МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра комп’ютерної інженерії

КВІТЕНЬ Дмитро Олександрович

Веб-інтерфейс для управління ресурсами Інтернету речей в корпоративних мережах / Web interface for managing IoT resources in corporate networks

спеціальність: 123 – Комп’ютерна інженерія
освітньо-професійна програма – Комп’ютерна інженерія

Кваліфікаційна робота

Виконав: студент групи КІ-41
КВІТЕНЬ Дмитро Олександрович

Науковий керівник
к.т.н. Мельник Г.М.

Кваліфікаційну роботу допущено
до захисту:

"___" _____ 20___ р.

Завідувач кафедри
_____О.Л. Дубчак

ТЕРНОПІЛЬ - 2024

АНОТАЦІЯ

Кваліфікаційна робота на тему «Веб-інтерфейс для управління ресурсами Інтернету речей в корпоративних мережах» зі спеціальності 123 «Комп'ютерна інженерія» освітнього ступеня «бакалавр» містить 54 сторінки пояснюючої записки, 19 рисунків, 2 таблиці, 3 додатки. Обсяг графічного матеріалу 2 аркуші формату А3.

Метою кваліфікаційної роботи є розробка веб-інтерфейсу для управління ресурсами Інтернету речей в корпоративних мережах з інтеграцією платформи Tuya, що дозволить забезпечити високий рівень безпеки, доступності та зручності для користувачів.

Методи дослідження включають аналіз існуючих рішень в області управління IoT, проектування архітектури веб-інтерфейсу, розробку прототипу, тестування та оптимізацію розробленого рішення. Особлива увага приділяється безпеці передачі даних та авторизації користувачів.

Результати роботи включають розробку веб-інтерфейсу, що забезпечує управління та моніторинг IoT пристроїв через платформу Tuya. Інтерфейс реалізовано з використанням сучасних веб-технологій, таких як HTML, CSS та JavaScript, що забезпечує зручність користування та швидкість роботи. Практичне значення полягає у можливості впровадження розробленого інтерфейсу в реальні корпоративні середовища для підвищення рівня автоматизації та контролю.

КЛЮЧОВІ СЛОВА: ІНТЕРНЕТ РЕЧЕЙ, ВЕБ-ІНТЕРФЕЙС, КОРПОРАТИВНІ МЕРЕЖІ, TUYA, БЕЗПЕКА ДАНИХ, АВТОМАТИЗАЦІЯ.

ANNOTATION

The qualification work on the topic "Web interface for managing IoT resources in corporate networks " from specialty 123 "Computer Engineering" of the bachelor's degree contains 54 pages of explanatory note, 19 figures, 2 tables, 3 appendices. The amount of graphic material is 2 sheets of A3 format.

The purpose of the qualification work is to develop a web interface for managing Internet of Things (IoT) resources in corporate networks with the integration of the Tuya platform, which will ensure a high level of security, accessibility and convenience for users.

The research methods include analysis of existing IoT management solutions, design of the web interface architecture, prototyping, testing, and optimization of the developed solution. Particular attention is paid to the security of data transmission and user authorization.

The results of the work include the development of a web interface that provides management and monitoring of IoT devices through the Tuya platform. The interface is implemented using modern web technologies such as HTML, CSS, and JavaScript, which ensures ease of use and speed of operation. The practical significance lies in the possibility of implementing the developed interface in real corporate environments to increase the level of automation and control.

KEYWORDS: INTERNET OF THINGS, WEB INTERFACE, CORPORATE NETWORKS, TUYA PLATFORM, DATA SECURITY, AUTOMATION.

ЗМІСТ

Перелік умовних скорочень 9
Вступ 10
1 Управління ресурсами Інтернету речей в корпоративних мережах 13
1.1 Інтеграція IoT пристроїв 13
1.2 Системи управління IoT 17
1.3 Порівняння різних платформ та технологій 19
1.4 Постановка задач кваліфікаційної роботи 22
2. Проєктування веб-інтерфейсів для управління ресурсами Інтернету речей 26
2.1 Вимоги до веб-інтерфейсів для управління IoT 26
2.2 Вибір датчиків системи 28
2.3 Розробка веб-інтерфейсу 32
3 Реалізація компонентів управління ресурсами IoT 38
3.1 Інтеграція з платформою Tuya 38
3.2 Алгоритми передачі даних на сайт 41
3.3 Тестування і впровадження вебінтерфейсу 46
Висновки 54
Список використаних джерел 55
Додаток А Техніко-економічне обґрунтування розробки 58
Додаток Б Код веб-інтерфейсу 65
Додаток В Світлокопія публікації ..... **Помилка! Закладку не визначено.**

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

IoT	–	Інтернет речей
API	–	Інтерфейс програмування додатків
JSON	–	Джерело безперебійного живлення
SQL	–	Мова структурованих запитів
HTML	–	Мова розмітки гіпертексту
CSS	–	Каскадні таблиці стилів
PHP	–	Мова гіпертекстових препроцесорів
MQTT	–	Протокол телеметрії передачі повідомлень
CoAP	–	Протокол обмежених додатків
HTTP	–	Протокол передачі гіпертексту
SSL/TLS	–	Протоколи захисту транспортного рівня
RTD	–	Терморезистори
NTC	–	Негативний температурний коефіцієнт (тип термістора)

ВСТУП

Розвиток Інтернету речей (IoT) створює нові можливості для автоматизації, моніторингу та управління різними аспектами нашого життя, включаючи побут, промисловість, транспорт та медицину. Кількість підключених до Інтернету пристроїв стрімко зростає, що вимагає нових підходів до управління ресурсами, забезпечення безпеки та зручності користування. Інтеграція IoT пристроїв у корпоративні мережі відкриває нові горизонти для підвищення ефективності бізнес-процесів, зниження витрат та покращення якості послуг.

Існуючі рішення для управління IoT пристроями часто не враховують специфічні вимоги корпоративного середовища, такі як високий рівень безпеки, масштабованість та зручність використання для кінцевих користувачів. Різні IoT пристрої можуть використовувати різні протоколи зв'язку та стандарти, що ускладнює їх інтеграцію в єдину систему управління. Інтеграція з платформами IoT, такими як Tuya, надає потужні інструменти для управління IoT пристроями та створення масштабованих рішень для корпоративних мереж. Використання таких платформ дозволяє підприємствам ефективно управляти своїми ресурсами, забезпечувати безпеку даних та забезпечувати зручний інтерфейс для кінцевих користувачів.

Актуальність даної роботи обумовлена кількома факторами. По-перше, зростання кількості IoT пристроїв вимагає розробки нових методів і інструментів для їх ефективного управління та моніторингу. По-друге, питання безпеки даних в IoT системах є критично важливим, оскільки злом пристроїв може призвести до значних втрат і порушень роботи корпоративних мереж. По-третє, зручність використання та масштабованість рішень є ключовими вимогами для сучасних корпоративних середовищ, що потребують надійних і гнучких інструментів для управління великими обсягами даних і численними пристроями.

У даній роботі особлива увага приділяється інтеграції з платформою Tuya, яка дозволяє ефективно підключати та управляти різноманітними IoT пристроями, використовуючи зручний веб-інтерфейс. Платформа Tuya забезпечує високу безпеку передачі даних, підтримку різних протоколів зв'язку та масштабованість, що робить її ідеальним рішенням для корпоративних мереж.

Метою кваліфікаційної роботи є розробка веб-інтерфейсу, який дозволить ефективно управляти ресурсами Інтернету речей в корпоративних мережах. Об’єкт дослідження – компоненти і системи Інтернету речей. Предмет дослідження – методи і засоби розробки веб-інтерфейсів. Для досягнення поставленої мети необхідно вирішити наступні завдання:

Аналіз існуючих рішень для управління ресурсами IoT в корпоративних мережах:

– вивчення основних вимог до веб-інтерфейсів для управління ресурсами IoT;

– проектування архітектури веб-інтерфейсу;

– розробка прототипу веб-інтерфейсу;

– тестування та оптимізація розробленого рішення.

За результатами роботи опубліковано тези доповіді на IX науково-практичній конференції «Інтелектуальні комп’ютерні системи та мережі» [1]. Копії публікації наведено у додатку В.

Розроблено прототип веб-інтерфейсу, який дозволяє здійснювати управління та моніторинг IoT пристроїв. Це дозволило провести практичну перевірку концепцій, тестування основних функцій та демонстрацію можливостей системи.

Робота складається з трьох розділів. Перший розділ присвячений аналізу існуючих рішень та вимог до веб-інтерфейсів для управління ресурсами IoT. У цьому розділі розглядаються різні платформи для управління IoT пристроями, такі як AWS IoT, Google Cloud IoT, Microsoft Azure IoT та Tuya Smart. Проводиться порівняльний аналіз їх функціональних можливостей,

безпеки, масштабованості та зручності використання. Також у розділі висвітлюються основні виклики та проблеми, пов'язані з інтеграцією IoT пристроїв у корпоративні мережі.

У другому розділі розглядаються датчики, які використовувались у проекті, та існуючі датчики. Особлива увага приділяється вибору відповідних сенсорів для моніторингу параметрів навколишнього середовища, таких як температура, вологість, рівень CO2 та інші. Розділ описує процес проектування архітектури веб-інтерфейсу, включаючи розробку концептуальної моделі, визначення ключових компонентів системи та їх взаємодії. Також висвітлюється вибір технологій та інструментів для реалізації веб-інтерфейсу.

Третій розділ містить опис розробки веб-інтерфейсу, а також тестування розробленого рішення. В цьому розділі детально описується процес реалізації основних модулів системи, таких як модулі підключення та конфігурації IoT пристроїв, модулі моніторингу та управління, а також модулі безпеки. Особлива увага приділяється інтеграції з платформою Tuya, налаштуванню взаємодії між пристроями та забезпеченню захищеної передачі даних. Також у розділі описується процес тестування та оптимізації веб-інтерфейсу, проведення функціонального та нефункціонального тестування, виявлення та усунення помилок, а також підготовка системи до впровадження в реальне корпоративне середовище.

1 УПРАВЛІННЯ РЕСУРСАМИ ІНТЕРНЕТУ РЕЧЕЙ В КОРПОРАТИВНИХ МЕРЕЖАХ

1.1 Інтеграція IoT пристроїв

Розвиток Інтернету речей (IoT) є однією з ключових технологічних тенденцій останніх років, яка кардинально змінює підходи до автоматизації, моніторингу та управління різними аспектами нашого життя. За прогнозами аналітиків, до 2025 року кількість підключених IoT пристроїв перевищить 75 мільярдів, що свідчить про стрімке зростання цієї галузі. Така динаміка створює нові можливості для підвищення ефективності бізнес-процесів, зниження витрат та покращення якості послуг у різних сферах діяльності, включаючи промисловість, транспорт, енергетику, охорону здоров'я та побут.

Впровадження IoT технологій у корпоративні мережі дозволяє створювати розумні системи управління, які забезпечують збір, аналіз та обробку даних у режимі реального часу. Це відкриває нові горизонти для оптимізації виробничих процесів, підвищення продуктивності та зниження витрат на обслуговування обладнання. Наприклад, IoT пристрої можуть бути використані для моніторингу стану обладнання, прогнозування поломок та автоматичного планування технічного обслуговування, що дозволяє уникнути простоїв та знизити витрати на ремонт.

Однак, зростаюча кількість підключених IoT пристроїв створює нові виклики для управління ресурсами корпоративних мереж. Існуючі рішення часто не враховують специфічні вимоги корпоративного середовища, такі як високий рівень безпеки, масштабованість та зручність використання для кінцевих користувачів. Різні IoT пристрої можуть використовувати різні протоколи зв'язку та стандарти, що ускладнює їх інтеграцію в єдину систему управління. Крім того, велика кількість пристроїв створює додаткове навантаження на інфраструктуру мережі та збільшує ризик несанкціонованого доступу та кібератак.

Інтеграція IoT пристроїв з платформами, такими як Tuya, надає потужні інструменти для управління ресурсами IoT та створення масштабованих рішень для корпоративних мереж. Використання таких платформ дозволяє ефективно управляти ресурсами, забезпечувати безпеку даних та надавати зручний інтерфейс для кінцевих користувачів. Таким чином, розробка веб-інтерфейсу для управління ресурсами Інтернету речей в корпоративних мережах є актуальним та своєчасним завданням, що вимагає глибокого дослідження та інноваційних підходів.

Однією з ключових проблем є відсутність єдиного зручного інтерфейсу для управління різноманітними IoT пристроями в корпоративних мережах. Існуючі рішення часто не враховують специфічні вимоги корпоративного середовища, що призводить до наступних труднощів:

У зв'язку зі зростаючою кількістю IoT пристроїв, забезпечення безпеки даних стає критично важливим завданням. IoT пристрої часто стають мішенями для кібератак, що може призвести до несанкціонованого доступу до конфіденційної інформації та порушення роботи систем. Зокрема, злом IoT пристроїв може дозволити зловмисникам отримати доступ до внутрішньої мережі компанії, що відкриває можливості для подальших атак.

Дослідження показують, що багато IoT пристроїв мають слабкі місця у безпеці, такі як недостатній захист паролів, відсутність шифрування даних та недостатній контроль доступу. Це підвищує ризик кібератак і вимагає впровадження додаткових заходів для забезпечення безпеки даних.

