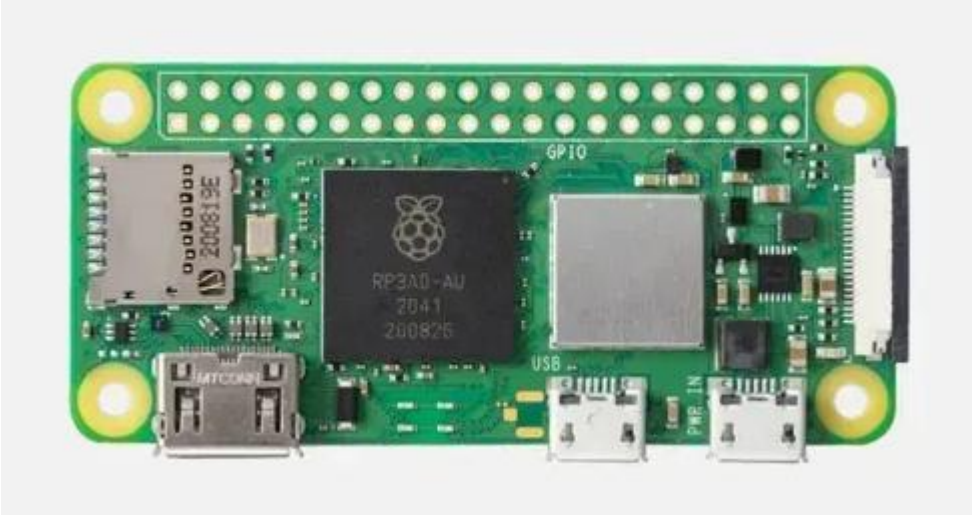

Рисунок 1.4 – Мікрокомп'ютер Raspberry Pi Zero 2 W [13]

Banana Pi M2 Zero (рис. 1.5) – є функціональним аналогом Raspberry Pi Zero 2 W, але базується на чипі Allwinner H2+, що забезпечує дещо вищу обчислювальну потужність. Пристрій має WiFi-модуль, однак Bluetooth може бути реалізований тільки зовнішнім адаптером через USB, що ускладнює реалізацію системи. Незважаючи на це, він має привабливу ціну та є доступним для придбання в Україні.

Рисунок 1.5 – Мікрокомп'ютер Banana Pi M2 Zero [2]

Onion Omega2 Plus (рис. 1.6) – мікрокомп’ютер, орієнтований на застосування в ІоТ-проєктах. Його переваги – низьке енергоспоживання, мініатюрні розміри та простота у використанні завдяки прошивці OpenWRT. Omega2 Plus має лише WiFi-модуль, тому підрахунок можливий виключно на основі WiFi-сигналів. Однак цей пристрій не підтримує Bluetooth, що дещо звужує його функціональність у порівнянні з Raspberry Pi.

Рисунок 1.6 – Мікрокомп'ютер Onion Omega2 Plus [12]

Усі перелічені варіанти підтримують роботу з Linux-подібними операційними системами (ОС), що дозволяє реалізовувати кастомні рішення, встановлювати необхідне програмне забезпечення та автоматизувати процеси аналізу даних.

Додатково до основного обчислювального модуля необхідно підібрати допоміжні компоненти. По-перше, це джерело живлення – автономний акумулятор, наприклад, Power Bank з виходом 5В 2А, або акумуляторна батарея з контролером заряду. Така система дозволить забезпечити кілька годин або навіть діб автономної роботи. По-друге, для збереження даних необхідно використовувати microSD-карту (мінімум 8 Гбайт) або, за потреби, USB-накопичувач. Також, якщо передбачено віддалену передачу даних, доцільно передбачити модуль GSM/4G (наприклад, SIM800L або Quectel EC25), що дозволить надсилати інформацію на сервер у випадках, коли немає стабільного доступу до Wi-Fi.

Варто також передбачити використання захисного корпусу для обчислювального модуля – як із міркувань безпеки електроніки, так і для зручності монтажу. Існує велика кількість готових пластикових та металевих корпусів для Raspberry Pi Zero 2 W, які забезпечують вентиляцію та захист пристрою від пилу й вологи.

Усі компоненти разом утворюють компактний та універсальний ІоТ-модуль, який може бути легко інтегрований у систему цивільного захисту, встановлений у бомбосховище та працювати в автоматичному режимі без постійного втручання людини. При правильному налаштуванні система може працювати тижнями без обслуговування, надсилаючи дані про кількість людей у приміщенні або зберігаючи їх локально.

У процесі порівняння декількох одноплатних комп’ютерів для реалізації системи підрахунку людей у бомбосховищі було визначено ключові критерії вибору: наявність бездротових інтерфейсів (Wi-Fi і Bluetooth), низьке енергоспоживання, компактність, доступність на ринку, а також наявність підтримки з боку спільноти розробників. Розглядалися моделі Raspberry Pi

Zero 2 W, Banana Pi M2 Zero та Onion Omega2 Plus, які потенційно могли б бути основою для створення IoT-модуля для підрахування людей у бомбосховищі.

У результаті технічного аналізу перевагу було надано Raspberry Pi Zero 2 W. Основна причина – наявність вбудованих модулів Wi-Fi та Bluetooth, що дозволяє здійснювати виявлення сигналів як по каналах Wi-Fi (через аналіз probe-запитів від мобільних пристроїв), так і по Bluetooth (BLE beacon’ів або просто активних пристроїв). Це розширює точність обліку та зменшує ймовірність помилкових пропусків, оскільки не всі користувачі мають постійно активовані Wi-Fi або Bluetooth – але один із цих інтерфейсів, як правило, увімкнений у більшості смартфонів.

Також Raspberry Pi Zero 2 W має компактний розмір, що дозволяє легко монтувати його у будь-якому місці приміщення бомбосховища, навіть за обмеженого простору. Його низьке енергоспоживання (менше 0.5 Вт у середньому режимі роботи) робить його придатним для тривалого автономного використання з живленням від PowerBank або акумуляторної батареї.

Ще однією вагомою перевагою Raspberry Pi Zero 2 W є велика спільнота розробників, наявність офіційної підтримки ОС Raspberry Pi OS, доступність бібліотек для роботи з бездротовими інтерфейсами, збору та обробки даних, а також інструментів автоматизації.

Інші моделі, зокрема Banana Pi M2 Zero, хоча й мають дещо вищу обчислювальну потужність, не оснащені вбудованим Bluetooth-модулем. Для повної реалізації функціоналу потрібне підключення зовнішнього адаптера, що збільшує споживання енергії та ускладнює конструкцію. Onion Omega2 Plus є чудовим IoT-рішенням з дуже малим споживанням енергії та простим адмініструванням на базі OpenWRT, проте не підтримує Bluetooth, що обмежує можливості виявлення пристроїв лише до Wi-Fi. Враховуючи ці недоліки, обидва варіанти були відкинуті як основна платформа для реалізації проєкту.

Таким чином, остаточним вибором для реалізації апаратної частини IoT-модуля підрахунку людей у бомбосховищі стала платформа Raspberry Pi Zero 2 W, яка забезпечує оптимальне співвідношення функціональності, енергоефективності, розміру та доступності.

1.3 Вибір і структура програмного забезпечення для IoT-модуля

Після визначення апаратної основи системи постає завдання створення програмного забезпечення, яке забезпечить збір, обробку та передачу даних про кількість людей у бомбосховищі. Основною метою програмної частини є реалізація механізму виявлення пристроїв з активованими інтерфейсами Bluetooth або Wi-Fi, обрахунок кількості унікальних пристроїв у певний часовий проміжок, а також передача цієї інформації до вебінтерфейсу або збереження її у локальній базі даних.

В якості ОС було обрано Raspberry Pi OS Lite, яка не містить графічного середовища, що зменшує навантаження на систему, дозволяє швидше завантажуватись і споживати менше пам’яті. Основні програмні компоненти будуть реалізовані за допомогою мови програмування Python, яка має великий набір бібліотек для роботи з бездротовими інтерфейсами та обміну даними.

Для реалізації виявлення пристроїв буде використано такі бібліотеки:

– bluetooth / bluepy або pybluez: для пошуку активних Bluetooth-пристроїв;

– sqlite3: для збереження результатів локально в міні-базу даних;

– requests або MQTT-клієнт: для відправки даних на сервер або вебінтерфейс;

– flask або fastapi: для створення локального вебінтерфейсу перегляду даних (опційно).

Структура програмного забезпечення передбачає наявність таких функціональних блоків:

– модуль збору даних, який періодично сканує навколишнє середовище та ідентифікує унікальні MAC-адреси пристроїв;

– фільтраційний модуль, що відкидає повторні сигнали протягом короткого інтервалу часу та усуває «фоновий шум» (наприклад, системні пристрої або постійно активні передавачі);

– модуль обробки та збереження, який формує звіт про кількість пристроїв у певний момент часу та записує інформацію у локальне сховище;

– модуль передачі даних, що передає результат на вебсервер або локальний сайт;

– модуль відображення (опційний), що дозволяє переглядати результати в режимі реального часу через браузер.

Оскільки система передбачає роботу в умовах нестабільного зв’язку або обмеженого доступу до Інтернету, передбачено реалізацію буферизації даних. Усі зібрані дані зберігаються локально, і при наявності з’єднання з мережею – синхронізуються з віддаленим сервером.

Загалом, програмне забезпечення має бути легким, стабільним, здатним працювати у фоновому режимі без втручання користувача, з можливістю автоматичного запуску після перезавантаження системи.

Висновки до розділу 1

У ході аналітичного огляду технологій підрахунку людей у приміщеннях, зокрема в умовах бомбосховищ, було розглянуто основні технічні підходи: відеоаналітику, інфрачервоні сенсори та методи на основі бездротових сигналів Bluetooth. Кожна з технологій має свої переваги та обмеження. Відеоспостереження забезпечує високу точність, але має високу вартість, потребує стабільного живлення та викликає занепокоєння щодо конфіденційності. Інфрачервоні сенсори є простими у впровадженні, але менш точні при високій інтенсивності трафіку. Безпровідні технології, зокрема аналіз Bluetooth-сигналів, демонструють оптимальне співвідношення автономності, конфіденційності, точності та вартості.

На основі порівняльного аналізу одноплатних комп’ютерів було обґрунтовано вибір Raspberry Pi Zero 2 W як базової апаратної платформи для

побудови системи підрахунку людей у бомбосховищі. Пристрій має вбудовані модулі бездротового зв’язку, низьке енергоспоживання, доступність на ринку та широку підтримку спільноти розробників. У поєднанні з відповідним програмним забезпеченням та автономним живленням Raspberry Pi Zero 2 W забезпечить ефективну, гнучку й масштабовану систему, що здатна працювати в реальному часі, враховуючи специфічні умови укриттів у період воєнного стану.

Таким чином, обґрунтовано доцільність розробки IoT-модуля на базі Raspberry Pi для автономного, недорогого та конфіденційного підрахунку людей у захисних спорудах.

2 АЛГОРИТМИ, МЕТОДИ ТА ЗАСОБИ РОЗРОБКИ КОМПЛЕКСУ ВІДСТЕЖЕННЯ КІЛЬКОСТІ ЛЮДЕЙ НА БАЗІ ІОТ-МОДУЛІВ

Щоб забезпечити ефективну роботу системи визначення кількості людей на основі IoT-модулів, важливо організувати надійну та швидку передачу даних від усіх модулів. Для цього слід розглянути використання повнодуплексних протоколів зв’язку, які дозволяють одночасну передачу та прийом інформації, а також звернути увагу на протоколи, пристосовані до роботи в умовах обмеженого або нестабільного зв’язку.

2.1 Алгоритм роботи протоколу WebSocket

WebSocket – це сучасний мережевий протокол, який забезпечує постійний двосторонній (повнодуплексний) обмін даними між клієнтом і сервером через одне TCP-з’єднання. Цей протокол був офіційно затверджений організацією IETF у 2011 році у вигляді стандарту RFC 6455. Його особливістю є здатність передавати дані в режимі реального часу без потреби у повторних запитах, що характерно для класичного HTTP-протоколу.

На відміну від традиційної моделі «запит-відповідь», де кожен запит створює нове підключення, WebSocket дозволяє підтримувати з’єднання відкритим після його встановлення. Завдяки цьому сервер може надсилати повідомлення клієнту, щойно з’являються нові дані, без необхідності періодичного опитування. Це значно покращує швидкодію системи й оптимізує використання мережевих і обчислювальних ресурсів.

Основні компоненти WebSocket-архітектури включають:

– клієнтську сторону, яка ініціює запит до сервера;

– шлюз WebSocket, який забезпечує канал обміну даними між клієнтом і сервером;

– сервер, що реагує на запити і відправляє повідомлення в реальному часі.

Основне призначення WebSocket – надати альтернативу HTTP, яка дозволяє більш ефективну взаємодію на основі TCP. Початкове з’єднання встановлюється через HTTP, використовуючи спеціальні заголовки (наприклад, Upgrade), після чого передача даних відбувається через протокол WebSocket. Замість стандартної адреси типу *http://*, тут використовується схема *ws://* або *wss://* (для безпечного з’єднання).

Переваги WebSocket над традиційними технологіями:

а) значне зменшення навантаження на мережу за рахунок усунення надмірних HTTP-заголовків — передаються лише ключові дані, що сприяє реальному часу взаємодії;

б) реалізація повнодуплексного каналу, що дає змогу обом сторонам обмінюватися даними в будь-який момент без додаткових запитів, усуваючи затримки;

в) робота через одне TCP-з’єднання, що мінімізує використання ресурсів;

г) підтримка постійного з’єднання, що робить можливим потокову передачу даних;

д) подолання обмежень, які притаманні застарілим рішенням, зокрема:

1) опитування: створює непотрібне навантаження, адже клієнт постійно звертається до сервера, навіть коли немає нових даних;

2) HTTP-потоки: хоча з’єднання не розривається, передача інформації все одно не є ефективною через зайві заголовки;

3) AJAX: дає змогу оновлювати частини сторінки, але працює в напівдуплексному режимі, потребує декількох TCP-з’єднань і створює додаткове навантаження на сервер.

Процес встановлення WebSocket-з’єднання передбачає так зване «рукостискання», яке виконується за допомогою спеціальних HTTP-заголовків. Для цього клієнт надсилає запит з такими заголовками, як:

– Connection: Upgrade;
– Upgrade: websocket;

– Sec-WebSocket-Key: випадковий рядок у форматі Base64;

– Sec-WebSocket-Version: вказує використовувану версію протоколу.

Це рукостискання є запитом типу HTTP GET. У відповідь сервер надсилає статус 101 Switching Protocols та додає заголовок Sec-WebSocket-Accept, який містить обчислений на основі отриманого ключа хеш — підтвердження того, що з’єднання дозволено.

