Міністерство освіти і науки України

Національний технічний університет
«Дніпровська політехніка»

Навчально-науковий
інститут електроенергетики
(інститут)

Факультет інформаційних технологій
(факультет)
Кафедра інформаційних технологій та комп'ютерної інженерії
(повна назва)

# ПОЯСНЮВАЛЬНА ЗАПИСКА
## кваліфікаційної роботи ступеня бакалавра

**здобувача** Южаков Олександр Дементійович
(ПІБ)

**академічної групи** 123-22ск-1
(шифр)

**спеціальності** 123 Комп'ютерна інженерія
(код і назва спеціальності)

**за освітньо-професійною програмою** 123 Комп'ютерна інженерія
(офіційна назва)

**на тему** "Розподілена IoT-система для контролю доступу та безпеки в офісному просторі"
(назва за наказом ректора)

| Керівники | Прізвище, ініціали | Оцінка за шкалою | | Підпис |
|---|---|---|---|---|
| | | рейтинговою | інституційною | |
| кваліфікаційної роботи | доц. Каштан В.Ю. | | | |
| загального розділу | доц. Каштан В.Ю. | | | |
| спеціального розділу | доц. Каштан В.Ю. | | | |
| **Рецензент** | | | | |
| **Нормоконтролер** | проф. Цвіркун Л.І. | | | |

**Дніпро**
**2025**

**ЗАТВЕРДЖЕНО:**
завідувач кафедри
інформаційних технологій
та комп'ютерної інженерії
(повна назва)
_____ Гнатушенко В.В.
(підпис) (прізвище, ініціали)

*"____" _____ 2025 року*

# ЗАВДАННЯ
## на кваліфікаційну роботу
## ступеня бакалавр

**здобувача** Южаков О.Д. **академічної групи** 123-22ск-1
(прізвище та ініціали) (шифр)

**спеціальності** 123 Комп'ютерна інженерія

**за освітньо-професійною програмою** Комп'ютерна інженерія
(офіційна назва)

**на тему** "Розподілена IoT-система для контролю доступу та безпеки в офісному просторі "

затверджену наказом ректора НТУ «Дніпровська політехніка» від 05.05.2025 № 336-с

| Розділ | Зміст | Термін виконання |
|---|---|---|
| Загальний розділ | На основі матеріалів виробничих практик, інших науково-технічних джерел показати актуальність завдання, сформулювати мету та задачі виконання кваліфікаційної роботи | 10.02.2025 |
| Спеціальний розділ | Сформулювати найменування й призначення розподіленої IoT-системи для контролю доступу та безпеки в офісному просторі, висунути технічні вимоги до неї | 15.03.2025 |
| | Реалізувати розподілену IoT-систему, використовуючи відповідні інструменти програмування та враховуючи принципи розробки надійних та ефективних програмних систем, що є частиною комп'ютерної інженерії | 07.05.2025 |
| | Протестувати розроблену розподілену IoT-систему для контролю доступу та безпеки в офісному просторі на предмет її функціональності, продуктивності обміну даними в реальному часі та стійкості до можливих збоїв | 31.05.2025 |

**Завдання видано** _____ доц. Каштан В.Ю.
(підпис керівника) (прізвище, ініціали)

**Дата видачі** 25.01.2025

**Дата подання до екзаменаційної комісії** _____

**Прийнято до виконання** _____ Южаков О.Д.
(підпис студента) (прізвище, ініціали)

# РЕФЕРАТ

Пояснювальна записка: 84 с., 25 рис., 3 табл., 1 додаток, 12 джерел.

Об'єктом дослідження є розподілені IoT-системи та їх застосування для підвищення рівня безпеки та оптимізації контролю доступу в офісних просторах.

Предметом дослідження є функціональні моделі окремих підсистем контролю доступу та безпеки (таких як RFID-замок, розумне вікно, система автономного живлення, система захисту від несанкціонованого проникнення, автоматизація їдальні, система відтворення музики, розумний вуличний ліхтар, автоматичні ворота гаража та система пожежної/димової сигналізації), їх моделювання в середовищі Cisco Packet Tracer, аналіз функціональності та ефективності, а також визначення шляхів інтеграції в єдину розподілену IoT-систему.

Метою роботи є розробка та дослідження концепції розподіленої IoT-системи для підвищення рівня безпеки та оптимізації контролю доступу в офісному просторі шляхом моделювання та аналізу окремих функціональних підсистем в середовищі Cisco Packet Tracer.

Для досягнення поставленої мети необхідно вирішити наступні задачі:

1. Розробити функціональні моделі окремих підсистем контролю доступу та безпеки в офісному просторі.

2. Здійснити моделювання розроблених підсистем в середовищі Cisco Packet Tracer.

3. Проаналізувати функціональність та ефективність змодельованих підсистем.

4. Визначити можливі шляхи інтеграції розроблених підсистем у єдину розподілену IoT-систему.

Ключові слова: IoT, RFID-системи, розумний офіс, управління доступом, контроль доступу.

# ЗМІСТ

# ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАК, ОДИНИЦЬ ТА ТЕРМІНІВ

IoT – Інтернет речей

MQTT – Message Queuing Telemetry Transport

RFID – технологія радіочастотної ідентифікації

MAC – Media Access Control

# ВСТУП

Технології Інтернету речей (IoT), що динамічно розвиваються, глибоко інтегруються в сучасне життя, революціонізуючи способи керування та автоматизації в численних сферах. Особливо актуальним є застосування IoT-систем у комерційних просторах, таких як офіси, де питання безпеки, контролю доступу та оптимізації енергоспоживання набувають першочергового значення.

Розвиток мікроелектроніки, бездротових мереж та хмарних обчислень створив сприятливі умови для розробки та впровадження розподілених IoT-систем, здатних забезпечити ефективне управління різноманітними аспектами офісного середовища. Такі системи відкривають нові можливості для підвищення рівня безпеки, автоматизації рутинних завдань, оптимізації використання ресурсів та створення комфортного робочого простору.

У зв'язку з цим, метою даної кваліфікаційної роботи є розробка та дослідження розподіленої IoT-системи для контролю доступу та забезпечення безпеки в офісному просторі. В рамках дослідження було спроєктовано та змодельовано в середовищі Cisco Packet Tracer ряд функціональних підсистем, що демонструють можливості інтеграції різноманітних датчиків, виконавчих механізмів та мережевих технологій для вирішення конкретних завдань.

Зокрема, в роботі реалізовано підсистеми: контролю доступу на основі RFID-ідентифікації, інтелектуального керування вікнами з урахуванням освітленості та опадів, автономного живлення на сонячних батареях, захисту від несанкціонованого проникнення, автоматизації роботи побутових приладів у їдальні, мультимедійна система для розваг, інтелектуального вуличного освітлення, автоматизованого керування гаражними воротами, а також система пожежної та димової сигналізації.

Практична реалізація зазначених підсистем у середовищі Cisco Packet Tracer дозволила продемонструвати принципи їхньої взаємодії в рамках єдиної розподіленої IoT-архітектури, а також оцінити потенційні переваги

їхнього впровадження для підвищення рівня безпеки та комфорту в офісному просторі.

Подальші розділи роботи будуть присвячені детальному опису розробленої IoT-системи, аналізу функціональності кожної підсистеми, опису процесу моделювання в Cisco Packet Tracer, а також обговоренню отриманих результатів та перспектив подальших досліджень у даній області.

# 1 ЗАГАЛЬНИЙ РОЗДІЛ

## 1.1 Роль і значення розподілених IoT-систем для контролю доступу та безпеки в офісному просторі

Концепція Інтернету речей (IoT) відкриває нові горизонти для автоматизації та інтелектуального управління різноманітними аспектами життєдіяльності людини, зокрема в комерційному секторі. Традиційна ідея "розумного дому", спрямована на підвищення комфорту та ефективності управління побутовими системами, еволюціонує в ширшу концепцію "розумного простору", яка знаходить своє застосування в офісних будівлях. Впровадження комплексних автоматизованих систем контролю в офісному середовищі має вирішальне значення для забезпечення безпеки, оптимізації доступу та підвищення загальної ефективності функціонування.

Розподілені IoT-системи є ключовим елементом у створенні інтелектуального офісного простору. На відміну від централізованих систем, розподілена архітектура передбачає використання мережі взаємопов'язаних пристроїв, датчиків та виконавчих механізмів, кожен з яких має певну автономність та здатність обмінюватися даними. Такий підхід забезпечує підвищену гнучкість, масштабованість та надійність системи в цілому.

Розширення концепції "розумного простору" в офісному середовищі є закономірним етапом розвитку технологій Інтернету речей (IoT). Якщо раніше фокус був переважно на автоматизації побутових завдань у межах приватного житла, то сьогодні інтелектуальні системи все активніше впроваджуються в комерційні будівлі, зокрема в офіси. Це зумовлено зростаючим розумінням того, що ефективне управління офісним простором напряму впливає на продуктивність співробітників, безпеку даних та фізичну безпеку об'єкту, а також на оптимізацію експлуатаційних витрат.

Ключову роль у створенні такого інтелектуального офісного середовища відіграють розподілені IoT-системи. Щоб краще зрозуміти їхнє значення, варто провести межу між ними та традиційними, централізованими

системами безпеки та контролю доступу. У централізованих системах обробка даних та прийняття рішень зазвичай відбуваються на одному або кількох центральних серверах. Всі датчики та виконавчі пристрої безпосередньо підключені до цих серверів, що може створювати вузькі місця в продуктивності та знижувати надійність системи в цілому. У випадку виходу з ладу центрального елемента, вся система може припинити функціонування.

На противагу цьому, розподілена IoT-архітектура передбачає мережу взаємопов'язаних інтелектуальних пристроїв, кожен з яких має певну обчислювальну потужність та здатність локально обробляти дані та приймати рішення. Ці пристрої обмінюються інформацією між собою та, за потреби, з центральним сервером або хмарною платформою. Така децентралізація забезпечує ряд важливих переваг [1]:

– підвищена гнучкість – додавання нових пристроїв та розширення функціональності системи стає значно простішим, оскільки не вимагає значних змін у центральній інфраструктурі. Нові датчики та виконавчі механізми можуть інтегруватися в існуючу мережу з меншими зусиллями. Наприклад, встановлення додаткової точки контролю доступу на новому поверсі може бути здійснено шляхом простого підключення нового RFID-зчитувача та контролера до існуючої мережі, без необхідності прокладання складних кабельних магістралей до центрального сервера;

– краща масштабованість – розподілені системи легше масштабуються для обслуговування великих офісних будівель або навіть кількох офісів, об'єднаних в одну мережу. Зі зростанням кількості приміщень та співробітників, до системи можуть додаватися нові пристрої без значного впливу на продуктивність вже існуючих. Наприклад, мережа датчиків температури та вологості може бути легко розширена на нові офісні кімнати без перевантаження центрального контролера;

– підвищена надійність – у випадку виходу з ладу окремого пристрою, вся система в більшості випадків продовжує функціонувати,

оскільки інші пристрої зберігають свою працездатність та можуть частково перебрати на себе функції несправного елемента. Наприклад, якщо один з датчиків руху вийде з ладу, інші датчики в тій же зоні можуть продовжувати виконувати функцію моніторингу;

– зменшення затримки – локальна обробка даних дозволяє зменшити час реакції системи на події. Наприклад, при зчитуванні RFID-картки, рішення про відкриття дверей може бути прийнято локальним контролером практично миттєво, без необхідності передачі даних на центральний сервер та отримання відповіді. Це особливо важливо для систем безпеки, де швидка реакція на загрозу може бути критичною.

Об'єктом вивчення даної кваліфікаційної роботи є саме розподілена IoT-система, призначена для вирішення критично важливих задач контролю доступу та забезпечення безпеки в офісному просторі.

Актуальність вирішуваної задачі - створення розподіленої IoT-системи для контролю доступу та безпеки в офісному просторі - зумовлена низкою важливих факторів, які відображають сучасні потреби та виклики в управлінні комерційною нерухомістю:

– сучасний бізнес стикається з різноманітними загрозами, починаючи від несанкціонованого проникнення та крадіжок, закінчуючи кіберзагрозами та необхідністю захисту конфіденційної інформації. Традиційні системи безпеки часто є фрагментованими та не забезпечують комплексного підходу до виявлення та запобігання цим загрозам. Розподілені IoT-системи дозволяють інтегрувати різні рівні захисту, використовуючи різноманітні методи ідентифікації (RFID, біометрія), постійний моніторинг (відеокамери, датчики руху, відчинення), а також системи оповіщення та реагування в реальному часі. Наприклад, інтеграція системи контролю доступу з системою відеоспостереження дозволяє не лише фіксувати факт проходу, але й візуально ідентифікувати особу, а у випадку спроби несанкціонованого проникнення - автоматично активувати сигналізацію та сповістити службу безпеки;

– сучасні офісні простори часто мають складну структуру з різними зонами доступу, вимогами до конфіденційності та графіками роботи. Розподілені IoT-системи надають можливість гнучко налаштовувати права доступу для різних категорій користувачів (співробітники різних відділів, тимчасові працівники, відвідувачі) до різних приміщень та в різний час. Централізоване управління правами доступу спрощує адміністрування та підвищує рівень безпеки. Наприклад, новому співробітнику може бути швидко надано доступ лише до необхідних йому приміщень, а при звільненні його права доступу можуть бути миттєво відкликані. Ведення детальних журналів подій дозволяє відстежувати переміщення співробітників та відвідувачів, що може бути корисним при розслідуванні інцидентів безпеки;

– офісні будівлі споживають значну кількість енергії на освітлення, опалення, вентиляцію та роботу різноманітного обладнання. Впровадження IoT-технологій дозволяє автоматизувати багато процесів, пов'язаних з безпекою та контролем доступу, що призводить до значної економії ресурсів та підвищення зручності використання офісного простору. Наприклад, інтеграція датчиків присутності з системою освітлення дозволяє автоматично вмикати світло лише в тих приміщеннях, де знаходяться люди, та вимикати його, коли приміщення порожнє. Аналогічно, моніторинг стану вікон та дверей може дозволити автоматично регулювати роботу системи кондиціонування;

– сучасні офісні будівлі часто оснащені різними інженерними системами (освітлення, клімат-контроль, пожежна сигналізація, відеоспостереження). Інтеграція цих систем на базі розподіленої IoT-архітектури дозволяє створити єдину екосистему управління офісним простором. Це не лише спрощує адміністрування та моніторинг, але й відкриває нові можливості для оптимізації роботи будівлі та підвищення рівня безпеки. Наприклад, при спрацюванні пожежної сигналізації, система контролю доступу може автоматично розблокувати всі евакуаційні виходи, а система вентиляції - увімкнути режим димовидалення.

## 1.2 Стандартизація та архітектура розподіленої IoT-системи контролю доступу в офісному просторі

Стандартизація систем Інтернету речей (IoT) є важливим аспектом для забезпечення їхньої широкої сумісності, інтероперабельності, безпеки та масштабованості. Хоча на початку розвитку IoT, особливо в період його значного зростання між 2010 і 2017 роками, спостерігався бурхливий розвиток різноманітних технологій і рішень, відсутність єдиних стандартів створювала певні перешкоди для масового впровадження та повної реалізації потенціалу IoT.

Період з 2010 по 2017 рік став переломним для IoT завдяки кільком ключовим факторам [2]:

– здешевлення мікроелектроніки та розвиток виробничих технологій призвели до появи широкого спектра недорогих сенсорів, мікроконтролерів та інших кінцевих пристроїв, що стало основою для створення різноманітних IoT-рішень;

– розвиток бездротових технологій (Wi-Fi, Bluetooth, Zigbee, LoRaWAN, NB-IoT) та протоколів зв'язку забезпечив можливість ефективної та, що важливо, безпечної передачі даних між пристроями та центральними системами;

– проникнення Інтернету та стандартизація IP-протоколів створили готову інфраструктуру для підключення та обміну даними між IoT-пристроями та хмарними сервісами.

Вищеописані фактори сприяли появі як простих, економічно ефективних рішень для окремих завдань, так і складних професійних систем, що охоплювали цілі підприємства та галузі (наприклад, концепції Індустрії 4.0 та ініціативи Консорціуму промислового Інтернету). Однак різноманітність технологій і підходів також виявила потребу в стандартизації для забезпечення сумісності та можливості інтеграції різних пристроїв і систем від різних виробників.

Незважаючи на думку деяких дослідників щодо потенційної

неможливості створення єдиного стандарту архітектури програмного забезпечення для IoT-застосунків через різноманітність контекстів застосування, зусилля з стандартизації активно ведуться на різних рівнях.

Консорціум промислового Інтернету (Industrial Internet Consortium - IIC) заснований у 2014 році, IIC (тепер частина Object Management Group - OMG) розробив методологію для промислових систем IoT. У 2015 році консорціумом було опубліковано рекомендації, що встановлюють стандарти для рівня побудови, складності, технічних деталей та особливостей впровадження IoT-систем у промислових умовах. Широке поширення цих документів серед західних компаній, що займаються розробкою промислових IoT-рішень, підкреслює їхню практичну цінність.

У межах впровадження розумних офісних середовищ важливу роль відіграє чітка структуризація IoT-системи, що забезпечує контроль доступу, моніторинг безпеки, обробку даних і збереження інформації. Загальна архітектура системи базується на трирівневій моделі IoT (perception – network – application), що відповідає сучасним підходам до стандартизації розподілених інформаційних систем (рис.1.1).

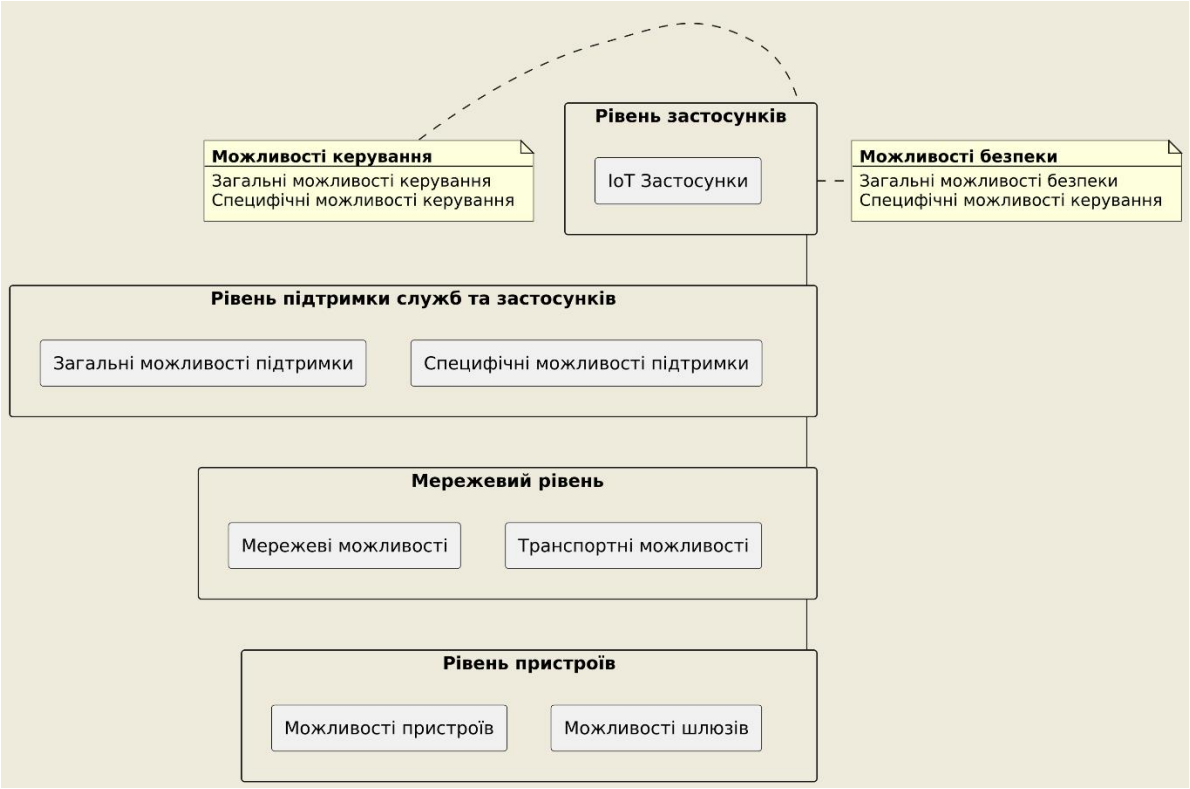



Рисунок 1.1 – Еталонна модель IoT

Рівень сприйняття (Perception Layer). Цей рівень включає фізичні пристрої, що безпосередньо взаємодіють з навколишнім середовищем [3]:

– кінцеві вузли (TerminalNode) – забезпечують зчитування параметрів (наприклад, руху, температури, вологості) і передачу інформації в мережу;

– бездротові сенсори (WirelessSensor) – виконують первинний збір даних, які передаються далі маршрутизуючими вузлами;

– маршрутизатори (RoutingNodes) – формують маршрути для передачі даних до вищого рівня;

– пристрої розумного офісу (SmartHomeDevices) – включають розумні замки, термостати, кондиціонери, розетки та монітори;

– інтелектуальний мобільний термінал (MobileTerminal) – мобільний застосунок користувача, що дозволяє дистанційно керувати доступом до офісного простору.