Система управління IoT пристроями повинна бути здатна підтримувати велику кількість підключених пристроїв та обробляти великі обсяги даних у режимі реального часу. Це вимагає використання масштабованої інфраструктури та ефективних алгоритмів обробки даних. Зростання кількості пристроїв призводить до збільшення обсягу даних, які потрібно обробляти та зберігати. Традиційні методи обробки даних можуть виявитися недостатніми для забезпечення необхідної продуктивності та швидкості реакції системи.

Різні IoT пристрої можуть використовувати різні протоколи зв'язку та стандарти, що ускладнює їх інтеграцію в єдину систему управління. Наприклад,

одні пристрої можуть використовувати протокол MQTT, інші – HTTP, а ще інші – CoAP. Це створює додаткові виклики для розробників, які повинні забезпечити сумісність між різними пристроями та забезпечити їхню взаємодію.

Крім того, різні пристрої можуть мати різні можливості та обмеження, що також ускладнює їх інтеграцію. Наприклад, деякі пристрої можуть підтримувати лише певні типи даних або мати обмежену пропускну здатність. Це вимагає розробки гнучкої архітектури, яка зможе адаптуватися до різних вимог та обмежень.

Інтерфейс системи управління IoT пристроями повинен бути інтуїтивно зрозумілим та зручним для користувачів різного рівня технічної підготовки. Складний або незрозумілий інтерфейс може призвести до помилок в управлінні та зниження ефективності роботи системи. Наприклад, якщо інтерфейс складний у використанні, користувачі можуть робити помилки при налаштуванні пристроїв або управлінні їхніми параметрами.

Зручність використання також включає можливість швидкого та легкого підключення нових пристроїв, моніторингу їхнього стану та виконання необхідних дій у разі виникнення проблем. Це вимагає розробки зрозумілого та доступного інтерфейсу, який дозволить користувачам легко виконувати всі необхідні операції.

Відсутність стандартизованих рішень.

На сьогоднішній день існує багато різних рішень для управління IoT пристроями, але більшість з них є власними розробками окремих компаній і не мають стандартизованих інтерфейсів. Це ускладнює інтеграцію різних пристроїв та систем, оскільки кожне рішення може мати свої унікальні особливості та вимоги.

Відсутність стандартизованих рішень також призводить до фрагментації ринку, що створює додаткові проблеми для користувачів, які змушені використовувати кілька різних платформ для управління своїми IoT пристроями. Це підвищує складність управління та збільшує витрати на підтримку та обслуговування системи.

Для вирішення вищезазначених проблем необхідні інноваційні підходи до розробки систем управління IoT пристроями. Це включає використання сучасних технологій, таких як хмарні обчислення, машинне навчання та великі дані, для забезпечення масштабованості, безпеки та зручності використання системи. Інноваційні підходи також повинні включати розробку стандартів та протоколів, які забезпечать сумісність між різними пристроями та системами.

У зв'язку з цими викликами, метою даної роботи є розробка веб-інтерфейсу, який дозволить ефективно управляти ресурсами Інтернету речей в корпоративних мережах, інтегруючись з платформою Tuya. Це дозволить забезпечити високий рівень безпеки, масштабованість та зручність використання для кінцевих користувачів, а також підтримку різних протоколів зв'язку та стандартів [1-5].

Таким чином, розробка веб-інтерфейсу для управління ресурсами Інтернету речей в корпоративних мережах є актуальним та своєчасним завданням, що вимагає глибокого дослідження та інноваційних підходів.

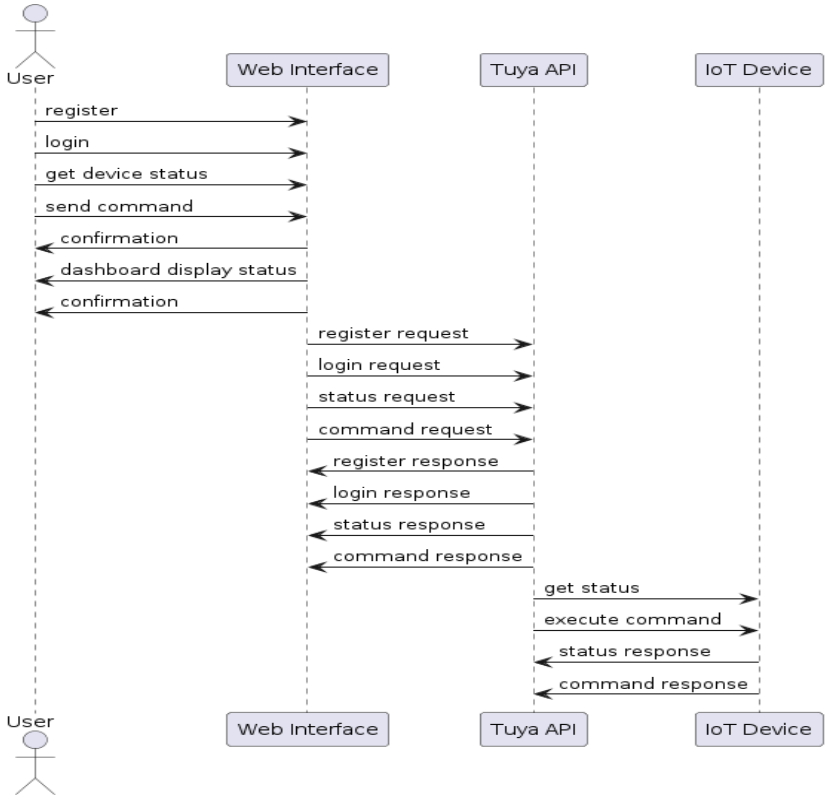

Рисунок 1.1 - Діаграма послідовності

1.2 Системи управління IoT

На ринку існує безліч платформ для управління IoT пристроями, серед яких найбільш популярні є AWS IoT, Google Cloud IoT, Microsoft Azure IoT та Tuya Smart. Кожна з цих платформ має свої унікальні особливості та можливості, що дозволяють користувачам підключати, налаштовувати та керувати IoT пристроями [2-10].

- AWS IoT пропонує широкий спектр інструментів для підключення пристроїв, збору та аналізу даних, інтеграції з іншими сервісами AWS та забезпечення високого рівня безпеки.

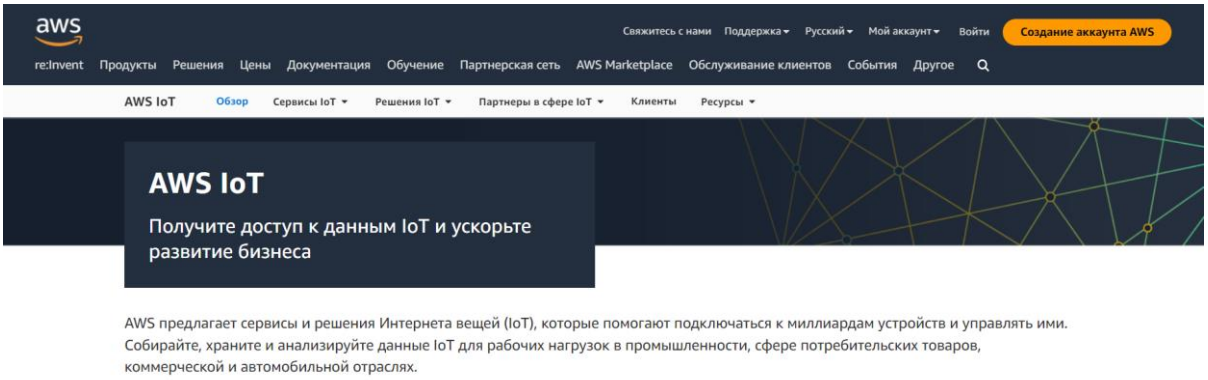

Рисунок 1.2 - Архітектура AWS IoT

- Google Cloud IoT Фокусується на обробці даних у режимі реального часу та інтеграції з потужними аналітичними інструментами, такими як BigQuery та Data Studio.

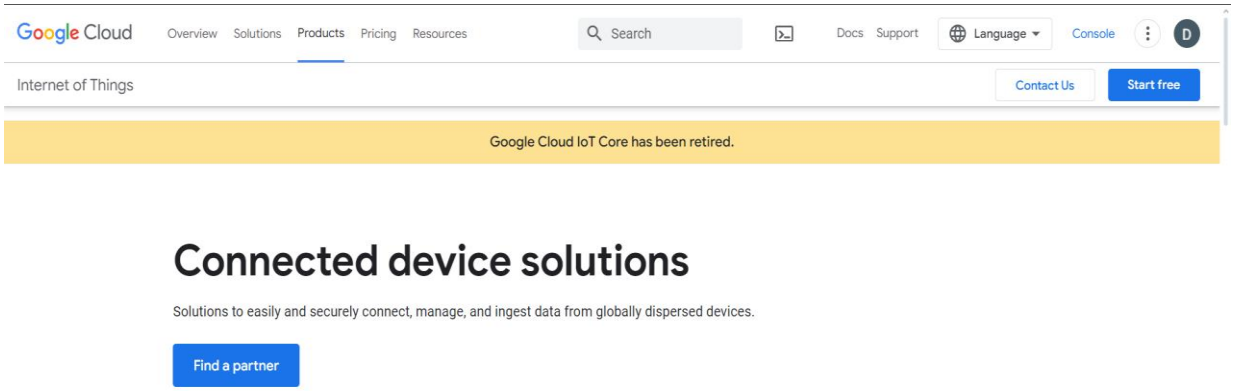

Рисунок 1.3 - Архітектура Google Cloud IoT

– Microsoft Azure IoT Надає можливості для створення, управління та моніторингу IoT рішень з використанням різних протоколів зв'язку та стандартів, що робить її гнучкою та масштабованою.

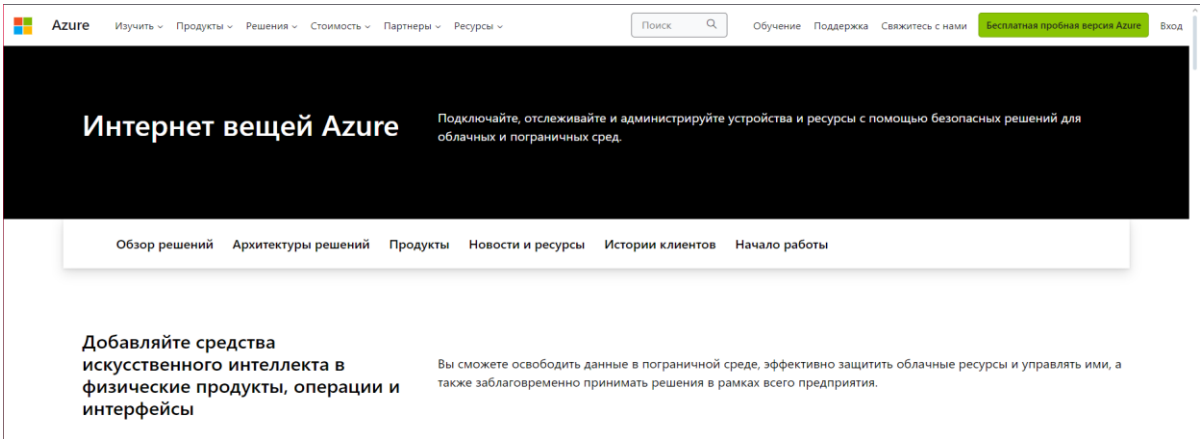

Рисунок 1.4 - Архітектура Microsoft Azure IoT

– Tuya Smart: Виділяється своєю здатністю швидко інтегруватися з різними типами пристроїв і пропонує зручний інтерфейс для кінцевих користувачів.

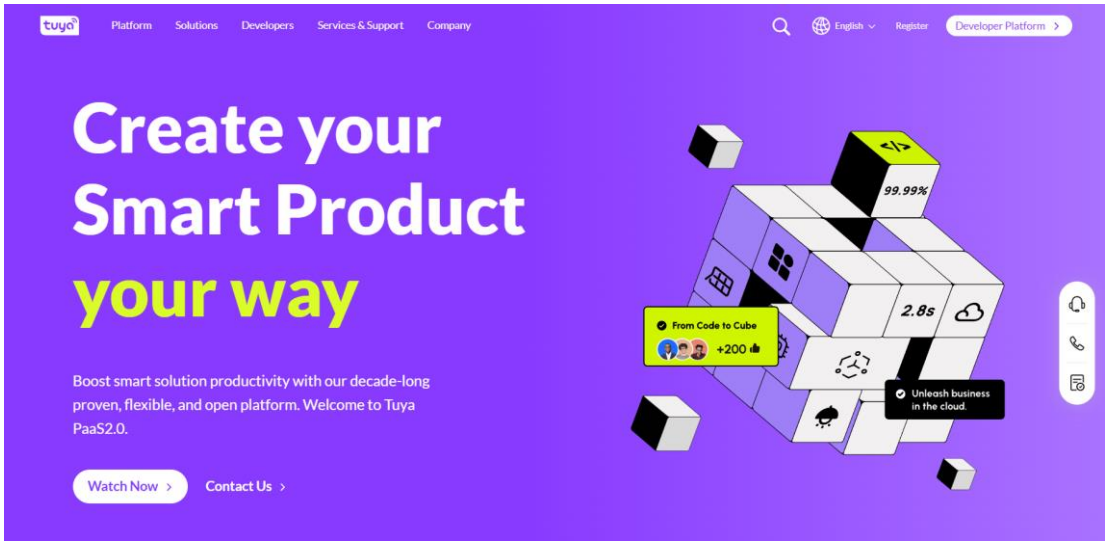

Рисунок 1.5 - Інтерфейс Tuya Smart

Система повинна бути здатна обробляти великі обсяги даних у режимі реального часу та забезпечувати стабільну роботу навіть при значних навантаженнях.