Рисунок 2.1 – Діаграма взаємодії в протоколі WebSocket [5]

Відповідно до специфікації IETF RFC 6455, протокол WebSocket підтримує декілька форматів передавання інформації у вигляді кадрів. Насправді, кожен кадр є послідовністю бітів, структура яких визначається рядом полів (рис. 2.2). Основні елементи цієї структури такі:

а) final (1 біт): цей біт встановлюється в значення «1», щоб позначити, що даний кадр є завершальним у поточній послідовності передавання. Іншими словами, він сигналізує про кінець фрагментації повідомлення;

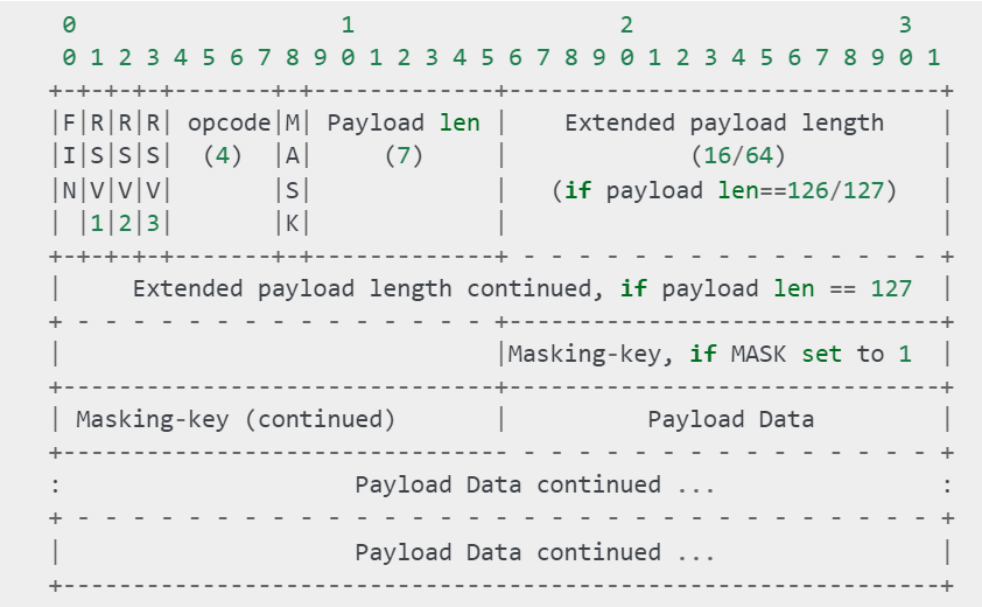

Рисунок 2.2 – Кадр даних у протоколі WebSocket [1]

б) Opcode – це поле довжиною 4 біти, яке вказує, який тип даних міститься в кадрі. Кожне значення має конкретне призначення:

– 0x0: використовується для фрагментів продовження повідомлення;

– 0x1: позначає текстовий формат даних;

– 0x2: вказує на бінарний тип інформації;

– 0x3 – 0x7: зарезервовані для майбутніх нестандартних застосувань;

– 0x8: застосовується для ініціації завершення з’єднання;

– 0x9: спеціальний кадр Ping, який використовується для перевірки активності;

– 0xA: кадр Pong – відповідь на Ping;

– 0xB – 0xF: зарезервовані для нових можливих контрольних операцій;

в) маска (Mask bit) – один біт, що сигналізує, чи маскуються передані дані. Якщо значення біта дорівнює «1», то дані корисного навантаження обов’язково маскуються і включають спеціальний ключ. За специфікацією, усі

дані, які надсилає клієнт на сервер, повинні проходити через процес маскування з міркувань безпеки та захисту від атак на рівні протоколу;

г) поле довжини навантаження (Payload Length) – залежно від розміру даних, поле може мати змінну довжину. Якщо обсяг інформації не перевищує 125 байт, то для її опису використовується лише 7 бітів. Якщо кількість байтів дорівнює 126, то додається ще 16 бітів, а у випадку, коли значення дорівнює 127, до структури кадру включаються додаткові 64 біти для точного опису довжини;

д) маскувальний ключ (Masking Key) – якщо біт маски активований, це поле містить 32 біти (4 байти), які використовуються для зворотного перетворення зашифрованих (маскованих) даних на стороні отримувача. Саме завдяки цьому ключу клієнтські дані можуть бути правильно прочитані сервером після демаскування;

е) навантаження (Payload Data) – це основний вміст кадру, який поділяється на дві складові:

1) додаткові дані (Extension Data) – не є обов’язковими, проте можуть містити інформацію про спосіб обробки даних, наприклад, тип стиснення чи шифрування. Це дозволяє оптимізувати передачу, особливо при великому обсязі інформації;

2) дані застосунку (Application Data) – це власне зміст повідомлення, яке передається між клієнтом і сервером. Воно може містити будь-яку інформацію, визначену логікою програми – від простих текстових повідомлень до складних двійкових структур.

Формат кадру WebSocket є гнучким та добре продуманим механізмом для ефективного обміну даними в реальному часі. Завдяки ретельно визначеним полям, як-от ознаки завершення передачі (final), типізація повідомлень (opcode), маскування даних, варіативна довжина навантаження, а також можливість включення допоміжних та прикладних даних, протокол демонструє високий рівень адаптивності до широкого спектра мережевих умов та сценаріїв використання.

Механізм маскування даних особливо актуальний для клієнтських запитів – він не лише виконує захисну функцію, а й знижує ризик атак, пов’язаних із передачею повторюваних шаблонів. Гнучка система обробки довжини навантаження дозволяє передавати як малі пакети, так і великі обсяги даних без суттєвих втрат у продуктивності. Також передбачена підтримка розширень, яка відкриває можливості для використання додаткових механізмів стиснення або шифрування без зміни основної логіки протоколу.

Завдяки повнодуплексному каналу зв’язку, відкритому з моменту рукостискання й до моменту явного завершення, WebSocket усуває необхідність у періодичному опитуванні сервера, що було типовим для попередніх вебтехнологій, як-от AJAX або HTTP polling. Це дозволяє зменшити затримки, оптимізувати споживання ресурсів як на клієнтській, так і на серверній стороні, а також мінімізувати навантаження на мережу.

Таким чином, WebSocket і його структура кадру дозволяють створювати високопродуктивні, масштабовані рішення, здатні обробляти велику кількість підключень у реальному часі з мінімальними витратами ресурсів. Протокол чудово підходить для інтерактивних додатків, таких як чат-системи, онлайн-ігри, біржові платформи, системи моніторингу IoT та будь-які сервіси, що вимагають швидкого реагування і постійного потоку даних. Його ефективна побудова дозволяє підтримувати стабільне з’єднання навіть в умовах нестабільного або обмеженого інтернету.

2.2 Алгоритм роботи протоколу MQTT

У сучасному світі кількість пристроїв, які підключаються до мережі Інтернет, щоденно невпинно зростає. Проте, не всі ці пристрої потребують високошвидкісного або стабільного з’єднання. У багатьох випадках, зокрема в умовах обмеженого доступу до мережі або при передачі невеликих обсягів даних, таке з’єднання навіть неможливе чи недоцільне. Саме для таких сценаріїв було розроблено окремий клас технологій, призначених для пристроїв з низькими ресурсами – так званий кластер рішень Internet of Things

(Інтернет речей), що орієнтується на передачу невеликих обсягів даних за мінімальних вимог до каналу зв’язку.

Щоб забезпечити ефективну комунікацію в таких обмежених умовах, ще на початку 2000-х років був створений спеціалізований протокол під назвою MQTT (Message Queuing Telemetry Transport). Його метою було забезпечення стабільного, легковагового та надійного обміну повідомленнями у середовищах з низькою пропускною здатністю, нестабільним зв’язком або значною затримкою. MQTT став особливо цінним у системах, де спілкування між машинами (M2M) має бути простим, швидким і маловитратним.

На відміну від класичної архітектури «клієнт-сервер», де пристрої взаємодіють напряму, MQTT базується на моделі публікації та підписки (pub/sub). Ця концепція дозволяє мінімізувати навантаження на мережу, уникати постійних запитів до кожного пристрою та зменшити енергоспоживання, що критично важливо для ІоТ-пристроїв на акумуляторах.

У даній моделі:

а) видавець (publisher) – пристрій, який генерує та надсилає дані;

б) підписник (subscriber) – пристрій або сервіс, який підписується на певні теми (topics) і отримує лише релевантну інформацію;

в) брокер (broker) – центральна ланка, що посередницькі передає повідомлення між публікаторами та підписниками.

Цей посередник повністю ізолює видавця від отримувача, дозволяючи їм працювати незалежно. Видавець не знає, хто отримає його повідомлення, і не повинен очікувати відповіді. З іншого боку, підписник отримує тільки ті дані, які відповідають його інтересам (зазначеним у темах підписки). Це значно оптимізує роботу мережі, підвищує масштабованість і спрощує адміністрування ІоТ-систем.

MQTT є ідеальним вибором для таких сфер, як розумне освітлення, моніторинг навколишнього середовища, системи безпеки, телеметрія транспорту та промислові датчики, де критичними є низьке споживання

енергії, невеликі об’єми даних та робота в нестабільних мережах (наприклад, через GSM, 3G або LPWAN).

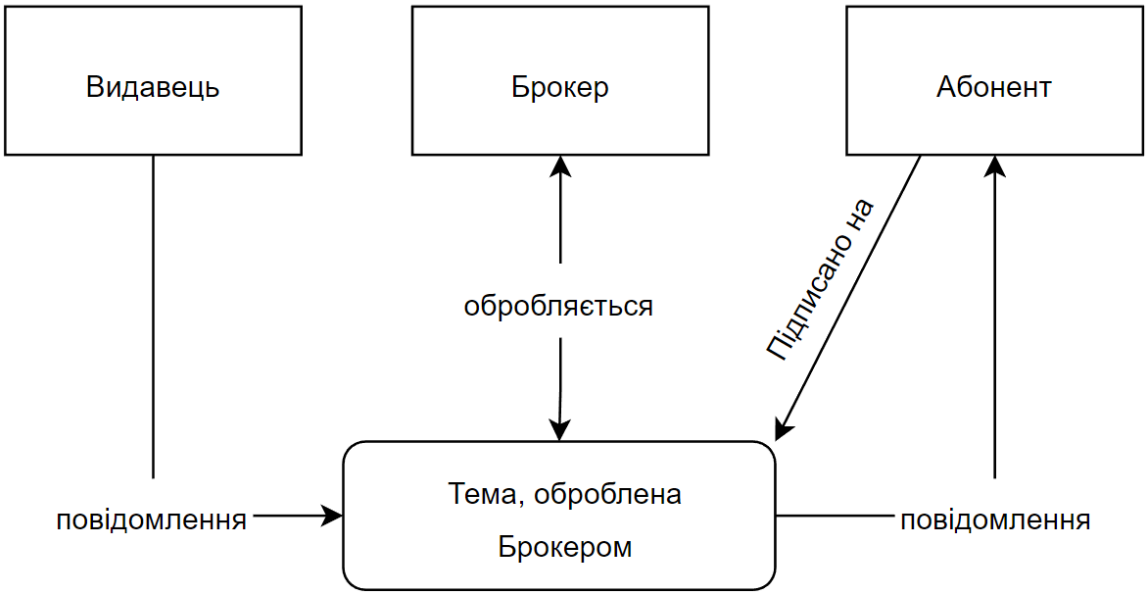

Рисунок 2.3 – Схема взаємодії протоколу MQTT [5]

У протоколі MQTT клієнти поділяються на дві основні категорії: видавці (publishers) та підписники (subscribers). Залежно від призначення пристрою, він може лише передавати повідомлення або лише отримувати їх. Однак нічого не заважає реалізувати обидві функції одночасно в одному клієнті — пристрій може як публікувати дані, так і підписуватись на інформацію від інших.

Коли пристрій має потребу передати інформацію, він виступає в ролі видавця, що надсилає повідомлення на центральний сервер – брокер MQTT. Цей брокер виступає як посередник, через якого реалізується вся комунікація між пристроями. Він виконує ключову функцію маршрутизації повідомлень: після отримання даних від одного клієнта, брокер аналізує тему (topic) повідомлення і надсилає його лише тим підписникам, які зареєстровані на відповідну тему.

Теми (topics) – це логічно впорядковані ієрархічні мітки, які використовуються для класифікації повідомлень. Вони дозволяють підписникам отримувати лише ті повідомлення, що відповідають їх інтересам

або ролі в системі. Фільтрація може здійснюватися як за назвою теми, так і за змістом або типом даних.

Важливо, що клієнтам не потрібно знати про існування один одного. Їхня взаємодія відбувається повністю анонімно — видавець не знає, хто отримає його дані, а підписник не знає, хто їх відправив. Такий підхід спрощує конфігурацію системи, дозволяє масштабувати її без додаткових витрат, і особливо добре підходить для систем з великою кількістю пристроїв. Механізм «один-до-багатьох», який застосовується в MQTT, забезпечує широкомовне розповсюдження повідомлень при мінімальному навантаженні на мережу та ресурси пристроїв.

MQTT також характеризується надзвичайно компактною структурою повідомлень, що оптимізує використання пропускної здатності. Кожне повідомлення включає фіксований заголовок обсягом у 2 байти, до якого за потреби додається змінна частина заголовка та корисне навантаження, яке може сягати до 256 мегабайт. Така структура дозволяє досягати балансу між мінімалізмом і функціональністю.

Ще однією критично важливою особливістю є підтримка трьох рівнів якості обслуговування (англ. Quality of Service, QoS), які визначають, як саме доставляється повідомлення і з якою надійністю:

1) QoS 0 – найпростіший рівень, коли повідомлення надсилається один раз без гарантії доставки. Якщо дані втрачаються по дорозі, вони не передаються повторно. Це найменш надійний, але найшвидший варіант;

2) QoS 1 – гарантує доставку повідомлення принаймні один раз. Брокер повторно надсилає повідомлення, якщо не отримує підтвердження від одержувача;

3) QoS 2 – найбільш надійний рівень, який гарантує доставку повідомлення лише один раз, використовуючи складний чотириетапний протокол підтвердження між клієнтом і брокером.

Рівень QoS 0, зображений на рис. 2.4, підходить для сценаріїв, де втрата одного повідомлення не є критичною. Він не вимагає підтверджень, не

зберігає стан і не виконує повторних спроб передачі – через що його часто називають принципом «надіслав і забув». Це робить його максимально легким для реалізації, швидким і ресурсозберігаючим.

Рисунок 2.4 – Схема роботи QoS 0 [5]

Рівень якості обслуговування QoS 1 забезпечує доставку повідомлення одержувачу щонайменше один раз (рис. 2.5). Відправник зберігає копію повідомлення до отримання підтвердження у вигляді пакета PUBACK від отримувача, що засвідчує успішне прийняття повідомлення. У разі відсутності підтвердження, повідомлення може бути надіслано повторно, тому можливе надходження одного й того ж повідомлення декілька разів.

Рисунок 2.5 – Схема роботи QoS 1 [5]

Відправник кожного пакета використовує унікальний ідентифікатор, який допомагає зіставити повідомлення PUBLISH з відповідним підтвердженням PUBACK. Якщо підтвердження не надходить протягом встановленого часу, відправник повторно надсилає пакет PUBLISH. Коли

одержувач отримує повідомлення з рівнем QoS 1, він може одразу його обробити: наприклад, якщо це брокер, він передає дані всім підписаним клієнтам, а потім надсилає підтвердження PUBACK. Якщо ж видавець повторно надсилає повідомлення, він позначає його прапором дублікату (DUP). У межах QoS 1 цей прапор служить лише для внутрішнього використання і не впливає на роботу брокера або клієнта, які відправляють підтвердження PUBACK незалежно від його встановлення.

Рівень QoS 2 – це найвищий і найнадійніший рівень якості обслуговування у MQTT (рис. 2.5). Він гарантує, що кожне повідомлення буде доставлено одержувачу лише один раз, уникаючи дублікатів. Цей рівень забезпечується складним чотириступеневим механізмом рукостискання між відправником і отримувачем. Для узгодження доставки використовується унікальний ідентифікатор пакета PUBLISH, який дозволяє сторонам координувати процес передачі і підтвердження повідомлення. Такий підхід робить QoS 2 найбільш надійним, хоч і менш швидким, порівняно з іншими рівнями.

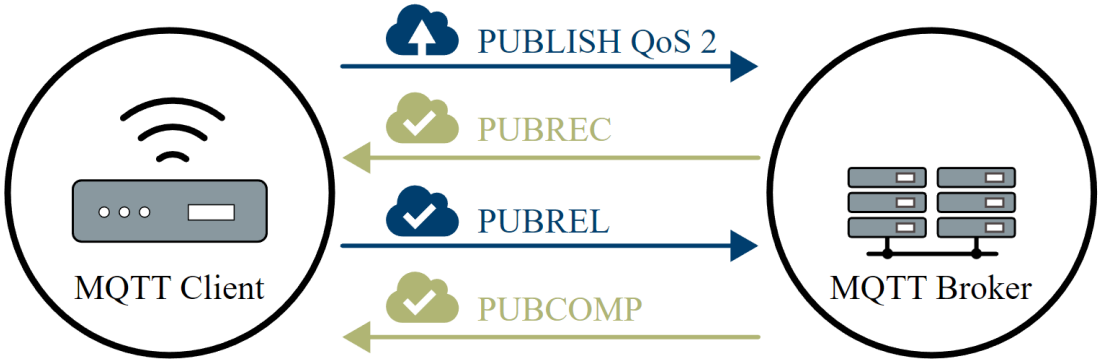

Рисунок 2.5 – Схема роботи QoS 2 [5]

Коли одержувач отримує пакет PUBLISH з рівнем QoS 2 від відправника, він обробляє повідомлення і надсилає у відповідь пакет PUBREC, що підтверджує прийняття PUBLISH. Якщо відправник не отримує підтвердження PUBREC, він повторно надсилає пакет PUBLISH з позначкою дублікату (DUP), доки не отримає підтвердження. Після отримання PUBREC

від одержувача, відправник може безпечно видалити початкове повідомлення PUBLISH, зберігаючи при цьому PUBREC і відправляючи у відповідь пакет PUBREL.