Мережевий рівень (Network Layer) забезпечує передачу,

маршрутизацію і попередню обробку даних:

– інтелектуальний шлюз (IntelligentGateway) – ключовий компонент, що об’єднує сенсорну мережу з базами даних і зовнішніми системами. Забезпечує шифрування, фільтрацію та агрегування інформації;

– керування мережею (NetworkManagement) – відповідає за конфігурацію та підтримку стабільності зв’язку між вузлами;

– керування передачею (Transmission Management) – модуль, що містить: основні дані (BasicData), параметри навколишнього середовища (EnvParameters), конфігурацію системи (SystemConfiguration), синтетичний інтерфейс вводу/виводу (SyntheticIOInterface).

Рівень застосування (Application Layer) забезпечує інтерпретацію та збереження отриманих даних, а також інтеграцію з користувацькими сервісами бази даних [2]:

– моніторинг навколишнього середовища (EnvMonitoringDB);

– віддалений моніторинг (RemoteMonitoringDB);

– домашня безпека (HomeSecurityDB);

– здоров'я сім'ї (FamilyHealthDB);

– економне управління (LeanManagementDB).

Базова архітектура розумного офісу представлена на рис.1.2 та являє собою багаторівневу систему, що включає в себе взаємодію фізичних пристроїв, мережевої інфраструктури, програмного забезпечення та інтерфейсів користувача для досягнення цілей автоматизації, безпеки, ефективності та комфорту в офісному просторі.

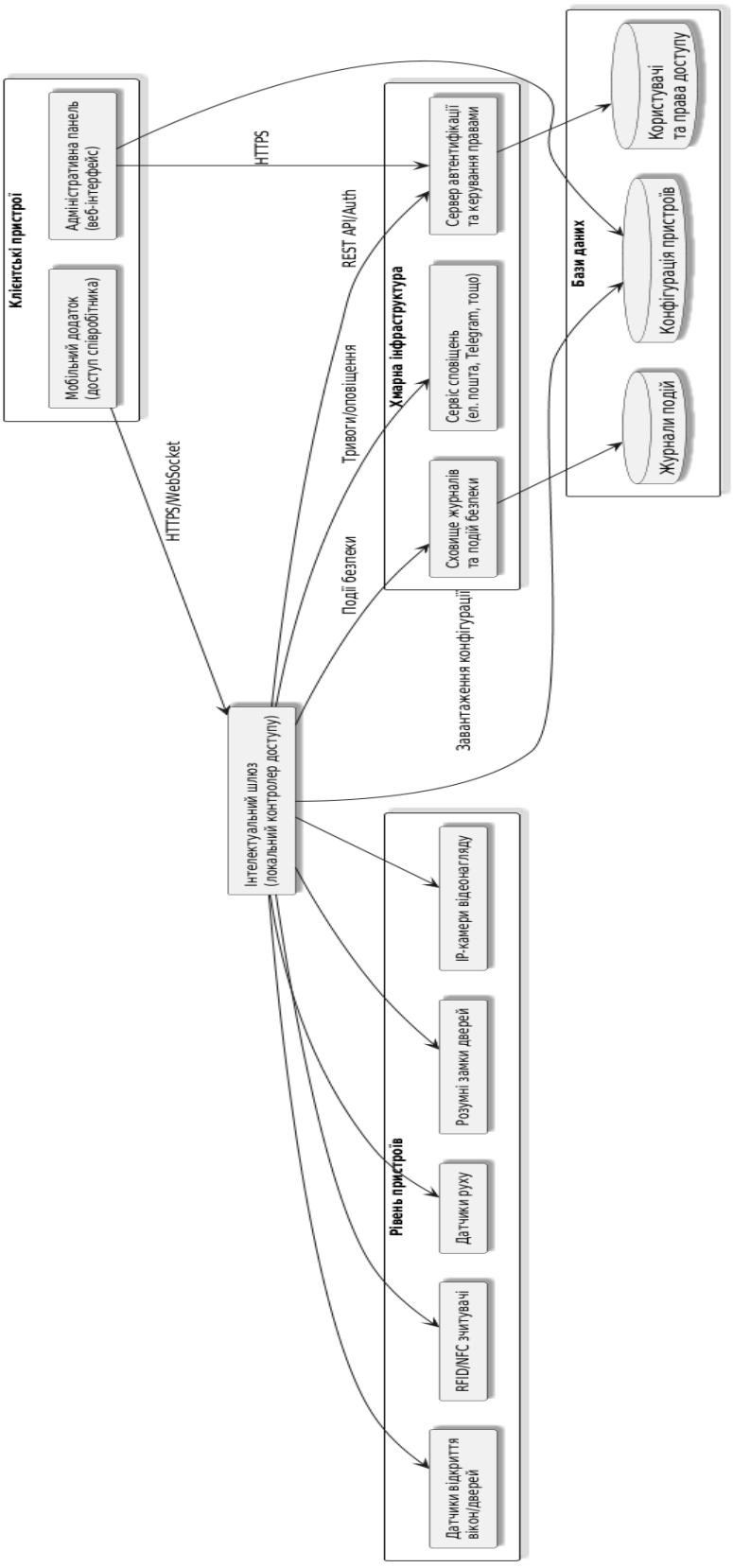



Рисунок 1.2 – Архітектура IoT-системи для контролю доступу до безпеки офісу

На рівні клієнтських пристроїв розміщено два основні інтерфейси:

мобільний додаток, який використовується співробітниками для здійснення доступу до офісу, та адміністративна панель, яка є веб-інтерфейсом для керування системою. Мобільний додаток з'єднується зі шлюзом через захищені протоколи (HTTPS або WebSocket), що дозволяє забезпечити автентифікацію користувача та здійснити перевірку доступу.

Інтелектуальний шлюз, який виступає як локальний контролер доступу, виконує роль центрального вузла, що взаємодіє з усіма фізичними пристроями системи безпеки, включаючи сенсори та виконавчі механізми. До нього підключені RFID/NFC-зчитувачі, датчики руху, розумні дверні замки, IP-камери відеонагляду, а також датчики відкриття вікон і дверей. Завдяки цьому шлюз отримує інформацію з сенсорів та ініціює відповідні дії в реальному часі.

Шлюз також обмінюється даними з хмарною інфраструктурою, яка складається з кількох ключових серверів. Сервер автентифікації та керування правами відповідає за перевірку користувачів і визначення їхніх прав доступу. Сховище журналів та подій безпеки накопичує дані про всі події в системі, що дозволяє забезпечити аудит і зворотній аналіз. Сервіс сповіщень передає тривоги та повідомлення через електронну пошту, месенджери (Telegram тощо) у випадку виникнення нештатних ситуацій.

На рівні баз даних система використовує три окремі сховища: базу даних користувачів і прав доступу, базу журналів подій та базу конфігурацій пристроїв. Сервер автентифікації взаємодіє з базою користувачів, сервер подій — з журналами, а адміністративна панель та шлюз – з конфігураціями пристроїв для їх актуалізації.

MQTT (Message Queuing Telemetry Transport) – це відкритий протокол обміну даними, який широко використовується у сфері Інтернету речей (IoT) завдяки своїй простоті, малій вимогливості до ресурсів та ефективності в умовах обмеженої пропускної здатності каналів зв’язку. MQTT працює поверх стеку TCP/IP і базується на моделі клієнт/сервер, де ключову роль відіграє брокер повідомлень, що функціонує як проміжний сервер.

У цій архітектурі всі пристрої (сенсори, приводи тощо) надсилають свої дані виключно брокеру та отримують повідомлення також лише від нього. Таким чином, брокер виступає центральним вузлом маршрутизації, який приймає повідомлення від клієнтів-видавців і пересилає їх клієнтам-передплатникам згідно з відповідними підписками. Така взаємодія реалізується через модель видавець–передплатник (publish–subscribe).

MQTT-брокер, який функціонує як TCP-сервер із динамічною базою даних, може також виконувати додаткові завдання, пов'язані з фільтрацією, аналізом або зберіганням отриманих даних. У складніших системах можлива взаємодія кількох брокерів між собою, коли один брокер підписується на повідомлення іншого для розширення функціональності або розподілу навантаження.

Основними об'єктами в мережі на основі MQTT є [4]:

– видавець (Publisher) – клієнт, який формує повідомлення на певну тему та надсилає його брокеру при настанні відповідної події;

– брокер (Broker) — центральний сервер MQTT, який приймає повідомлення від видавців і передає їх передплатникам. У великих системах брокер може також виконувати операції з обробки та агрегації даних;

– передплатник (Subscriber) — клієнт, що підписується на одну або кілька тем і постійно прослуховує брокера, отримуючи повідомлення, які відповідають його інтересам.

Таким чином, MQTT забезпечує ефективну, масштабовану та надійну передачу повідомлень у системах з великою кількістю вузлів. Принципова схема взаємодії між видавцем, брокером і передплатником подана на рис.1.3, де проілюстровано базову логіку функціонування протоколу. На схемі показано принцип функціонування протоколу MQTT за моделлю "видавець–брокер–передплатник". Видавці публікують повідомлення за певними темами до брокера, який маршрутизує їх до відповідних передплатників, підписаних на ці теми. Така архітектура забезпечує асинхронну доставку даних і розділення ролей у системах з великою кількістю пристроїв.

У лівій частині схеми показані видавці (Publisher) — це клієнти MQTT, які формують і надсилають повідомлення на певні теми (Topic), що їх стосуються. Кожен видавець ініціює передачу даних до центрального вузла — брокера MQTT (MQTT broker), розміщеного в центрі схеми. Повідомлення публікуються у вигляді пар "тема–повідомлення" (Publish('Topic', message)).

Центральним елементом є MQTT-брокер, який приймає повідомлення від усіх підключених видавців, зберігає їх у черзі повідомлень і пересилає далі відповідним абонентам. Брокер виступає посередником у мережі, забезпечуючи повну логіку маршрутизації та розповсюдження даних, не вимагаючи прямого зв'язку між джерелами та отримувачами.

У правій частині схеми знаходяться передплатники (Subscriber) — клієнти, які підписані на конкретні теми за допомогою команд типу Subscribe('Topic'). Передплатники постійно очікують надходження нових повідомлень і автоматично їх отримують, коли брокер отримує релевантну публікацію.

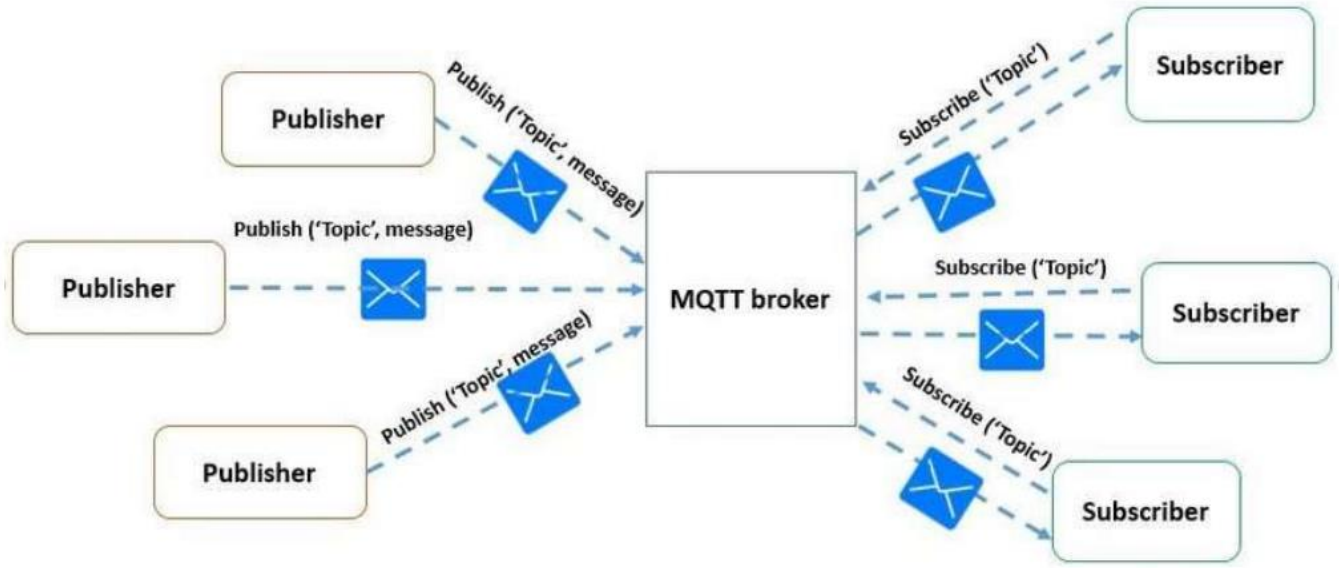



Рисунок 1.3 – Основна структура брокера MQTT

## 1.3 Огляд існуючих методів, технологій, аналогів систем

Сучасні системи контролю доступу та безпеки в офісних приміщеннях

активно розвиваються в контексті інтеграції технологій Інтернету речей (IoT). Основні підходи ґрунтуються на використанні RFID/NFC-карток, біометричних засобів і мобільних додатків для ідентифікації користувачів, а також застосуванні сенсорних технологій для виявлення руху, несанкціонованого проникнення та інших загроз безпеці.

В офісі системи контролю доступу та безпеки функціонально можуть бути поділені на такі основні частини (рис.1.4): система контролю доступу, система безпеки та система інформаційного забезпечення.

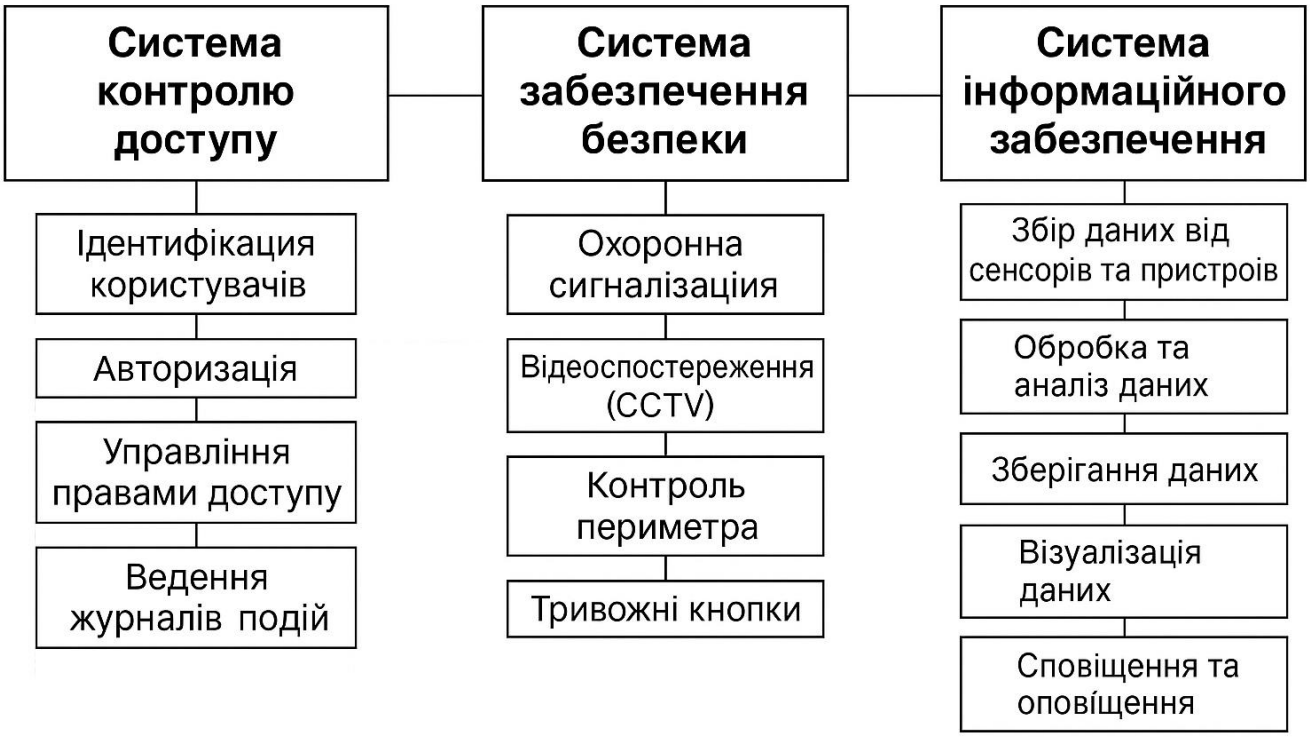



Рисунок 1.4 – Функціональна структура систем контролю доступу та безпеки в офісному просторі

Система контролю доступу забезпечує ідентифікацію та авторизацію осіб для входу в офіс загалом або до окремих його зон (кабінетів, серверних кімнат, складів тощо). Основні функції включають [4]:

–розпізнавання особи за допомогою різних методів (RFID-картки, PIN-коди, біометричні дані, мобільні додатки);

–перевірка прав доступу користувача до конкретних зон або ресурсів на основі встановлених правил та ролей;

–адміністрування списків авторизованих користувачів та їхніх рівнів доступу;

–реєстрація всіх спроб доступу, часу входу/виходу, спрацювань системи;

–фіксація часу приходу та відходу співробітників.

Система забезпечення безпеки спрямована на запобігання несанкціонованому проникненню, крадіжкам, вандалізму та іншим загрозам. Основні функції включають:

– виявлення спроб вторгнення за допомогою різних датчиків (руху, відчинення дверей/вікон, розбиття скла);

– візуальний моніторинг офісних приміщень та прилеглої території, запис подій для подальшого аналізу;

– забезпечення безпеки зовнішніх меж офісу (огорожі, ворота);

– забезпечення можливості швидкого виклику служби безпеки у випадку надзвичайної ситуації;

– координація дій у випадку пожежі.

Система інформаційного забезпечення забезпечує збір, обробку, зберігання та візуалізацію даних, пов'язаних з контролем доступу та безпекою. Основні функції включають:

– отримання інформації про стан дверей, рух, події доступу, відеопотоки тощо;

– фільтрація, агрегація та аналіз отриманих даних для виявлення аномалій, формування звітів та прийняття рішень;

– забезпечення надійного зберігання журналів подій, відеоархівів та іншої важливої інформації;

– представлення інформації у зручному для користувача форматі (панелі моніторингу, звіти);

– надсилання повідомлень про тривожні події або інші важливі статуси відповідальним особам.

У контексті IoT, системи контролю доступу та безпеки в офісі можуть

будуватися за різними архітектурними моделями. Централізовані системи (рис.1.5) передбачають підключення всіх датчиків, зчитувачів та виконавчих пристроїв до єдиного центрального контролера або сервера, який відповідає за обробку даних, прийняття рішень та управління всією системою.