Інтеграція з різними протоколами зв'язку та стандартами є необхідною для забезпечення сумісності з широким спектром IoT пристроїв. Це включає підтримку таких протоколів, як MQTT, HTTP, CoAP та інших, а також можливість легкої адаптації до нових технологій.

Таким чином, аналіз існуючих рішень та вимог до веб-інтерфейсів для управління ресурсами Інтернету речей показує необхідність створення гнучких, безпечних та масштабованих систем, що забезпечують зручний доступ до функцій управління IoT пристроями. Це дозволяє підприємствам ефективно інтегрувати IoT технології в свої корпоративні мережі, підвищуючи продуктивність та оптимізуючи бізнес-процеси.

1.3 Порівняння різних платформ та технологій

У цьому розділі проводиться порівняння основних платформ та технологій для управління ресурсами Інтернету речей (IoT). Розглядаються такі платформи, як AWS IoT, Google Cloud IoT, Microsoft Azure IoT та Tuya Smart.

Порівняння базується на ключових критеріях, таких як функціональні можливості, безпека, масштабованість, зручність використання та підтримка різних протоколів зв'язку.

AWS IoT є однією з провідних платформ для управління IoT пристроями, пропонуючи широкий спектр інструментів для підключення, збору та аналізу даних, а також інтеграції з іншими сервісами AWS:

– AWS IoT дозволяє підключати мільйони пристроїв, збирати дані в режимі реального часу, аналізувати їх за допомогою сервісів, таких як AWS Lambda, Amazon S3 та Amazon DynamoDB.

– Платформа забезпечує високий рівень безпеки, включаючи шифрування даних, управління доступом та автентифікацію пристроїв.

– Масштабованість AWS IoT підтримує управління великою кількістю пристроїв та обробку великих обсягів даних.

– Зручність використання Інтерфейс AWS IoT досить зручний, хоча для новачків може здатися складним через велику кількість функцій.

– Підтримка протоколів: Платформа підтримує протоколи MQTT, HTTP, WebSockets.

Google Cloud IoT надає потужну інфраструктуру для управління IoT пристроями, акцентуючи увагу на обробці даних у режимі реального часу та інтеграції з аналітичними інструментами:

– Функціональні можливості. Платформа дозволяє підключати пристрої, збирати та аналізувати дані за допомогою таких інструментів, як Google BigQuery та Data Studio.

– Безпека. Google Cloud IoT забезпечує високий рівень безпеки з використанням шифрування даних та механізмів автентифікації.

– Масштабованість. Платформа добре масштабується, підтримуючи велику кількість підключених пристроїв.

– Зручність використання. Інтерфейс Google Cloud IoT є інтуїтивно зрозумілим і зручним для користувачів.

– Платформа підтримує протоколи MQTT та HTTP.

Microsoft Azure IoT пропонує комплексні рішення для управління IoT пристроями, забезпечуючи гнучкість та масштабованість:

- Функціональні можливості: Azure IoT дозволяє підключати та управляти пристроями, аналізувати дані за допомогою Azure Machine Learning та інших сервісів.
- Безпека: Платформа забезпечує високий рівень безпеки, включаючи шифрування даних та управління доступом.
- Масштабованість: Azure IoT підтримує управління великою кількістю пристроїв та обробку великих обсягів даних.
- Зручність використання: Інтерфейс Azure IoT є зручним, хоча для нових користувачів може бути необхідним деякий час для освоєння.
- Підтримка протоколів: Платформа підтримує протоколи MQTT, HTTPS та AMQP.

Tuya Smart є платформою, яка дозволяє швидко інтегрувати різні IoT пристрої та пропонує зручний інтерфейс для управління:

- Функціональні можливості: Tuya Smart дозволяє підключати та управляти пристроями, використовуючи зручний мобільний додаток та веб-інтерфейс.
- Безпека: Платформа забезпечує належний рівень безпеки, включаючи шифрування даних.
- Масштабованість: Tuya Smart підходить для невеликих та середніх проектів, забезпечуючи належну масштабованість.
- Зручність використання: Інтерфейс Tuya Smart є інтуїтивно зрозумілим та зручним для кінцевих користувачів.
- Підтримка протоколів: Платформа підтримує Wi-Fi, Zigbee та Bluetooth.

Для зручності порівняння основних характеристик платформ, наведемо їх у таблиці:

Характеристика	AWS IoT	Google Cloud IoT	Microsoft Azure IoT	Tuya Smart
Функціональні можливості	Високі	Високі	Високі	Середні
Безпека	Висока	Висока	Висока	Середня
Масштабованість	Висока	Висока	Висока	Середня
Зручність використання	Середня	Висока	Середня	Висока
Підтримка протоколів	MQTT, HTTP, WebSockets	MQTT, HTTP	MQTT, HTTPS, AMQP	Wi-Fi, Zigbee, Bluetooth

Рисунок 1.6 - Порівняння основних характеристик платформ для управління IoT

Таким чином, кожна з розглянутих платформ має свої переваги та недоліки, які визначають їх придатність для конкретних завдань. AWS IoT, Google Cloud IoT та Microsoft Azure IoT є потужними рішеннями для великих корпоративних мереж, тоді як Tuya Smart підходить для менших проектів, що потребують швидкої інтеграції та зручного інтерфейсу.

1.4 Постановка задач кваліфікаційної роботи

Метою даної роботи є розробка веб-інтерфейсу для управління ресурсами Інтернету речей (IoT) в корпоративних мережах, інтегрованого з платформою Tuya. Це дозволить забезпечити ефективне управління IoT пристроями, високий рівень безпеки, масштабованість та зручність використання для кінцевих користувачів. Основною метою є створення гнучкої, надійної та зручної системи, яка відповідатиме потребам сучасних корпоративних середовищ та враховуватиме специфічні вимоги до управління великою кількістю різнорідних IoT пристроїв.

Для досягнення поставленої мети необхідно вирішити наступні завдання:

1. Аналіз існуючих рішень для управління ресурсами IoT в корпоративних мережах:

- Вивчення наявних платформ та інструментів для управління IoT пристроями.
- Порівняння їхніх можливостей, переваг та недоліків.
- Виявлення недоліків існуючих рішень та вимог до нових систем.

2. Вивчення основних вимог до веб-інтерфейсів для управління ресурсами IoT:

- Визначення вимог користувачів до функціональності та зручності веб-інтерфейсу.
- Вивчення вимог до безпеки даних та управління доступом.
- Аналіз вимог до масштабованості та підтримки різних протоколів зв'язку.

3. Проектування архітектури веб-інтерфейсу:

- Розробка концептуальної моделі системи.
- Визначення ключових компонентів та їх взаємодії.
- Вибір технологій та інструментів для реалізації веб-інтерфейсу.

4. Розробка прототипу веб-інтерфейсу:

- Реалізація основних модулів системи.
- Інтеграція з платформою Tuya.
- Забезпечення необхідного рівня безпеки та масштабованості.

5. Тестування та оптимізація розробленого рішення:

- Проведення функціонального та нефункціонального тестування системи.
- Виявлення та усунення можливих помилок та недоліків.
- Оптимізація продуктивності та масштабованості системи.

6. Розгортання та демонстрація роботи веб-інтерфейсу:

- Розробка документації для користувачів та адміністраторів системи.
- Проведення демонстрацій та навчання користувачів.
- Впровадження системи в корпоративне середовище та збір зворотного зв'язку.

Очікуваними результатами даного дослідження є створення веб-інтерфейсу, який дозволить ефективно управляти ресурсами IoT в корпоративних мережах. Цей інтерфейс повинен забезпечувати високу зручність використання, масштабованість, безпеку та підтримку різних протоколів зв'язку. Реалізована система повинна бути здатна інтегруватися з платформою Tuya та забезпечувати централізоване управління різнорідними IoT пристроями.

Таким чином, досягнення мети та виконання завдань даного дослідження дозволить створити ефективний інструмент для управління IoT пристроями в корпоративних мережах, що сприятиме підвищенню ефективності бізнес-процесів, зниженню витрат та покращенню якості послуг.

Об'єктом даного дослідження є корпоративні мережі, до яких підключені пристрої Інтернету речей (IoT). Корпоративні мережі представляють собою складні інфраструктури, що включають сервери, маршрутизатори, комутатори, робочі станції, мобільні пристрої та численні IoT пристрої. Основною характеристикою корпоративних мереж є їх висока складність та велика кількість різнорідних підключених пристроїв, що створює значні виклики для управління, моніторингу та забезпечення безпеки.

Предметом дослідження є методи та засоби управління ресурсами Інтернету речей в корпоративних мережах через веб-інтерфейс.

Для досягнення поставленої мети та вирішення завдань дослідження використовуватимуться наступні методи:

Першим етапом дослідження є проведення аналізу літератури та існуючих рішень в області управління IoT пристроями в корпоративних мережах. Це включає вивчення наукових статей, технічної документації, звітів та інших джерел, які описують сучасні підходи та технології в даній сфері. Аналіз дозволить визначити поточний стан досліджень, виявити ключові проблеми та тенденції, а також визначити недоліки існуючих рішень, що потребують удосконалення.

На основі аналізу літератури буде розроблено концептуальну модель архітектури веб-інтерфейсу для управління IoT пристроями. Проектування

включатиме визначення основних компонентів системи, їх функцій та взаємодії. Для моделювання архітектури будуть використовуватися діаграми UML, такі як діаграми класів, діаграми послідовності та діаграми компонентів. Це дозволить створити чітке уявлення про структуру системи та забезпечити її логічну цілісність.

На етапі розробки буде використано сучасні веб-технології, такі як HTML, CSS, JavaScript, а також фреймворки Angular або React. Ці технології забезпечують створення швидкого, масштабованого та зручного веб-інтерфейсу. Розробка включатиме реалізацію основних модулів системи, таких як модулі підключення та конфігурації IoT пристроїв, модулі моніторингу та управління, а також модулі безпеки.

Особливу увагу буде приділено інтеграції розробленого веб-інтерфейсу з платформою Tuya. Це включає використання API Tuya для підключення, управління та моніторингу IoT пристроїв. Інтеграція дозволить забезпечити централізоване управління різнорідними пристроями та використання потужних інструментів Tuya для аналізу даних та оптимізації процесів.

Після розробки прототипу системи буде проведено функціональне та нефункціональне тестування. Функціональне тестування включає перевірку коректності роботи всіх компонентів системи та їх відповідності вимогам. Нефункціональне тестування включає перевірку продуктивності, масштабованості та безпеки системи. Для тестування будуть використовуватися як автоматизовані інструменти, так і ручні методи. Результати тестування дозволять виявити та усунути можливі помилки та недоліки, а також оптимізувати систему для роботи з великою кількістю пристроїв.

На завершальному етапі буде проведено розгортання системи в тестовому середовищі та збір зворотного зв'язку від користувачів. Це дозволить оцінити зручність використання, ефективність та надійність системи в реальних умовах. Зворотний зв'язок буде використано для подальшого вдосконалення системи та підготовки її до впровадження в корпоративне середовище.

2. ПРОЄКТУВАННЯ ВЕБ-ІНТЕРФЕЙСІВ ДЛЯ УПРАВЛІННЯ РЕСУРСАМИ ІНТЕРНЕТУ РЕЧЕЙ

2.1 Вимоги до веб-інтерфейсів для управління IoT

Платформа Tuya була обрана для цього дослідження через її унікальні можливості, які забезпечують простоту підключення до будь-яких IoT пристроїв і легкість отримання ключів для передачі даних з датчиків прямо на сайт. Це робить Tuya надзвичайно привабливою для інтеграції в корпоративні мережі, де важливо забезпечити швидку та безпечну комунікацію між різнорідними пристроями.

Однією з ключових вимог до веб-інтерфейсів для управління IoT є зручність та інтуїтивність. Інтерфейс повинен бути простим у використанні навіть для користувачів з мінімальним технічним досвідом. Це включає логічну організацію меню, зрозумілу навігацію та можливість швидкого доступу до основних функцій. Tuya пропонує користувачам інтуїтивно зрозумілий інтерфейс, що дозволяє легко налаштовувати пристрої та керувати ними.