Коли одержувач приймає пакет PUBREL, він звільняє всі збережені дані про цей сеанс і відправляє пакет PUBCOMP, що сигналізує про завершення обробки. Аналогічно, після отримання PUBCOMP від одержувача, відправник може звільнити ідентифікатор пакета PUBLISH для подальшого використання. Цей механізм допомагає запобігти повторній обробці одного й того ж повідомлення.

Завдяки такому чотириступеневому процесу рукостискання, що завершується отриманням PUBCOMP, обидві сторони отримують впевненість у тому, що повідомлення доставлено саме один раз і підтвердження доставки отримано. Якщо будь-який пакет загубиться під час передачі, відправник несе відповідальність за повторну відправку протягом розумного проміжку часу. Це правило застосовується як для клієнтів MQTT, так і для брокерів, які повинні відповідно реагувати на всі командні повідомлення.

2.3 Топологія мережі кінцевого комплексу з IoT-модулів

Перш за все потрібно визначити, яким чином будуть взаємодіяти між собою внутрішні програмні компоненти кінцевої системи на базі IoT-модулів. Оскільки інтерфейс і обчислювальні блоки IoT-модулів реалізовані з використанням різних технологій, а інтерфейс представлено у вигляді вебзастосунку, було прийнято рішення застосувати протокол WebSocket для обміну даними (рис. 2.6). Такий вибір дозволяє практично усунути затримки, які могли б виникати при використанні, наприклад, протоколу HTTP. Крім того, завдяки повнодуплексній природі WebSocket, стає можливою організація двонаправленої комунікації між компонентами.

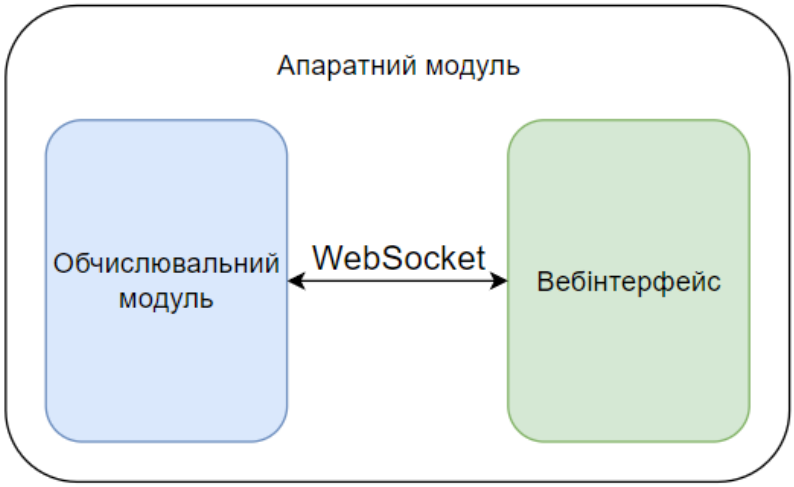

Рисунок 2.6 – Комунікація програмних модулів з вебінтерфейсом диспетчерського центру [5]

Для організації зв’язку між IoT-модулями та вебінтерфейсом диспетчеризації, що відповідає за збір інформації про кількість людей у мережі бомбосховищ, було обрано протокол MQTT, який найкраще підходить для роботи в умовах обмеженого зв’язку. Оскільки для підключення до мережі передбачається використання мобільного зв’язку, який може перериватися в будь-який момент, вибір цього протоколу забезпечує надійність передачі даних навіть при нестабільному зв’язку. Таким чином, у центральній частині всього комплексу має працювати сервер із двома шлюзами (рис. 2.7): MQTT-брокером та WebSocket-шлюзом.

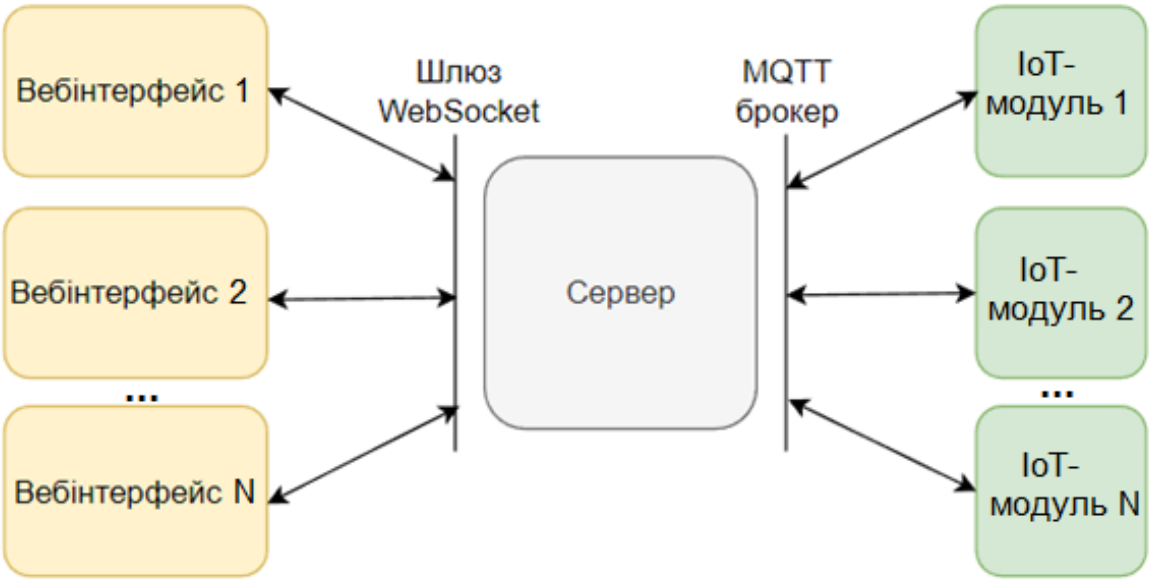

Рисунок 2.7 – Топологія мережі кінцевого комплексу
з сукупності ІоТ-модулів [5]

Далі MQTT-брокер підключає ІоТ-модулі за схемою «один до багатьох». Для отримання актуальної інформації від апаратних модулів у вебінтерфейсі застосовується протокол WebSocket, який також працює за принципом «один до багатьох».

Таким чином, можливо створення комплексу із сукупності ІоТ-модулів, в межах котрого можна узагальнювати дані з сукупності ІоТ-модулів, розміщених в різних бомбосховищах. Такий підхід може допомогти у плануванні роботи підрозділів ДСНС України, бригад екстреної медичної допомоги й т. п.

2.4 Алгоритм функціонування системи моніторингу присутності в бомбосховищі

У межах реалізації системи моніторингу присутності людей у бомбосховищі було розроблено алгоритм роботи ІоТ-модуля, який базується на виявленні Bluetooth-пристроїв (рис. 2.8). Цей ІоТ-модуль дозволяє в реальному часі оцінювати кількість людей у приміщенні на основі кількості

унікальних MAC-адрес, що передаються пристроями користувачів (переважно смартфонами).

Для реалізації даного підходу було використано мікрокомп'ютер Raspberry Pi Zero 2 W, який оснащений вбудованим Bluetooth-модулем. Перевагами такого методу є:

– безконтактне виявлення без встановлення додаткових датчиків;
– швидка обробка та збирання даних;
– можливість інтеграції з вебсервісами та хмарними платформами.

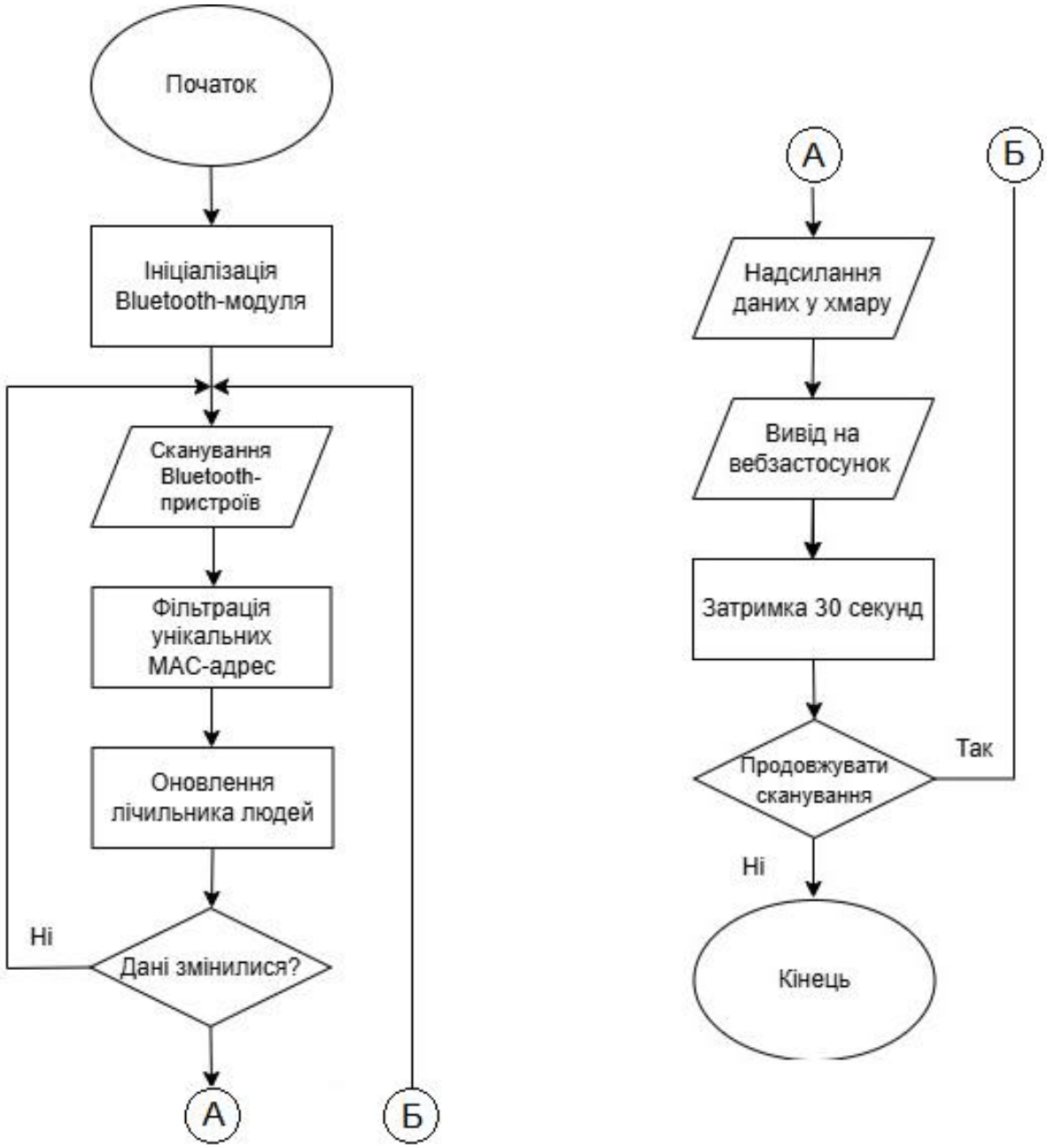

Рисунок 2.8 – Блок-схема алгоритму роботи ІоТ-модуля

Алгоритм роботи системи базується на циклічному скануванні простору приміщення на наявність активних Bluetooth-пристроїв. Ключові етапи цього процесу:

– ініціалізація Bluetooth-модуля забезпечує готовність системи до роботи після запуску;

– сканування пристроїв дозволяє виявити всі активні Bluetooth-пристрої в радіусі дії;

– фільтрація унікальних MAC-адрес виключає повтори та дозволяє ідентифікувати унікальні пристрої;

– оновлення лічильника людей базується на кількості унікальних MAC-адрес;

– умова перевірки змін – якщо кількість змінюється, то відбувається надсилання даних у хмару;

– результати виводяться на вебзастосунок, де їх можна переглядати з будь-якого пристрою;

– затримка між циклами (30 секунд) зменшує навантаження на пристрій і дозволяє оптимізувати енергоспоживання.

Розроблений алгоритм IoT-системи на базі Raspberry Pi Zero W дозволяє ефективно здійснювати підрахунок кількості людей у приміщенні шляхом безконтактного виявлення Bluetooth-пристроїв. Використання унікальних MAC-адрес як ідентифікаторів забезпечує простоту реалізації, гнучкість, а також знижує потребу в дорогих сенсорах. Система працює у циклічному режимі, автоматично фіксуючи зміни, надсилаючи інформацію у хмару та виводячи її у вебінтерфейс, що дозволяє оперативно реагувати на зміни в заповненості бомбосховища. Таким чином, запропонований підхід є практичним, економічним та придатним для використання в умовах надзвичайної ситуації.

Висновки до розділу 2

У цьому розділі було розглянуто основні принципи, технології та алгоритми, що лежать в основі функціонування IoT-модуля для підрахунку людей у бомбосховищі. Основну увагу приділено архітектурі передачі даних та механізмам взаємодії між окремими компонентами системи, зокрема між апаратною частиною на базі Raspberry Pi Zero 2 W, хмарною інфраструктурою та користувацьким вебінтерфейсом.

Було обґрунтовано вибір протоколів MQTT та WebSocket для забезпечення надійного й швидкого обміну інформацією в умовах обмежених ресурсів. MQTT виступає ефективним засобом для публікації даних зі сторони пристрою у хмару, де брокер розподіляє повідомлення всім зацікавленим підписникам. Це дозволяє мінімізувати навантаження на мережу та забезпечити масштабованість системи при розширенні. WebSocket, у свою чергу, забезпечує миттєву двосторонню передачу даних між сервером і вебінтерфейсом, дозволяючи користувачам у реальному часі бачити оновлення без необхідності перезавантаження сторінки чи повторних запитів.

У межах даного розділу було також представлено блок-схему алгоритму роботи IoT-модуля, яка наочно демонструє послідовність дій пристрою: запуск системи, активація Bluetooth-модуля, сканування найближчих пристроїв, збір та фільтрація унікальних MAC-адрес, визначення кількості людей у приміщенні та передача цих даних у хмару. У випадку змін, інформація автоматично оновлюється й передається до користувача. Циклічний характер роботи системи дозволяє виконувати підрахунок у фоновому режимі з заданим інтервалом, наприклад, кожні 30 секунд.

Варто підкреслити, що застосування Bluetooth-технології як основного сенсора дозволяє уникнути потреби у складному або дорогому обладнанні, що робить запропоновану систему доступною для реалізації в умовах обмеженого фінансування або екстреної інсталяції. Водночас, забезпечується достатній рівень точності, якщо врахувати типові сценарії – коли у кожної людини є смартфон або інший Bluetooth-пристрій.

Загалом, поєднання описаних протоколів, апаратних засобів та алгоритмічної логіки дозволяє створити функціональну, гнучку та енергоощадну IoT-систему, призначену для моніторингу присутності людей у захисних спорудах. Така система має реальні перспективи впровадження у громадських та приватних укриттях, надаючи органам цивільного захисту та відповідальним особам актуальну інформацію про кількість людей у реальному часі.

3 РОЗРОБКА IOT-МОДУЛЯ ВІДСТЕЖЕННЯ КІЛЬКОСТІ ЛЮДЕЙ У БОМБОСХОВИЩІ НА БАЗІ RASPBERRY PI ZERO 2 W

Для досягнення поставленої мети необхідно розробити IoT-модуль, що складається з однієї апаратної платформи Raspberry Pi Zero 2 W та програмного забезпечення для збору, обробки й передачі даних. Програмна частина містить два основні модулі: модуль виявлення та підрахунку пристроїв у радіусі дії, а також модуль взаємодії з користувачем, реалізований у вигляді вебінтерфейсу для перегляду статистики та адміністрування пристрою.

3.1 Налаштування апаратної платформи

З обраних в першому розділі апаратних компонентів необхідно зібрати повноцінний IoT-модуля, а також забезпечити його програмним забезпеченням.

3.1.1 Збірка апаратної частини

У процесі створення апаратного модуля для підрахунку людей у приміщенні на базі Raspberry Pi Zero 2 W виникла необхідність реалізувати зручне та універсальне підключення, яке б дозволяло одночасно подавати живлення на пристрій та здійснювати обмін даними з керуючою системою. З урахуванням компактних габаритів плати та обмеженої кількості доступних портів, було прийнято рішення використати сучасний роз’єм USB Type-C як основний інтерфейс підключення.