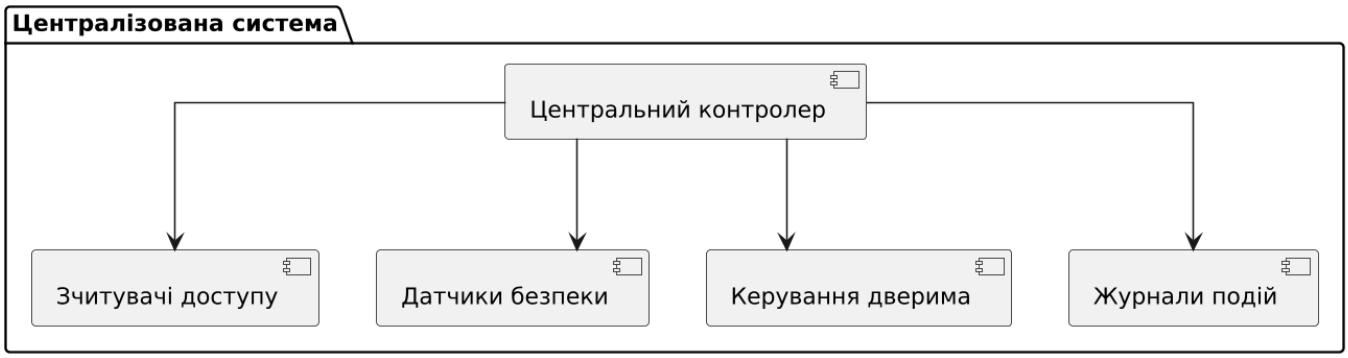



Рисунок 1.5 – Централізована система

На противагу цьому, децентралізовані (розподілені) системи характеризуються наявністю власного інтелекту у кожного пристрою або групи пристроїв, що дозволяє їм приймати локальні рішення, а зв'язок з центральною системою (за наявності) здійснюється через мережу (рис.1.6).

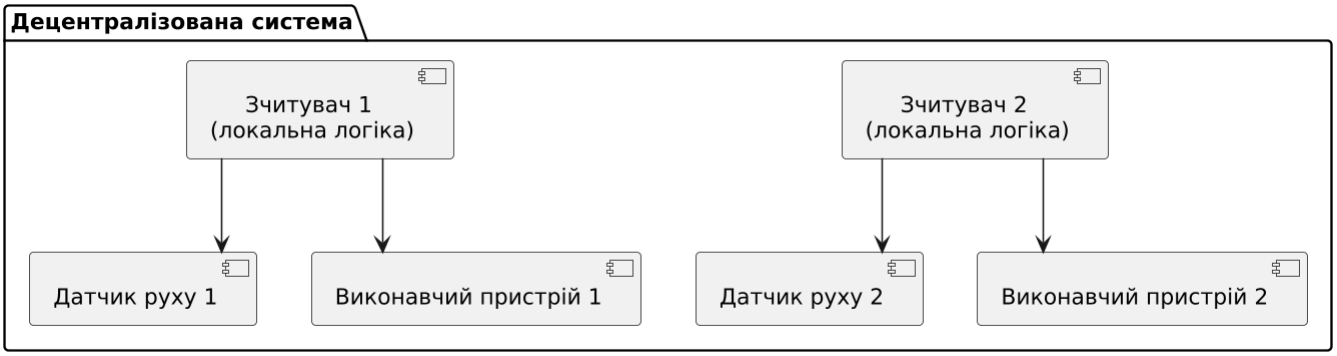



Рисунок 1.6 – Децентралізована система

Змішані системи поєднують елементи обох підходів, де частина функціональності виконується локально, а частина – на центральному рівні (рис.1.7).

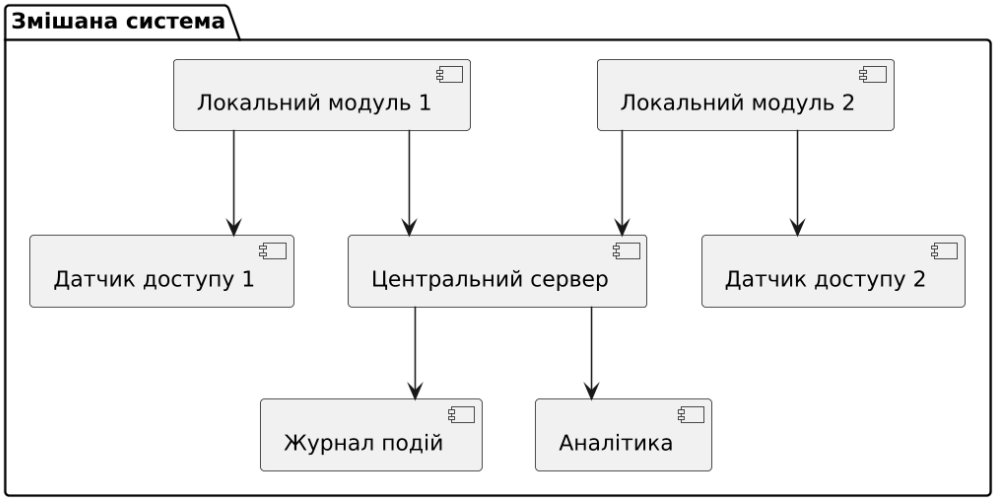



Рисунок 1.7 – Змішана система

Серед поширених архітектур домінує модель з централізованим сервером або хмарною платформою, яка виконує функції автентифікації, управління правами доступу, зберігання подій та аналітики. У такому підході сенсори та виконавчі пристрої комунікують з центральним вузлом через локальні шлюзи або безпосередньо через мережу Інтернет. Суттєвими перевагами таких рішень є можливість масштабування та централізоване управління, проте вони можуть мати обмежену стійкість до відмов або кіберзагроз у разі втрати зв’язку з основним сервером.

Microsoft Azure є хмарною платформою, яка поєднує інфраструктурні рішення як послугу (IaaS) (сервери, сховища даних, мережі, операційні системи) з широким спектром інструментів та сервісів, що полегшують розробку та розгортання хмарних застосунків (платформа як послуга або PaaS). Ці рішення дозволяють розробникам швидко створювати, розгортати та керувати великомасштабними програмами, адаптованими до конкретних потреб організацій. Служби Microsoft Azure допомагають клієнтам уникнути складнощів і додаткових витрат, пов'язаних із заміною існуючого обладнання та управлінням інфраструктурою, що сприяє ефективнішому управлінню IT-бюджетом. Клієнти платять лише за ті ресурси, які їм потрібні. 16 березня 2015 року корпорація Microsoft анонсувала Azure IoT Suite — хмарне рішення, призначене для допомоги корпоративним користувачам в адаптації

до зростаючого ландшафту Інтернету речей (IoT). Це рішення містить набір хмарних послуг, інтегрованих у середовище Azure, що дозволяє компаніям ініціювати власні проєкти IoT. Azure IoT Suite полегшує підключення різноманітних електронних продуктів з мережевими можливостями до хмари Microsoft Azure. Це забезпечує можливість збору, керування, зберігання та аналізу даних з цих підключених пристроїв, що дозволяє приймати важливі бізнес-рішення та автоматизувати процеси.

Microsoft Azure IoT Suite — це комплексне рішення корпоративного рівня, яке пропонує попередньо налаштовані та розширювані рішення для швидкого розгортання. Ці рішення підходять для типових сценаріїв Інтернету речей, таких як віддалений моніторинг і прогнозне обслуговування. Попередньо налаштовані компоненти, що входять до пакету Azure IoT, включають [5]:

–віртуальні пристрої – це готові до використання віртуальні пристрої, необхідні для початку роботи в середовищі IoT;

–попередньо налаштовані служби Azure – набір містить попередньо налаштовані служби Azure, які легко інтегруються з рішеннями IoT;

–консоль керування – кожне рішення постачається з окремою консоллю керування, розробленою для цього конкретного рішення.

Ці попередньо налаштовані рішення містять перевірений і готовий до використання код, який можна налаштувати та розширити відповідно до конкретних сценаріїв IoT.

В основі Azure IoT Suite лежить служба Azure IoT Hub, яка виконує роль платформи обміну повідомленнями між пристроями та хмарою. Вона служить шлюзом до хмари та інших ключових служб у пакеті IoT Suite. IoT Hub забезпечує надсилання та отримання повідомлень від пристроїв у реальному часі, доставку команд на пристрої та керування ними.

Azure Stream Analytics використовується для аналізу оперативних даних у межах попередньо налаштованих рішень. Він керує обробкою вхідних телеметричних даних, їх агрегацією та виявленням подій. Stream

Analytics застосовується для обробки інформаційних повідомлень, що містять метадані або відповіді на команди пристрою, полегшуючи обробку повідомлень пристроїв та їх доставку до інших служб.

Можливості зберігання даних забезпечуються службою Azure Document DB для зберігання метаданих пристроїв та надання функціональних можливостей керування пристроями в рамках рішень. Для зберігання телеметричних даних та їх підготовки до аналізу попередньо налаштовані рішення використовують сховище Blob.

Візуалізація даних здійснюється за допомогою веб-застосунків Microsoft Power BI. Power BI пропонує гнучкість у створенні інтерактивних панелей моніторингу на основі даних з Azure IoT Suite.

Окрім Azure IoT Suite, існує інша платформа під назвою AWS IoT Core, яка є керованою хмарною платформою, наданою Amazon Web Services (AWS) для роботи з пристроями IoT. AWS IoT Core підтримує мільярди пристроїв і трильйони повідомлень, полегшуючи підключення пристроїв до хмари або встановлення з'єднань між ними. AWS IoT Core пропонує широкий вибір операційних систем, мов програмування, платформ інтернет-застосунків, баз даних та інших необхідних сервісів.

Основні особливості AWS IoT Core включають можливість фільтрації та трансформації даних пристроїв, збереження останнього стану пристроїв, комплексну безпеку інфраструктури та інтеграцію з різними службами AWS і Amazon, такими як AWS Lambda, Amazon Kinesis, Amazon S3, Amazon SageMaker, Amazon DynamoDB, Amazon CloudWatch, AWS CloudTrail, Amazon QuickSight і Alexa Voice Service. AWS IoT Core спрощує розробку застосунків IoT для збору, обробки, аналізу та автоматизації даних без необхідності керувати основною інфраструктурою.

Для офісного простору розподілена архітектура IoT має ряд значних переваг. Масштабованість забезпечує легке додавання нових пристроїв та розширення системи на більшу площу без значних змін в існуючій інфраструктурі. Гнучкість дозволяє адаптувати систему до специфічних

потреб різних функціональних зон офісу. Надійність підвищується за рахунок того, що відмова окремого пристрою не призводить до зупинки роботи всієї системи. Швидкість реакції локальних контролерів на події є значно вищою, оскільки вони можуть приймати рішення та виконувати дії без затримки, пов'язаної зі зв'язком з центральним сервером.

В IoT-системах контролю доступу та безпеки в офісному просторі використовується широкий спектр технологій. Датчики включають RFID-зчитувачі, зчитувачі біометричних даних (відбитки пальців, розпізнавання обличчя), датчики руху, магнітні датчики дверей/вікон, датчики розбиття скла, камери відеоспостереження, теплові камери та датчики диму/газу. Виконавчими пристроями є електронні замки, турнікети, шлагбауми, сирени та світлові оповіщувачі. Мережеві пристрої представлені локальними контролерами (мікроконтролери, одноплатні комп'ютери), шлюзами IoT, мережевими комутаторами та маршрутизаторами. Для забезпечення зв'язку використовуються різноманітні протоколи: IoT-протоколи, такі як MQTT, CoAP, Zigbee, Bluetooth LE, LoRaWAN, NB-IoT (для зв'язку між кінцевими пристроями та локальними контролерами/шлюзами), та IP-протоколи, такі як Ethernet та Wi-Fi (для зв'язку між локальними контролерами/шлюзами та центральним сервером, а також для доступу користувачів). Програмне забезпечення включає централізовані рішення для управління правами доступу, моніторингу подій, аналізу даних, візуалізації та інтеграції з іншими системами, а також мобільні додатки та веб-інтерфейси для користувачів та адміністраторів. Опціонально можуть використовуватись хмарні платформи для централізованого управління, зберігання даних та аналітики.

Впровадження розподілених IoT-систем контролю доступу та безпеки в офісному просторі надає значні переваги. Підвищений рівень безпеки досягається завдяки комплексному моніторингу та швидкому реагуванню на потенційні загрози. Гнучке управління доступом дозволяє надавати диференційовані права доступу для різних категорій співробітників та відвідувачів. Автоматизація рутинних процесів, таких як автоматичне

відкриття/закриття дверей або увімкнення/вимкнення сигналізації за розкладом або за певними подіями, підвищує ефективність. Система забезпечує покращену ефективність використання офісного простору та робочого часу. Збір та аналіз даних надає цінну інформацію про використання офісного простору та патерни переміщення співробітників, що може бути використано для подальшої оптимізації. Зручність використання забезпечується інтуїтивно зрозумілими інтерфейсами. Нарешті, в довгостроковій перспективі можливе досягнення економічної ефективності за рахунок зниження витрат на охорону та енергоспоживання.

### 1.4 Обґрунтування вибраного напрямку вирішення задачі

Обраний напрямок вирішення задачі в даній кваліфікаційній роботі полягає у проєктуванні та дослідженні розподіленої IoT-системи для контролю доступу та безпеки в офісному просторі. Це рішення ґрунтується на аналізі існуючих методів, технологій та аналогів систем, представлених у попередньому розділі, а також на розумінні специфічних потреб та викликів, пов’язаних із забезпеченням безпеки та контролю в сучасному офісному середовищі.

Обґрунтування вибраного напрямку – по, перше, на відміну від централізованих систем, розподілена архітектура IoT пропонує підвищену гнучкість, масштабованість та надійність. Додавання нових функціональних можливостей або розширення системи на більшу площу може бути здійснено шляхом інтеграції нових кінцевих пристроїв без значного впливу на основну інфраструктуру. У випадку виходу з ладу окремого пристрою, працездатність системи в цілому зберігається.

По-друге, технології IoT надають широкий спектр можливостей для інтеграції різноманітних сенсорів, виконавчих механізмів та комунікаційних протоколів. Це дозволяє створити комплексну систему, яка одночасно забезпечує контроль доступу, моніторинг безпеки, автоматизацію рутинних

процесів та збір даних для подальшого аналізу та оптимізації.

По-третє, сучасні офісні простори вимагають гнучких та ефективних рішень для забезпечення безпеки співробітників, майна та інформації. Розподілена IoT-система може адаптуватися до специфічних потреб різних офісних зон, надаючи персоналізований рівень контролю та безпеки.

По-четверте, розроблена система передбачає можливість інтеграції з іншими інтелектуальними системами будівлі (наприклад, системами управління освітленням, HVAC, пожежною сигналізацією), створюючи єдину екосистему управління офісним простором.

По-п’яте, використання доступних IoT-пристроїв та бездротових технологій може зробити розподілену систему більш економічно вигідною в порівнянні з традиційними пропрієтарними системами контролю доступу та безпеки.

Основна ідея роботи полягає у розробці концепції та демонстрації працездатності розподіленої IoT-системи для офісного простору (рис.1.5), яка забезпечує багаторівневий контроль доступу та комплексний моніторинг безпеки. Система базується на інтеграції різноманітних датчиків (RFID, руху, відчинення вікон/дверей, диму), виконавчих механізмів (електронні замки, сирена) та комунікаційних технологій (протоколи IoT, Ethernet). Ключовим аспектом є розподілена архітектура, де значна частина логіки та обробки даних відбувається на рівні локальних контролерів та пристроїв, а центральний сервер виконує функції координації, управління правами доступу та збору агрегованих даних [6].

Кінцеві пристрої (Датчики, виконавчі механізми) збирають інформацію про фізичне середовище (стан дверей, наявність RFID-міток, рух, задимлення тощо). Приклади: RFID-зчитувачі, датчики руху, магнітні датчики дверей/вікон, датчики диму. Виконавчі механізми виконують команди, отримані від мережевих пристроїв або центрального сервера. Приклади: Електронні замки, сирени.

Мережеві пристрої (Локальні контролери, шлюзи) обробляють дані,

отримані від підключених до них кінцевих пристроїв, приймають локальні рішення (наприклад, відкрити двері при зчитуванні дійсної RFID-мітки) та передають інформацію на центральний сервер. Приклади: Мікроконтролери (ESP32, Raspberry Pi Pico) з відповідними інтерфейсами.

Шлюзи забезпечують зв'язок між локальною мережею кінцевих пристроїв (що використовують протоколи IoT) та IP-мережею офісу, через яку здійснюється зв'язок з центральним сервером.

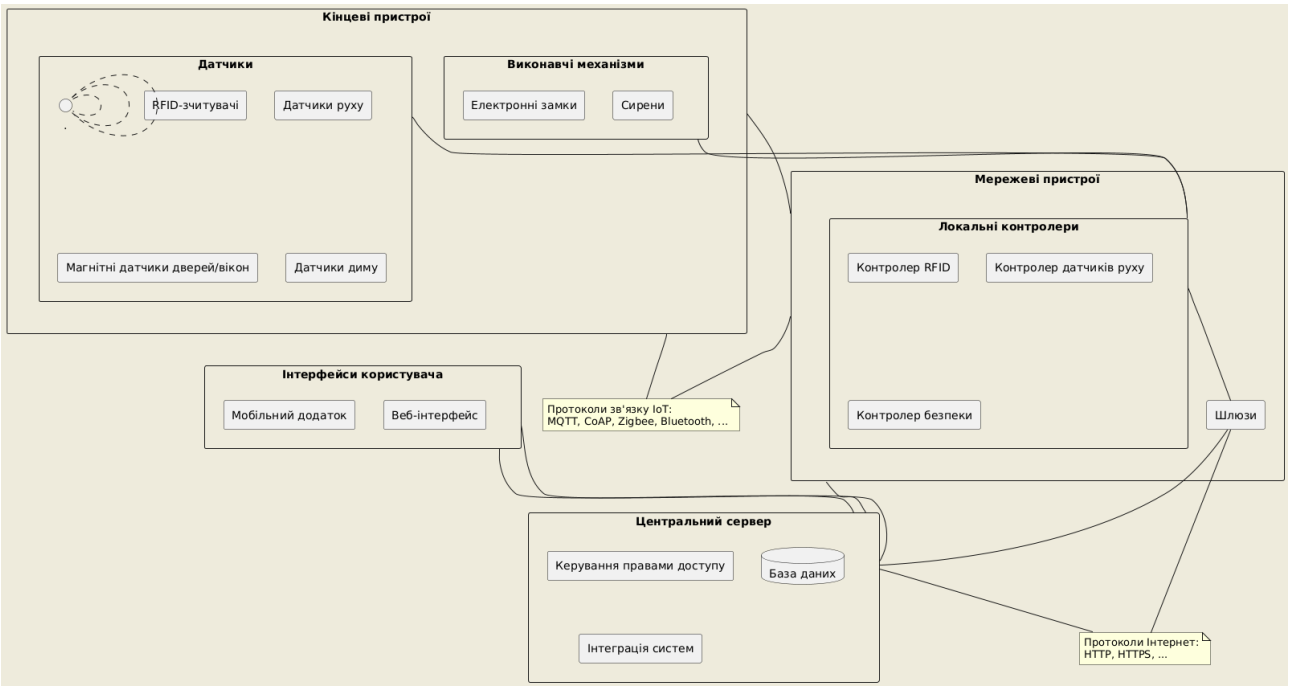



Рисунок 1.8 – Функціональна структура розподіленої IoT-системи для офісного простору

Центральний сервер (Керування, база даних, інтеграція) зберігає інформацію про користувачів та їхні права доступу до різних зон офісу.

База даних зберігає журнали подій (час входу/виходу, спрацювання датчиків), конфігураційні дані та іншу інформацію про роботу системи.

Інтеграція забезпечує взаємодію з інтерфейсами користувача та, за потреби, з іншими офісними системами.

Протоколи зв'язку IoT використовуються для обміну даними між кінцевими пристроями та локальними контролерами/шлюзами. Приклади: MQTT, CoAP, Zigbee, Bluetooth.

Протоколи Інтернет використовуються для зв'язку між шлюзами та центральним сервером, а також між центральним сервером та інтерфейсами користувача. Приклади: HTTP, HTTPS.

Інтерфейси користувача (веб-інтерфейс) надають користувачам (співробітникам, адміністраторам, службі безпеки) можливість взаємодії з системою: переглядати стан пристроїв, отримувати сповіщення, керувати правами доступу (для адміністраторів) тощо.

### 1.5 Мета і задачі роботи

Метою даної роботи є розробка та дослідження концепції розподіленої IoT-системи для підвищення рівня безпеки та оптимізації контролю доступу в офісному просторі шляхом моделювання та аналізу окремих функціональних підсистем в середовищі Cisco Packet Tracer.