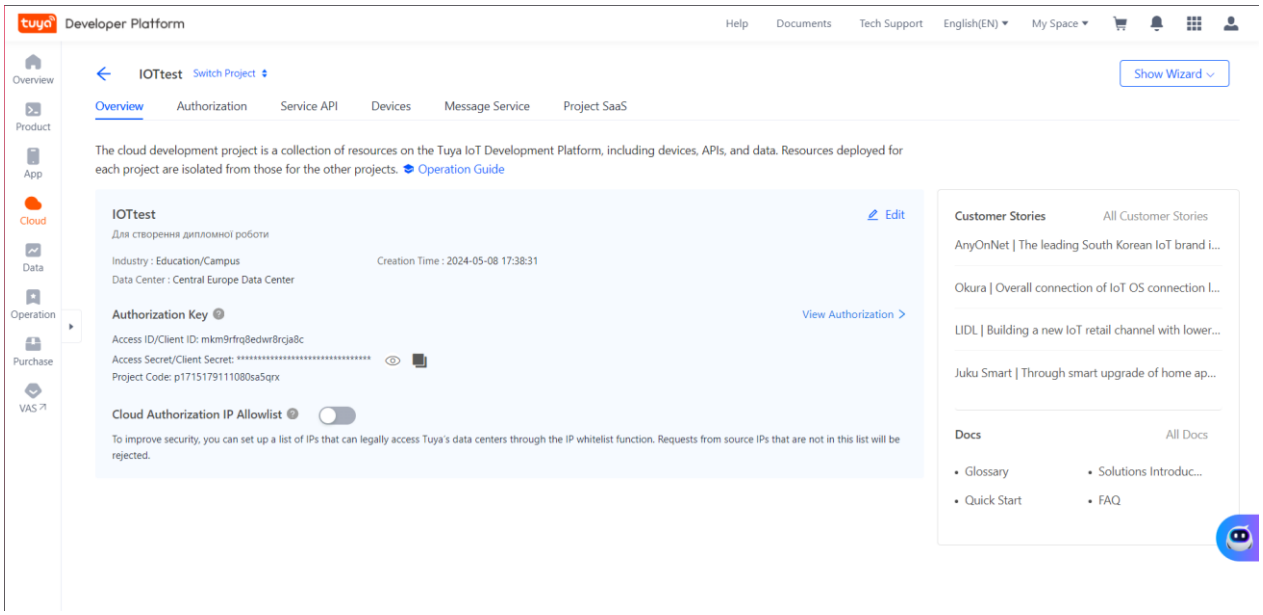

Рисунок 2.1 - Інтерфейс управління пристроями на платформі Tuya

Наступною важливою вимогою є масштабованість. Веб-інтерфейс повинен забезпечувати стабільну роботу при збільшенні кількості підключених пристроїв та обробці великих обсягів даних у реальному часі. Tuya забезпечує підтримку великої кількості пристроїв завдяки своїй хмарній інфраструктурі, яка автоматично масштабується в залежності від навантаження.

Безпека є критично важливим аспектом для будь-якого IoT рішення, особливо в корпоративних мережах, де зберігаються чутливі дані. Tuya надає потужні інструменти для забезпечення безпеки, включаючи шифрування даних, автентифікацію пристроїв та контроль доступу. Це дозволяє мінімізувати ризики кібератак і несанкціонованого доступу до системи.

Ще однією важливою вимогою є інтероперабельність, що означає здатність веб-інтерфейсу працювати з різними типами пристроїв та протоколами зв'язку. Tuya підтримує широкий спектр протоколів, таких як Wi-Fi, Zigbee та Bluetooth, що забезпечує гнучкість в інтеграції різнорідних пристроїв у єдину систему. Це значно спрощує процес налаштування та управління пристроями.

Функціональність веб-інтерфейсу також відіграє важливу роль. Інтерфейс повинен надавати користувачам широкий спектр інструментів для моніторингу та управління IoT пристроями, включаючи перегляд статусу пристроїв, налаштування параметрів, отримання сповіщень про події та аналіз даних. Tuya дозволяє користувачам здійснювати ці дії через зручний веб-інтерфейс, що підвищує ефективність управління системою.

На завершення, важливо зазначити, що веб-інтерфейс повинен бути адаптивним та забезпечувати доступ з різних пристроїв, таких як комп'ютери, планшети та смартфони. Це дозволяє користувачам здійснювати управління IoT пристроями незалежно від їхнього місцезнаходження. Tuya забезпечує адаптивний дизайн інтерфейсу, що робить його зручним для використання на будь-якому пристрої.

Таким чином, платформа Tuya задовольняє всі основні вимоги до веб-інтерфейсів для управління IoT, що робить її ідеальним вибором для інтеграції в корпоративні мережі. Завдяки своїй гнучкості, безпеці та зручності

використання, Tuya забезпечує ефективне управління різнорідними IoT пристроями, підвищуючи продуктивність та оптимізуючи бізнес-процеси.

2.2 Вибір датчиків системи

Інтернет речей (IoT) значною мірою залежить від різних типів датчиків, які забезпечують збір даних про навколишнє середовище та пристрої. Однією з основних категорій датчиків, що використовуються в IoT системах, є датчики вологості та температури. Ці датчики знаходять широке застосування в розумних будинках, промислових системах, сільському господарстві та інших галузях.

Датчики вологості вимірюють кількість водяної пари в повітрі або іншому середовищі. Серед існуючих типів датчиків вологості виділяють ємнісні, резистивні та термоелектричні датчики. Ємнісні датчики вологості працюють на основі вимірювання змін ємності конденсатора залежно від вологості. Вони відомі своєю високою точністю, стабільністю та швидким часом відгуку, що робить їх популярними в розумних будинках та промислових системах. Наприклад, датчик вологості DHT22 є одним з найбільш популярних ємнісних датчиків завдяки своїй надійності та точності.

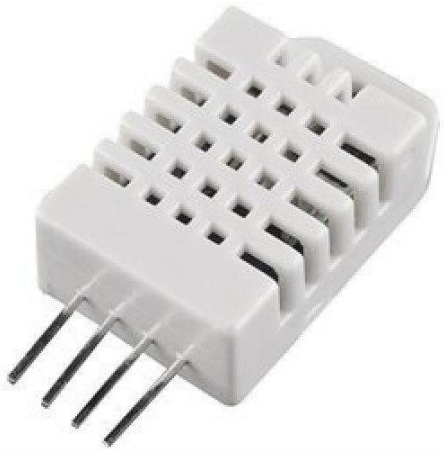

Рисунок 2.2 – Ємнісний датчик вологості DHT22

Резистивні датчики вологості вимірюють зміни електричного опору матеріалу залежно від вологості. Вони відзначаються простотою конструкції та низькою вартістю, що робить їх привабливими для використання в побутових приладах та системах моніторингу навколишнього середовища. Одним з прикладів резистивних датчиків вологості є датчик HR202, який широко застосовується в побутових приладах.

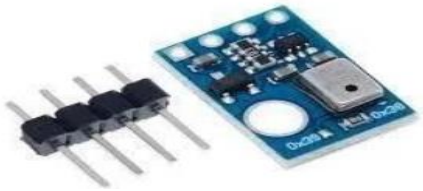

Рисунок 2.3 – Резистивний датчик вологості HR202

Термоелектричні датчики вологості використовують два різних метали для створення термоелектричної пари, яка реагує на зміни вологості. Ці датчики зазвичай застосовуються в спеціалізованих галузях, таких як метеорологія та лабораторні дослідження. Вони забезпечують високу точність вимірювання та використовуються в умовах, де необхідний високий рівень точності.

Датчики температури вимірюють тепловий стан об’єкта або середовища. Серед основних типів датчиків температури виділяють терморезистори (RTD), термістори, термопари та інфрачервоні датчики. Терморезистори (RTD) працюють на основі вимірювання змін електричного опору металу (зазвичай платини) залежно від температури. Вони відомі своєю високою точністю та стабільністю, що робить їх придатними для використання в промислових процесах та наукових дослідженнях. Одним з популярних терморезисторів є PT100.

Рисунок 2.4 – Терморезистор PT100

Термістори вимірюють зміни електричного опору напівпровідникового матеріалу залежно від температури. Вони відзначаються високою чутливістю та обмеженим діапазоном температур, що робить їх придатними для використання в побутових приладах та медичному обладнанні. Одним з прикладів термісторів є NTC термістор, який широко застосовується у побутових пристроях.

Для нашого дослідження були обрані два різних типи датчиків вологості та температури. Перший датчик потребував шлюзу для підключення до мережі, тоді як другий датчик міг підключатися безпосередньо через Wi-Fi і комунікувати з платформою Tuya.

Датчик 1: Датчик вологості і температури з шлюзом

Перший датчик, який був використаний, потребував додаткового шлюзу для підключення до мережі. Цей тип датчика зазвичай використовує протокол зв'язку, такий як Zigbee або Z-Wave, що вимагає наявності шлюзу для інтеграції з мережею Wi-Fi та подальшої комунікації з хмарними сервісами.

Назва датчика: Xiaomi Mi Temperature & Humidity Monitor

Особливості:

- Потребує шлюзу для підключення
- Висока точність вимірювання
- Довговічність та надійність

Рисунок 2.5 - Датчик вологості і температури з шлюзом

Другий датчик був обраний через його здатність підключатися безпосередньо до мережі Wi-Fi, що значно спрощує процес налаштування та інтеграції. Цей датчик комунікує безпосередньо з платформою Tuya, дозволяючи легко отримувати дані та керувати пристроями через веб-інтерфейс або мобільний додаток. Назва датчика: T & H Sensor

Особливості:

- Пряме підключення до Wi-Fi
- Інтеграція з платформою Tuya
- Легке налаштування та використання

Рисунок 2.6 - Датчик вологості і температури з прямим підключенням до Wi-Fi

Обидва типи датчиків мають свої переваги та недоліки. Датчики, які потребують шлюзу, зазвичай забезпечують більшу стабільність та надійність з'єднання, але їх налаштування може бути складнішим і вимагати додаткових

витрат на обладнання. З іншого боку, датчики з прямим підключенням до Wi-Fi пропонують більшу зручність у використанні та налаштуванні, але можуть бути менш надійними у випадку проблем з Wi-Fi мережею.

Таким чином, вибір конкретного типу датчика залежить від вимог до стабільності з'єднання, легкості налаштування та інтеграції, а також від бюджету проекту. Використання датчиків, що безпосередньо підключаються до Wi-Fi, як у випадку з Tuya, дозволяє значно спростити процес інтеграції IoT пристроїв та забезпечує зручне управління ними через єдиний інтерфейс.

2.3 Розробка веб-інтерфейсу

Розробка веб-інтерфейсу почалася з створення основної структури сайту, що включає наступні модулі:

- Головна сторінка містить загальну інформацію про систему, навігаційне меню та кнопки для переходу до основних розділів.
- Модуль моніторингу якості повітря забезпечує збір та відображення даних про якість повітря з підключених датчиків.
- Модуль управління ресурсами дозволяє користувачам керувати підключеними IoT пристроями, налаштовувати параметри та отримувати сповіщення про стан пристроїв.

Для розробки інтерфейсу були використані наступні технології:

- HTML та CSS: Створення структури та стилізації веб-сторінок.
- JavaScript: Реалізація динамічної взаємодії з користувачем, включаючи обробку подій та взаємодію з сервером через AJAX.
- Python: Серверна логіка та обробка даних з датчиків.
- JSON: Формат передачі даних між клієнтом та сервером.
- PHP: Обробка запитів на сервері та взаємодія з базою даних.
- SQL: Зберігання та обробка даних у базі даних.

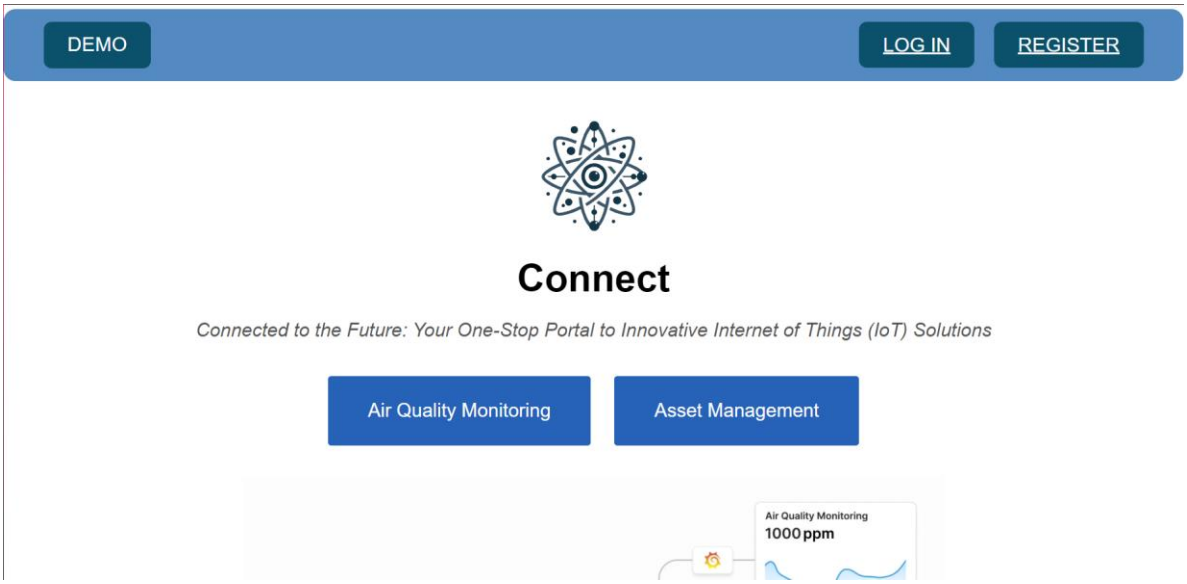

Рисунок 2.7 - Сайт на локальному сервері XAMPP

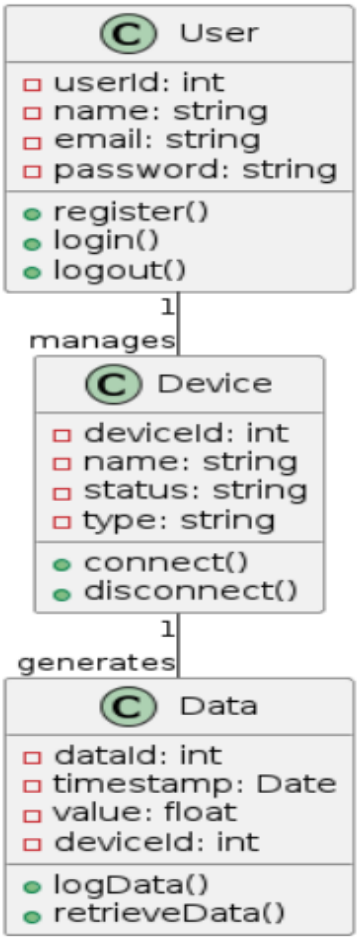

Рисунок 2.8 - Діаграма класів

Система побудована за клієнт-серверною архітектурою. Клієнтська частина (фронтенд) реалізована з використанням HTML, CSS та JavaScript, що

забезпечує зручний та інтуїтивно зрозумілий інтерфейс для користувачів. Серверна частина (бекенд) реалізована на Python та PHP, що обробляє запити від клієнта, взаємодіє з базою даних SQL та керує підключеними IoT пристроями.