Проте перед початком будь-яких монтажних робіт важливо провести ретельне вивчення плати з метою визначення оптимальних точок для підключення додаткових компонентів. Зокрема, необхідно ідентифікувати контактні площадки або вільні пінові виходи, до яких можливо безпечно під’єднати живлення для вентилятора, а також реалізувати підключення нового роз’єму USB Type-C. Такий підхід дозволяє уникнути пошкодження плати або конфліктів з існуючими електричними ланцюгами, а також

забезпечує стабільну роботу всіх підключених модулів у межах заданих параметрів. Лише після проведення цієї підготовчої фази можна переходити до етапу паяння та монтажу додаткових елементів (рис. 3.1).

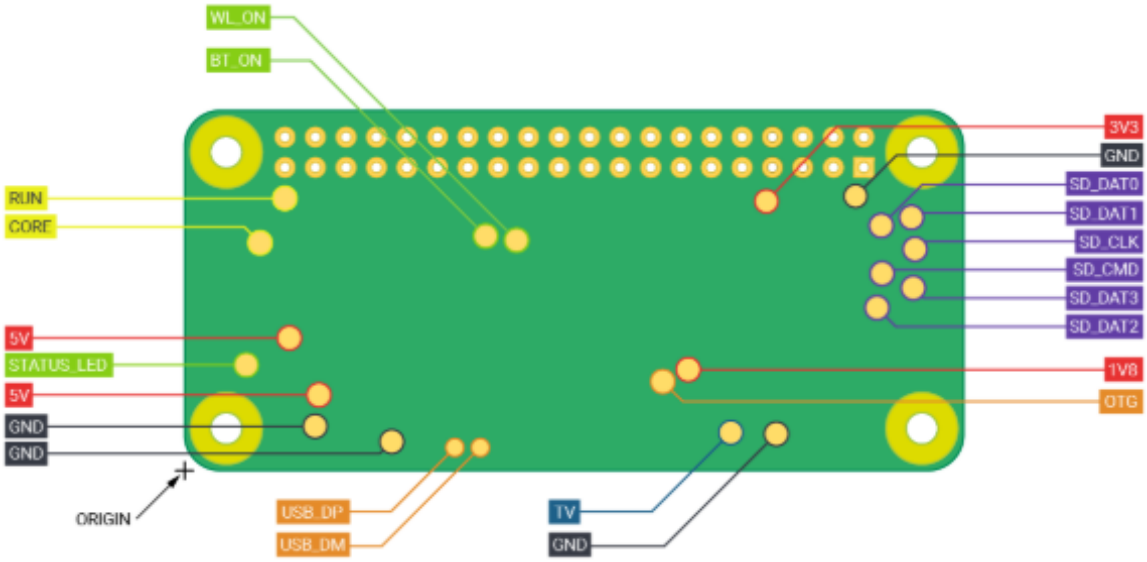

Рисунок 3.1 – Аналіз зворотної сторони Raspberry Pi Zero 2 W для підключення вентилятора та роз’єму USB Type-C [13]

На даному зображенні показано зворотну сторону плати Raspberry Pi Zero 2 W, яка використовується в проєкті для реалізації IoT-модуля підрахунку людей у бомбосховищі. Дослідження цієї частини плати дозволяє виявити місця, де можна здійснити фізичне підключення додаткових компонентів, таких як система охолодження (вентилятор) та роз’єм USB Type-C.

У процесі роботи над проєктом було помічено, що плата має властивість помірно нагріватися під час виконання обчислень та активної роботи Bluetooth-модуля. З огляду на це, було ухвалено рішення встановити кулер, який допоможе стабілізувати температурний режим усередині корпусу, виготовленого на 3D-принтері. Для реалізації цього рішення необхідно знайти доступні контактні площадки з напругою 5V та заземленням GND, які й будуть використовуватися для живлення вентилятора.

Крім того, для зручності живлення та передачі даних плата була дообладнана роз’ємом USB Type-C (рис. 3.2). Це дозволяє реалізувати одночасну подачу живлення та передачу інформації через сучасний інтерфейс,

що робить пристрій зручнішим у щоденному використанні. Таке рішення вимагає визначення місць, до яких можливо припаяти дроти живлення й даних від нового роз’єму, не пошкодивши заводське компонування плати.

Загалом, детальне вивчення зворотної сторони плати є критично важливим етапом у процесі апаратного моделювання пристрою, оскільки дозволяє точно спланувати підключення всіх додаткових елементів без ризику перевантаження схеми або механічного пошкодження.

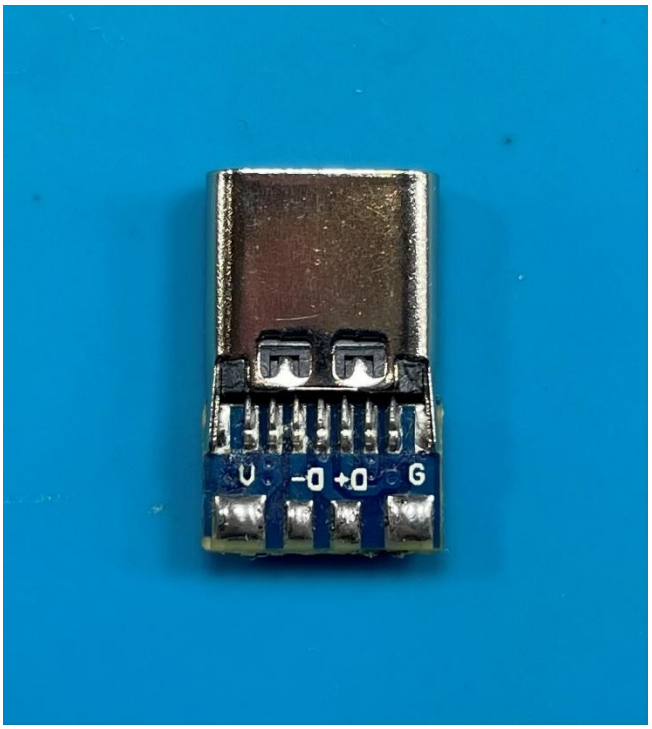

Рисунок 3.2 – Роз’єм USB Type-C у збільшеному масштабі з виділеними контактами для живлення та передачі даних, підготовленими для підпаювання

Зважаючи на потребу у компактності, універсальності й сучасності рішення, було прийнято рішення використати роз’єм USB Type-C. Цей стандарт забезпечує не лише надійне подавання живлення, а й можливість передачі даних за допомогою одного компактного кабелю.

На зображенні представлено роз’єм Type-C у великому плані, що дозволяє детально розглянути структуру контактів. Зокрема, було обрано чотири основні паяльні точки, дві з яких відповідають за подавання напруги

(VCC і GND), а дві інші – за передачу даних (D+ і D−). Це дозволяє ефективно реалізувати як енергопостачання пристрою, так і зв’язок із зовнішніми модулями або інтерфейсами.

Особливу увагу було приділено захисту ліній даних від можливих завад. Для цього відповідні проводи, які відповідають за передачу даних, були скручені у вигляді витої пари, що є перевіреним методом зменшення впливу електромагнітних перешкод. Така техніка дозволяє значно підвищити надійність зв’язку, особливо в умовах, коли мікрокомп’ютер працює в середовищі з потенційними джерелами шумів – наприклад, поруч із живленням кулера або іншими електронними модулями.

Окрім цього, використання роз’єму Type-C у конструкції дає змогу уніфікувати кабелі для живлення та обслуговування пристрою. Це значно спрощує подальшу експлуатацію та обслуговування IoT-модуля, оскільки усуває необхідність використовувати кілька різних інтерфейсів.

Для цього до плати Raspberry Pi Zero 2 W було вручну припаяно роз’єм USB Type-C, який з'єднує два наявних на платі micro-USB порти: micro-USB Power (живлення) та micro-USB OTG (передача даних). З'єднання здійснено таким чином, що лінії живлення (VCC та GND) від роз’єму Type-C підключено до порту живлення, а сигнальні лінії (D+ та D−) – до порту OTG. Це дозволило об’єднати два окремі інтерфейси в єдиний сучасний роз’єм, що значно підвищує зручність експлуатації пристрою (рис. 3.3).

Переваги такого підходу:

1) скорочення кількості кабелів, необхідних для підключення: замість двох (для живлення і даних) використовується один кабель Type-C;

2) підвищення надійності з’єднання завдяки використанню сучасного стандарту;

3) сумісність з новими пристроями, що вже масово використовують USB Type-C;

4) спрощення конструкції корпусу пристрою – достатньо одного отвору для одного роз’єму.

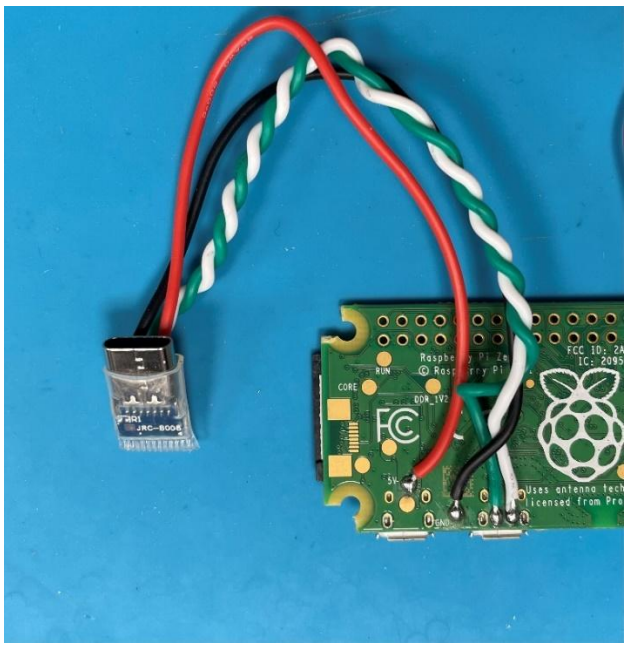

Рисунок 3.3 – Схема підключення живлення та передачі даних через USB Type-C

Таким чином, реалізація з'єднання через USB Type-C не тільки модернізує інтерфейс підключення Raspberry Pi Zero 2 W, а й оптимізує фізичне та функціональне розміщення компонентів пристрою.

Перегрів електронних компонентів негативно впливає на стабільність роботи, продуктивність та термін служби пристрою. З огляду на ці фактори, виникла потреба у впровадженні системи активного охолодження.

Одним з найефективніших та водночас простих способів відведення тепла є використання кулера – невеликого вентилятора, призначеного для примусового охолодження електронних компонентів. Кулер сприяє зниженню температури процесора та інших елементів плати за рахунок постійного руху повітря, що циркулює навколо плати. Це особливо важливо в умовах, коли пристрій буде розміщено в закритому пластиковому корпусі, виготовленому за допомогою 3D-друку.

Принцип роботи кулера полягає в обертанні лопатей за допомогою електродвигуна, живлення якого подається через два основні контакти – VCC (живлення) та GND (земля). При підключенні до плати вентилятор створює потік повітря, який дозволяє ефективно зменшувати нагрівання основних

компонентів. Для більш ефективної циркуляції повітря в корпусі були передбачені вентиляційні отвори, розміщені в верхній частині кришки корпусу. Вони забезпечують вихід гарячого повітря назовні, запобігаючи накопиченню тепла всередині.

Під час монтажу кулера особливу увагу було приділено вибору точок підключення на платі, щоб не перевантажити живлення і не створити перешкод для роботи інших компонентів. Також враховувалася компактність, оскільки Raspberry Pi Zero 2 W має обмежений простір для встановлення додаткових елементів.

Загалом, встановлення кулера суттєво покращило тепловий режим роботи пристрою, що забезпечує більшу стабільність, тривалість експлуатації та знижує ризики збоїв у роботі IoT-модуля при тривалому навантаженні.

У процесі практичних випробувань апаратного модуля на базі Raspberry Pi Zero 2 W було зафіксовано істотне підвищення температури плати при тривалій безперервній роботі. Особливо це стало помітним у режимах підвищеного навантаження, пов’язаних із збором, обробкою та передачею даних. Враховуючи конструктивні особливості плати та той факт, що в рамках реалізації проєкту передбачається створення індивідуального корпусу методом 3D-друку, постала необхідність додатково продумати систему охолодження (рис. 3.4).

Закритий пластиковий корпус без природної вентиляції може призвести до утворення зони перегріву, що в свою чергу потенційно здатне негативно вплинути на стабільність роботи пристрою, зменшити термін служби електронних компонентів або викликати спрацювання захисту від перегріву. Для уникнення подібних проблем було прийнято рішення інтегрувати до конструкції активну систему охолодження.

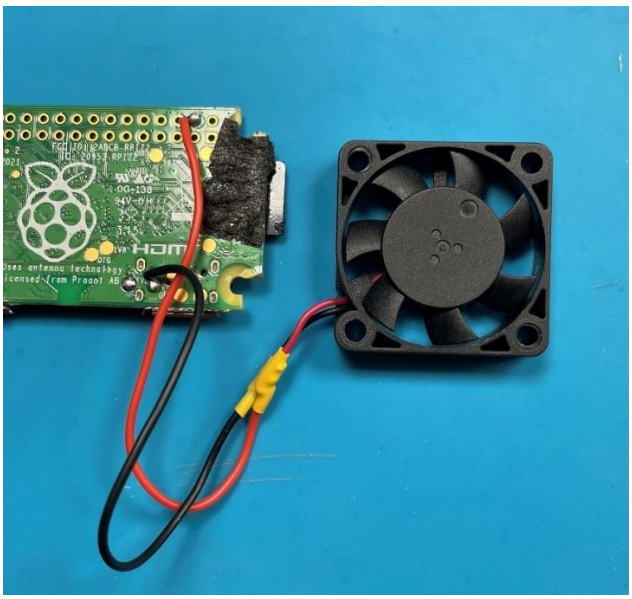

Рисунок 3.4 – Встановлення кулера для охолодження плати

Як оптимальне рішення обрано встановлення компактного мініатюрного кулера, який забезпечить ефективне відведення тепла з поверхні плати. Такий підхід дозволить зберігати робочу температуру пристрою в безпечному діапазоні, забезпечити стабільність функціонування всіх компонентів та підвищити надійність системи загалом. У конструкції корпусу передбачається відповідне місце для монтажу кулера та вентиляційні отвори для циркуляції повітря.

У процесі створення апаратного модуля, одним із важливих етапів є вибір матеріалу для виготовлення захисного корпусу пристрою. Корпус виконує не лише естетичну функцію, але й забезпечує фізичний захист електроніки від механічних пошкоджень, перегріву, пилу та вологи. Тому до матеріалу, з якого він виготовляється, висуваються певні вимоги: міцність, термостійкість, хімічна інертність, простота обробки та сумісність з технологією 3D-друку. У цьому підрозділі розглянуто найбільш поширені типи пластикових ниток, що використовуються в FDM-друці, а також проведено їх порівняння з метою вибору оптимального варіанта для конкретних умов експлуатації корпусу IoT-модуля.

PLA є одним із найпопулярніших матеріалів для 3D-друку. Він виготовляється з біорозкладаної сировини (кукурудзяного крохмалю або цукрової тростини), тому є екологічно безпечним.

Переваги:

– простий у друці, не вимагає нагріву платформи;
– відсутність неприємного запаху під час друку;
– висока точність і якість деталей;
– висока жорсткість.

Недоліки:

– крихкість і низька ударостійкість;
– погана термостійкість (розм’якшується вже при 55–60°C);
– погано підходить для використання в конструкціях з електронікою, що нагрівається.

ABS є одним із перших пластикових матеріалів, які використовувалися в FDM-друці. Міцний, термостійкий і стійкий до механічних впливів.

Переваги:

– висока механічна міцність;
– відмінна термостійкість (до 100°C);
– добра ударостійкість;
– піддається постобробці (шліфуванню, розчиненню в ацетоні тощо).

Недоліки:

– важкий у друці: потребує закритої камери та підігрітої платформи;
– виділяє шкідливі пари при нагріванні;
– має тенденцію до деформацій (англ. warping) при охолодженні.