Для досягнення поставленої мети необхідно вирішити наступні задачі:

– розробити функціональні моделі окремих підсистем контролю доступу та безпеки в офісному просторі: реалізувати модель дверного замка, що функціонує на основі технології радіочастотної ідентифікації (RFID), забезпечуючи доступ лише авторизованим користувачам. Створити модель розумного вікна, яке автоматично відкривається за сприятливих погодних умов (ранок, відсутність дощу) з використанням датчиків освітленості та дощу, та закривається вночі або під час дощу. Розробити модель системи автономного живлення вентилятора та освітлення від сонячної енергії з контролем рівня заряду акумулятора. Створити модель системи захисту від несанкціонованого проникнення через вікно з використанням датчика спрацьовування та звукової сигналізації. Реалізувати модель автоматичного ввімкнення вентилятора та кавоварки в їдальні при виявленні присутності людини за допомогою датчика руху. Створити модель відтворення музики через Bluetooth на портативній колонці як елемент створення комфортного середовища. Розробити модель розумного вуличного ліхтаря, що

автоматично вмикається в темний час доби за допомогою фотосенсора для енергозбереження. Створити модель автоматичного керування воротами гаража на основі виявлення диму від заведеного автомобіля. Реалізувати модель системи пожежної та димової сигналізації з використанням відповідних датчиків для своєчасного оповіщення про загрозу;

– здійснити моделювання розроблених підсистем в середовищі Cisco Packet Tracer: Налаштувати необхідні віртуальні пристрої, сенсори, виконавчі механізми та логіку їхньої взаємодії в межах кожної підсистеми;

– проаналізувати функціональність та ефективність змодельованих підсистем: Оцінити працездатність кожної підсистеми у змодельованих сценаріях та визначити їхній потенційний внесок у забезпечення контролю доступу та безпеки в офісному просторі;

– визначити можливі шляхи інтеграції розроблених підсистем у єдину розподілену IoT-систему: Розглянути потенційні протоколи обміну даними та підходи до централізованого управління та моніторингу.

# 2 СПЕЦІАЛЬНИЙ РОЗДІЛ

## 2.1 Технічні вимоги до розподіленої IoT-система для контролю доступу та безпеки в офісному просторі

### 2.1.1 Найменування і призначення розподіленої IoT-система

Розподілена IoT-система для контролю доступу та безпеки в офісному просторі призначена для демонстрації окремих функціональних підсистем контролю доступу та безпеки в офісному просторі, змодельованих в середовищі Cisco Packet Tracer.

### 2.1.2 Вимоги до структури і функціонування розподіленої IoT-системи

Розподілена IoT-система для контролю доступу та безпеки в офісному просторі складається з наступних функціональних підсистем, змодельованих в середовищі Cisco Packet Tracer [6]:

– підсистема контролю доступу на основі RFID: RFID-зчитувач для ідентифікації користувачів; електронний замок для керування доступом до дверей; логічний контролер для обробки даних з RFID-зчитувача та керування електронним замком; емулятор RFID-карток для імітації ідентифікації користувачів;

– підсистема автоматичного керування вікном: фотодатчик для визначення рівня освітленості (день/ніч); датчик дощу для фіксації опадів; електропривід для відкриття/закриття вікна та логічний контролер для обробки сигналів з датчиків та керування електроприводом;

– підсистема автономного живлення: сонячна панель для генерації електроенергії; акумуляторна батарея для накопичення енергії; вентилятор як демонстраційне навантаження; світлодіодна лампа як демонстраційне навантаження; контролер заряду для керування процесом зарядки та розряду акумулятора та логічний контролер для керування живленням навантажень

залежно від рівня заряду батареї та освітленості;

– підсистема захисту від крадіжки: датчик спрацьовування (контактний або вібраційний) для фіксації несанкціонованого проникнення через вікно; сирена для звукового оповіщення про тривогу; логічний контролер для обробки сигналу з датчика спрацьовування та активації сирени;

– підсистема автоматичного керування вентилятором та кавоваркою: датчик руху для виявлення присутності людини в їдальні; віртуальний вентилятор як демонстраційне навантаження; віртуальна кавоварка як демонстраційне навантаження та логічний контролер для обробки сигналу з датчика руху та керування живленням вентилятора та кавоварки;

– підсистема відтворення музики;

– підсистема розумного вуличного освітлення;

– підсистема автоматизованого керування гаражними воротами: датчик диму для виявлення вихлопних газів автомобіля; електропривід для відкриття/закриття гаражних воріт та логічний контролер для обробки сигналу з датчика диму та керування електроприводом;

– підсистема пожежної та димової сигналізації: пожежний датчик (тепловий) для виявлення підвищення температури; датчик диму для виявлення задимлення; сирена для звукового оповіщення про тривогу та логічний контролер для обробки сигналів з датчиків та активації сирени.

### 2.1.3 Вимоги до показників призначення

Розподілена IoT-система для контролю доступу та безпеки в офісному просторі та її функціональні підсистеми повинні забезпечувати наступні показники призначення. Підсистема контролю доступу на основі RFID повинна забезпечувати зчитування унікального ідентифікатора з віртуальної RFID-картки, його порівняння з попередньо визначеним списком дійсних

ідентифікаторів, генерацію сигналу для відкриття електронного замка при збігу ідентифікатора, блокування електронного замка у випадку недійсної RFID-картки, а також візуальну індикацію статусу доступу (дозволено/заборонено).

Підсистема автоматичного керування вікном повинна забезпечувати визначення стану "день" та "ніч" на основі рівня освітленості, визначення стану "є дощ" та "немає дощу" за сигналом датчика дощу, автоматичне відкриття вікна при стані "день" та "немає дощу", автоматичне закриття вікна при стані "ніч" або "є дощ", а також візуальну індикацію стану вікна (відкрито/закрито).

Підсистема автономного живлення повинна забезпечувати генерацію віртуальної електроенергії сонячною панеллю при наявності достатнього освітлення, накопичення віртуальної енергії в акумуляторній батареї, автоматичне живлення віртуального вентилятора та світлодіодної лампи при достатньому рівні заряду батареї, припинення живлення навантажень при низькому рівні заряду батареї, а також візуальну індикацію рівня заряду батареї та стану живлення навантажень (увімкнено/вимкнено).

Підсистема захисту від крадіжки повинна забезпечувати генерацію віртуального сигналу тривоги від датчика спрацьовування при імітації несанкціонованого проникнення, активацію віртуальної сирени при отриманні сигналу тривоги та візуальну індикацію стану системи захисту (активовано/деактивовано) та спрацювання тривоги [4].

Підсистема автоматичного керування вентилятором та кавоваркою повинна забезпечувати виявлення віртуальної присутності людини за допомогою датчика руху, автоматичне ввімкнення віртуального вентилятора та кавоварки при виявленні руху, автоматичне вимкнення через заданий проміжок часу після припинення руху, а також візуальну індикацію стану вентилятора та кавоварки (увімкнено/вимкнено).

Підсистема відтворення музики повинна забезпечувати імітацію встановлення Bluetooth-з'єднання між віртуальним музичним плеєром та

портативною колонкою, відтворення віртуального звукового сигналу через віртуальну колонку при активації плеєра, а також візуальну індикацію статусу з'єднання Bluetooth та відтворення музики.

Підсистема розумного вуличного освітлення повинна забезпечувати визначення стану "темно" на основі рівня освітленості, автоматичне ввімкнення віртуального вуличного ліхтаря при настанні стану "темно", автоматичне вимкнення при достатньому рівні освітленості, а також візуальну індикацію стану вуличного ліхтаря (увімкнено/вимкнено).

Підсистема автоматизованого керування гаражними воротами повинна забезпечувати виявлення віртуального диму, автоматичне відкриття віртуальних гаражних воріт при виявленні диму, автоматичне закриття після припинення виявлення диму через заданий проміжок часу, а також візуальну індикацію стану гаражних воріт (відкрито/закрито).

Підсистема пожежної та димової сигналізації повинна забезпечувати генерацію віртуального сигналу тривоги від пожежного датчика при імітації підвищення температури, генерацію віртуального сигналу тривоги від датчика диму при імітації задимлення, активацію віртуальної сирени при отриманні сигналу тривоги від будь-якого з датчиків, а також візуальну індикацію стану системи сигналізації (активовано/деактивовано) та спрацювання тривоги (пожежа/дим).

## 2.2 Розробка апаратної частини

### 2.2.1 Розробка структури комплексу технічних засобів IoT системи

Структура комплексу технічних засобів IoT-системи на рівні пристроїв включає різноманітні фізичні та віртуальні компоненти, що безпосередньо взаємодіють з офісним середовищем та виконують функції контролю доступу, забезпечення безпеки та автоматизації окремих процесів (рис.2.1).

Рівень пристроїв поділяється на підрівні "Вулиця" та "Приміщення", що відображає їхнє фізичне розташування та специфічні завдання.

Підрівень "Вулиця" охоплює технічні засоби, розташовані поза основним офісним простором або на його зовнішньому периметрі. До нього належать: RFID Card (Віртуальна RFID-картка) – електронний ідентифікатор, що імітує фізичну RFID-картку співробітника або відвідувача. Містить унікальний ідентифікаційний номер, який використовується для контролю доступу.

RFID Reader (Віртуальний RFID-зчитувач) – пристрій, призначений для безконтактного зчитування інформації з RFID-картки. При піднесенні дійсної картки зчитувач передає її ідентифікатор на наступний рівень для перевірки прав доступу.

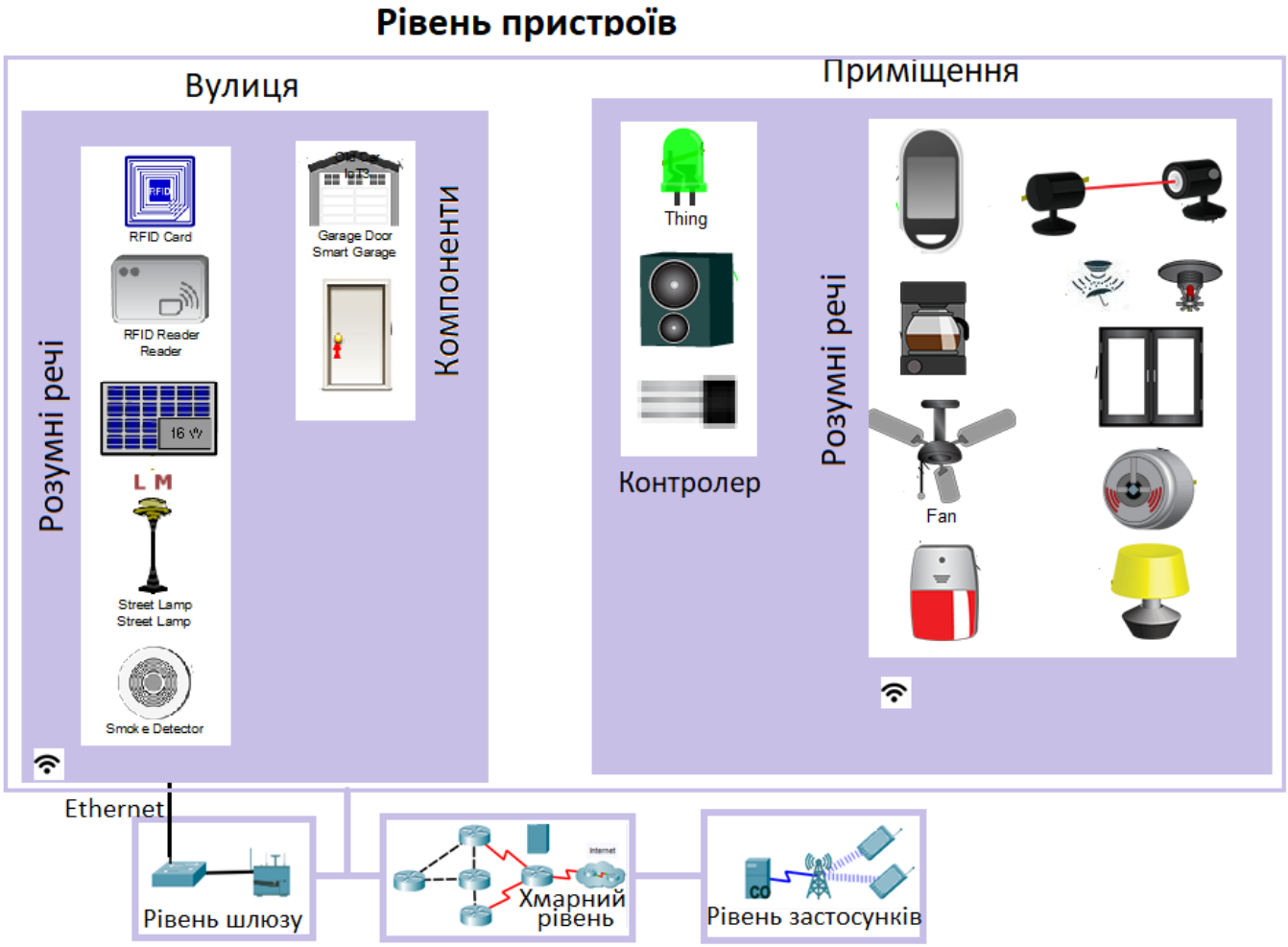



Рисунок 2.1 – Структурна схема компонентів IoT-системи

Garage Door Smart Garage (Віртуальний розумний гараж та ворота) –

комплекс, що включає віртуальні гаражні ворота та систему їх автоматизованого керування. Ворота можуть відкриватися та закриватися на основі сигналу від датчика диму (імітація заїзду/виїзду автомобіля) або за командою з центральної системи.

Street Lamp (Віртуальний вуличний ліхтар) – пристрій зовнішнього освітлення, обладнаний віртуальним фотосенсором. Автоматично вмикається при зниженні рівня освітленості (настанні темряви) та вимикається вдень, забезпечуючи енергоефективне освітлення прилеглої території.

Smoke Detector (Віртуальний датчик диму) – сенсор, призначений для виявлення задимлення. При перевищенні встановленого порогу концентрації диму генерує сигнал тривоги, який передається на центральний рівень для відповідного реагування (наприклад, активації сирени).

Зв'язок між пристроями підрівня "Вулиця" та рівнем шлюзу в проєкті Cisco Packet Tracer імітується переважно бездротовим з'єднанням (символ Wi-Fi), що відображає типові сценарії розгортання зовнішніх IoT-пристроїв.

Підрівень "Приміщення" включає технічні засоби, розташовані безпосередньо у внутрішньому офісному просторі. До нього належать: Thing (Віртуальний універсальний пристрій) – узагальнений віртуальний пристрій, який може представляти різноманітні сенсори або виконавчі механізми, функціональність яких може бути налаштована в залежності від конкретного сценарію (наприклад, датчик температури, вологості тощо).

Speaker (Віртуальний гучномовець) – пристрій для відтворення звукових сигналів, наприклад, музики (через імітацію Bluetooth-з'єднання з віртуальним музичним плеєром) або сповіщень.

Coffee Maker (Віртуальна кавоварка) – електроприлад, обладнаний віртуальним механізмом автоматичного ввімкнення при виявленні руху в зоні їдальні.

Fan (Віртуальний вентилятор) – електромеханічний пристрій, обладнаний віртуальним механізмом автоматичного ввімкнення при виявленні руху в зоні їдальні або за командою з центральної системи.

Break Sensor (Віртуальний датчик спрацьовування) – сенсор, призначений для виявлення фізичного впливу, що може свідчити про спробу несанкціонованого проникнення (наприклад, розбиття вікна). При спрацюванні генерує сигнал тривоги.

Window (Віртуальне вікно з електроприводом) – вікно, обладнане віртуальним електроприводом для автоматичного відкриття та закриття на основі даних з віртуальних датчиків світла та дощу.

Fire Detector (Віртуальний пожежний датчик) – сенсор, призначений для виявлення різкого підвищення температури, що є ознакою пожежі. При виявленні небезпеки генерує сигнал тривоги.

Lamp (Віртуальна лампа внутрішнього освітлення) – пристрій внутрішнього освітлення, яким можна керувати автоматично (наприклад, на основі даних з датчика руху) або віддалено.

Контролер (Віртуальний локальний контролер) – представлений як окремий пристрій, імовірно, виконує функцію збору даних з внутрішніх "розумних речей" та керування ними. Він може здійснювати первинну обробку даних перед їх передачею на рівень шлюзу.

Зв'язок між пристроями підрівня "Приміщення" та рівнем шлюзу в проєкті Cisco Packet Tracer також імітується переважно бездротовим з'єднанням (символ Wi-Fi), що відображає зручність та гнучкість бездротових технологій для внутрішнього розгортання IoT-систем.

На рівні шлюзу в представленій IoT-системі для офісного простору (проєкт в Cisco Packet Tracer) ключову роль відіграє віртуальний пристрій, що забезпечує інтеграцію різнорідних мереж та протоколів, використовуваних на рівні пристроїв, з IP-орієнтованою інфраструктурою хмарного рівня. Центральним елементом цього рівня є віртуальний IoT-шлюз. Цей пристрій виконує ряд важливих функцій, серед яких агрегація даних, що полягає у зборі інформації від усіх кінцевих пристроїв, розташованих як на підрівні "Вулиця", так і на підрівні "Приміщення". Шлюз встановлює з'єднання з цими пристроями через відповідні бездротові

інтерфейси (імітація Wi-Fi, Bluetooth тощо). Важливою функцією є перетворення протоколів (імітація), оскільки, хоча детальна демонстрація цього процесу може бути обмежена можливостями Cisco Packet Tracer, віртуальний шлюз забезпечує передачу даних, отриманих за різними протоколами рівня пристроїв, у форматі, придатному для подальшої передачі по IP-мережі. Крім того, шлюз може виконувати первинну обробку даних, таку як базова фільтрація, агрегація або тимчасове зберігання даних перед їх відправкою на хмарний рівень, що може включати відкидання надлишкової інформації або усереднення значень датчиків. Керування пристроями також є однією з ключових функцій шлюзу, оскільки він забезпечує канал зв'язку для передачі команд керування від хмарного рівня до кінцевих пристроїв, отримуючи команди та перетворюючи їх у формат, зрозумілий для відповідного виконавчого пристрою. Нарешті, на рівні шлюзу можуть бути реалізовані базові механізми безпеки, такі як фільтрація трафіку або просте шифрування даних, що передаються [7].

Віртуальний шлюз здійснює зв'язок з іншими рівнями системи. З рівнем пристроїв він взаємодіє з різноманітними "розумними речами" через бездротові з'єднання (імітація Wi-Fi для більшості внутрішніх та деяких зовнішніх пристроїв), приймаючи від них дані сенсорів, ідентифікатори користувачів (з RFID-зчитувача) та інші службові повідомлення. З хмарним рівнем віртуальний шлюз встановлює IP-з'єднання (імітація Ethernet або Wi-Fi підключення до віртуальної мережі) з віртуальним сервером або іншими компонентами хмарного рівня, через яке здійснюється передача зібраних даних для подальшої обробки та отримання команд керування.

### 2.2.2 Розробка специфікації апаратних засобів IoT системи

Для реалізації проєкту розподіленої IoT-системи контролю доступу та безпеки в офісному просторі на реальному обладнанні можна розглянути наступні конкретні пристрої та датчики (табл.2.1), які відповідають

функціональності віртуальних компонентів, використаних у моделюванні Cisco Packet Tracer.

В якості центрального контролера та шлюзу системи пропонується використовувати Raspberry Pi 4 Model B від Raspberry Pi Foundation. Цей одноплатний комп'ютер володіє достатньою обчислювальною потужністю завдяки чотириядерному процесору Broadcom BCM2711, до 8 ГБ оперативної пам'яті, вбудованими модулями Wi-Fi та Bluetooth, Gigabit Ethernet портом та різноманітними інтерфейсами, включаючи GPIO для підключення датчиків та USB для периферії. Raspberry Pi підтримує операційну систему Raspberry Pi OS (на базі Linux), що забезпечує гнучкість у розробці програмного забезпечення для збору, обробки даних та керування пристроями, а також для зв'язку з хмарною платформою [7].

Для ідентифікації користувачів у системі контролю доступу буде використано RFID-зчитувач HID Global OMNIKEY 5321 CL. Цей пристрій підтримує поширені стандарти безконтактних карток, включаючи ISO 14443 A/B, MIFARE та DESFire EV1, має USB інтерфейс для підключення до контролера та забезпечує надійне зчитування ідентифікаторів з відстані до кількох сантиметрів.