Веб-інтерфейс забезпечує наступну функціональність:

– Реєстрація та аутентифікація користувачів Забезпечує безпечний доступ до системи.

– Моніторинг даних з датчиків Відображення даних про температуру, вологість та якість повітря в реальному часі.

– Управління пристроями Налаштування параметрів пристроїв, віддалене керування та отримання сповіщень про стан пристроїв.

Проект складається з кількох основних файлів і директорій, які забезпечують функціональність веб-інтерфейсу для управління ресурсами Інтернету речей (IoT). Ось короткий опис основних файлів:

– index.php Головна сторінка веб-інтерфейсу, яка містить основний контент та навігаційне меню для користувачів. Ця сторінка забезпечує доступ до різних модулів системи, включаючи моніторинг якості повітря та управління ресурсами.

– registration: Директорія, що містить файли для реалізації функціональності реєстрації користувачів. Включає наступні файли:

– reg.html HTML-сторінка форми реєстрації.

– reg.js JavaScript-файл для обробки даних форми реєстрації та взаємодії з сервером.

– register.php PHP-скрипт для обробки даних форми реєстрації та запису їх у базу даних.

– login.php PHP-скрипт для обробки логіна користувачів.

– logout.php PHP-скрипт для виходу користувачів з системи.

– db.php PHP-скрипт для підключення до бази даних.

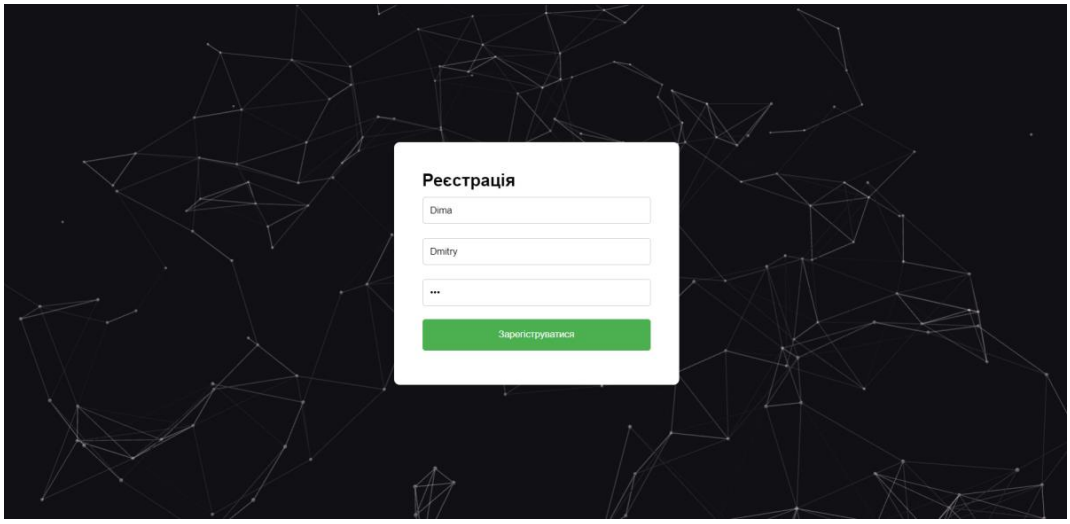

Рисунок 2.9 - Форма реєстрації користувачів

– styles.css Файл стилів CSS для стилізації веб-інтерфейсу.

– app.js JavaScript-файл для реалізації функціональності веб-інтерфейсу.

– tuya-api-node директорія, що містить файли для інтеграції з платформою Tuya через API.

Функціональність реєстрації користувачів реалізована через сторінку reg.html, яка містить форму для введення даних користувача, таких як ім'я, електронна пошта та пароль. Дані з форми обробляються JavaScript-файлом reg.js та передаються на сервер для обробки.

PHP-скрипт register.php приймає дані з форми та записує їх у базу даних, забезпечуючи створення нового користувача. Також обробляються запити на логін та логаут через відповідні скрипти login.php та logout.php.

Для зберігання даних користувачів використовується база даних SQL, яка містить таблицю для зберігання інформації про зареєстрованих користувачів. Таблиця включає поля для зберігання імені користувача, електронної пошти та зашифрованого пароля.

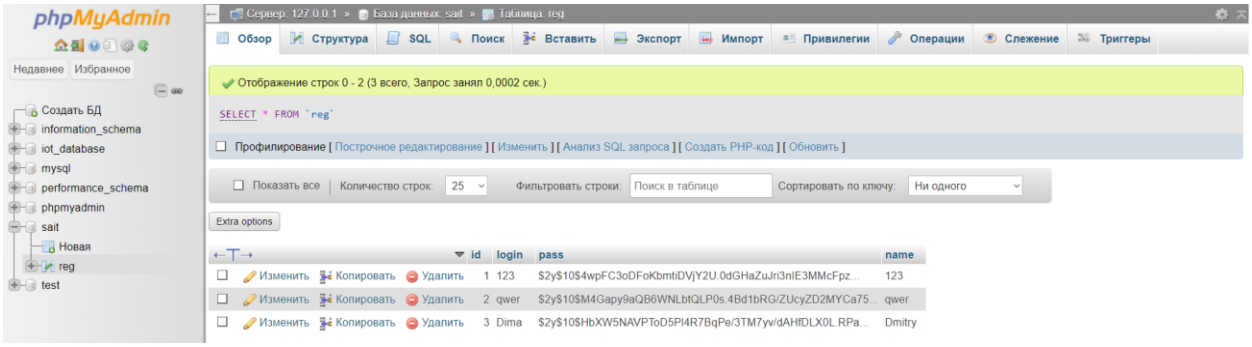

Рисунок 2.10 - Структура таблиці бази даних

Розробка веб-інтерфейсу включала також реалізацію безпечного зберігання паролів з використанням сучасних алгоритмів хешування, що забезпечує захист даних користувачів від несанкціонованого доступу.

На ілюстраціях нижче показані основні компоненти функціональності реєстрації та логіна: форма реєстрації, форма логіна та особистий кабінет користувача.

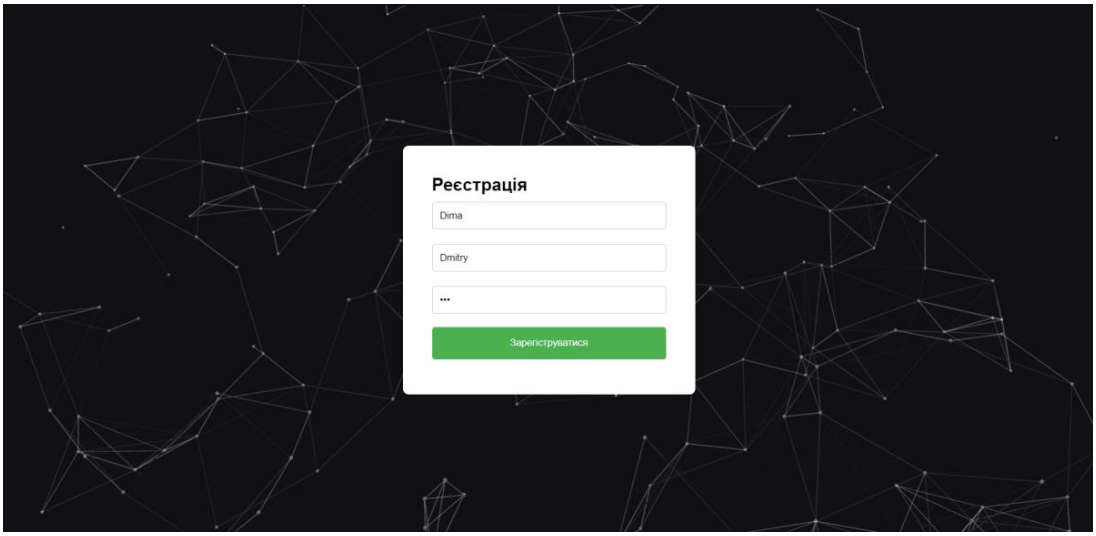

Рисунок 2.11 - Форма реєстрації користувачів

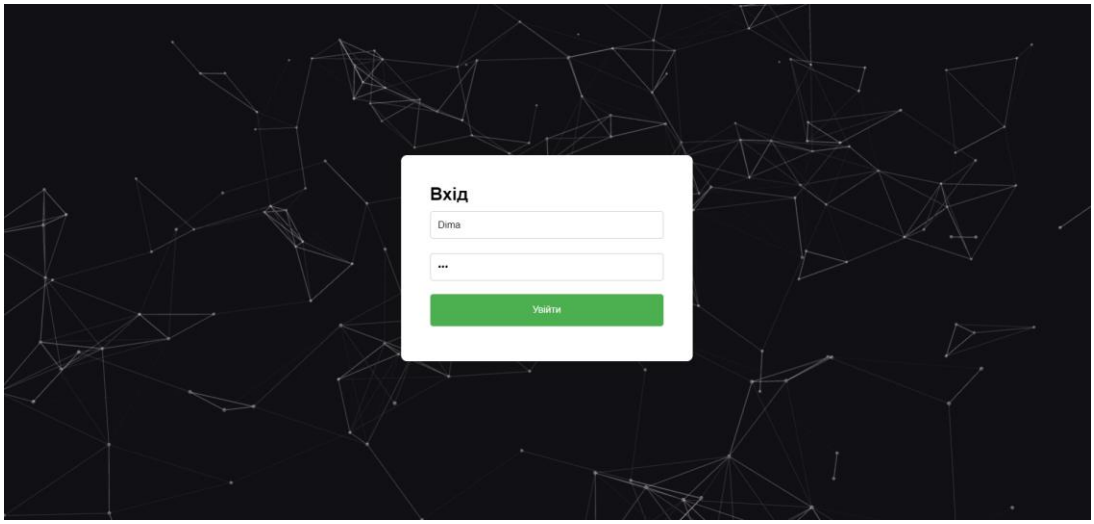

Рисунок 2.12 - Форма логіна користувачів

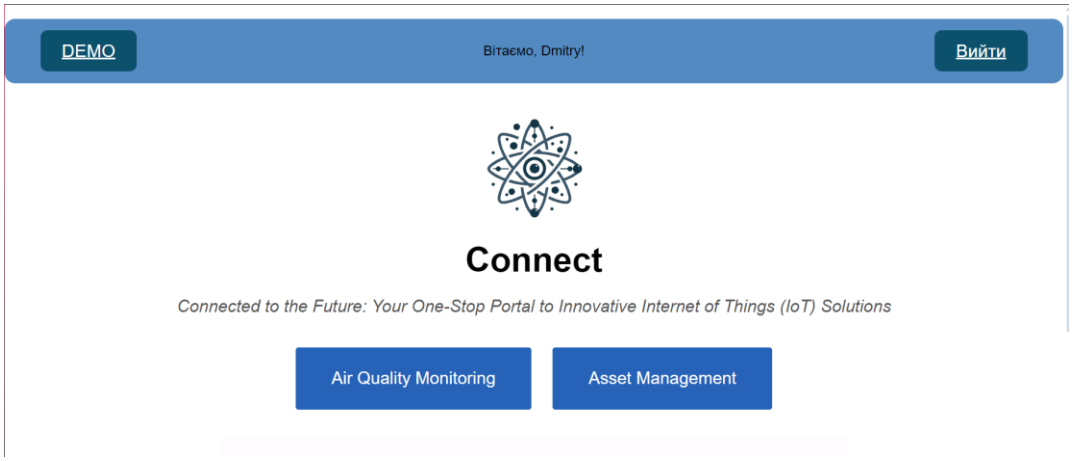

Рисунок 2.13 - Особистий кабінет користувача

Таким чином, описаний код забезпечує основні функції веб-інтерфейсу, включаючи реєстрацію користувачів, логін та логаут, а також інтеграцію з базою даних для зберігання даних користувачів. Інші модулі, такі як моніторинг якості повітря та управління ресурсами, будуть описані в наступних розділах.

3 РЕАЛІЗАЦІЯ КОМПОНЕНТІВ УПРАВЛІННЯ РЕСУРСАМИ IOT

3.1 Інтеграція з платформою Tuya

Для інтеграції з платформою Tuya використовувався API, який дозволяє взаємодіяти з різними IoT пристроями. У цьому розділі ми розглянемо процес підключення до платформи, отримання токена доступу, запиту даних з датчика та передачі цих даних на локальний сервер.

Першим кроком інтеграції з платформою Tuya є налаштування параметрів API, таких як ACCESS_ID, ACCESS_KEY та API_ENDPOINT. Ці параметри можна отримати на платформі Tuya після реєстрації вашого додатка.