PETG є модифікованою версією звичайного PET (того, що використовується у пляшках), доповненою гліколем для покращення друкованих властивостей. Це напівгнучкий, хімічно стійкий матеріал, що поєднує властивості PLA та ABS.

Переваги:

– висока ударостійкість;
– гарна термостійкість (до 75–80°C);
– стійкість до вологи та хімічних речовин;
– легше друкується, ніж ABS (не потребує повністю закритої камери);
– не має неприємного запаху;
– напівпрозорість і гарний зовнішній вигляд.

Недоліки:

– менш жорсткий, ніж PLA;
– потребує точного налаштування температури екструдера;
– трохи «струниться» (англ. stringing) при переміщеннях друкуючої голівки.

З огляду на це, вибір матеріалу для 3D-друку корпусу має ґрунтуватися на ряді технічних і експлуатаційних критеріїв: достатня міцність, термостійкість, стійкість до деформацій, простота обробки, а також екологічна безпечність. Після аналізу поширених видів пластикових ниток для FDM-друку – таких як PLA, ABS, PETG, TPU – було прийнято рішення використовувати саме PETG. На відміну від PLA, який хоч і легко друкується, але має низьку термостійкість, або ABS, що потребує закритої камери та виділяє неприємний запах під час друку, PETG демонструє стабільну поведінку навіть при друкуванні на відкритому 3D-принтері, має хорошу адгезію шарів, достатню гнучкість, а також стійкість до високих температур. Зважаючи на особливості проєкту, а саме: наявність постійного нагріву під час роботи (через встановлену плату Raspberry Pi Zero 2 W та кулер), потребу в міцності конструкції, стабільності до зовнішніх впливів та спрощеного процесу друку – PETG був обраний як оптимальний матеріал, що забезпечує баланс між функціональністю, довговічністю та простотою виготовлення.

У рамках реалізації апаратного корпусу для ІоТ-модуля було обрано сучасний 3D-принтер Bambu Lab X1 Carbon, який вирізняється високою точністю, швидкістю друку та підтримкою широкого спектра філаментів,

включно з PETG, ABS, PLA, TPU та композитними матеріалами. Принтер обладнаний повністю закритим корпусом, системою автоюстування та лідарами вбудованих сенсорів, що забезпечують стабільність та якість друку.

Процес 3D-друку починається з підготовки цифрової моделі в слайсері (напр., Bambu Studio), де встановлюються параметри друку: температура сопла, температура стола, швидкість друку, товщина шару, тип підтримки, інфіл тощо (рис. 3.5). Далі файл з розширенням .3mf або .gcode передається на принтер за допомогою SD-карти або бездротової передачі.

Перед початком друку принтер автоматично виконує калібрування (напр., PID-регулювання нагріву сопла, перевірку рівності платформи). Після завершення підготовки екструдер нагрівається до заданої температури, після чого починається друк моделі шар за шаром. Завдяки своїй конструкції, Bambu Lab X1 Carbon може працювати на швидкостях понад 500 мм/с, при цьому зберігаючи точність і деталізацію.

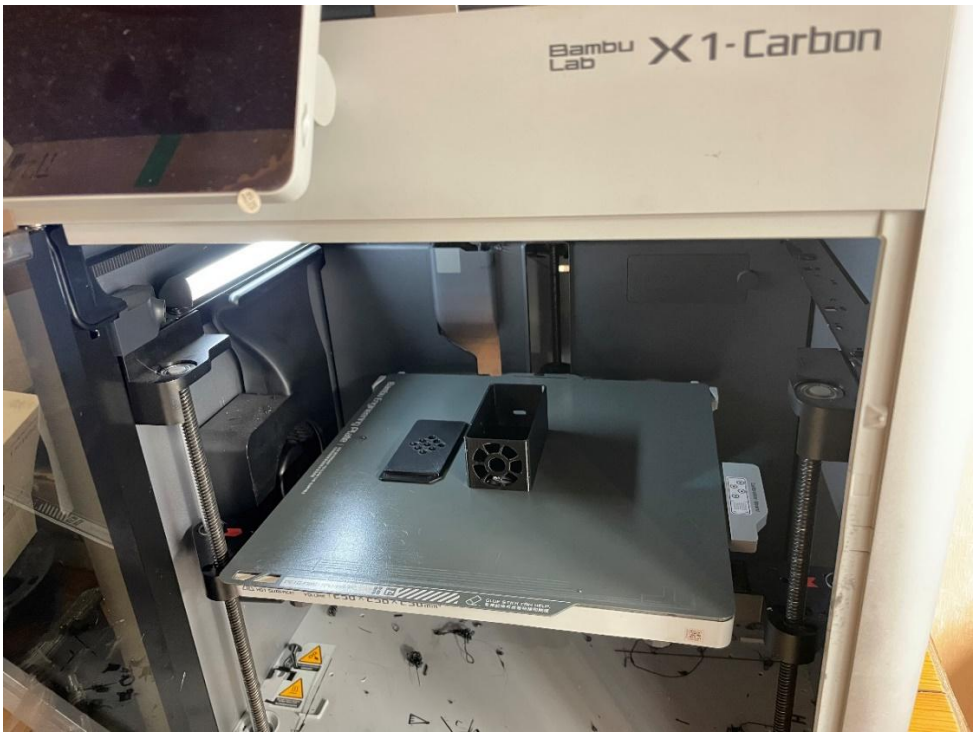

Рисунок 3.5 – Процес друку корпусу модуля
на принтері Bambu Lab X1 Carbon

У процесі розробки корпусу для апаратного модуля було створено тестову 3D-модель у програмному середовищі Untitled. На початковому етапі моделювання було вирішено створити базову версію корпусу без верхньої кришки з метою оцінки габаритів, відповідності роз’ємів і компонування внутрішніх елементів, зокрема плати Raspberry Pi Zero 2 W та системи охолодження (рис. 3.6).

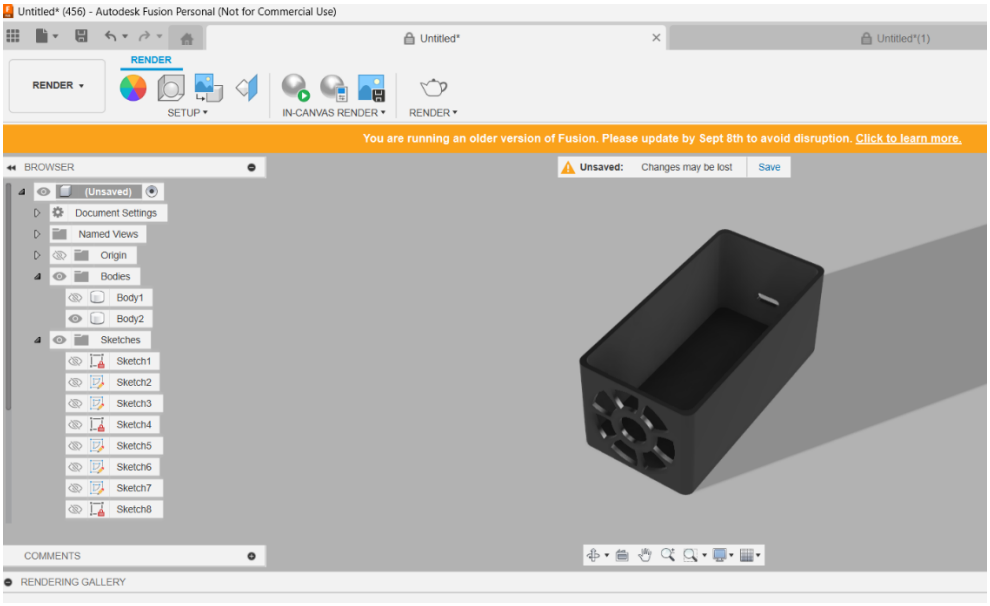

Рисунок 3.6 – Тестова 3D-модель корпусу апаратного модуля в середовищі Untitled без кришки

Така тестова модель дозволяє виявити потенційні помилки конструкції, перевірити зручність розміщення елементів та забезпечити доступ до необхідних інтерфейсів пристрою. Надалі планується доопрацювання модель з додаванням верхньої кришки.

Після створення базової тестової моделі корпусу було виконано розміщення всіх ключових компонентів апаратного модуля всередині 3D-моделі. У корпус були інтегровані: плата Raspberry Pi Zero 2 W, кулер для охолодження, USB-перехідники, а також провід для живлення з роз'ємом Type-C, що замінює стандартні micro-USB входи (рис. 3.7).

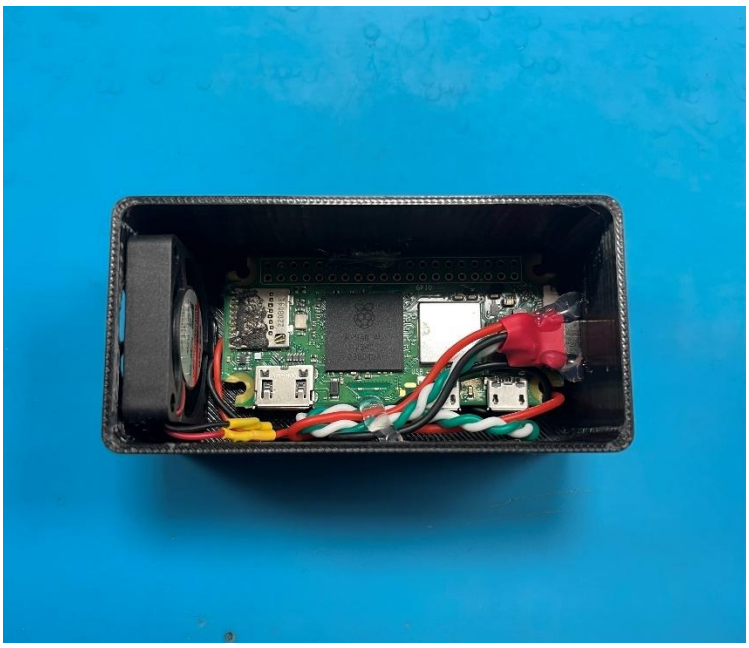

Рисунок 3.7 – Розміщення основних компонентів модуля у 3D-корпусі

Розміщення компонентів проводилося з урахуванням зручності монтажу, відведення тепла, а також забезпечення вільного доступу до портів для подальшого обслуговування чи оновлення прошивки. Конструкція корпусу також враховує можливість подальшого доопрацювання – додавання кріплень або монтажу на стіну чи інші поверхні.

Наступним етапом розробки стало моделювання верхньої кришки для корпусу апаратного модуля. Основною метою було забезпечення надійного захисту внутрішніх компонентів від пилу та механічних пошкоджень, а також покращення вентиляції. Для цього в конструкції кришки були передбачені спеціальні вентиляційні отвори, розташовані над областю, де розміщується кулер і плата Raspberry Pi Zero 2 W (рис. 3.8).

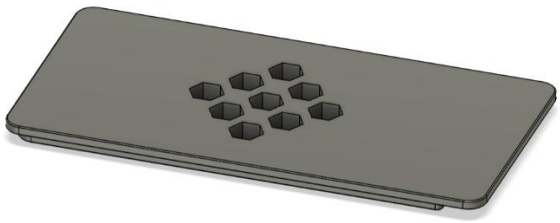

Рисунок 3.8 – 3D-модель кришки корпусу з вентиляційними отворами

Такий підхід дозволяє ефективно відводити тепло, що утворюється під час роботи пристрою, та знижує ризик перегрівання в умовах обмеженої циркуляції повітря всередині корпусу. Модель кришки розроблена з урахуванням можливості її легкого зняття або заміни, що спрощує обслуговування пристрою.

Завершальним етапом стало остаточне складання всіх компонентів пристрою всередині надрукованого корпусу. Усі елементи – плата Raspberry Pi Zero 2 W, кулер для охолодження, а також допоміжні з'єднання (Type-C роз'єм, кабелі живлення та передавання даних) – були надійно розміщені та закріплені. Корпус був спеціально змодельований таким чином, щоб враховувати простір для вентиляції, доступ до інтерфейсів підключення, а також забезпечити зручне технічне обслуговування у майбутньому (рис. 3.9).

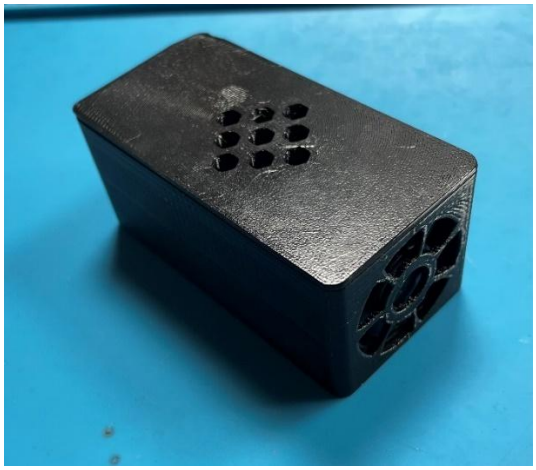

Рисунок 3.9 – Готовий апаратний модуль у зібраному корпусі

Повністю зібраний пристрій демонструє компактність, функціональність і готовність до експлуатації в умовах бомбосховища. Завдяки використанню сучасного підходу до 3D-моделювання корпус вдалося зробити легким, міцним і відповідним до поставлених технічних вимог.

3.2 Налаштування операційної системи та збірка образу

З метою забезпечення стабільної та повторюваної роботи апаратного модуля на базі Raspberry Pi Zero 2 W, було прийнято рішення одноразово встановити і налаштувати ОС, після чого створити її оптимізований образ для

подальшого розгортання на інших пристроях. В якості базової ОС була обрана Raspberry Pi OS Lite – офіційна мінімалістична версія на базі Debian 11 (Bullseye), яка характеризується високою продуктивністю та відсутністю графічного середовища, що ідеально підходить для роботи в умовах обмежених апаратних ресурсів.

Встановлення починається із запису ОС на SD-карту за допомогою утиліти Raspberry Pi Imager. Після завершення запису, до повернення SD-карти у пристрій, необхідно здійснити низку підготовчих дій. Для активації SSH-доступу потрібно створити порожній файл з назвою *ssh* у директорії /boot. Далі, у цій же директорії створюється файл *wpa_supplicant.conf*, в якому зазначаються параметри доступу до бездротової мережі Wi-Fi, завдяки чому модуль зможе автоматично під’єднуватись до мережі при кожному ввімкненні.

Після вставлення картки в Raspberry Pi та підключення живлення, пристрій автоматично завантажиться, під’єднається до мережі Wi-Fi і стане доступним для віддаленого керування через SSH. Таким чином, фізичне підключення монітора або інших пристроїв введення не потрібне – усі подальші дії виконуються дистанційно через термінал.

Після першого запуску пристрою та успішного підключення до локальної мережі через Wi-Fi, необхідно налаштувати пріоритети мережевих інтерфейсів. Це дає змогу модулю в першу чергу використовувати бездротове з’єднання, а лише у випадку його відсутності – резервне через USB. Для цього відкривається конфігураційний файл за допомогою команди:

```
sudo nano /etc/dhcpcd.conf
```

І в кінець файлу додаються наступні рядки:

```
interface wlan0
metric 200
interface usb0
metric 300
```

Збереження змін у конфігураційному файлі та подальше перезавантаження системи дозволяє застосувати внесені параметри маршрутизації, що забезпечить коректний пріоритет мережевих інтерфейсів при завантаженні Raspberry Pi. У результаті цього налаштування обчислювальний модуль спочатку спробує встановити з’єднання через Wi-Fi (інтерфейс *wlan*0), який був попередньо налаштований у файлі *wpa_supplicant.conf*. Лише у випадку відсутності з'єднання з бездротовою мережею пристрій автоматично перемкнеться на альтернативний варіант – інтерфейс usb0, що забезпечує доступ до мережі через з’єднання з комп’ютером або іншим пристроєм.

Цей підхід дозволяє гарантувати стабільну роботу пристрою у випадках, коли він використовується автономно, без постійного підключення до комп’ютера, і залежить від локальної WiFi-мережі для передачі даних. Також таке налаштування особливо важливе в умовах, де доступ до пристрою обмежений або фізичне втручання у процес підключення небажане. Перезавантаження системи після збереження файлу конфігурації */etc/dhcpcd.conf* є необхідною умовою для того, щоб нові значення метрик вступили в силу та були враховані мережею при кожному запуску системи.