Для реалізації функціональності "розумного вікна" буде використано фотодатчик Vishay Semiconductors VEML7700 для вимірювання рівня освітленості та датчик дощу Seeed Studio Grove - Rain Sensor для фіксації опадів. VEML7700 відрізняється високою точністю та широким діапазоном вимірювання освітленості, має інтерфейс $I^2C$ для підключення до GPIO Raspberry Pi. Датчик дощу від Seeed Studio є простим у використанні та надає цифровий сигнал про наявність або відсутність опадів, також підключаючись до GPIO. Керування електроприводом вікна здійснюватиметься через релейний модуль, підключений до GPIO, на основі даних з цих датчиків.

Для виявлення присутності в офісних приміщеннях буде використано пасивний інфрачервоний (PIR) датчик руху Panasonic EKMB1303111. Цей датчик характеризується низьким енергоспоживанням та надійним

виявленням руху в радіусі до 5 метрів з кутом огляду до 110°, підключається до GPIO Raspberry Pi та використовується для автоматизації освітлення (через розумні розетки), систем клімат-контролю (через ІЧ-контролери або інтеграцію з BMS) та для активації тривожних сценаріїв у системі безпеки.

Для забезпечення захисту від несанкціонованого проникнення через вікна пропонується використовувати датчик вібрації SW-420. Цей простий та широкодоступний електромагнітний датчик генерує цифровий сигнал при виявленні вібрації або удару, підключається до GPIO Raspberry Pi та може бути використаний для активації сигналізації.

Для раннього виявлення задимлення буде використано оптичний датчик диму Honeywell MORNING PRIDE SD2-24DC (або аналогічний з цифровим виходом). Цей датчик відповідає стандартам пожежної безпеки та забезпечує надійне виявлення диму, надаючи цифровий сигнал тривоги для контролера.

Для виявлення пожежі за ознакою підвищення температури буде використано тепловий датчик System Sensor 5201. Цей датчик спрацьовує при досягненні фіксованої температури (наприклад, 57°C) та надає цифровий сигнал тривоги для контролера [8].

Таблиця 2.1 – Технічні характеристики пристроїв датчиків

| Пристрій/Датчик | Виробник | Модель/Серія | Основні характеристики | Інтерфейс підключення до контролера |
|---|---|---|---|---|
| Контролер | Raspberry Pi | 4 Model B | Чотириядерний процесор, до 8 ГБ RAM, Wi-Fi, Bluetooth, Gigabit Ethernet, GPIO, USB | - |
| RFID-зчитувач | HID Global | OMNIKEY 5321 CL | Підтримка ISO 14443 A/B, дальність зчитування до 5 см | USB |
| Фотодатчик | Vishay | VEML7700 | Діапазон | I²C |

| | | | | |
|---|---|---|---|---|
| | Semiconductors | | вимірювання 0 lx до 120 klx, висока точність | |
| Датчик дощу | Seeed Studio | Grove - Rain Sensor | Цифровий вихід (є/немає дощу) | Цифровий (GPIO) |
| Датчик руху (PIR) | Panasonic | EKMB1303111 | Дальність виявлення до 5 м, кут огляду до 110°, низьке енергоспоживання | Цифровий (GPIO) |
| Датчик спрацьовування (вібрації) | Китайські виробники | SW-420 | Цифровий вихід (замкнуто/розімкнуто при вібрації), регульована чутливість | Цифровий (GPIO) |
| Датчик диму | Honeywell | MORNING PRIDE SD2-24DC | Оптичний, цифровий вихід (тривога/норма), відповідає стандартам UL 217 | Цифровий |

Продовження до таблиці 2.1

| Пристрій/Датчик | Виробник | Модель/Серія | Основні характеристики | Інтерфейс підключення до контролера |
|---|---|---|---|---|
| Пожежний датчик (тепловий) | System Sensor | 5201 | Термістор, фіксована температура спрацювання (наприклад, 57°C), цифровий вихід (тривога/норма) | Цифровий |
| Розумна розетка | TP-Link | Tapo P100 | Керування через Wi-Fi, мобільний додаток | IP (через Wi-Fi) |
| Релейний модуль | Різні виробники | Залежно від потреб | Керування електроживлен | GPIO контролера |

| | | | | |
|---|---|---|---|---|
| | | | ням виконачих пристроїв (замки, приводи) | |
| Хмарна платформа | Amazon | AWS IoT Core | Збір, зберігання, обробка даних, віддалене керування, візуалізація, безпека | IP (через Інтернет) |
| Пристрій/Датчик | Виробник | Модель/Серія | Основні характеристики | Інтерфейс підключення до контролера |

Керування виконавчими пристроями, такими як електронні замки, розумні розетки (наприклад, TP-Link Tapo P100), сирени та приводи гаражних воріт, здійснюватиметься через релейні модулі, підключені до GPIO Raspberry Pi, або через API розумних пристроїв (у випадку Wi-Fi пристроїв). Керування вуличним освітленням також може бути реалізовано через релейний модуль на основі даних з фотодатчика.

Для хмарної інтеграції планується використання платформи AWS IoT Core, яка забезпечить безпечне та масштабоване підключення пристроїв, збір, зберігання та аналіз даних, а також віддалене керування. Клієнтські застосунки для моніторингу та керування системою будуть розроблені у вигляді веб-інтерфейсу (наприклад, за допомогою Flask на Raspberry Pi або на AWS) та мобільного додатку (наприклад, за допомогою Flutter або React Native) для зручного доступу користувачів.

### 2.2.3 Розробка переліку вхідних та вихідних сигналів

Для забезпечення належного функціонування та керування IoT-системою офісного простору необхідно чітко визначити перелік вхідних сигналів, які надходять до системи від віртуальних датчиків та пристроїв, а

також вихідних сигналів, які система генерує для керування віртуальними виконавчими механізмами.

Вхідні сигнали є первинною інформацією, яку система збирає з навколишнього віртуального середовища та використовує для аналізу, прийняття рішень та ініціювання відповідних дій. До переліку вхідних сигналів розробленої IoT-системи входять дані від різноманітних віртуальних сенсорів, що імітують реальні фізичні датчики. Ці сигнали відображають поточний стан контрольованих параметрів офісного простору, таких як рівень освітленості, наявність опадів, рух, вібрація, рівень задимлення та температура. Крім того, до вхідних сигналів належить інформація від пристрою ідентифікації користувачів (RFID-зчитувача), що є ключовим елементом підсистеми контролю доступу. Детальний опис кожного вхідного сигналу, його ідентифікатор у середовищі Cisco Packet Tracer, джерело сигналу, функціональне призначення, тип сигналу та його можливі значення представлено у таблиці 2.2 [9].

Таблиця 2.2 – Вхідні сигнали IoT-системи

| № | Назва вхідного сигналу | Ідентифікатор (в CP) | Джерело сигналу (в CP) | Функція | Тип сигналу | Опис сигналу |
|---|---|---|---|---|---|---|
| 1 | RFID-зчитувач (ID картки) | RFID Reader | RFID Card | Авторизація доступу | Цифровий | Рядок символів (наприклад, "A1B2C3D4") |
| 2 | Фотодатчик | Photo Sensor | Віртуальне оточення | Вимірювання рівня освітленості | Аналоговий | Ціле число в діапазоні 0–255 |
| 3 | Датчик дощу | Rain Sensor | Віртуальне оточення | Визначення наявності дощу | Цифровий | 0 (немає дощу), 1 (є дощ) |
| 4 | Датчик руху | Motion Sensor | Віртуальне оточення | Виявлення руху | Цифровий | 0 (немає руху), 1 (рух виявлено) |
| 5 | Датчик вібрації | Break Sensor | Віртуальне оточення | Фіксація вібрації | Цифровий | 0 (немає вібрації), 1 (виявлено) |
| 6 | Датчик диму | Smoke Detector | Віртуальне оточення | Вимірювання рівня задимлення | Аналоговий | Ціле число в діапазоні 0–255 |

Вихідні сигнали є командами керування, які генеруються центральним контролером IoT-системи на основі обробки вхідних сигналів та відповідно до заданої логіки роботи системи. Ці сигнали спрямовані на віртуальні виконавчі пристрої, що моделюють реальні керовані компоненти офісного простору. До переліку вихідних сигналів належать команди для керування електронним замком дверей, електроприводом вікна, вентилятором, внутрішнім освітленням, кавоваркою, сиреною, гаражними воротами, вуличним ліхтарем, музичним плеєром та портативною колонкою. Кожен вихідний сигнал має свій ідентифікатор у середовищі Cisco Packet Tracer, визначає приймач сигналу (конкретний віртуальний пристрій), описує функцію керування, вказує на тип сигналу та його значення, що відповідають різним станам керованого пристрою. Детальна інформація про вихідні сигнали представлена у таблиці 2.3.

Таблиця 2.3 – Вихідні сигнали IoT-системи

| № | Назва вихідного сигналу | Ідентифікатор (в CP) | Приймач сигналу (в CP) | Функція керування | Тип сигналу | Опис сигналу |
|---|---|---|---|---|---|---|
| 1 | Електронний замок | Door | Електронний замок | Відкрити / закрити | Цифровий | 0 – закрито, 1 – відкрито |
| 2 | Вікно (електропривод) | Window | Електропривод вікна | Відкрити / закрити | Цифровий | 0 – закрито, 1 – відкрито |
| 3 | Вентилятор | Fan | Віртуальний вентилятор | Увімкнути / вимкнути | Цифровий | 0 – вимкнено, 1 – увімкнено |
| 4 | Лампа (внутрішнє освітлення) | Lamp | Віртуальна лампа | Увімкнути / вимкнути | Цифровий | 0 – вимкнено, 1 – увімкнено |
| 5 | Кавоварка | Coffee Maker | Віртуальна кавоварка | Увімкнути / вимкнути | Цифровий | 0 – вимкнено, 1 – увімкнено |
| 6 | Сирена | Siren | Віртуальна | Активува | Цифров | 0 – |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | сирена | ти / деактивувати | ий | деактивовано, 1 – активовано |
| 7 | Гаражні ворота | Garage Door | Електропривод воріт | Відкрити / закрити | Цифровий | 0 – закрито, 1 – відкрито |
| 8 | Вуличний ліхтар | Street Lamp | Віртуальний ліхтар | Увімкнути / вимкнути | Цифровий | 0 – вимкнено, 1 – увімкнено |
| 9 | Музичний плеєр (відтворення) | Music Player | Віртуальний плеєр | Відтворити / зупинити | Цифровий | 0 – зупинено, 1 – відтворює |
| 10 | Портативна колонка (живлення) | Wireless Speaker | Віртуальна колонка | Увімкнути / вимкнути живлення | Цифровий | 0 – вимкнено, 1 – увімкнено |

### 2.2.4 Розробка принципової схеми

Принципова схема (рис.2.2) реалізує IoT-систему керування множинними модулями зберігання, центральним елементом якої є мікроконтролер ESP32. Кожен модуль обладнується електромагнітним замком, а керування їхнім станом здійснюється централізовано через ESP32. Важливою особливістю системи є можливість віддаленого керування доступом до модулів через вебінтерфейс або спеціалізований мобільний застосунок, що забезпечує зручність для користувачів. Взаємодія між клієнтськими пристроями та ESP32 здійснюється за допомогою бездротового з'єднання Wi-Fi, інтегрованого в мікроконтролер [8].

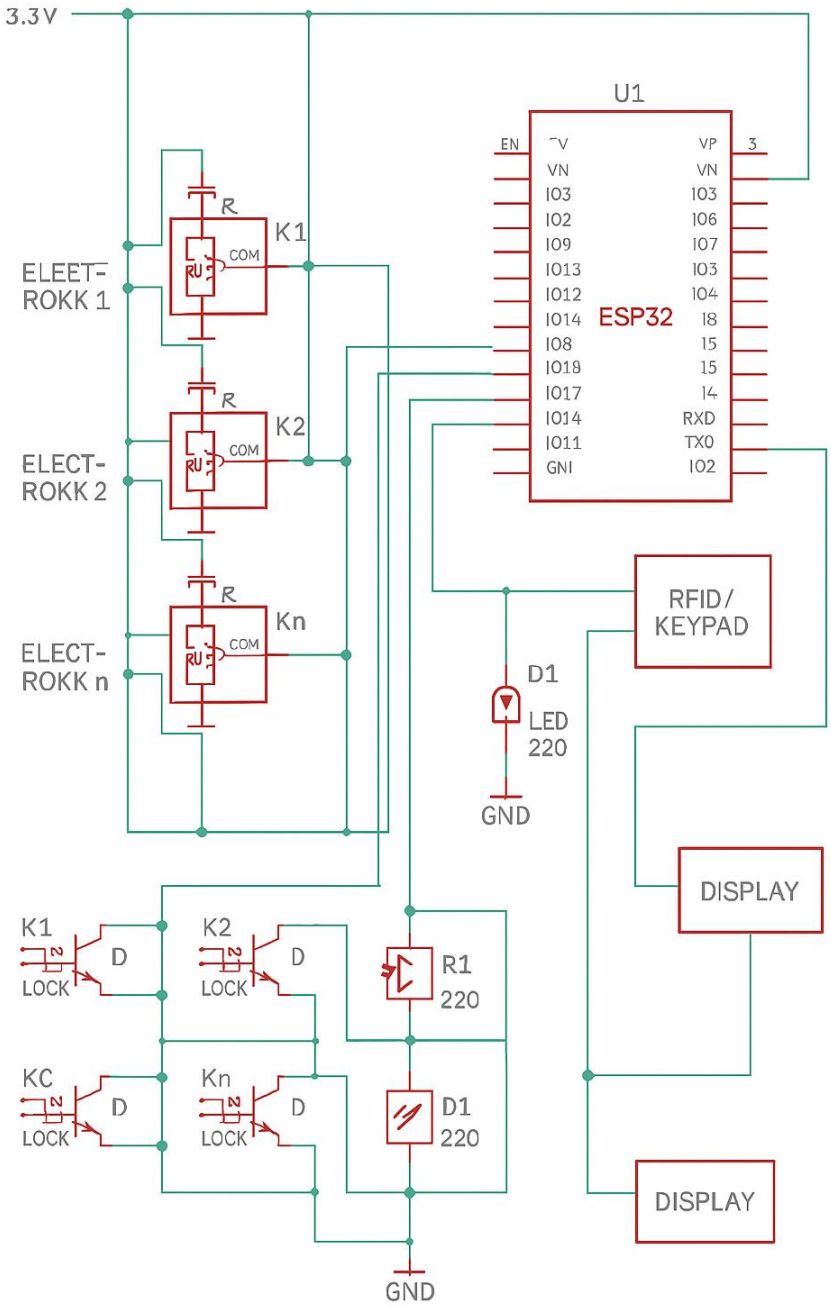
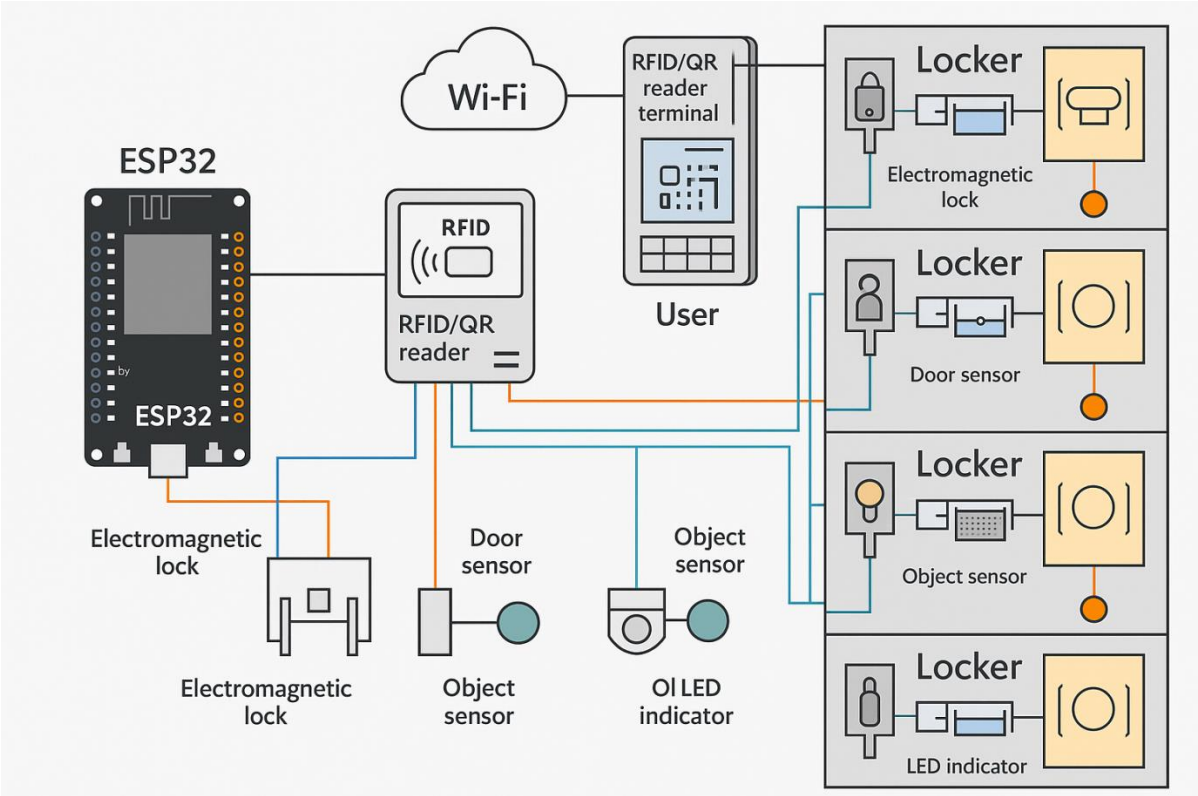



Рисунок 2.2 – Принципова схема IoT-система для контролю доступу та безпеки в офісному просторі

Основними компонентами принципової схеми є: центральний контролер ESP32, який відповідає за керування всіма вихідними сигналами, що надходять до електромагнітних замків, встановлених на кожному модулі. ESP32 має вбудований модуль Wi-Fi, що забезпечує його підключення до локальної мережі. Електромагнітні замки (Lock1, Lock2, і так далі для

кожного модуля) є виконавчими механізмами, що блокують доступ до вмісту модулів. Живлення на замки подається через керуючі реле. Реле (Relay1, Relay2, і так далі) виступають в ролі електронних ключів, які керують ланцюгом живлення відповідного замка. Активація реле відбувається за сигналом з відповідного GPIO-виходу ESP32, що призводить до замикання кола живлення замка та його відкриття. Джерело живлення є критично важливим елементом схеми, що забезпечує енергією всі компоненти. Зазвичай використовуються джерела на 5В або 12В, залежно від вимог електромагнітних замків. У схемі передбачаються окремі гілки живлення для ESP32 та для силових елементів, таких як замки. Для захисту ESP32 від зворотних викидів напруги, що виникають при вимкненні індуктивних елементів (котушок реле), паралельно кожній котушці реле підключаються діоди (наприклад, 1N4007). Резистори в поєднанні зі світлодіодами виконують функцію візуальних індикаторів стану реле, дозволяючи оперативно діагностувати роботу схеми, відображаючи, чи активоване відповідне реле в даний момент. Wi-Fi з'єднання, хоча фізично не відображене на електричній схемі, є ключовим елементом функціональності системи та реалізується на рівні прошивки ESP32, забезпечуючи бездротове керування замками через мобільний застосунок або веббраузер.

Принцип роботи підсистеми полягає в наступному: користувач, використовуючи свій смартфон або комп'ютер, надсилає через Wi-Fi команду на ESP32, вказуючи, наприклад, необхідність відкрити модуль №3. ESP32, отримавши команду, активує відповідний GPIO-вихід, який підключений до керуючої котушки реле, відповідального за модуль №3. Активація GPIO призводить до спрацювання реле, яке замикає електричне коло живлення електромагнітного замка цього модуля. В результаті замок відкривається на заданий проміжок часу (наприклад, 5 секунд), після чого ESP32 деактивує GPIO-вихід, реле розмикає коло живлення, і замок автоматично зачиняється. Світлодіодний індикатор, підключений до ланцюга керування реле або живлення замка, візуально сигналізує про процес активації замка.