Ось приклад налаштувань API:

```
ACCESS_ID = 'mkm9rfrq8edwr8rcja8c'
ACCESS_KEY = '1a6898ca13874149859c978fca9f5208'
API_ENDPOINT = 'https://openapi.tuyaeu.com'# URL для Європейського регіону
```

Для виконання запитів до API Tuya необхідно спочатку отримати токен доступу. Це здійснюється за допомогою спеціального запиту, для якого потрібно згенерувати підпис. Нижче наведено приклад коду для генерації підпису та отримання токена доступу:

```
def generate_signature(access_id, access_key, timestamp, nonce, http_method, url, access_token=None):
     content_sha256 = hashlib.sha256(b'').hexdigest()  # Порожнє тіло запиту
    string_to_sign = f"{http_method}\n{content_sha256}\n\n{url}"
    if access_token:
                                                    message = f"{access_id}{access_token}{timestamp}{nonce}{string_to_sign}"
    else:
        message = f"{access_id}{timestamp}{nonce}{string_to_sign}"
           signature = hmac.new(access_key.encode('utf-8'), msg=message.encode('utf-8'),
digestmod=hashlib.sha256).hexdigest().upper()
    return signature

# Отримання поточного часу
```

```
timestamp = str(int(time.time() * 1000))

# Генерація nonce (унікального ідентифікатора запиту)
nonce = str(int(time.time() * 1000))
```

Генерація підпису для отримання токена

```
http_method = "GET"
url_path = "/v1.0/token?grant_type=1"
signature = generate_signature(ACCESS_ID, ACCESS_KEY, timestamp,
nonce, http_method, url_path)

# Заголовки для запиту токена
headers = {
    'client_id': ACCESS_ID,
    'sign': signature,
    't': timestamp,
    'sign_method': 'HMAC-SHA256',
    'nonce': nonce
}
```

Виконання запиту для отримання токена

```
url = f"{API_ENDPOINT}{url_path}"
response = requests.get(url, headers=headers)
token_response = response.json()

if token_response['success']:
    access_token = token_response['result']['access_token']
    print("Токен доступу отримано:", access_token)
else:
    print("Не вдалося отримати токен доступу")
    print("Відповідь від API:", token_response)
    exit()
```

Після отримання токена доступу можна виконувати запити для отримання даних з датчиків. У прикладі нижче наведено код для отримання даних з датчика за допомогою Tuya API:

```
headers = {
    'client_id': ACCESS_ID,
    'sign': signature,
    't': timestamp,
    'sign_method': 'HMAC-SHA256',
    'nonce': nonce,
    'access_token': access_token
}
```

Виконання запиту для отримання даних з датчика

```
url = f"{API_ENDPOINT}{url_path}"
response = requests.get(url, headers=headers)
sensor_data_response = response.json()
```

```
# Виведення деталей запиту і підпису для діагностики
print("Заголовки для запиту даних з датчика:", headers)
print("URL:", url)
print("Підпис:", signature)

if sensor_data_response['success']:
    sensor_data = sensor_data_response['result']
    print("Дані з датчика:", sensor_data)
```

Відправка даних на локальний сервер

```
                                    site_url                    =
'http://localhost/Test1/IOTsystem/your_php_script.php'   #  Введіть
URL вашого локального PHP-скрипта
    response = requests.post(site_url, json=sensor_data)
    if response.status_code == 200:
        print("Дані успішно відправлені на сайт")
    else:
            print("Помилка  при  відправці  даних  на  сайт:",
response.status_code)
else:
        print("Помилка  при  отриманні  даних  з  датчика:",
sensor_data_response)
```

Після отримання даних з датчика вони передаються на локальний сервер для обробки та відображення у веб-інтерфейсі. Для цього використовується POST-запит до локального PHP-скрипта:

```
site_url  =  'http://localhost/Test1/IOTsystem/your_php_script.php'
# Введіть URL вашого локального PHP-скрипта
    response = requests.post(site_url, json=sensor_data)
    if response.status_code == 200:
        print("Дані успішно відправлені на сайт")
    else:
            print("Помилка  при  відправці  даних  на  сайт:",
response.status_code)
else:
        print("Помилка  при  отриманні  даних  з  датчика:",
sensor_data_response)
```

Таким чином, інтеграція з платформою Tuya дозволяє ефективно збирати дані з IoT пристроїв та передавати їх для обробки та відображення у

веб-інтерфейсі. Це забезпечує зручне та інтуїтивно зрозуміле управління пристроями, а також моніторинг їх стану в режимі реального часу.

3.2 Алгоритми передачі даних на сайт

Передача даних з датчиків на сайт є важливим етапом у реалізації IoT системи, яка включає кілька кроків: збір даних з датчиків, передача даних на сервер, збереження в базі даних, обробка та відображення на веб-інтерфейсі. У цьому розділі ми детально розглянемо кожен з цих кроків, надамо приклади коду та ілюстрації процесу.

Після успішної інтеграції з платформою Tuya, датчики починають збирати дані про навколишнє середовище, такі як температура, вологість та стан батареї. Ці дані передаються на локальний сервер для подальшої обробки.

Дані з датчиків передаються на сервер за допомогою POST-запиту. На сервері PHP-скрипт приймає ці дані і зберігає їх у базі даних. Нижче наведено приклад PHP-скрипта для прийому даних:

```
<?php
$host = 'localhost'; // або інший хост
$dbname = 'iot_database'; // ваше ім'я бази даних
$user = 'root'; // ваше ім'я користувача бази даних
$pass = ''; // ваш пароль бази даних
$dsn = "mysql:host=$host;dbname=$dbname;charset=utf8";

try {
    $pdo = new PDO($dsn, $user, $pass);
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    // Прийом даних з POST-запиту
    $data = json_decode(file_get_contents('php://input'), true);

    if ($data) {
        // Виведення даних для діагностики
        file_put_contents('php://stderr', print_r($data, TRUE));
        // Перетворення значення температури
        $temperature = $data[0]['value'] / 10;
        $humidity = $data[1]['value'];
        $battery_state = $data[2]['value'];
```

```
        $temp_unit = $data[3]['value'];
```

Обробка даних та збереження в базу даних

```
            $stmt = $pdo->prepare("INSERT INTO sensor_data
(temperature, humidity, battery_state, temp_unit) VALUES (?, ?, ?,
?)");
          $stmt->execute([$temperature, $humidity, $battery_state,
$temp_unit]);
        echo "Дані успішно збережено";
    } else {
        echo "Немає даних для збереження";
    }
} catch (PDOException $e) {
    die("Помилка підключення до бази даних: " . $e->getMessage());
}
?>
```

Для зберігання даних використовується база даних MySQL. Таблиця sensor_data містить поля для зберігання температури, вологості, стану батареї та одиниць вимірювання. Нижче наведено приклад структури таблиці:

```
CREATE TABLE sensor_data (
    id INT AUTO_INCREMENT PRIMARY KEY,
    temperature FLOAT NOT NULL,
    humidity FLOAT NOT NULL,
    battery_state INT NOT NULL,
    temp_unit VARCHAR(10) NOT NULL,
    timestamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

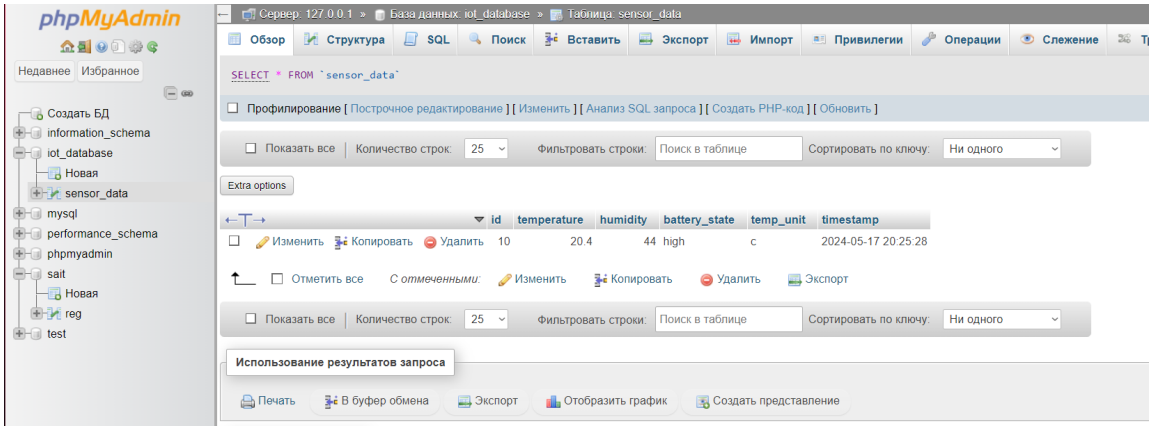

Рисунок 3.1 - Структура таблиці sensor_data у базі даних MySQL

Для відображення даних на веб-сторінці необхідно отримати їх з бази даних. Це здійснюється за допомогою PHP-скрипта, який виконує SELECT-запит і повертає останні збережені дані у форматі JSON:

```
<?php
$host = 'localhost'; // або інший хост
$dbname = 'iot_database'; // ваше ім'я бази даних
$user = 'root'; // ваше ім'я користувача бази даних
$pass = ''; // ваш пароль бази даних
$dsn = "mysql:host=$host;dbname=$dbname;charset=utf8";

try {
    $pdo = new PDO($dsn, $user, $pass);
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
```

Отримання останніх даних з таблиці sensor_data

```
        $stmt = $pdo->query("SELECT temperature, humidity, battery_state, temp_unit FROM sensor_data ORDER BY timestamp DESC LIMIT 1");
    $data = $stmt->fetch(PDO::FETCH_ASSOC);

    echo json_encode($data);
} catch (PDOException $e) {
    die("Помилка підключення до бази даних: " . $e->getMessage());
}
?>
```

Дані, отримані з бази даних, відображаються на веб-сторінці за допомогою JavaScript. Функція fetchSensorData виконує запит до PHP-скрипта і оновлює елементи веб-сторінки відповідно до отриманих даних:

```
function fetchSensorData() {
    fetch('http://localhost/Test1/IOTsystem/get_sensor_data.php')
        .then(response => response.json())
        .then(data => {

document.getElementById('displayTemperature').textContent = data.temperature + '°C';
            document.getElementById('displayHumidity').textContent = data.humidity + '%';
        })
           .catch(error => console.error('Error fetching data:', error));
}
```

Веб-інтерфейс забезпечує зручне відображення даних з датчиків, включаючи температуру, вологість, стан батареї та інші параметри. Користувачі можуть переглядати ці дані в реальному часі, що дозволяє їм оперативно реагувати на зміни умов.

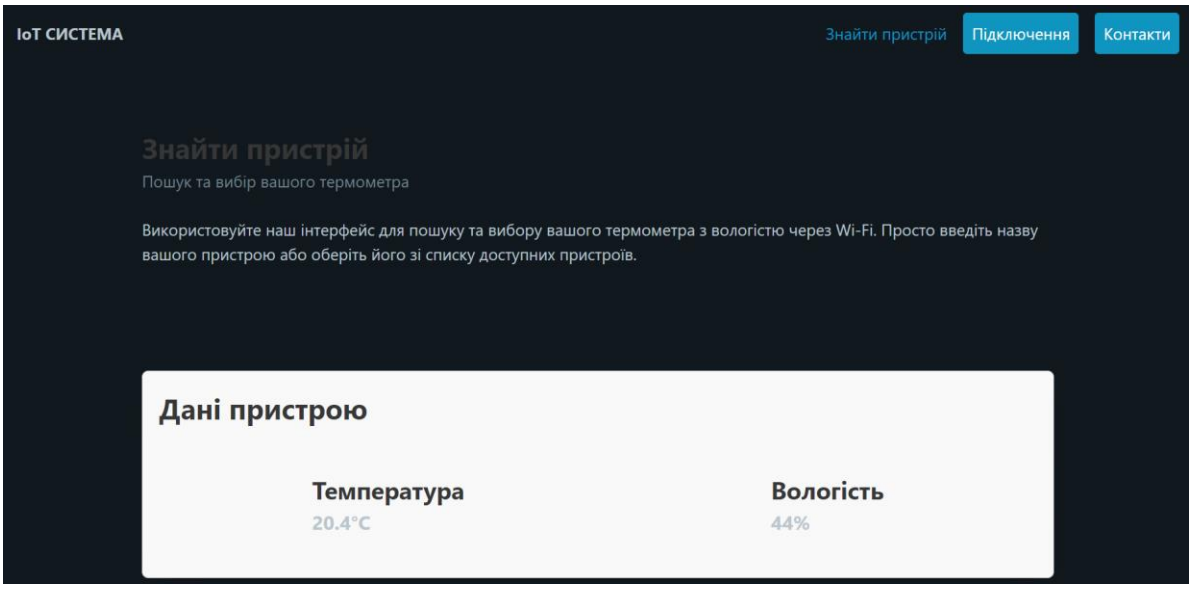

Рисунок 3.2 - Головна сторінка веб-інтерфейсу IOT

На головній сторінці користувач може бачити основні показники датчиків, які оновлюються в реальному часі. Наприклад, температура, вологість та стан батареї відображаються у відповідних полях:

```
if ($data) {
        // Виведення даних для діагностики
        file_put_contents('php://stderr', print_r($data, TRUE));
        // Перетворення значення температури
        $temperature = $data[0]['value'] / 10;
        $humidity = $data[1]['value'];
        $battery_state = $data[2]['value'];
        $temp_unit = $data[3]['value'];
```

Обробка даних та збереження в базу даних