3.2.1 Встановлення необхідного програмного забезпечення

Перед тим як перейти до реалізації функціоналу для підрахунку людей та передачі даних через локальну мережу, необхідно підготувати середовище, в якому буде працювати пристрій. Основним завданням на цьому етапі є встановлення всього необхідного програмного забезпечення, яке забезпечить роботу Bluetooth-сканування, обробку даних та взаємодію з вебінтерфейсом.

Оскільки модуль Raspberry Pi Zero 2 W має обмежені апаратні ресурси, особливу увагу потрібно приділити легкості та оптимізації програмних рішень. Перевагу надається мінімалістичним, швидким і стабільним інструментам, які можна запустити на Raspberry Pi OS Lite – полегшеній версії ОС на базі Debian.

На цьому етапі встановлюються такі компоненти:

– системні утиліти для оновлення, налаштування мережі та обслуговування ОС;

– інструменти для сканування Bluetooth-пристроїв (наприклад, bluetoothctl, bluez, hcitool);

– мережеві бібліотеки для взаємодії з вебінтерфейсом (через HTTP або WebSocket);

– менеджери пакетів (наприклад, *pip*) для встановлення необхідних Python-бібліотек;

– вебсервер nginx для локального відображення вебінтерфейсу;

– додаткові Python-модулі, які забезпечують зв’язок з MQTT-брокером або іншими сервісами, якщо проєкт цього вимагає.

Після встановлення та налаштування всі ці компоненти працюватимуть узгоджено, забезпечуючи стабільну та автономну роботу апаратного модуля в умовах обмеженого підключення та відсутності дисплея.

3.3 Налаштування вебсерверу для передачі даних через локальну мережу

Оскільки пристрій функціонує без підключеного екрана, всі дані, які він збирає в режимі реального часу, повинні відображатися на віддаленому пристрої (наприклад, ноутбуці або смартфоні користувача). Для цього необхідно реалізувати просту, але надійну систему віддаленого доступу до інформації через вебінтерфейс. Найкращим способом організації такого підключення є розгортання локального вебсерверу, який працюватиме на самому Raspberry Pi та забезпечуватиме доступ до вебсторінки з інформаційною панеллю, використовуючи бездротове WiFi-з’єднання у локальній мережі.

У цьому проєкті для реалізації вебінтерфейсу обрано легкий і продуктивний вебсервер nginx, який чудово підходить для пристроїв з обмеженими обчислювальними ресурсами, таких як Raspberry Pi Zero 2 W.

Його основні переваги – низьке споживання пам’яті, висока продуктивність при обробці статичних файлів, а також широкі можливості для налаштування.

Крок 1: Оновлення системи

Перед початком встановлення нового програмного забезпечення слід переконатися, що всі системні пакети оновлені до актуальних версій. Це дозволить уникнути конфліктів залежностей та підвищити стабільність роботи системи.

```
sudo apt update && sudo apt upgrade -y
```

Ця команда оновлює список доступних пакетів (apt update) та одразу встановлює всі наявні оновлення (apt upgrade -y), автоматично підтверджуючи кожен крок без запиту в користувача (-y).

Крок 2: Встановлення *nginx*

Після оновлення системи переходимо до встановлення самого вебсерверу. Щоб не перевантажувати систему непотрібними компонентами, встановлюємо nginx з мінімальними залежностями:

```
sudo apt install --no-install-recommends nginx -y
```

Цей підхід дозволяє встановити лише критично важливі файли, без додаткових пакетів, які зазвичай рекомендуються автоматично. Це зменшує розмір системи та прискорює її запуск.

Крок 3: Налаштування *nginx*

Після встановлення вебсерверу потрібно вказати йому, з якої директорії брати HTML-файли для відображення вебінтерфейсу. Для цього відкриваємо конфігураційний файл за допомогою текстового редактора:

```
sudo nano /etc/nginx/sites-enabled/default
```

Замість типового вмісту вставляємо таку конфігурацію:

```
server {
    listen 127.0.0.1:80;
    root /var/www/html;
    index index.html;

    location / {
        try_files $uri $uri/ /index.html;
    }
}
```

Це налаштування вказує серверу *nginx* слухати запити лише з локального інтерфейсу (тобто з самого пристрою), використовувати директорію */var/www/html* як кореневу для вебсайту та відображати файл index.html як стартову сторінку. Опція *try_files* дозволяє обробляти запити до файлів, яких фізично не існує, і перенаправляти їх на головну сторінку, що особливо корисно при створенні SPA.

Крок 4: Перезапуск вебсерверу

Після збереження змін конфігурації потрібно перезапустити nginx, щоб нові налаштування вступили в дію:

```
sudo systemctl restart nginx
```

Після цього вебсервер починає обробку HTTP-запитів. Якщо пристрій підключено до WiFi-мережі, користувач може ввести IP-адресу Raspberry Pi у браузері на іншому пристрої та отримати доступ до вебінтерфейсу, розміщеного в директорії */var/www/html*.

3.4 Встановлення Python-модулів і налаштування системної служби

На цьому етапі реалізації програмної частини необхідно встановити всі бібліотеки та інструменти, що забезпечать взаємодію з Bluetooth-модулем Raspberry Pi Zero 2 W, обробку зібраних даних, а також їх передачу до вебінтерфейсу через локальну мережу. Зважаючи на те, що основна логіка реалізована мовою Python, для зручного керування зовнішніми бібліотеками доцільно встановити менеджер пакетів *pip*, який є стандартом у середовищі Python.

Встановлення менеджера пакетів *pip*

Першим кроком є встановлення менеджера пакетів командою:

```
sudo apt install python3-pip -y
```

Цей менеджер дозволяє легко додавати сторонні Python-модулі, необхідні для роботи з серійним інтерфейсом, Bluetooth, мережею та обміном даними через MQTT або WebSocket-протоколи.

Встановлення потрібних бібліотек

Після встановлення *pip* за його допомогою встановлюємо всі необхідні Python-модулі, що будуть використовуватись у програмному скрипті. Це виконується командою:

```
sudo pip3 install pyserial websocket-server paho-mqtt
```

– pyserial: дозволяє взаємодіяти з серійними портами, якщо використовуються зовнішні пристрої;

– websocket-server: забезпечує двосторонню комунікацію між Raspberry Pi та вебінтерфейсом у реальному часі;

– paho-mqtt: популярна бібліотека для роботи з протоколом MQTT, що може використовуватись для інтеграції з IoT-системами, якщо такі плануються в подальшому.

Створення системної служби

Для того, щоб не запускати програму вручну кожного разу після перезавантаження пристрою, доцільно створити спеціальну системну службу. Вона дозволяє автоматизувати запуск скрипту під час старту ОС.

Створюємо новий конфігураційний файл служби:

```
sudo nano /etc/systemd/system/asset.service
```

У вікно редагування вставляємо наступну конфігурацію:

```
[Unit]
Description=Python-сервіс збору та передачі даних
After=multi-user.target

[Service]
Type=simple
Restart=always
ExecStart=/usr/bin/python3 /home/pi/app/app.py

[Install]
WantedBy=multi-user.target
```

– ExecStart: вказує на шлях до головного скрипту app.py, який розташований у директорії /home/pi/app/;

– Restart=always: забезпечує перезапуск скрипту у випадку аварійного завершення;

– After=multi-user.target: гарантує запуск після того, як система готова до роботи в багатокористувацькому режимі.

Активація служби

Після збереження файлу конфігурації потрібно перезавантажити конфігурацію *systemd* і активувати службу:

```
sudo systemctl daemon-reload
sudo systemctl enable asset.service
```

З цього моменту Python-програма буде автоматично запускатись після кожного увімкнення Raspberry Pi, без необхідності втручання з боку користувача.

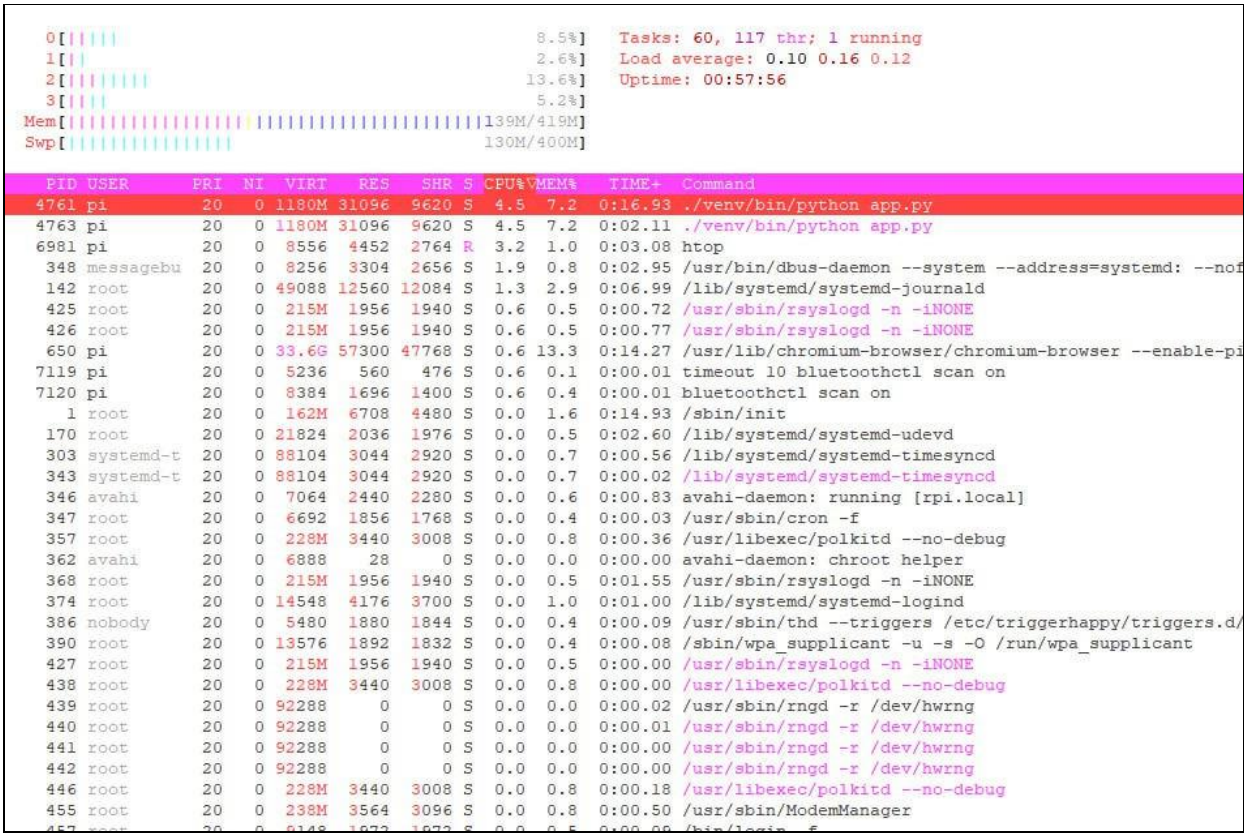

Рисунок 3.10 – Перший запуск апаратного модуля Raspberry Pi Zero 2 W у зібраному корпусі

У результаті цього налаштування досягається повна автономність роботи пристрою: модуль автоматично підключається до Wi-Fi, активує Bluetooth-адаптер, виконує сканування, обробку й передачу даних, які у реальному часі можна побачити у вебінтерфейсі через локальну мережу. Такий підхід значно спрощує процес використання пристрою у реальних умовах, наприклад, у бомбосховищі, де швидкий старт і мінімальна взаємодія з системою є критично важливими.

3.5 Розробка вебінтерфейсу для моніторингу пристроїв

У межах розробки IoT-модуля для підрахунку кількості людей у бомбосховищі було реалізовано повноцінний вебінтерфейс, який відображає результати сканування Bluetooth-пристроїв у реальному часі. Цей інтерфейс є ключовим елементом системи, що дозволяє адміністраторам або відповідальним особам оперативно контролювати кількість присутніх осіб у сховищі через WiFi-підключення до Raspberry Pi Zero 2 W.

3.5.1 Загальний вигляд та функціональність

Інтерфейс має сучасний, інтуїтивно зрозумілий дизайн із адаптивною верстkою, що дозволяє користуватися ним як із комп’ютерів, так і з мобільних пристроїв. Основний блок із заголовком містить назву системи Enhanced Bluetooth People Counter та короткий опис її функцій. В центрі сторінки розміщується панель з ключовими статистичними показниками, такими як:

- поточна кількість активних пристроїв;
- загальна кількість виявлених пристроїв;
- оцінена кількість присутніх людей;
- кількість пристроїв з випадковою MAC-адресацією;
- кількість BLE-пристроїв (якщо активовано розширений режим сканування).

3.5.2 Технічна реалізація інтерфейсу

Інтерфейс побудований на HTML, CSS та JavaScript. Основна верстка реалізована без використання сторонніх фреймворків, проте з використанням адаптивної сітки *grid* для ефективного розташування елементів. Колірна палітра підібрана у стилі neumorphism, що забезпечує м’який вигляд і візуальний комфорт для користувача.

Режими сканування:

1) *Basic Mode* – стандартне виявлення пристроїв у радіусі дії Bluetooth-модуля;

2) *Unified Mode* – розширений режим, що включає обробку MAC-рандомізації, кластеризацію за часом, Kalman-фільтрацію, BLE-відстеження.

У верхній частині інтерфейсу присутній індикатор активного режиму, що підтягується з API */api/scan_mode*.

Дані пристроїв. Таблиця з пристроями будується динамічно з відповідей API */api/devices*. У ній відображаються:

– MAC-адреса;
– назва пристрою;
– метод виявлення (стандартний або розширений);
– відстань або впевненість;
– час першого/останнього виявлення;
– статус пристрою (активний, неактивний тощо).

Для оновлення даних використовуються функції JavaScript з *fetch()* та регулярним інтервалом оновлення кожні 3 секунди (*setInterval*).

Взаємодія з бекендом. Інтерфейс обмінюється даними з бекендом через REST API, реалізоване на стороні Raspberry Pi. Основні точки доступу:

– */api/start_scan* – запуск сканування;
– */api/stop_scan* – зупинка процесу сканування;
– */api/stats* – отримання агрегованої статистики;
– */api/people_count* – оцінка кількості присутніх осіб;

– */api/devices* – масив об’єктів пристроїв;

– */api/ble_stats* – інформація про BLE-аналіз.

Завдяки використанню асинхронного *JavaScript* та *Web API*, всі дані завантажуються без перезавантаження сторінки, що забезпечує ефективність та зручність у користуванні.

Особливості дизайну (рис. 3.11):

– статистичні картки (cards) реалізовані як самостійні блоки з анімацією та ефектами наведення;

– індикатор надійності оцінки: графічна шкала, що відображає впевненість системи у розрахунку кількості людей;

– адаптивність: усі елементи змінюють свій розмір залежно від ширини екрану, що дозволяє комфортно використовувати інтерфейс зі смартфону.