## 2.3 Налаштування IoT-система для контролю доступу та безпеки в офісному просторі

### 2.3.1 Налаштування RFID

Для налаштування RFID-зчитувача використано JavaScript API для системи контролю доступу, призначеної для використання в офісному просторі, змодельованому в середовищі Cisco Packet Tracer. Ініціалізує RFID-зчитувач як зареєстрований пристрій в екосистемі IoE, визначаючи його тип та можливі стани: ідентифікатор зчитаної RFID-мітки (Card ID) та поточний статус зчитувача (Status), який може бути "Valid" (0), "Invalid" (1) або "Waiting" (2). Зчитувач налаштовується на обробку вхідних даних, що надходять від сервера IoE, за допомогою функції processData (рис.2.3).

Таблиця 2.3 – Вихідні сигнали IoT-системи

| № | Назва вихідного сигналу | Ідентифікатор (в СР) | Приймач сигналу (в СР) | Функція керування | Тип сигналу | Опис сигналу |
|---|---|---|---|---|---|---|
| 1 | Електронний замок | Door | Електронний замок | Відкрити / закрити | Цифровий | 0 – закрито, 1 – відкрито |
| 2 | Вікно (електропривод) | Window | Електропривод вікна | Відкрити / закрити | Цифровий | 0 – закрито, 1 – відкрито |
| 3 | Вентилятор | Fan | Віртуальний вентилятор | Увімкнути / вимкнути | Цифровий | 0 – вимкнено, 1 – увімкнено |
| 4 | Лампа (внутрішнє освітлення) | Lamp | Віртуальна лампа | Увімкнути / вимкнути | Цифровий | 0 – вимкнено, 1 – увімкнено |
| 5 | Кавоварка | Coffee Maker | Віртуальна кавоварка | Увімкнути / вимкнути | Цифровий | 0 – вимкнено, 1 – увімкнено |
| 6 | Сирена | Siren | Віртуальна сирена | Активувати / деактивувати | Цифровий | 0 – деактивовано, 1 – активовано |
| 7 | Гаражні ворота | Garage Door | Електропривод воріт | Відкрити / закрити | Цифровий | 0 – закрито, 1 – відкрито |
| 8 | Вуличний ліхтар | Street Lamp | Віртуальний ліхтар | Увімкнути / вимкнути | Цифровий | 0 – вимкнено, 1 – увімкнено |
| 9 | Музичний плеєр (відтворення) | Music Player | Віртуальний плеєр | Відтворити / зупинити | Цифровий | 0 – зупинено, 1 – відтворює |
| 10 | Портативна колонка (живлення) | Wireless Speaker | Віртуальна колонка | Увімкнути / вимкнути живлення | Цифровий | 0 – вимкнено, 1 – увімкнено |

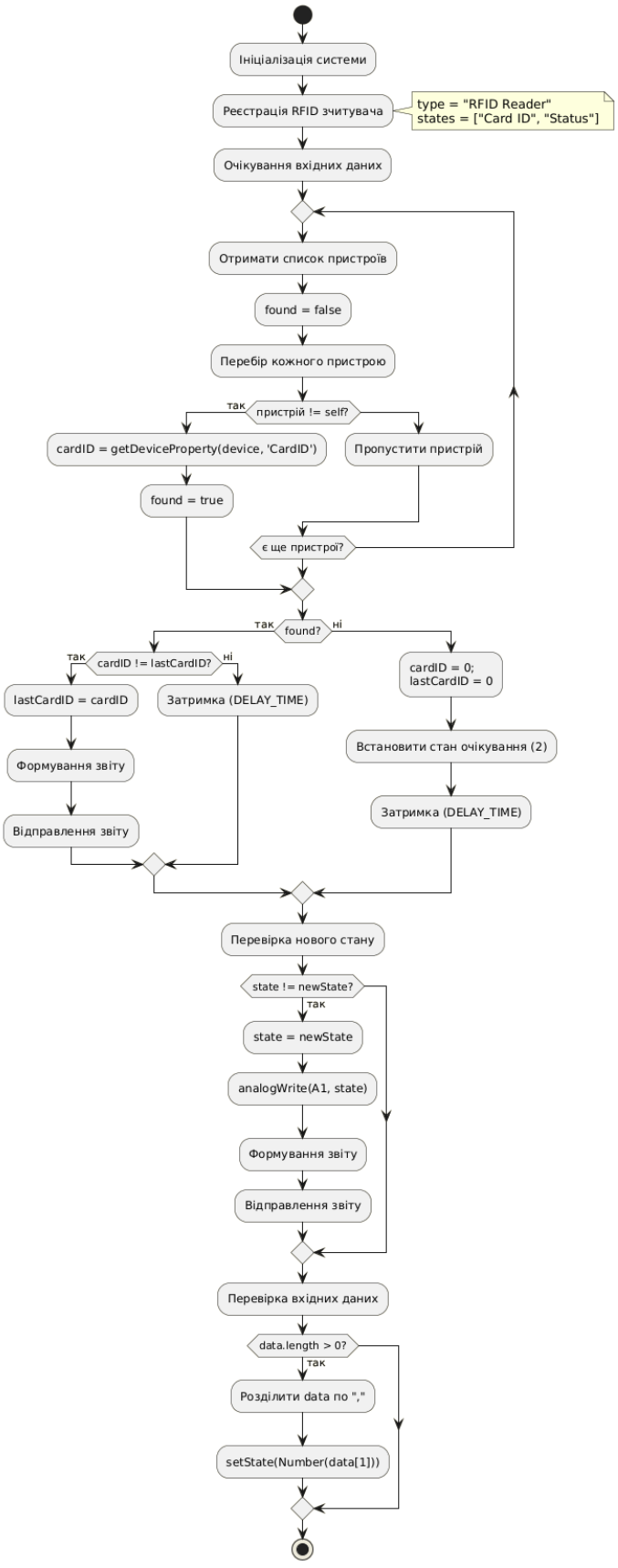



Рисунок 2.3 – Блок-схема алгоритму RFID-зчитувача

Основний цикл роботи програми, що виконується у функції loop(),

полягає в періодичному скануванні навколишнього віртуального середовища на наявність RFID-міток у визначеному радіусі. При виявленні RFID-картки зчитується її ідентифікатор (CardID). Для запобігання надмірному звітуванню про одну й ту саму картку, поточний ID порівнюється з останнім зчитаним ID (lastCardID). У випадку виявлення нової картки, її ID зберігається, і на сервер IoE відправляється звіт, що містить ідентифікатор картки та поточний стан зчитувача. Якщо в області сканування не знайдено жодної RFID-мітки, поточний та останній зчитані ID скидаються до нуля, а стан зчитувача встановлюється на "Waiting". Між ітераціями циклу передбачена затримка в одну секунду.

Функція setState(newState) відповідає за зміну внутрішнього стану RFID-зчитувача. Вона приймає новий стан як аргумент і, якщо цей стан відрізняється від поточного, оновлює значення змінної state, а також здійснює запис цього значення на аналоговий вихід A1 (що може використовуватися для локальної індикації статусу). Після зміни стану на сервер IoE відправляється відповідний звіт. Функція sendReport() формує рядок звіту, що складається зі зчитаного ідентифікатора RFID-мітки та поточного стану зчитувача, розділених комою, і відправляє цей звіт на сервер IoE за допомогою функції IoEClient.reportStates(). Нарешті, функція processData(data, bIsRemote) обробляє вхідні дані, отримані від сервера IoE. Вона перевіряє, чи дані не є порожніми, розділяє їх на окремі значення за комою та використовує друге значення (індекс 1) для встановлення нового стану RFID-зчитувача. Перше значення (індекс 0), яке потенційно може містити ID картки для валідації, в поточній реалізації функції ігнорується (рис.2.4).

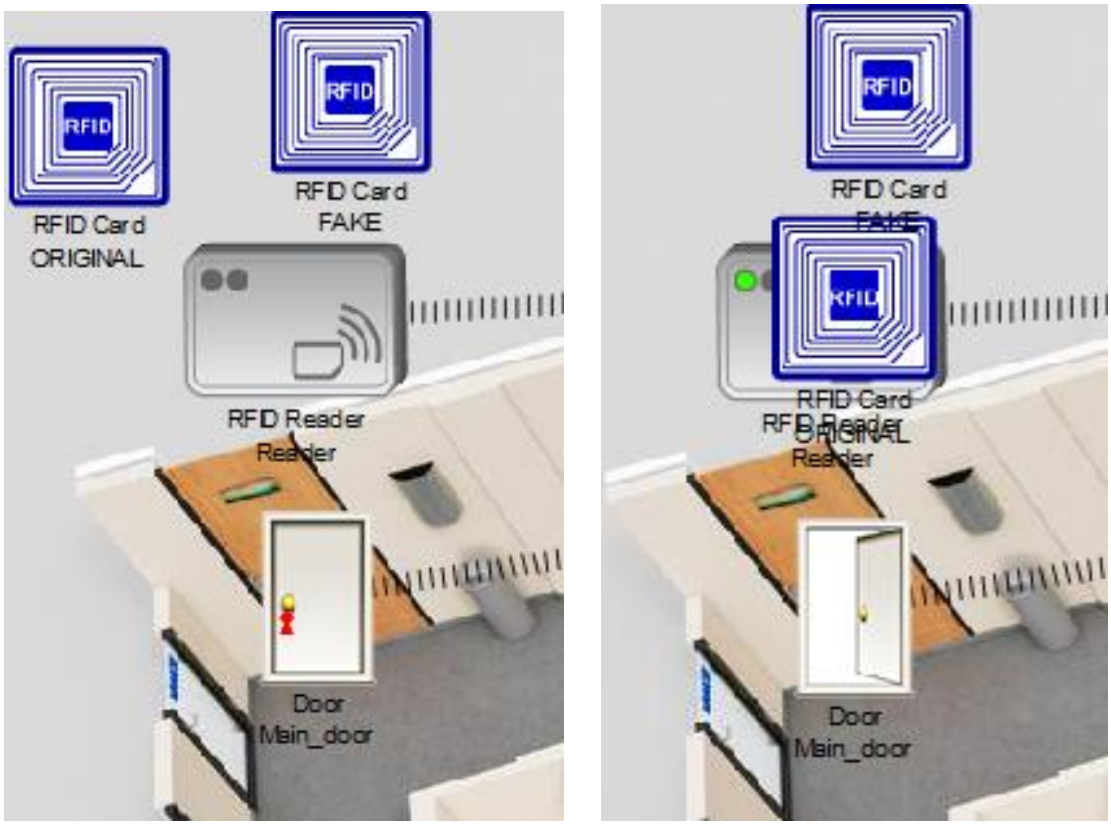



Рисунок 2.4 – Результат налаштування RFID-зчитувача

### 2.3.2 Налаштування «розумного вікна»

Пристрій "Вікно" реєструється на сервері IoE та має контрольований булевий стан "On", що представляє його відкритий (1) або закритий (0) стан. Процес налаштування включає імпорт необхідних бібліотек, визначення констант, що регулюють вплив вікна на екологічні параметри, ініціалізацію початкового стану вікна, а також встановлення обробників подій для отримання команд від сервера (onInputReceiveDone) та локальних віртуальних подій (клік миші mouseEvent та зміна стану віртуального пін-коду detect). Попередній стан вікна відновлюється при запуску з використанням збережених властивостей пристрою (рис.2.5) [11].

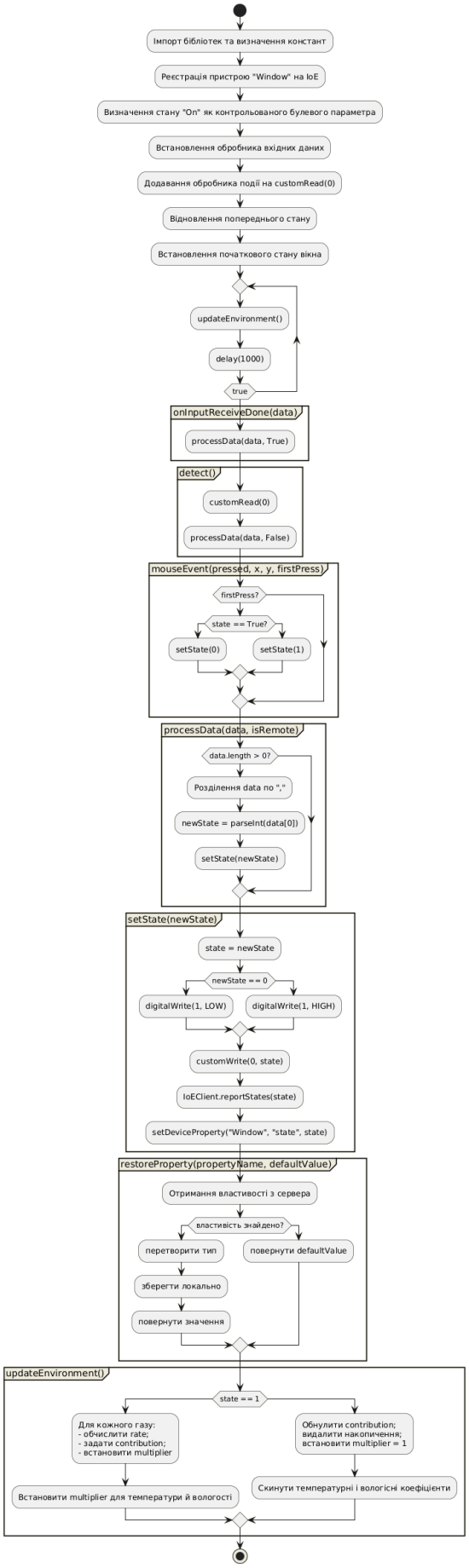



Рисунок 2.5 – Блок-схема алгоритму «розумне вікно»

Основний принцип роботи полягає в керуванні станом вікна та моделюванні його впливу на віртуальне офісне середовище. Стан вікна може змінюватися дистанційно через команди з сервера IoE або локально через взаємодію користувача (клік миші або зміна віртуального пін-коду). Функція setState(newState) відповідає за встановлення нового стану, відображаючи його на віртуальному цифровому виході та повідомляючи про зміну на сервер IoE, а також зберігаючи стан локально. Циклічно виконувана функція updateEnvironment() моделює вплив відчиненого вікна на концентрацію різних газів (згідно зі списком ENVIRONMENTS), температуру ("Ambient Temperature") та вологість ("Humidity") у віртуальному офісі. При відчиненому вікні швидкість зміни цих параметрів збільшується на визначені множники, а також встановлюється внесок відчиненого вікна у зміну концентрації газів. При зачиненому вікні вплив на екологічні параметри скасовується шляхом встановлення відповідних множників та внесків на нуль.

Функція processData(data, bIsRemote) обробляє вхідні команди від сервера IoE, отримуючи рядок даних, розділений комами, та використовуючи перше значення для встановлення нового стану вікна. Функції onInputReceiveDone() та detect() служать проміжними обробниками подій, передаючи отримані дані до processData(). Функція mouseEvent() забезпечує локальне керування станом вікна при кліку на нього у віртуальному середовищі. Таким чином, "розумне" вікно не лише може дистанційно відкриватися та закриватися, але й динамічно впливає на модельовані екологічні умови офісного простору в Cisco Packet Tracer (рис.2.6).

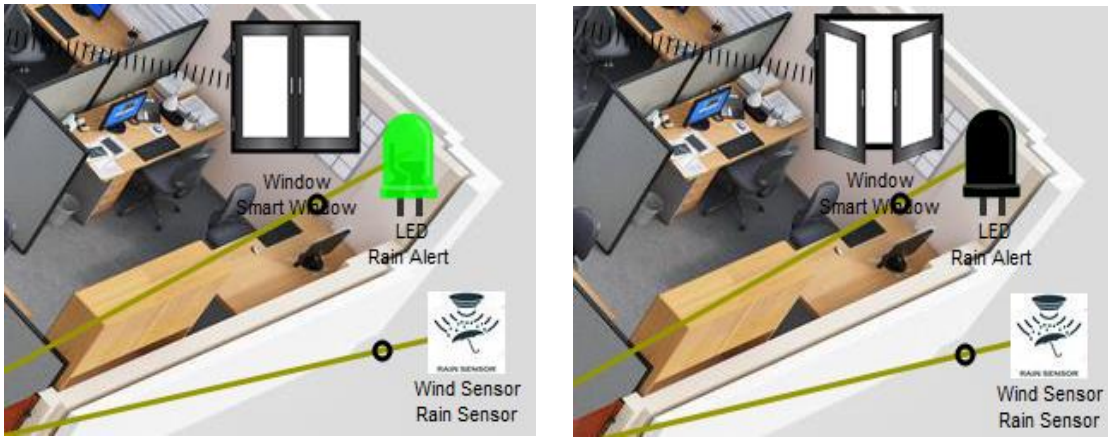



Рисунок 2.6 – Результат налаштування RFID-зчитувача

Рисунок 2.7 демонструє графік зміни двох екологічних параметрів протягом 24-годинного періоду, відображених у віртуальному середовищі Cisco Packet Tracer. По горизонтальній осі відкладено час, починаючи з 00:00 і закінчуючи 23:00, з годинними інтервалами. По вертикальній осі відображено значення цих параметрів у певних одиницях, що варіюються від 0 до 98.

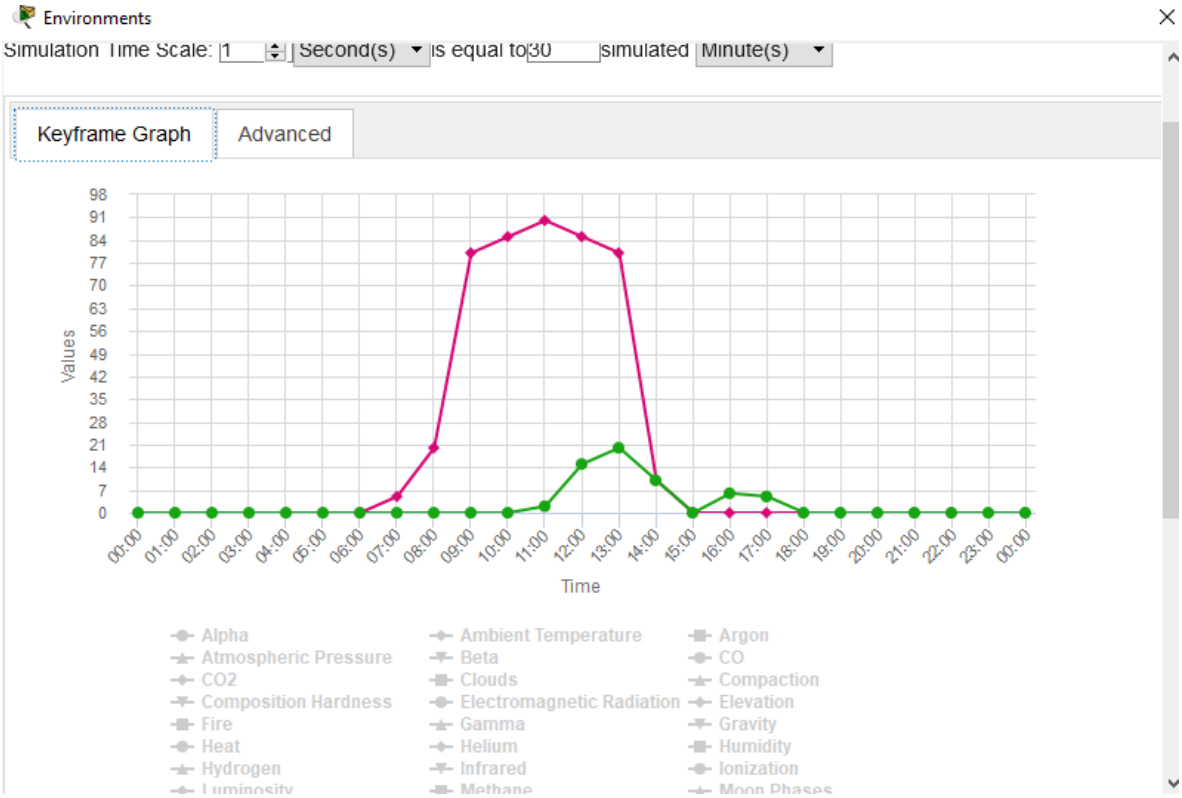



Рисунок 2.7 – Графік екологічних параметрів

На графіку представлені дві криві, що відображають динаміку двох різних параметрів. Одна крива, позначена рожевим кольором, демонструє різке зростання значення приблизно з 07:00, досягаючи пікового значення близько 85 у період між 11:00 та 12:00. Після цього спостерігається поступове зниження значення, яке стає близьким до початкового рівня (0) приблизно до 15:00 і залишається таким протягом решти дня.