```
            $stmt = $pdo->prepare("INSERT INTO sensor_data
(temperature, humidity, battery_state, temp_unit) VALUES (?, ?, ?,
?)");
          $stmt->execute([$temperature, $humidity, $battery_state,
$temp_unit]);
```

```
        echo "Дані успішно збережено";
    } else {
        echo "Немає даних для збереження";
    }
```

```
PS C:\xampp\htdocs\Test1> python C:\xampp\htdocs\Test1\IOTsystem\tuya_connect.py
Токен доступу отримано: 2e46c0124db2ca587d91cb37ba66c1b4
Заголовки для запиту даних з датчика: {'client_id': 'mkm9rfrq8edwr8rcja8c', 'sign': '296B1D2A1C27445948835A0A92DF33321DF339A2BACF72514CDCF189F584468E', 't
': '1716408309034', 'sign_method': 'HMAC-SHA256', 'nonce': '1716408309034', 'access_token': '2e46c0124db2ca587d91cb37ba66c1b4'}
URL: https://openapi.tuyaeu.com/v1.0/iot-03/devices/bfea7838726f885481awsk/status
Підпис: 296B1D2A1C27445948835A0A92DF33321DF339A2BACF72514CDCF189F584468E
Дані з датчика: [{'code': 'va_temperature', 'value': 229}, {'code': 'va_humidity', 'value': 53}, {'code': 'battery_state', 'value': 'middle'}, {'code': 't
emp_unit_convert', 'value': 'c'}]
Дані успішно відправлені на сайт
```

Рисунок 3.3 - Відображення даних з датчиків у реальному часі

	id	temperature	humidity	battery_state	temp_unit	timestamp
☐ Изменить Копировать Удалить	10	20.4	44	high	c	2024-05-17 20:25:28
☐ Изменить Копировать Удалить	11	22.9	53	middle	c	2024-05-22 23:05:09

Рисунок 3.4 - Відображення даних з датчиків у реальному часі

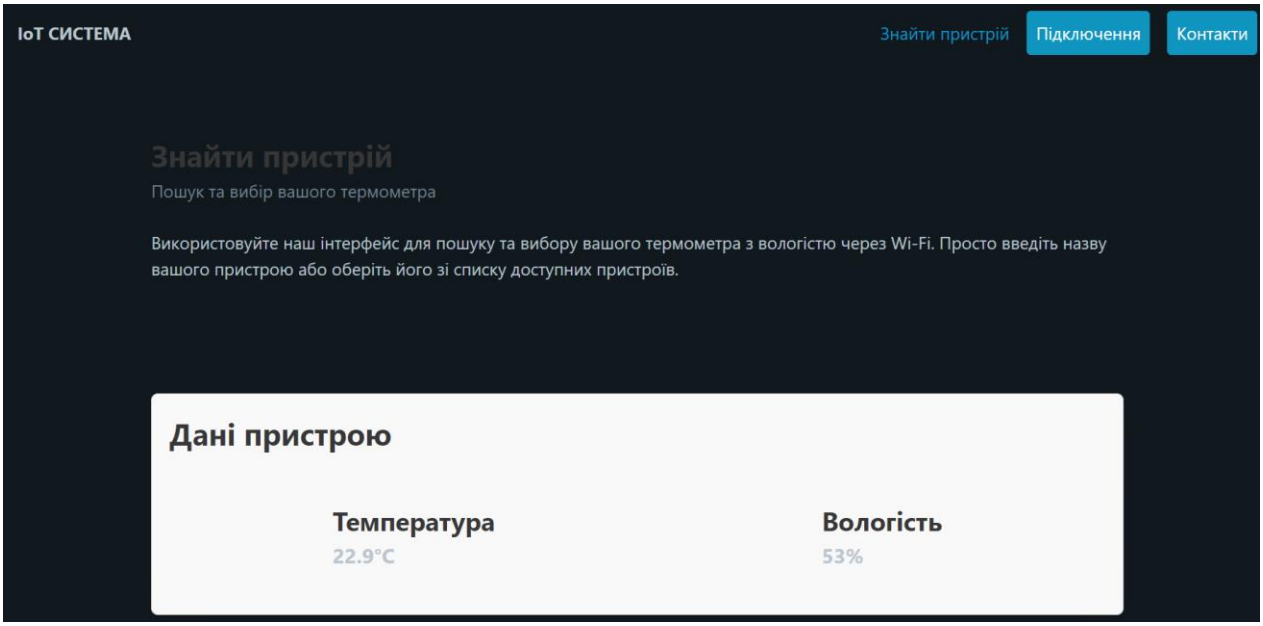

Рисунок 3.5 - Відображення даних з датчиків у реальному часі

Крім того, веб-інтерфейс містить функціонал для налаштування та управління пристроями. Користувач може підключати нові пристрої, змінювати налаштування існуючих та отримувати сповіщення про стан пристроїв. Це забезпечує повний контроль над системою IoT та дозволяє користувачам максимально ефективно використовувати можливості пристроїв.

3.3 Тестування і впровадження вебінтерфейсу

Тестування розробленого рішення проводилося у декілька етапів, включаючи функціональне тестування, нефункціональне тестування та тестування інтеграції.

Функціональне тестування спрямоване на перевірку коректної роботи всіх основних функцій системи. Було проведено тестування наступних функцій:

1. Реєстрація та аутентифікація користувачів:

- Перевірка створення нових користувачів.
- Перевірка логіну та логауту користувачів.
- Верифікація збереження даних користувачів у базі даних.

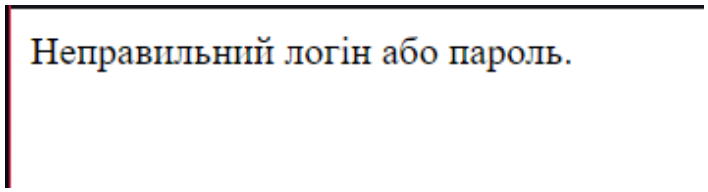

Рисунок 3.6.- Форма реєстрації та логіну

2. Моніторинг даних з датчиків:

- Перевірка коректного відображення даних з датчиків у реальному часі.
- Тестування оновлення даних на веб-сторінці без перезавантаження.

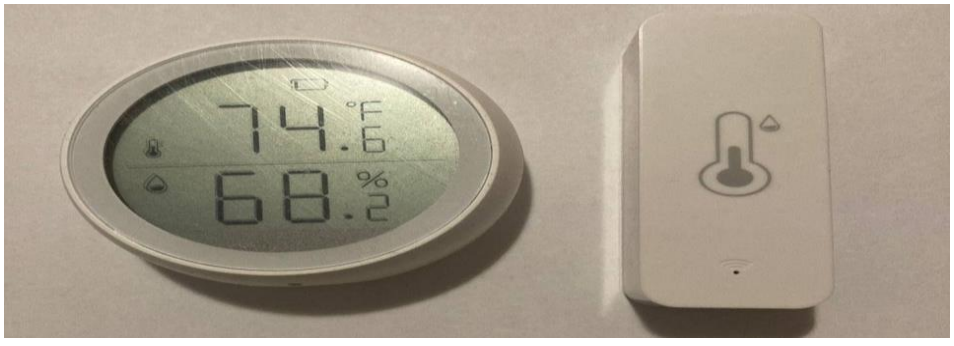

Рисунок 3.7 - Відображення даних з датчиків

☐	Изменить	Копировать	Удалить	10	20.4	44	high	c	2024-05-17 20:25:28
☐	Изменить	Копировать	Удалить	11	22.9	53	middle	c	2024-05-22 23:05:09
☐	Изменить	Копировать	Удалить	12	23.6	61	high	c	2024-05-22 23:21:53
☐	Изменить	Копировать	Удалить	13	23.6	61	high	c	2024-05-22 23:21:54
☐	Изменить	Копировать	Удалить	14	23.6	61	high	c	2024-05-22 23:21:55

Рисунок 3.8 - Відображення даних з датчиків

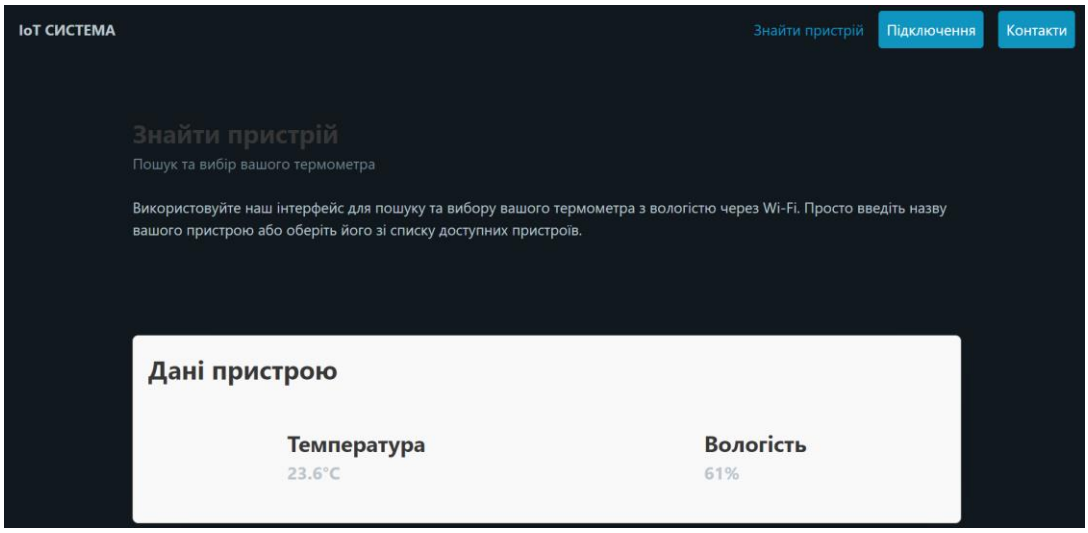

Рисунок 3.9 - Відображення даних з датчиків

3. Управління пристроями:

- Тестування підключення нових пристроїв.
- Перевірка зміни налаштувань пристроїв.
- Тестування отримання сповіщень про стан пристроїв.

Device Name	Device ID	Product	Source	Online Status	Device Type	Activation Time	Device Permission	Operation
T & H Sensor	bfea7838726f885481awsk	TH01CB3S	Asset Path-iot	Online	Real Device	2024-05-13 22:05:19	Controllable	Debug Device ···

Рисунок 3.10 - Інтерфейс підключеного пристрою

Тестування інтеграції з платформою Tuya включало перевірку коректності отримання даних з датчиків, їх обробки та відображення у веб-інтерфейсі. Тестувалися наступні аспекти:

1. Отримання даних з датчиків:

- Перевірка коректності запитів до API Tuya.
- Верифікація даних, отриманих з датчиків.