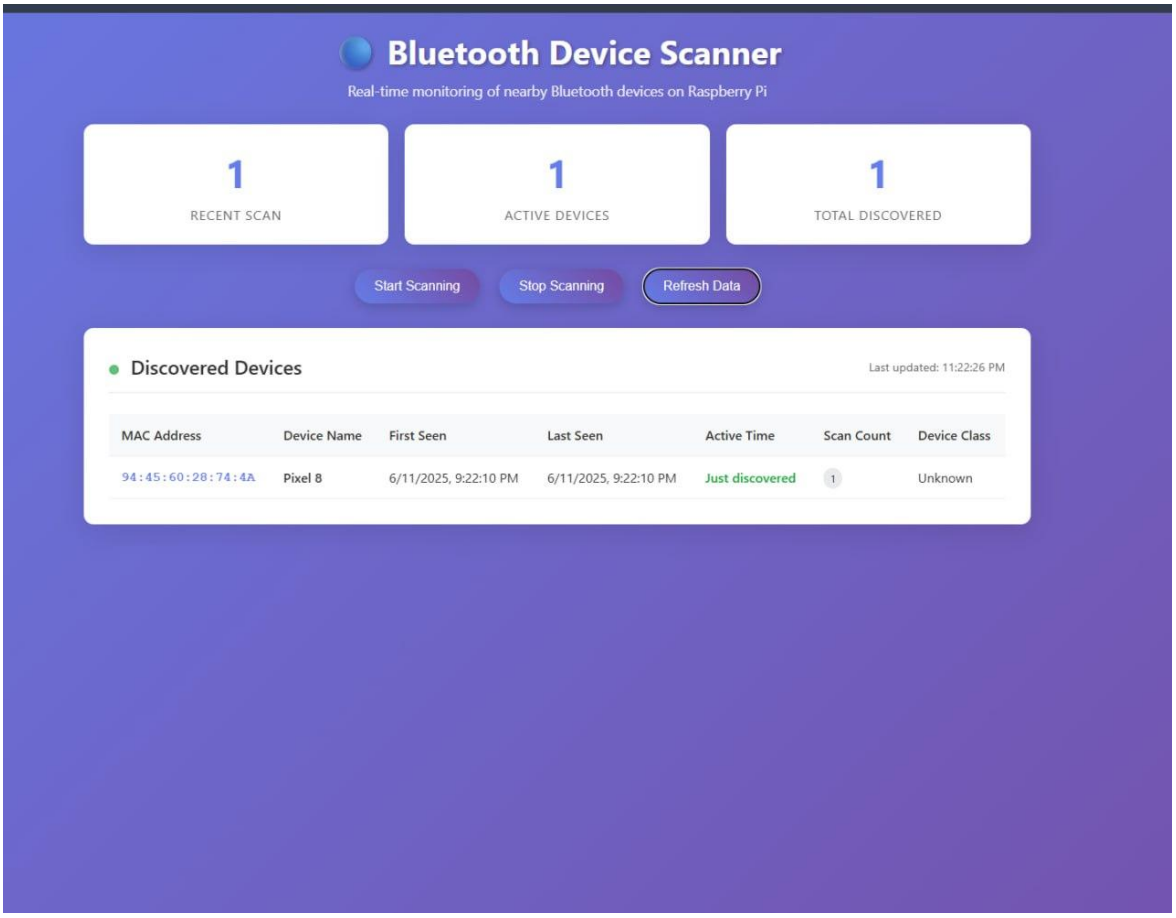

Рисунок 3.11 – Вебінтерфейс моніторингу Bluetooth-пристроїв у бомбосховищі

Переваги реалізованого інтерфейсу:

– автономність: інтерфейс працює локально на самій платі Raspberry Pi, не потребуючи зовнішнього інтернету;

– живе оновлення: завдяки fetch і setInterval користувач завжди бачить актуальні дані;

– розширення: структуру легко масштабувати або інтегрувати з базами даних та централізованими системами моніторингу;

– конфіденційність: відсутність відеонагляду чи обробки персональних даних.

Висновки до розділу 3

У межах третього розділу дипломної роботи було виконано повний цикл створення ІоТ-модуля для підрахунку кількості людей у приміщенні бомбосховища на базі одноплатного комп’ютера Raspberry Pi Zero 2 W. Кожен етап реалізації був продуманий з урахуванням вимог до автономності, енергоефективності, зручності монтажу та простоти використання в умовах реальних обмежень – зокрема, у притулках без постійного живлення, інтернету або професійного технічного обслуговування.

Першим кроком стала фізична модифікація плати Raspberry Pi Zero 2 W: було припаяно USB Type-C роз’єм, що виконує функції як живлення, так і передачі даних. Такий підхід дозволяє використовувати сучасні кабелі живлення та забезпечує більшу надійність у порівнянні зі стандартними microUSB. Крім того, для вирішення проблеми перегріву під час тривалої безперервної роботи було встановлено невеликий кулер охолодження, підключений безпосередньо до контактів плати. Це дозволило стабілізувати температуру під час навантаження та забезпечити довготривалу безперебійну роботу модуля.

Наступним кроком стало створення 3D-моделі корпусу. Для цього було проведено аналіз різних типів пластиків, порівняно їх властивості щодо міцності, температурної стабільності, гнучкості, зокрема PLA, ABS, PETG, і в

результаті було обрано пластик PETG як найкращий варіант для умов, у яких буде експлуатуватись пристрій. Сам корпус був спроєктований у середовищі 3D-моделювання (Untitled app) та надрукований на 3D-принтері Bambu Lab X1 Carbon. Було передбачено вентиляційні отвори у кришці корпусу, що доповнюють функціональність кулера та забезпечують вільну циркуляцію повітря.

Паралельно з апаратною частиною було розгорнуто легку серверну ОС Raspberry Pi OS Lite, яка була налаштована для автоматичного запуску при живленні та з’єднання з WiFi-мережею. Особливу увагу було приділено автоматизації всіх процесів: створено системну службу, яка після запуску автоматично активує скрипт Bluetooth-сканування, а зібрані дані передаються через локальну мережу до вебінтерфейсу. Таке рішення дозволяє використовувати модуль навіть у віддалених укриттях без необхідності підключення до монітора, миші чи клавіатури.

Було також розроблено повноцінний вебінтерфейс для відображення зібраних даних у реальному часі. Інтерфейс реалізовано на HTML, CSS та JavaScript, а його компоненти дозволяють візуально контролювати кількість знайдених Bluetooth-пристроїв, відображати MAC-адреси, назви пристроїв, час їх появи, статус активності тощо. Додатково реалізовані режими роботи сканера, включаючи стандартний та розширений, які відрізняються рівнем точності та глибиною аналізу.

4 ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІЖЕННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ РОЗРОБКИ ІОТ-МОДУЛЯ

У цьому розділі подано результати створення ІоТ-модуля, а також більш детально розглянуто окремі особливості його практичного застосування.

4.1 Огляд процесу підрахування людей у бомбосховищі

З метою перевірки працездатності розробленого ІоТ-модуля в умовах, наближених до реального використання, було обрано одне з доступних бомбосховищ, де регулярно перебуває значна кількість людей. Це дало змогу не лише оцінити стабільність роботи системи, але й перевірити її точність у реальному середовищі з великою кількістю бездротових пристроїв, різними рівнями сигналу та умовами просторового розташування людей.

Саме у цьому середовищі було розгорнуто Raspberry Pi Zero 2 W з налаштованим програмним забезпеченням, після чого активовано режим сканування та зібрано статистичні дані, які далі було виведено на вебінтерфейс для подальшого аналізу.

На рис. 4.1 продемонстровано результат запуску вебінтерфейсу Enhanced Bluetooth People Counter, розгорнутого на модулі Raspberry Pi Zero 2 W. Система працює в режимі Unified Mode, який включає обробку випадкових MAC-адрес, безперервне BLE-сканування, фільтрацію шумів (Kalman filtering) та статистичний аналіз.

У центральній частині інтерфейсу показано, що система виявила 2 присутні особи, ґрунтуючись на даних 10 знайдених пристроїв, з яких 7 мають рандомізовані MAC-адреси. Поточний рівень довіри системи до оцінки становить 53 %, що свідчить про достатню точність у режимі роботи в реальних умовах.

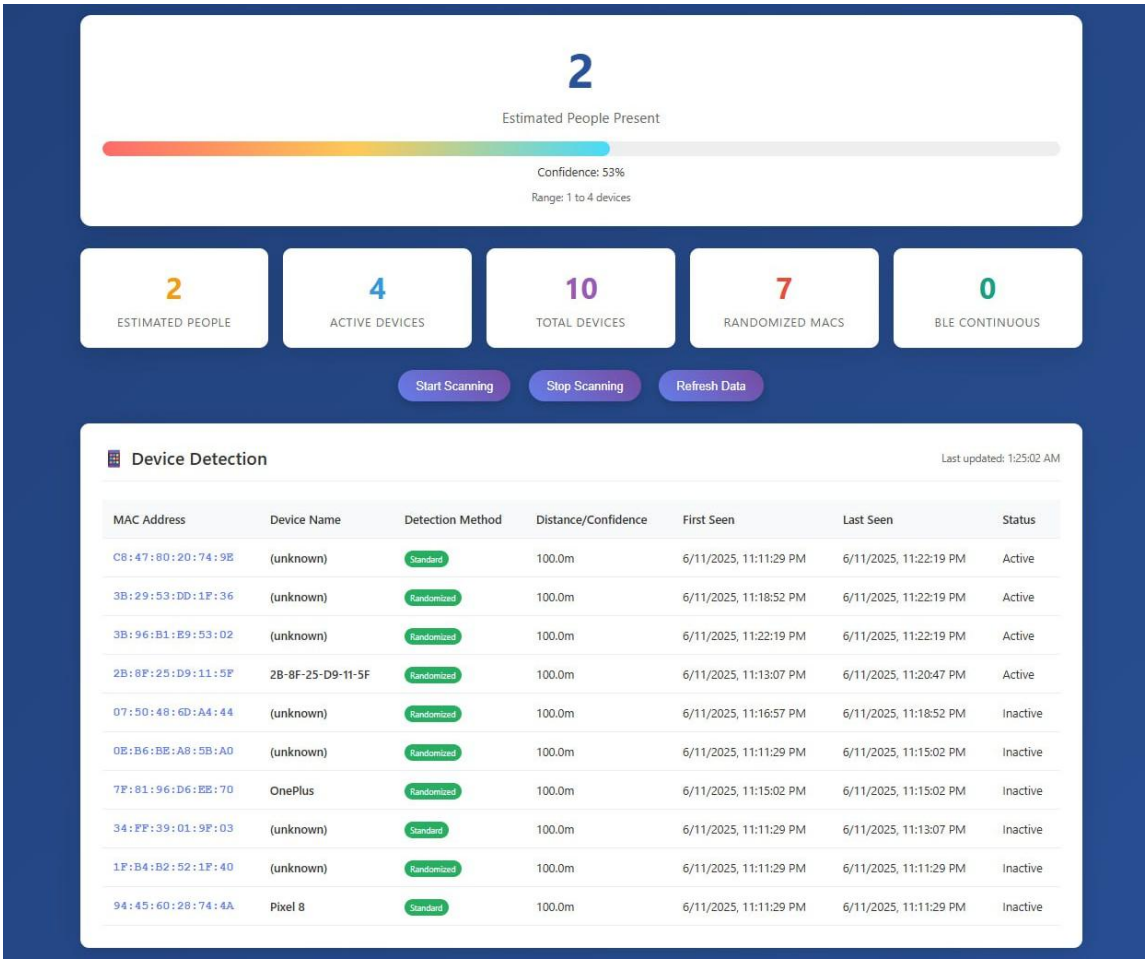

Рисунок 4.1 – Вебінтерфейс у розширеному режимі Unified Mode для підрахунку людей за Bluetooth-сигналами

У нижній частині виводиться розширена таблиця, в якій детально описані параметри кожного знайденого пристрою: MAC-адреса, назва пристрою (якщо доступна), метод виявлення (стандартний або з MAC-рандомізацією), відстань, час першої/останньої появи та поточний статус (активний чи неактивний).

Ці результати демонструють повну працездатність розробленого IoT-модуля, включно з передачею даних через локальну мережу, їх обробкою та виведенням у зручній формі в реальному часі.

4.2 Аналіз експериментальної роботи IoT-модуля

З метою перевірки працездатності та ефективності створеного IoT-модуля для підрахунку кількості людей у приміщенні було проведено

експериментальне тестування в умовах, наближених до реальних. Для цього апаратний модуль, зібраний на базі Raspberry Pi Zero 2 W, був поміщений у змодельований 3D-корпус, під’єднаний до джерела живлення та попередньо налаштований на автоматичний запуск усіх необхідних служб.

IoT-модуль був запрограмований на безперервне сканування Bluetooth-пристроїв у межах радіусу дії адаптера. Увімкнене програмне забезпечення здійснювало обробку отриманих сигналів, зокрема MAC-адрес, які надходили від мобільних пристроїв користувачів. Зібрані дані аналізувалися за допомогою алгоритмів, які дозволяли не лише виявити унікальні пристрої, але й визначити кількість людей на основі логіки фільтрації дублікатів, випадкових MAC-адрес і BLE-маячків.

Під час запуску система працювала в режимі Unified Mode, що забезпечує розширену функціональність сканера. В результаті експерименту було виявлено 10 пристроїв, з яких 7 були з рандомізованими MAC-адресами, а 2 пристрої були класифіковані системою як потенційні присутні особи. При цьому рівень довіри до результату становив 53 %, що вважається прийнятним у складних умовах без постійного інтернету, у середовищі з різними типами пристроїв і випадковими сигналами. Із розрахунку «1 пристрій на 1 людину» можна зробити висновок, що було виявлено 10 людей.

Сканер також правильно розрізнив активні пристрої від неактивних, а також зафіксував перший і останній час виявлення кожного об’єкта, що дозволяє вести повноцінний облік руху в укритті.

Слід відзначити, що дані передавались і виводились через вебінтерфейс, який автоматично відкривається в локальній мережі без потреби підключення до інтернету. Інтерфейс забезпечив зручну візуалізацію: кількість знайдених пристроїв, кількість активних, оцінка кількості людей, рівень довіри, а також таблицю детекції пристроїв із такими полями як MAC-адреса, назва пристрою, метод виявлення, час активності та статус.

Таким чином, результати експериментального запуску підтверджують повну функціональність розробленого IoT-модуля: від збирання та обробки

даних до виводу у зручному вигляді на вебінтерфейсі. Всі компоненти працюють узгоджено, система автоматично запускається при подачі живлення, не потребує втручання користувача, не використовує відеокамери та зберігає приватність осіб, що перебувають у зоні дії пристрою.

Отримані результати демонструють доцільність і перспективність застосування такого підходу для обліку людей у захисних спорудах та інших ізольованих приміщеннях, де важливі автономність, простота розгортання та ефективне використання ресурсів.

Висновки до розділу 4

У результаті проведеного експериментального дослідження було підтверджено працездатність розробленого ІоТ-модуля для автоматичного підрахунку людей у бомбосховищі. Модуль успішно продемонструвала свою здатність у режимі реального часу виявляти бездротові пристрої поблизу, аналізувати їхню присутність, визначати ймовірну кількість осіб на основі зібраних Bluetooth-сигналів та надавати користувачеві візуалізовану інформацію через локальний вебінтерфейс.

Під час тестування ІоТ-модуль показав стабільну роботу в автономному режимі, що особливо важливо в умовах обмеженого електроживлення та відсутності інтернет-з’єднання, які можуть бути типовими для укриттів. Вбудований інтерфейс дозволяє оперативно переглядати поточну кількість присутніх, історію виявлених пристроїв, рівень довіри до оцінки та інші параметри, що є критично важливими для швидкого прийняття рішень у надзвичайних ситуаціях.

Також було підтверджено ефективність реалізованих алгоритмів – розроблений ІоТ-модуль зміг обробити рандомізовані MAC-адреси, класифікувати типи пристроїв та надати узагальнений прогноз присутності людей із допустимою похибкою.

Загалом, результати експерименту свідчать про готовність прототипу до практичного використання, а обрані підходи – про їхню доцільність, простоту

впровадження та адаптивність до різних умов експлуатації. Розроблений IoT-модуль може бути використаний як окремо, так і як частина більших систем цивільного захисту, а в подальшому – масштабований або доповнений новими функціями (наприклад, підрахунком за декількома технологіями одночасно або централізованим збиранням статистики).

ВИСНОВКИ

У межах даної дипломної роботи було здійснено повний цикл розробки IoT-модуля для автоматичного підрахунку кількості людей у захисних спорудах цивільного призначення. Поставлена мета – створення недорогого, енергоефективного, автономного та зручного у впровадженні рішення – була досягнута шляхом реалізації системи на базі одноплатного комп’ютера Raspberry Pi Zero 2 W з використанням технологій Bluetooth-сканування та локального вебінтерфейсу.

Виконані завдання:

– проаналізовано існуючі методи підрахунку людей у приміщеннях;

– обрано апаратну платформу для IoT-модуля та засоби збору даних;

– реалізовано програмну частину для виявлення та підрахунку персональних гаджетів, за якими визначається кількість людей у бомбосховищі;

– забезпечено відображення зібраної інформації у зручному вигляді;

– перевірено працездатність розробленого IoT-модуля..

Першим етапом стало вивчення існуючих підходів до підрахунку людей у закритих приміщеннях. Проаналізовано методи відеоспостереження, інфрачервоні сенсори, а також бездротові технології. Було визначено, що найбільш доцільним у контексті бомбосховищ є саме використання Bluetooth/WiFi-аналізу як конфіденційного, автономного та простого в реалізації методу.

У процесі проєктування було відібрано оптимальні апаратні компоненти, зокрема Raspberry Pi Zero 2 W завдяки його компактності, наявності вбудованих модулів Wi-Fi і Bluetooth, низькому енергоспоживанню та активній спільноті підтримки. Для зменшення перегріву пристрою було встановлено кулер, а для підключення живлення – впаяно роз’єм USB Type-C.