### 2.3.3 Налаштування зарядки сонячної батареї

Основною метою є симуляція виробництва електроенергії на основі рівня сонячного освітлення у віртуальному середовищі та передача даних про вироблену енергію на сервер IoE. Процес налаштування (рис.2.8) включає визначення ключових констант, таких як назва параметра освітлення, коефіцієнт масштабування, максимальна потужність панелі, її ефективність та основа логарифма для масштабування вихідного сигналу.

Також ініціалізуються змінні стану пристрою та поточного рівня виробленої електроенергії. У функції setup() відбувається реєстрація пристрою "Solar" на сервері IoE з визначенням стану "Status" (вироблена енергія у ват-годинах) як параметра лише для читання та встановлюється обробник вхідних даних (хоча в поточній версії коду він не використовується). Початковий звіт про стан пристрою відправляється на сервер [10].

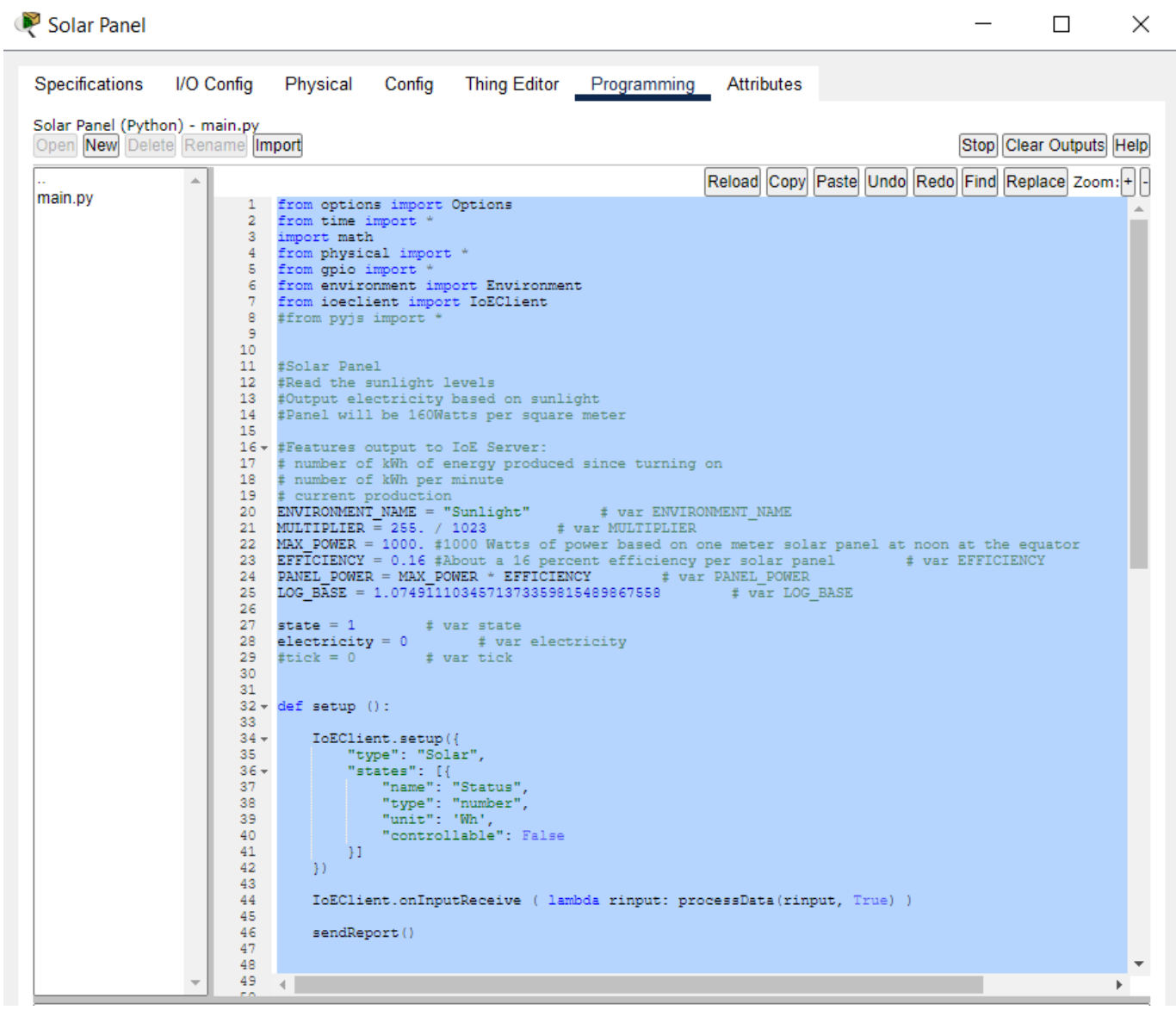



Рисунок 2.8 – Фрагмент програмного коду налаштування сонячної батареї

Основний цикл роботи програми, що виконується у функції loop(), полягає в періодичному (кожної секунди) оновленні значення виробленої електроенергії на основі поточного рівня сонячного освітлення, отриманого з віртуального середовища за допомогою функції getElectricityProduction(). Розраховане значення відображається у віртуальному інтерфейсі пристрою за допомогою displayElectricity(), передається на сервер IoE через sendReport() і використовується для керування рівнем аналогового виходу 0 за допомогою outputElectricity(). Функція getElectricityProduction() обчислює вироблену

потужність пропорційно рівню освітлення та максимальній потужності панелі з урахуванням її ефективності. displayElectricity() виводить текстове значення поточної виробленої енергії на екрані пристрою. sendReport() передає це значення як стан пристрою на сервер IoE та зберігає його як властивість пристрою. outputElectricity() здійснює нелінійне (логарифмічне) масштабування значення виробленої електроенергії для виведення на аналоговий вихід, що дозволяє ефективно представляти широкий діапазон значень у обмеженому діапазоні вихідного сигналу. Таким чином, віртуальна сонячна панель в режимі реального часу реагує на зміни рівня освітлення, моделюючи процес виробництва електроенергії та надаючи відповідні дані для моніторингу в IoT-системі.

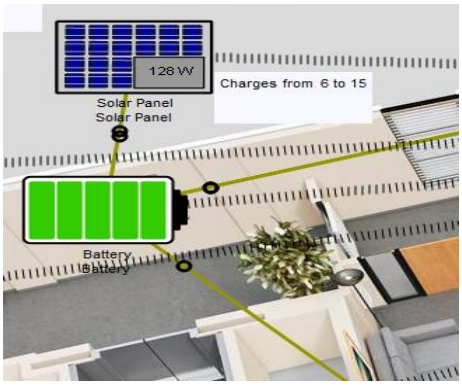



Рисунок 2.9 – Результат налаштування сонячної батареї

### 2.3.4 Налаштування системи несанкціонованого доступу

Для забезпечення захисту від несанкціонованого проникнення через вікна в офісному просторі передбачено використання датчика спрацьовування. Цей датчик, інтегрований в систему безпеки, виконує функцію виявлення спроб злому, зокрема розбиття вікна з метою проникнення всередину приміщення. У випадку фіксації подібної події, датчик спрацьовування миттєво ініціює активацію сирени. Сирена, у свою чергу, слугує гучним звуковим сигналом тривоги, який має на меті відлякати потенційного зловмисника та сповістити про інцидент осіб, що знаходяться

поблизу або відповідальних за безпеку об'єкта. Таким чином, комбінація датчика спрацьовування та сирени утворює ефективний механізм раннього виявлення та реагування на спроби крадіжки через вікна (рис.2.10).

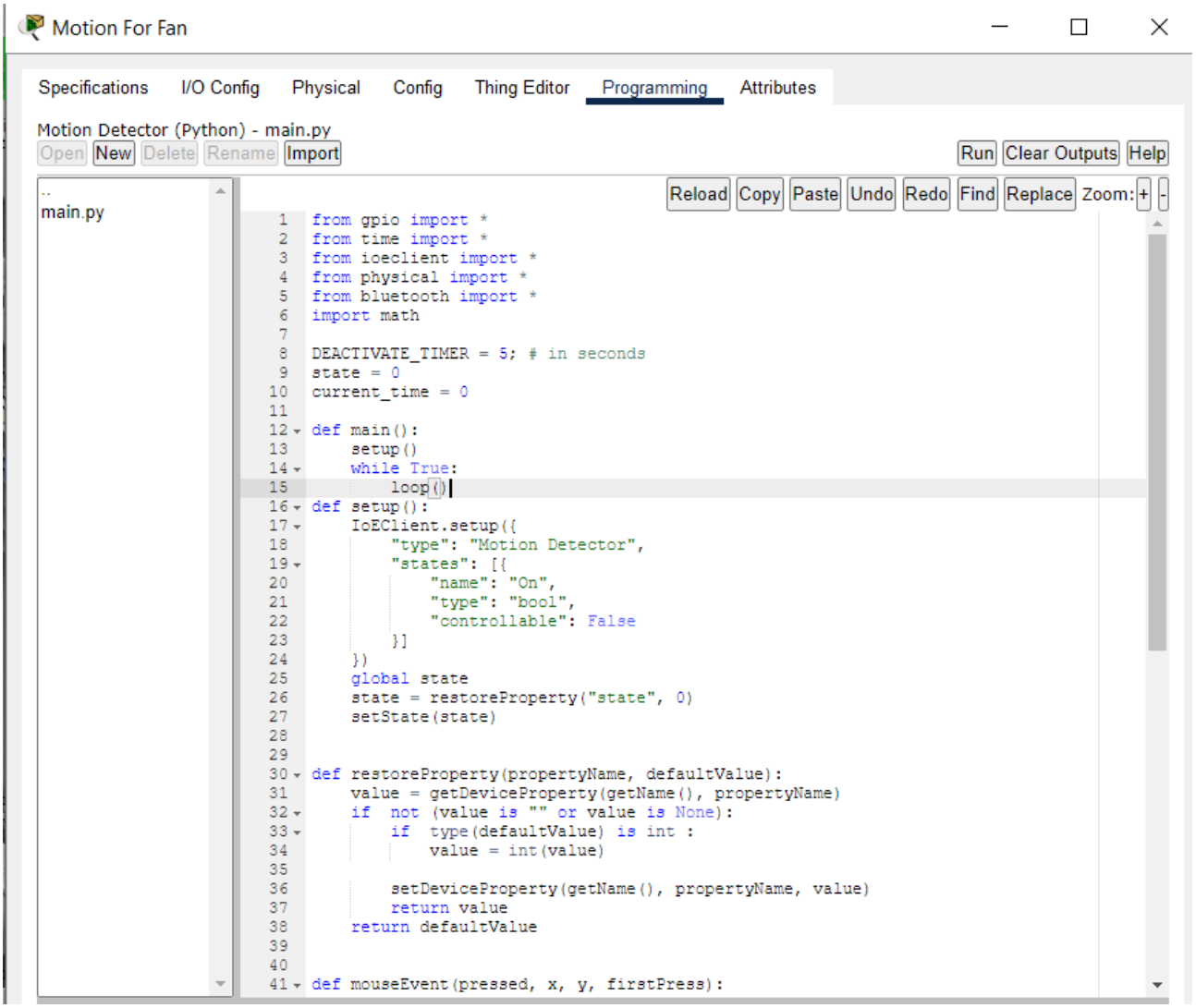



Рисунок 2.10 – Налаштування сповіщення від несанкціонованого доступу

### 2.3.5 Налаштування кавоварки та музичного плеєра

Основна функціональність кавоварки полягає у керуванні її станом (увімкнено/вимкнено) як дистанційно, так і локально, а також у звітуванні про свій поточний стан на сервер IoE (рис.2.11) [12].

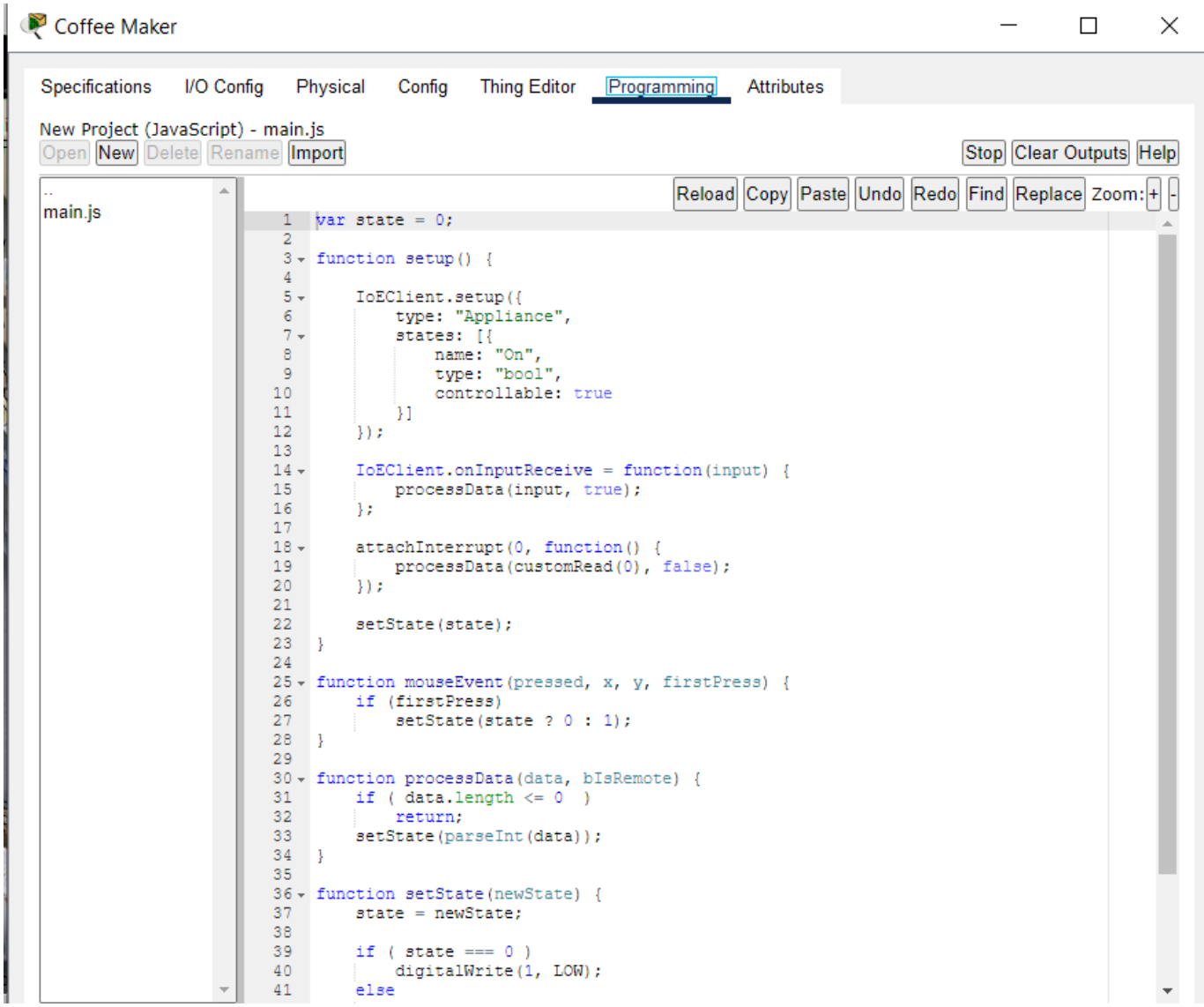



Рисунок 2.11 – Налаштування кавоварки

Процес налаштування в функції setup() включає реєстрацію пристрою на сервері IoE з визначенням стану "On" булевого типу, яким можна керувати дистанційно (controllable: true). Також встановлюються обробники подій: анонімна функція для обробки переривання на віртуальному піні 0, яка викликає processData() з локальними даними, та функція зворотного виклику для обробки вхідних даних від сервера IoE (IoEClient.onInputReceive), яка також передає дані до processData(). Початковий стан кавоварки встановлюється за допомогою функції setState(state), де state ініціалізовано як 0 (вимкнено).

Основний принцип роботи кавоварки полягає у зміні її стану. Локально

стан змінюється при кліку миші на віртуальному пристрої, що обробляється функцією mouseEvent(). При кожному першому натисканні стан інвертується (якщо було вимкнено, стає увімкнено, і навпаки), і викликається функція setState() для застосування нової зміни. Дистанційне керування станом здійснюється через сервер IoE, команди від якого приймаються та обробляються функцією processData(). Ця функція перевіряє наявність вхідних даних і, якщо вони є, передає їх цілочисельне значення до функції setState().

Функція setState(newState) відповідає за фактичну зміну стану кавоварки. Вона оновлює глобальну змінну state відповідно до newState. Якщо newState дорівнює 0 (вимкнено), на віртуальний цифровий вихід 1 встановлюється низький рівень (digitalWrite(1, LOW)), що символізує вимкнений стан. Якщо newState дорівнює 1 (увімкнено), на вихід встановлюється високий рівень (digitalWrite(1, HIGH)), що символізує увімкнений стан. Поточний стан також записується у віртуальну властивість пристрою (customWrite(0, state)), повідомляється на сервер IoE (IoEClient.reportStates(state)) та зберігається локально (setDeviceProperty(getName(), "state", state)). Таким чином, код забезпечує базове керування віртуальною кавоваркою та відображення її стану в середовищі Cisco Packet Tracer (рис.2.12).


Рисунок 2.12 – Налаштування кавоварки

Основна функціональність плеєра полягає у керуванні його станом

(увімкнено/вимкнено), передачі команди відтворення/зупинки через Bluetooth на інший віртуальний пристрій, а також у звітуванні про свій поточний стан на сервер IoE (рис.2.13).

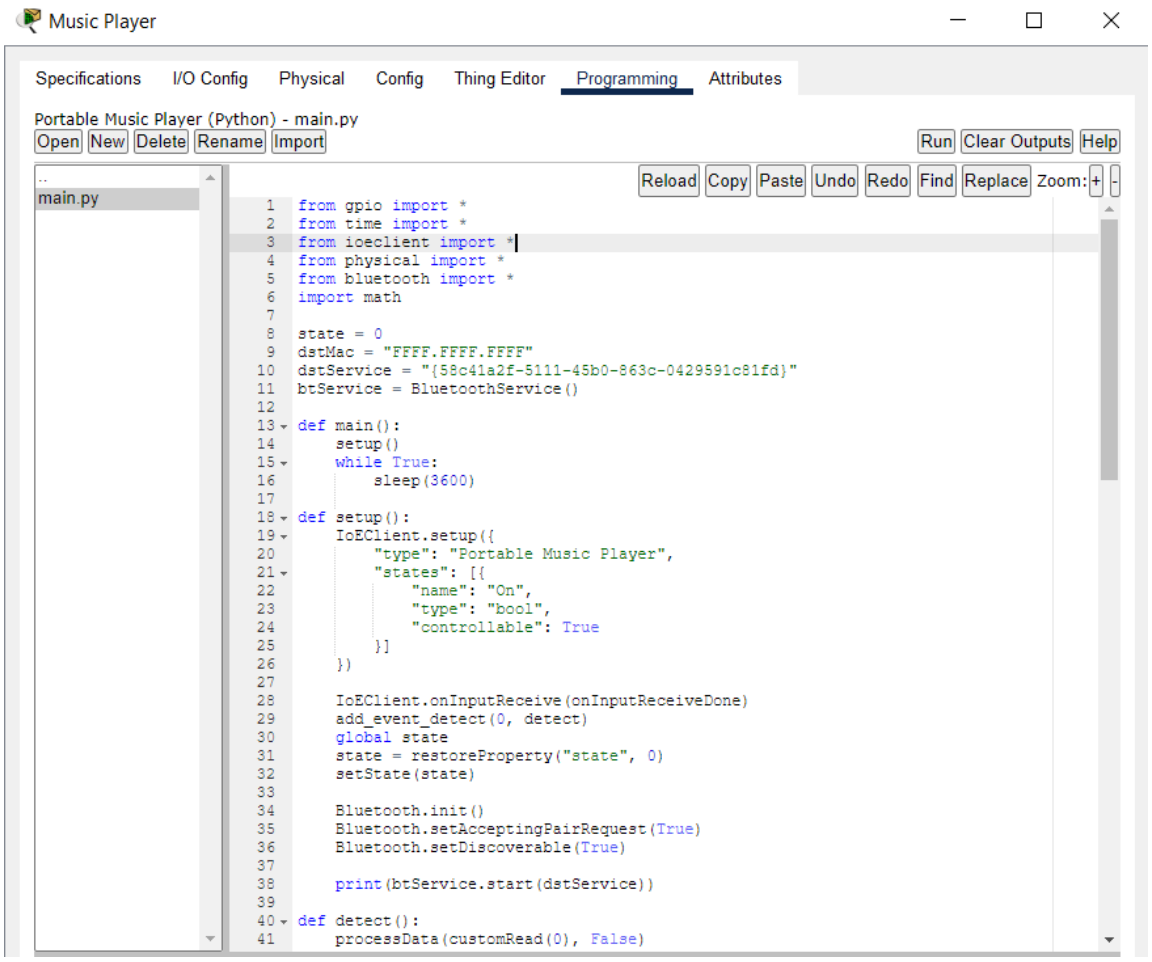



Рисунок 2.13 – Налаштування плеєра

Процес налаштування в функції setup() включає реєстрацію пристрою на сервері IoE з визначенням стану "On" булевого типу, яким можна керувати дистанційно (controllable: true). Також встановлюються обробники подій: onInputReceiveDone() для обробки вхідних даних від сервера IoE та detect() для реагування на зміну віртуального пін-коду 0. Початковий стан плеєра відновлюється зі збережених властивостей пристрою за допомогою функції restoreProperty() та встановлюється за допомогою setState().