Особливу увагу приділено створенню корпусу пристрою. Було проведено порівняльний аналіз різних типів пластикових матеріалів для 3D-друку, і в результаті вибрано PETG як найбільш термостійкий, міцний та

придатний для умов експлуатації в укриттях. Сам корпус було спроєктовано індивідуально та надруковано на 3D-принтері Bambu Lab X1 Carbon.

На програмному рівні було налаштовано Raspberry Pi OS Lite, підготовлено автоматичний запуск скриптів Bluetooth-сканування та створено вебінтерфейс, що дозволяє переглядати зібрані дані через локальну мережу. Особливістю інтерфейсу є його інтуїтивна візуалізація: користувач бачить кількість активних пристроїв, MAC-адреси, назви, статуси та час виявлення.

Під час експериментального тестування система показала стабільну роботу: розроблений IoT-модуль коректно визначав кількість пристроїв поблизу, ідентифікував випадкові MAC-адреси, оцінював орієнтовну кількість людей з достатньо високою точністю та виводив усі дані у реальному часі. Робота комплексу не потребує постійного доступу до інтернету чи монітору, що робить його зручним для розгортання в реальних умовах.

Загалом, розроблений IoT-модуль:

– забезпечує автономний підрахунок людей у бомбосховищі;
– не порушує приватність осіб;
– працює у режимі реального часу;
– має зручний інтерфейс для перегляду інформації;
– є масштабованим та придатним до інтеграції в більші системи моніторингу.

Отримані результати підтверджують, що запропонований підхід є ефективним та актуальним в умовах надзвичайних ситуацій та може бути рекомендований для створення комплексу із сукупності IoT-модулів визначення кількості людей у декількох бомбосховищах та впровадження запропонованого рішення у системах цивільного захисту.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. A brief overview of the websocket protocol – noio_ws 0.0.5 documentation. noio_ws. URL: https://noio-ws.readthedocs.io/en/latest/overview_of_websockets.html (Last accessed: 14.06.2025).

2. Banana Pi BPI-M2 Zero. RKS Components - RADIOMAG. URL: https://www.rcscomponents.kiev.ua/product/banana%20pi%20bpi-m2%20zero.html (Last accessed: 14.06.2025).

3. Buy LILYGO ESP32 and LoRa modem SX1276 868MHz TTGO V1.3 development board in Kiev and Ukraine. Arduino in Ukraine. URL: https://arduino.ua/ru/prod2901-plata-rozrobnika-lilygo-esp32-i-lora-modema-sx1276 -868mgc-ttgo-v1-3?srsltid=AfmBOor8ZiiekfcxwmB6QftL04DGm_rhOaQD_ST4yr51i-n6-kwOWOKU (Last accessed: 14.06.2025).

4. Documentation – Leaflet – a JavaScript library for interactive maps. URL: https://leafletjs.com/reference.html#popupevent (Last accessed: 08.05.2025).

5. Graceful degradation of websocket connections to nonpersistent HTTP-based communications : Сполучені Штати Америки (США) : US9143550B2. Заявл. 01.12.2012 ; опубл. 22.09.2015.

6. Information technology – Message Queuing Telemetry Transport (MQTT) (ISO/IEC 20922:2016). ISO. URL: https://www.iso.org/standard/69466.html (Last accessed: 08.05.2025).

7. Kondratenko Y., Kondratenko G., Sidenko I. Multi-criteria selection of the wireless communication technology for specialized IoT network. *Proc. of the 14th Interna-tional Conference on ICT in Education, Research and Industrial Applications. Inte-gration, Harmonization and Knowledge Transfer, Workshops* (ICTERI), Kyiv, Ukraine. 2018. Vol. 2104. P. 501–516.

8. Krainyk Y. Information technology of university class internet-of-things-module. *CEUR Workshop Proceedings*. 2019. Vol. 2516. P. 58–68.

9. MQTT (Quality of Service) QoS levels explained | Cedalo. Cedalo. URL: https://cedalo.com/blog/understanding-mqtt-qos/ (Last accessed: 14.06.2025).

10. Mykhaylova R., Antonenko I., Ryabova S. Modern BOMB shelter: current tasks and modernization. *Proc. of the V International Scientific and Practical Conference "Current Problems of Modern Design"*, Kyiv, KNUTD, April 27, 2023. P. 217–219. URL: https://er.knutd.edu.ua/handle/123456789/24751 (Last accessed: 14.06.2025).

11. National Marine Electronics Association. Pratiques et Techniques de la Plaisance. URL: https://www.plaisance-pratique.com/IMG/pdf/NMEA0183-2.pdf (Last accessed: 08.05.2025).

12. Omega2+. Onion – Compute Platform for IoT. URL: https://onion.io/store/omega2p/ (Last accessed: 14.06.2025).

13. Raspberry Pi Datasheets. URL: https://datasheets.raspberrypi.com/rpizero2/raspberry-pi-zero-2-w-product-brief.pdf (Last accessed: 14.06.2025).

14. Raspberry Pi Documentation – Configuration. Raspberry Pi. URL: https://www.raspberrypi.com/documentation/computers/configuration.html (Last accessed: 08.05.2025).

15. Raspberry Pi Kiosk using Chromium. Pi My Life Up. URL: https://pimylifeup.com/raspberry-pi-kiosk/ (Last accessed: 08.05.2025).

16. Raspberry Pi WiFi Setup: Here are 5 Methods for Raspbian. Raspberry Expert. URL: https://raspberryexpert.com/raspberry-pi-wifi-setup/ (Last accessed: 14.06.2025).

17. Raspberry Pi wins MacRobert Award from the Royal Academy of Engineering: University of Cambridge news. URL: https://www.staff.admin.cam.ac.uk/awards/raspberry-pi-wins-macrobert-award-from-the-royal-academy-of-engineering/ (Last accessed: 08.05.2025).

18. SIM7600G-H 4G HAT (B) – Waveshare Wiki. Waveshare Electronics. URL: https://www.waveshare.com/wiki/SIM7600G-H_4G_HAT_(B) (Last accessed: 08.05.2025).

19. TK-01 WI-FI | Visitor counters and anti-theft systems. TK Systems. URL: https://tksystems.com.ua/visitor-counters/tk-01-wifi (Last accessed: 14.06.2025).

20. Viatec. DS-2CD6825G0/C-IVS(B) 2МП (2мм). IP video camera Hikvision. VIATEC.UA. URL: https://viatec.ua/ru/product/DS-2CD6825G0-C-IVS-B?srsltid=AfmBOoqgN33aFIrH8AU2_fa5sJ3-JWbCrmCdeUkLki21fz-2T1MdH1E4 (Last accessed: 14.06.2025).

21. Web sockets vs Ajax . EDUCBA. URL: https://www.educba.com/web-sockets-vs-ajax/ (Last accessed: 14.06.2025).

ДОДАТОК А

Код програмного забезпечення апаратної частини IoT-модуля

bluetooth_scanner.py

```
#!/usr/bin/env python3
import sqlite3, subprocess, time, threading, json, math, re, logging
from datetime import datetime, timedelta
from collections import deque

# Простий Калманівський фільтр для згладжування RSSI
class KalmanFilter:
    def __init__(self, process_var=1e-3, measure_var=0.1):
        self.pv, self.mv = process_var, measure_var
        self.estimate, self.error_estimate = 0.0, 1.0
        self.initialized = False
    def update(self, measurement):
        if not self.initialized:
            self.estimate = measurement
            self.initialized = True
            return measurement
        priori = self.estimate
        error = self.error_estimate + self.pv
        k = error / (error + self.mv)
        self.estimate = priori + k * (measurement - priori)
        self.error_estimate = (1 - k) * error
        return self.estimate

# Клас для відстеження пристроїв
class DeviceTracker:
    def __init__(self, mac, first_seen, rssi=None, name="Unknown"):
        self.mac, self.name = mac, name
        self.first_seen = self.last_seen = first_seen
        self.rssi_filter = KalmanFilter()
        self.filtered_rssi = rssi or -100
        self.appearances = [(first_seen, rssi)]

    def update(self, timestamp, rssi=None):
        self.last_seen = timestamp
        self.appearances.append((timestamp, rssi))
        if rssi: self.filtered_rssi = self.rssi_filter.update(rssi)

    def get_distance(self):
        tx_power = -59  # Стандартне значення для BLE
        n = 2.0         # Ступінь загасання
        return math.pow(10, (tx_power - self.filtered_rssi) / (10 * n))

# Основний клас сканування
class BluetoothScanner:
    def __init__(self, db_path='devices.db'):
        self.db_path = db_path
        self.trackers = {}
        self.init_db()

    def init_db(self):
        conn = sqlite3.connect(self.db_path)
        cursor = conn.cursor()
        cursor.execute('''
            CREATE TABLE IF NOT EXISTS devices (
                mac TEXT PRIMARY KEY,
                name TEXT,
                first_seen TIMESTAMP,
                last_seen TIMESTAMP,
                rssi INTEGER,
                distance REAL
            )
        ''')
        conn.commit()
```

```
        conn.close()

    def scan(self):
        devices = []
        subprocess.run(['sudo', 'hciconfig', 'hci0', 'up'])
        result = subprocess.run(['sudo', 'hcitool', 'scan'], capture_output=True,
text=True)
        for line in result.stdout.split('\n')[1:]:
            if line.strip():
                parts = line.strip().split('\t')
                if len(parts) >= 2:
                    mac, name = parts[0], parts[1]
                    devices.append((mac, name))
        return devices

    def process_scan(self):
        current_time = datetime.now()
        for mac, name in self.scan():
            if mac not in self.trackers:
                self.trackers[mac] = DeviceTracker(mac, current_time, name=name)
            self.trackers[mac].update(current_time)

    def save_to_db(self):
        conn = sqlite3.connect(self.db_path)
        cursor = conn.cursor()
        for mac, tracker in self.trackers.items():
            distance = tracker.get_distance()
            cursor.execute('''
                INSERT OR REPLACE INTO devices
                (mac, name, first_seen, last_seen, rssi, distance)
                VALUES (?, ?, ?, ?, ?, ?)
            ''', (mac, tracker.name, tracker.first_seen, tracker.last_seen,
                  tracker.filtered_rssi, distance))
        conn.commit()
        conn.close()

    def run(self, interval=30):
        while True:
            self.process_scan()
            self.save_to_db()
            print(f"[{datetime.now()}] Активних пристроїв: {len(self.trackers)}")
            time.sleep(interval)

if __name__ == '__main__':
    scanner = BluetoothScanner()
    try:
        scanner.run()
    except KeyboardInterrupt:
        print("Сканування завершено.")
```

ДОДАТОК Б

Код інтерфейсу IoT-модуля

dashboard.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8"/>
<meta content="width=device-width, initial-scale=1.0" name="viewport"/>
<title>Enhanced Bluetooth People Counter</title>
<style>
        * {
            margin: 0;
            padding: 0;
            box-sizing: border-box;
        }

        body {
            font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, Arial,
sans-serif;
            background: linear-gradient(135deg, #1e3c72 0%, #2a5298 100%);
            min-height: 100vh;
            color: #333;
        }

        .container {
            max-width: 1400px;
            margin: 0 auto;
            padding: 20px;
        }

        .header {
            text-align: center;
            margin-bottom: 30px;
            color: white;
        }

        .header h1 {
            font-size: 2.5rem;
            margin-bottom: 10px;
            text-shadow: 2px 2px 4px rgba(0,0,0,0.3);
        }

        .header p {
            font-size: 1.1rem;
            opacity: 0.9;
        }

        .mode-indicator {
            background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
            color: white;
            padding: 10px 20px;
            border-radius: 25px;
            margin: 10px auto;
            display: inline-block;
            font-weight: 500;
        }

        .people-count-section {
            background: white;
            border-radius: 15px;
            padding: 30px;
            margin-bottom: 30px;
            box-shadow: 0 10px 30px rgba(0,0,0,0.1);
            text-align: center;
        }
```

```
.people-count-big {
    font-size: 4rem;
    font-weight: bold;
    color: #2a5298;
    margin-bottom: 10px;
}

.people-count-label {
    font-size: 1.2rem;
    color: #666;
    margin-bottom: 20px;
}

.confidence-bar {
    width: 100%;
    height: 20px;
    background: #eee;
    border-radius: 10px;
    overflow: hidden;
    margin-bottom: 10px;
}

.confidence-fill {
    height: 100%;
    background: linear-gradient(90deg, #ff6b6b 0%, #feca57 50%, #48dbfb 100%);
    border-radius: 10px;
    transition: width 0.3s ease;
}

.stats-grid {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));
    gap: 20px;
    margin-bottom: 30px;
}

.stat-card {
    background: white;
    border-radius: 12px;
    padding: 25px;
    text-align: center;
    box-shadow: 0 8px 32px rgba(0,0,0,0.1);
}

.stat-number {
    font-size: 2.5rem;
    font-weight: bold;
    margin-bottom: 10px;
}

.stat-label {
    font-size: 1rem;
    color: #666;
    text-transform: uppercase;
    letter-spacing: 1px;
}

.randomized { color: #e74c3c; }
.static { color: #27ae60; }
.active { color: #3498db; }
.total { color: #9b59b6; }
.people { color: #f39c12; }

.controls {
    text-align: center;
    margin-bottom: 30px;
}

.btn {
    background: linear-gradient(45deg, #667eea, #764ba2);
    color: white;
```

```
    border: none;
    padding: 12px 24px;
    border-radius: 25px;
    font-size: 1rem;
    cursor: pointer;
    margin: 0 10px;
    transition: all 0.3s ease;
    box-shadow: 0 4px 15px rgba(0,0,0,0.2);
}

.btn:hover {
    transform: translateY(-2px);
    box-shadow: 0 6px 20px rgba(0,0,0,0.3);
}

.devices-section {
    background: white;
    border-radius: 12px;
    padding: 30px;
    box-shadow: 0 8px 32px rgba(0,0,0,0.1);
}

.devices-header {
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin-bottom: 25px;
    padding-bottom: 15px;
    border-bottom: 2px solid #f0f0f0;
}

.devices-title {
    font-size: 1.5rem;
    color: #333;
    font-weight: 600;
}

.last-updated {
    color: #666;
    font-size: 0.9rem;
}

.devices-table {
    width: 100%;
    border-collapse: collapse;
    margin-top: 20px;
}

.devices-table th,
.devices-table td {
    padding: 15px;
    text-align: left;
    border-bottom: 1px solid #eee;
}

.devices-table th {
    background: #f8f9fa;
    font-weight: 600;
    color: #333;
    position: sticky;
    top: 0;
}

.devices-table tr:hover {
    background: #f8f9fa;
}

.mac-address {
    font-family: 'Courier New', monospace;
    font-weight: bold;
    color: #667eea;
```

```
        }

        .device-name {
            font-weight: 500;
        }

        .method-badge {
            padding: 4px 8px;
            border-radius: 12px;
            font-size: 0.7rem;
            color: white;
        }

        .method-advanced { background: #e74c3c; }
        .method-standard { background: #27ae60; }

        .loading {
            text-align: center;
            padding: 40px;
            color: #666;
        }

        .spinner {
            border: 3px solid #f3f3f3;
            border-top: 3px solid #667eea;
            border-radius: 50%;
            width: 30px;
            height: 30px;
            animation: spin 1s linear infinite;
            margin: 0 auto 20px;
        }

        @keyframes spin {
            0% { transform: rotate(0deg); }
            100% { transform: rotate(360deg); }
        }

        .info-panel {
            background: linear-gradient(135deg, #fff3cd 0%, #fefefe 100%);
            border: 1px solid #ffc107;
            border-radius: 12px;
            padding: 20px;
            margin-bottom: 30px;
        }

        .info-content {
            color: #856404;
            line-height: 1.6;
        }

        @media (max-width: 768px) {
            .container {
                padding: 10px;
            }

            .header h1 {
                font-size: 2rem;
            }

            .stats-grid {
                grid-template-columns: 1fr;
            }

            .people-count-big {
                font-size: 3rem;
            }
        }
    </style>
</head>
<body>
```

```
<!-- Скорочений фрагмент HTML-коду вебінтерфейсу -->

<!-- ...решта коду опущена для стислості... -->
</body>
</html>
```