Важливою частиною налаштування є ініціалізація Bluetooth-модуля віртуального пристрою. Встановлюється можливість приймати запити на

парування (Bluetooth.setAcceptingPairRequest(True)), вмикається режим виявлення (Bluetooth.setDiscoverable(True)), та запускається Bluetooth-сервіс з визначеним UUID (dstService). Також визначається MAC-адреса пристрою призначення (dstMac), на який будуть надсилатися команди. Обробники подій Bluetooth (для запитів на парування, успішного/невдалого парування, підключення та відключення) також налаштовуються для виведення відповідних повідомлень у консоль (рис.2.14).


Рисунок 2.14 – Результат роботи плеєра

Основний принцип роботи плеєра полягає у зміні його стану. Локально стан змінюється при кліку миші на віртуальному пристрої, що обробляється функцією mouseEvent(), інвертуючи поточний стан. Дистанційне керування станом здійснюється через сервер IoE, команди від якого приймаються та обробляються функцією processData(), встановлюючи новий стан плеєра.

Функція setState(newState) відповідає за фактичну зміну стану плеєра. Вона оновлює глобальну змінну state. Якщо newState дорівнює 0 (вимкнено), на віртуальний цифровий вихід 1 встановлюється низький рівень, і через Bluetooth на пристрій з MAC-адресою dstMac та сервісом dstService надсилається пусте повідомлення (ймовірно, команда "зупинити"). Якщо newState дорівнює 1 (увімкнено), на віртуальний цифровий вихід 1 встановлюється високий рівень, і через Bluetooth надсилається команда на відтворення звукового файлу "crickets.wav" з віртуального шляху

"/../Sounds/". Поточний стан плеєра також повідомляється на сервер IoE та зберігається локально. Таким чином, код забезпечує базове керування віртуальним музичним плеєром, включаючи дистанційне керування, локальне керування через мишу та передачу команд відтворення/зупинки через Bluetooth на інший віртуальний пристрій.

### 2.3.6 Налаштування пожежної сигналізації

Налаштування пожежної сигналізації в віртуальному середовищі Cisco Packet Tracer (IoE JavaScript API) [12] передбачає створення віртуальних пристроїв, які імітують роботу відповідних датчиків та виконавчих механізмів, а також програмування логіки їхньої взаємодії. Оскільки конкретний код налаштування не надано, я опишу загальний підхід та можливі елементи такого налаштування, базуючись на принципах роботи IoT-пристроїв та можливостях IoE API (рис.2.15).

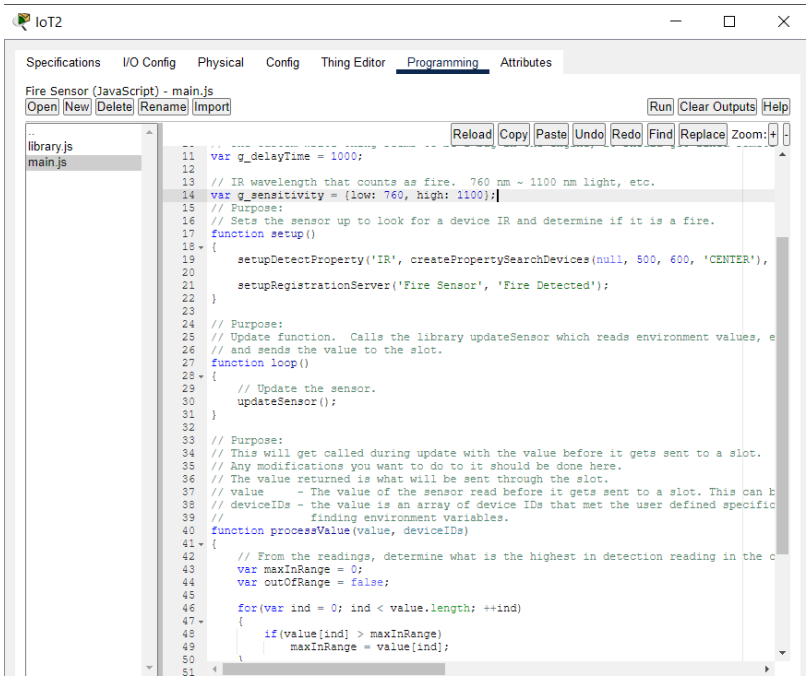



Рисунок 2.15 – Фрагмент коду налаштування пожежної сигналізації

У функції setup(), датчик налаштовується на виявлення інфрачервоного випромінювання ('IR') від інших віртуальних пристроїв, що знаходяться в

радіусі 500x600 віртуальних одиниць навколо центру поточного датчика. Результат виявлення передається як цифровий сигнал (HIGH або LOW) на віртуальний цифровий вихід 0. Також налаштовується реєстрація цього датчика на сервері IoE як "Fire Sensor" зі станом "Fire Detected" булевого типу.

Принцип роботи полягає в циклічному виконанні функції loop(), яка викликає updateSensor() з бібліотеки. Ця функція збирає дані про рівень інфрачервоного випромінювання від усіх пристроїв у визначеній області. Потім функція processValue(value, deviceIDs) обробляє отримані значення. Вона визначає максимальний рівень інфрачервоного випромінювання серед усіх знайдених пристроїв та порівнює його із заданим діапазоном чутливості датчика (760-1100 одиниць). Якщо максимальне значення потрапляє в цей діапазон, функція повертає HIGH, сигналізуючи про виявлення можливого полум'я. В іншому випадку повертається LOW. Це оброблене значення потім передається на віртуальний цифровий вихід 0, і стан "Fire Detected" (true для HIGH, false для LOW) повідомляється на сервер IoE, дозволяючи іншим компонентам віртуальної системи реагувати на потенційну пожежу (рис.2.16).


Рисунок 2.16 – Результат роботи пожежної сигналізації

### 2.3.6 Моделювання IoT-системи офісного простору

Однією з ключових підсистем є контроль доступу, представлена RFID-зчитувачем (RFID Reader) та, ймовірно, електромагнітним замком дверей (нечітко видно, але передбачається). Ця система забезпечує авторизований доступ до офісних приміщень за допомогою RFID-карток.

Для забезпечення енергоефективності в проєкті присутня сонячна панель (Solar Panel), яка, ймовірно, використовується для живлення деяких пристроїв або для моніторингу генерації енергії. Також можна помітити розумний ліхтар вуличного освітлення (Smart Garage/Street Lamp), який може автоматично регулювати свою яскравість залежно від часу доби або наявності руху.

Моніторинг навколишнього середовища представлений різними датчиками, такими як датчик температури та вологості (Temp/Hum Sensor), датчик якості повітря (Air Quality Sensor), а також датчики газів (CO, CO2, Smoke). Ці датчики збирають дані про стан офісного середовища, що може використовуватися для автоматичного регулювання клімату, вентиляції або для сповіщення про небезпечні рівні забруднення.

Для забезпечення безпеки в проєкті передбачена система відеонагляду (IP Camera), яка здійснює моніторинг офісних приміщень. Також присутній датчик спрацьовування (Trip Sensor), який може реагувати на відкриття дверей або вікон, та сирена (Siren) для звукового оповіщення у випадку тривоги. Додатково, є датчик пожежі (Fire Sensor), який виявляє задимлення або підвищення температури.

Для комфорту та автоматизації офісного простору присутні розумне вікно (Smart Window), яке може автоматично відкриватися або закриватися для регулювання температури та вентиляції, а також розумна кавоварка (Smart Coffee Maker), якою можна керувати дистанційно або за розкладом. Музичний плеєр (Portable Music Player) також може бути частиною системи розваг або сповіщень. Система поливу рослин (Water Sprinkler/Soil Sensor) автоматично підтримує оптимальний рівень вологості ґрунту (рис.2.17).


Рисунок 2.17 – Моделювання IoT-системи офісного простору в Cisco PT

# ВИСНОВКИ

У кваліфікаційній роботі розроблено та досліджено розподілену IoT-систему, призначену для забезпечення контролю доступу та підвищення рівня безпеки в сучасному офісному просторі. Проведене дослідження підтвердило актуальність та перспективність використання технологій Інтернету речей для вирішення завдань безпеки та управління доступом, демонструючи їхній потенціал у створенні гнучких, масштабованих та інтелектуальних систем.

Розроблена розподілена архітектура системи, що включає взаємодію між різними типами IoT-пристроїв (RFID-зчитувачами, датчиками спрацьовування, камерами відеонагляду, сиренами), локальною мережею та централізованою платформою управління, довела свою ефективність у забезпеченні комплексного підходу до контролю доступу та безпеки. Використання протоколів зв'язку IoT дозволило забезпечити надійну та ефективну передачу даних між пристроями та сервером, а модульність розробленого програмного забезпечення сприяє легкому розширенню та адаптації системи до майбутніх потреб офісного середовища.

Результати моделювання та експериментальної перевірки окремих компонентів системи в середовищі Cisco Packet Tracer підтвердили працездатність запропонованих рішень та їхню здатність виконувати поставлені завдання. Зокрема, було продемонстровано ефективність системи контролю доступу на основі RFID-технології, можливість виявлення несанкціонованого проникнення за допомогою датчиків спрацьовування та своєчасне оповіщення про потенційні загрози через систему тривоги. Інтеграція з системою відеонагляду забезпечує додатковий рівень безпеки та можливість візуального підтвердження подій.

Розглянуті в роботі аспекти енергоефективності, можливості інтеграції з іншими офісними системами (освітлення, клімат-контроль) та потенціал використання штучного інтелекту для аналізу даних безпеки відкривають

нові напрямки для подальшого розвитку та вдосконалення подібних IoT-систем.

Таким чином, розроблена розподілена IoT-система для контролю доступу та безпеки в офісному просторі є перспективним рішенням, що поєднує в собі переваги технологій Інтернету речей для створення інтелектуального, безпечного та ефективного робочого середовища. Результати даної роботи можуть слугувати основою для подальших досліджень та практичного впровадження подібних систем в реальних офісних просторах.

# ПЕРЕЛІК ПОСИЛАНЬ

1. Z., Houhamdi, & B., Athamena, (2020). Identity identification and management in the internet of things. International. Arab Journal of Information Technology, 17(4A), 645-654.

2. Keke, M., J. Egerega, and I. D. Emeyazia. "Design, development and performance evaluation of a smart office automation system." Applied Journal of Physical Science 3.2 (2021): 81-85.

3. Sunchu, R., Palli, S., Datta, V.S.R. and Shanmugasundaram, M., 2019, April. Intelligent system for office environment using internet of things. In 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI) (pp. 717-721).

4. Sharma, D., Sharma, H. and Panchal, D., 2020, December. Automatic Office Environment System for Employees Using IoT and Computer Vision. In 2020 IEEE 17th India Council International Conference (INDICON) (pp. 1-6).

5. Prentice, C.T. and Karakonstantis, G., 2018, October. Smart office system with face detection at the edge. In 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (pp. 88-93).

6. Internet of Things Global Standards Initiative [Електронний ресурс] – Режим доступу до ресурсу: http://www.itu.int.

7. Tang G., Yan Y., Shen C., Jia X., Zinn M., Trivedi Z., Yingling A., Westover K., Jiang S. Development of a real-time indoor location system using Bluetooth low energy technology and deep learning to facilitate clinical applications. (2020), pp. 3277-3285.

8. Candanedo, L.M., Feldheim, V. Accurate occupancy detection of an office room from light, temperature, humidity and CO2 measurements using statistical learning models (2016), pp. 28-39.

9. Smys, S. A Survey on Internet of Things (IoT) based Smart Systems.

Journal of ISMAC 2, no. 04 (2020): 181-189

10. Smart Office: Wireless Sensor Network for Energy Monitoring and User Profiling-PFC_memòria [Електронний ресурс] – Режим доступу до ресурсу: http://hdl.handle.net/2099.1/11094.

11. The Complete Guide to UML Diagram Types with Examples [Електронний ресурс] – Режим доступу до ресурсу: http://creately.com/blog/diagrams/uml-diagramtypes-examples/#Activity Diagram.

12. Комп'ютерна академія Cisco [Електронний ресурс] – Режим доступу: https://www.netacad.com.

# ДОДАТОК А

Текст програми розподіленої IoT-системи для контролю доступу та безпеки в офісному просторі

**Міністерство освіти і науки України**
**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ**
**“ДНІПРОВСЬКА ПОЛІТЕХНІКА”**

**ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ**

# РОЗПОДІЛЕНА ІОТ-СИСТЕМА ДЛЯ КОНТРОЛЮ ДОСТУПУ ТА БЕЗПЕКИ В ОФІСНОМУ ПРОСТОРІ

Текст програми

804.02070743.25018-01 12 01

Листів 8

2025

# АНОТАЦІЯ

Дана робота представляє собою концепцію розподіленої IoT-системи, призначеної для значного підвищення рівня безпеки та ефективності контролю доступу в сучасному офісному просторі.

Система розроблена для комплексного вирішення задач автоматизації та безпеки, охоплюючи широкий спектр функціональних підсистем: від інтелектуального управління доступом за допомогою RFID-замків, розумного керування вікнами та освітленням, що реагує на погодні умови та час доби, до автономних систем живлення, захисту від несанкціонованого проникнення та пожежної/димової сигналізації. Вона також включає елементи комфорту, такі як автоматизація їдальні та відтворення музики. Основна ідея полягає в інтеграції цих підсистем у єдину, взаємопов'язану мережу, що дозволяє централізовано моніторити та керувати офісним середовищем.

Проєкт реалізовано шляхом моделювання в середовищі Cisco Packet Tracer, що забезпечує гнучке та візуальне представлення роботи системи та її окремих компонентів. Це дозволяє детально аналізувати взаємодію пристроїв, ефективність запропонованих рішень та їх потенційний внесок у створення безпечного, енергоефективного та комфортного офісного простору. Робота може бути використана для навчальних цілей, вдосконалення знань у галузі IoT та системної інтеграції, а також як основа для розробки реальних рішень для "розумних" будівель.

# ЗМІСТ

Solar Panel

```
from options import options
from time import *
import math
from physical import *
from gpio import *
from environment import environment
from ioeclient import ioeclient
#from pyjs import *


#solar panel
#read the sunlight levels
#output electricity based on sunlight
#panel will be 160watts per square meter

#features output to ioe server:
# number of kwh of energy produced since turning on
# number of kwh per minute
# current production
environment_name = "sunlight"        # var environment_name
multiplier = 255. / 1023        # var multiplier
max_power = 1000. #1000 watts of power based on one meter solar panel at noon
at the equator        # var max_power
efficiency = 0.16 #about a 16 percent efficiency per solar panel        # var
efficiency
panel_power = max_power * efficiency        # var panel_power
log_base = 1.074911103457137335981548986755 8        # var log_base

state = 1        # var state
electricity = 0        # var electricity
#tick = 0        # var tick


def setup ():

    ioeclient.setup({
        "type": "solar",
        "states": [{
            "name": "status",
            "type": "number",
            "unit": 'wh',
            "controllable": false
```

```
    }]
})

ioeclient.oninputreceive ( lambda rinput: processdata(rinput, true) )

sendreport()



def loop ():
    global electricity
    ##    if  (tick++ % 10) is 0 )    # is tick consistent across devices?
    ##    {
    electricity = int(getelectricityproduction())
    ##print(electricity)
    displayelectricity()
    sendreport()
    outputelectricity()
    delay(1000)
    ##



def displayelectricity ():
    setcustomtext(70, 45, 1000, 1000, str(int(electricity)) + '\tw')



def getelectricityproduction ():
    return panel_power * environment.get(environment_name) / 100



def sendreport ():
    report = state # comma seperated states        # var report
    ioeclient.reportstates(electricity)
    setdeviceproperty(getname(), "level", electricity)



def outputelectricity ():
    el_log = math.floor(math.log(electricity) / math.log(log_base))        # var el_log
    if el_log < 0:
```

```
        el_log = 0
    elif el_log > 255:
        el_log = 255
    ##    print(el_log)
    analogwrite(0, el_log)



if __name__ == "__main__":
    setup()
    while true:
        loop()
        sleep(0)
```

Speaker

```
from gpio import *
from time import *
from ioeclient import *
from physical import *
from bluetooth import *
import math

dstservice = "{58c41a2f-5111-45b0-863c-0429591c81fd}"
btservice = bluetoothservice()
state = 0
active = 0

def setup ():
    ioeclient.setup({
        "type": "bluetooth speaker",
        "states": [{
            "name": "connected",
            "type": "bool",
            "controllable": false
        },
        {
            "name": "playing",
            "type": "bool",
            "controllable": false
        }]
    })
    global state
    global active
    global dstservice
    global bluetooth
    state = restoreproperty("state", 0)
    active = restoreproperty("active", 0)

    destroysounds()
    bluetooth.init()
    bluetooth.setacceptingpairrequest(true)
    bluetooth.setdiscoverable(true)
    print btservice.start(dstservice)


def restoreproperty(propertyname, defaultvalue):
    value = getdeviceproperty(getname(), propertyname)
```

```
    if  not (value == "" or value == none):
        if  type(defaultvalue) is int :
            value = int(value)

        setdeviceproperty(getname(), propertyname, value)
        return value
    return defaultvalue

def main ():
    setup()
    while true:
        updatestate()
        delay(1000)


def updatestate ():
    global state
    global active
    if float(active) == 0:
        digitalwrite(1, low)
    else:
        digitalwrite(1, high)

    report = str(state) + "," + str(active)
    ioeclient.reportstates(report)
    setdeviceproperty(getname(), "state", state)
    setdeviceproperty(getname(), "active", active)


def playmusic (sound):
    global active
    destroysounds()
    addsound("music", sound)
    playsound("music", -1)
    active = 1
    digitalwrite(1, high)


def stopmusic ():
    global active
    destroysounds()
    active = 0
    digitalwrite(1, low)
```

```
def onacceptpairrequestdone(mac, devicename):
    print "accepting pair request: " + str(mac)
    bluetooth.acceptpairrequest(mac, devicename)


def ondevicepairdone(mac):
    global state
    print "paired: " + str(mac)
    state = 1


def ondeviceunpairdone(mac):
    global state
    print "unpaired: " + str(mac)
    stopmusic()
    state = 0


def ondeviceconnectdone(mac):
    global state
    print "connected: " + str(mac)
    state = 1


def ondevicedisconnectdone(mac):
    global state
    print "disconnected: " + str(mac)
    stopmusic()
    state = 0


def onreceivedone(srcmac, srcservice, dstmac, dstservice, data):
    print "received from " + srcmac + ":" + srcservice + ": " + data
    if len(data) > 0:
        playmusic(data)
    else:
        stopmusic()

btservice.onreceive(onreceivedone)
bluetooth.ondevicedisconnect(ondevicedisconnectdone)
bluetooth.ondeviceconnect(ondeviceconnectdone)
bluetooth.ondevicepair(ondevicepairdone)
bluetooth.ondeviceunpair(ondeviceunpairdone)
bluetooth.onpairrequest(onacceptpairrequestdone)
```

```
if __name__ == "__main__":
    main()
```