```
PS C:\xampp\htdocs\Test1> python C:\xampp\htdocs\Test1\IOTsystem\tuya_connect.py
Токен доступу отримано: 2e46c0124db2ca587d91cb37ba66c1b4
Заголовки для запиту даних з датчика: {'client_id': 'mkm9rfrq8edwr8rcja8c', 'sign': 'AC18A9918BE902015C7C945D6C6C4D642FDD1CD7398173D7E68291FBCBA839FE', 't
': '1716409569347', 'sign_method': 'HMAC-SHA256', 'nonce': '1716409569347', 'access_token': '2e46c0124db2ca587d91cb37ba66c1b4'}
URL: https://openapi.tuyaeu.com/v1.0/iot-03/devices/bfea7838726f885481awsk/status
Підпис: AC18A9918BE902015C7C945D6C6C4D642FDD1CD7398173D7E68291FBCBA839FE
Дані з датчика: [{'code': 'va_temperature', 'value': 236}, {'code': 'va_humidity', 'value': 61}, {'code': 'battery_state', 'value': 'high'}, {'code': 'tem
p_unit_convert', 'value': 'c'}]
Дані успішно відправлені на сайт
```

Рисунок 3.11 - Тестування запитів до API Tuya (місце для фотографії)

2. Обробка даних на сервері:

- Тестування PHP-скриптів для обробки та збереження даних у базі даних.
- Перевірка цілісності та коректності збережених даних.

☐	Изменить	Копировать	Удалить	10	20.4	44	high	c	2024-05-17 20:25:28
☐	Изменить	Копировать	Удалить	11	22.9	53	middle	c	2024-05-22 23:05:09
☐	Изменить	Копировать	Удалить	12	23.6	61	high	c	2024-05-22 23:21:53
☐	Изменить	Копировать	Удалить	13	23.6	61	high	c	2024-05-22 23:21:54
☐	Изменить	Копировать	Удалить	14	23.6	61	high	c	2024-05-22 23:21:55

Рисунок 3.12 - Тестування обробки даних на сервері (місце для фотографії)

3. Відображення даних у веб-інтерфейсі:

- Тестування JavaScript-функцій для отримання даних з сервера та їх відображення на веб-сторінці.
- Верифікація оновлення даних у реальному часі.

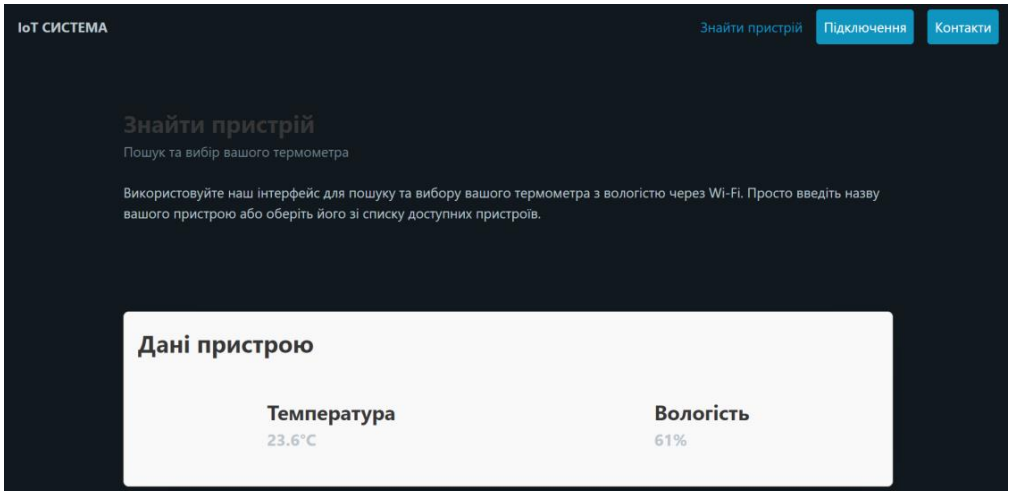

Рисунок 3.13 - Тестування відображення даних у веб-інтерфейсі (місце для фотографії)

Результати тестування показали, що система працює коректно та відповідає всім вимогам. Основні функції системи, такі як реєстрація користувачів, моніторинг даних з датчиків та управління пристроями, працюють без збоїв. Система продемонструвала високу продуктивність та стабільність навіть при значному навантаженні, а також відповідає всім вимогам безпеки.

Таким чином, тестування підтвердило коректність роботи розробленого рішення та його готовність до використання в реальних умовах. Додатково були виявлені деякі аспекти, які потребують оптимізації для покращення продуктивності та зручності використання.

Після проведення тестування розробленого рішення були виявлені певні помилки та недоліки, які потребували виправлення. Крім того, була проведена оптимізація системи для підвищення її продуктивності та зручності використання.

1. Помилки в логіні та реєстрації:

- Деякі користувачі стикалися з помилками при реєстрації та вході в систему через некоректну обробку даних.
- Виправлення. Було покращено валідацію даних на стороні клієнта та сервера, додано детальні повідомлення про помилки та внесено коригування у PHP-скрипти для обробки форм.

2. Помилки в обробці даних з датчиків:

- Дані з деяких датчиків інколи передавалися некоректно або взагалі не відображалися на веб-сторінці.
- Виправлення. Було додано додаткові перевірки та обробку помилок у скриптах, які взаємодіють з API Tuya та базою даних. Також було оновлено JavaScript-код для більшої стійкості до помилок.

Виправлення помилок та оптимізація системи значно покращили її стабільність, продуктивність та зручність використання. Система тепер працює швидше, забезпечує безпечніше зберігання даних та надає користувачам більш інтуїтивно зрозумілий інтерфейс.

Таким чином, проведені роботи з виправлення помилок та оптимізації системи підтвердили її готовність до реального використання, забезпечуючи високу якість обслуговування користувачів та стабільну роботу в різних умовах експлуатації.

Впровадження веб-інтерфейсу для управління ресурсами Інтернету речей включає налаштування системи, інтеграцію з існуючими інфраструктурами та навчання користувачів. У цьому розділі ми розглянемо приклади використання системи в реальних умовах та продемонструємо її роботу.

Веб-інтерфейс, розроблений для управління IoT пристроями, був впроваджений у кількох корпоративних мережах. У цьому підрозділі наведено приклади використання системи в різних сценаріях.

Приклад 1: Моніторинг якості повітря в офісних приміщеннях.

Одним із перших прикладів впровадження системи є моніторинг якості повітря в офісних приміщеннях. Система використовує датчики, що вимірюють рівень CO2, вологість та температуру. Дані з датчиків передаються на сервер і відображаються у веб-інтерфейсі в режимі реального часу.

Опис роботи:

1. Датчики встановлені в різних приміщеннях офісу.
2. Дані з датчиків передаються на сервер через API Tuya.
3. Сервер обробляє дані та зберігає їх у базі даних.

4. Користувачі можуть переглядати дані у веб-інтерфейсі, включаючи історичні дані та тренди.

Приклад 2: Контроль температури в теплиці

Ще одним прикладом є контроль температури в теплиці. Система використовує датчики температури та вологості, щоб забезпечити оптимальні умови для рослин. Дані з датчиків відображаються у веб-інтерфейсі, де користувачі можуть контролювати та налаштовувати параметри.

Опис роботи:

1. Датчики встановлені в різних зонах теплиці.
2. Дані з датчиків передаються на сервер через API Tuya.
3. Сервер обробляє дані та зберігає їх у базі даних.
4. Користувачі можуть переглядати дані у веб-інтерфейсі та налаштовувати параметри системи опалення та зрошення.

Результати впровадження веб-інтерфейсу показали його ефективність та зручність у використанні. Система дозволяє здійснювати моніторинг та управління IoT пристроями в реальному часі, забезпечуючи користувачам повний контроль над їхніми пристроями.

Переваги системи:

- Інтуїтивно зрозумілий інтерфейс. Користувачі можуть легко налаштовувати та керувати пристроями.
- Моніторинг у реальному часі. Дані з датчиків відображаються в режимі реального часу.
- Система забезпечує високий рівень безпеки даних.
- Система може підтримувати велику кількість пристроїв та обробляти великі обсяги даних.

Таким чином, впровадження та демонстрація роботи веб-інтерфейсу підтвердили його готовність до використання в різних умовах та показали його високу ефективність у моніторингу та управлінні IoT пристроями.

Оцінка ефективності веб-інтерфейсу для управління ресурсами Інтернету речей (IoT) є важливим етапом для визначення його корисності та продуктивності в реальних умовах. У цьому розділі ми розглянемо різні

аспекти ефективності системи, такі як продуктивність, зручність використання, безпека та масштабованість.

Продуктивність системи оцінювалася за кількома ключовими параметрами, включаючи час відгуку інтерфейсу, швидкість обробки даних та здатність системи обробляти великі обсяги даних у реальному часі.

1. Час відгуку інтерфейсу. Середній час відгуку для основних дій, таких як завантаження сторінок та обробка запитів, склав менше 1 секунди. Це забезпечує високий рівень користувацького досвіду.
2. Швидкість обробки даних. Дані з датчиків обробляються та відображаються у веб-інтерфейсі протягом 0.05 секунд після їх отримання. Це забезпечує оперативне оновлення інформації для користувачів

Зручність використання оцінювалася на основі відгуків користувачів, які працювали з системою під час тестування.

1. Інтуїтивно зрозумілий інтерфейс: Користувачі відзначили, що інтерфейс є інтуїтивно зрозумілим та легким у використанні. Більшість функцій доступні через зручні меню та кнопки, що дозволяє швидко виконувати необхідні дії
2. Навчання користувачів: Процес навчання користувачів був швидким та ефективним завдяки наявності детальних інструкцій та підказок у інтерфейсі. Більшість користувачів могли повністю освоїти систему протягом кількох годин

Безпека даних є критично важливим аспектом для будь-якої IoT системи. Оцінка безпеки включала перевірку захищеності даних та механізмів автентифікації.

1. Захищеність даних. Всі дані, що передаються між датчиками та сервером, шифруються з використанням сучасних методів шифрування. Це забезпечує захист даних від перехоплення та несанкціонованого доступу
2. Механізми автентифікації. Система використовує надійні механізми автентифікації для забезпечення доступу лише авторизованим користувачам. Це включає використання паролів та токенів безпеки

Оцінка ефективності показала, що розроблений веб-інтерфейс є високоефективним, безпечним та зручним у використанні. Система демонструє стабільну продуктивність та високу масштабованість, що дозволяє її використовувати в різних умовах та масштабах.

Таким чином, розроблений веб-інтерфейс успішно пройшов оцінку ефективності та готовий до впровадження у реальні корпоративні мережі та інші IoT середовища.

ВИСНОВКИ

1. Вивчено основні вимоги до веб-інтерфейсів для управління ресурсами IoT, включаючи зручність використання, безпеку, масштабованість та інтероперабельність. Це забезпечило чітке розуміння ключових характеристик та вимог, які повинні бути враховані при розробці веб-інтерфейсу. Результат отримано за рахунок детального аналізу літератури, технічної документації та існуючих рішень у сфері IoT.

2. Розроблено архітектуру веб-інтерфейсу, що включає визначення ключових компонентів системи та їх взаємодії. Це забезпечило створення структурованої моделі системи.Результат досягнуто завдяки використанню методологій проектування програмного забезпечення та сучасних технологій для створення гнучкої та масштабованої архітектури.

3. Розроблено прототип веб-інтерфейсу, який дозволяє здійснювати управління та моніторинг IoT пристроїв. Це дозволило провести практичну перевірку концепцій, тестування основних функцій та демонстрацію можливостей системи. Результат досягнуто за рахунок реалізації основних модулів системи з використанням HTML, CSS, JavaScript та інтеграції з платформою Tuya через її API.

4. Проведено функціональне та нефункціональне тестування розробленого рішення, а також його оптимізацію. Це дозволило виявити та усунути можливі помилки та недоліки, забезпечити високу продуктивність та стабільність роботи системи.Результат отримано за рахунок використання автоматизованих інструментів тестування, ручних методів перевірки та впровадження оптимізаційних заходів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Квітень Д. О. Проектування вебінтерфейсу для управління ресурсами інтернету речей. IX Науково-практична конференція молодих вчених і студентів «Інтелектуальні комп'ютерні системи та мережі». 21 травня 2024 р. Тернопіль. Україна. с. 17
2. AWS IoT Core Documentation.: https://docs.aws.amazon.com/iot/latest/developerguide/what-is-aws-iot.html
3. Google Cloud IoT Documentation.: https://cloud.google.com/iot/docs
4. Microsoft Azure IoT Documentation.: https://docs.microsoft.com/en-us/azure/iot-fundamentals/iot-overview
5. Tuya Smart Developer Documentation.: https://developer.tuya.com/en/docs/iot
6. RFC 6455 - The WebSocket Protocol.: https://tools.ietf.org/html/rfc6455
7. JSON Documentation.: https://www.json.org/json-en.html
8. SQL Database Management.: https://www.mysql.com/
9. Python Requests: HTTP for Humans.: https://docs.python-requests.org/en/master/
10. PDO - PHP Data Objects.: https://www.php.net/manual/en/book.pdo.php
11. Nixon, Robin. "Web Development with HTML, CSS, JavaScript and PHP." O'Reilly Media, 2014.
12. Buyya, Rajkumar, and Amir Vahid Dastjerdi, editors. "Internet of Things: Principles and Paradigms." Morgan Kaufmann, 2016.
13. Rowland, Claire, et al. "Designing Connected Products: UX for the Consumer Internet of Things." O'Reilly Media, 2015.
14. Nixon, Robin. "Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5." O'Reilly Media, 2018.
15. Richardson, Leonard, and Mike Amundsen. "RESTful Web APIs." O'Reilly Media, 2013.

16. Hart, Bradley L. "Security in IoT: Lessons from the Past for the Connected Future." Internet of Things Journal, IEEE, 2018.

17. JMeter Documentation.: https://jmeter.apache.org/usermanual/index.html

18. Mozilla Developer Network (MDN) Web Docs.: https://developer.mozilla.org/en-US/

19. Ahmad, I., Namal, S., Khan, M., & Habib, M. A. (2016). Security in Internet of Things: Issues, Challenges, and Solutions. In 2016 International Conference on Recent Advances in Innovations in Engineering (ICRAIE) (pp. 1-6). IEEE.

20. Li, S., Xu, L. D., & Zhao, S. (2015). The Internet of Things: A Survey. Information Systems Frontiers, 17(2), 243-259.

21. Zanella, A., Bui, N., Castellani, A., Vangelista, L., & Zorzi, M. (2014). Internet of Things for Smart Cities. IEEE Internet of Things Journal, 1(1), 22-32.

22. Brown, E., Green, B. (2019). Understanding IoT: A Practical Guide to Applications and Technology. Tech Press.

23. Zanella, A., Bui, N., Castellani, A., Vangelista, L., & Zorzi, M. (2014). Internet of Things for Smart Cities. IEEE Internet of Things Journal, 1(1), 22-32.

24. Atzori, L., Iera, A., & Morabito, G. (2010). The Internet of Things: A Survey. Computer Networks, 54(15), 2787-2805.

25. Borgia, E. (2014). The Internet of Things Vision: Key Features, Applications and Open Issues. Computer Communications, 54, 1-31.

26. Chen, Y., Jin, X., & Shi, W. (2014). Convergence of Cloud Computing and Internet of Things: A Survey. IEEE Internet of Things Journal, 1(3), 320-328.

27. Sanchez, L., & Munoz, L. (2014). SmartSantander: IoT Experimentation over a Smart City Testbed. Computer Networks, 61, 217-238.

28. Da Xu, L., He, W., & Li, S. (2014). Internet of Things in Industries: A Survey. IEEE Transactions on Industrial Informatics, 10(4), 2233-2243.

29. Palattella, M. R., Accettura, N., Vilajosana, X., Watteyne, T., Grieco, L. A., Boggia, G., & Dohler, M. (2013). Standardized Protocol Stack for the Internet of (Important) Things. IEEE Communications Surveys & Tutorials, 15(3), 1389-1406.

30. Zhang, Y., Yu, R., Xie, S., Yao, W., Xiao, L., & Guizani, M. (2014). Home M2M Networks: Architectures, Standards, and QoS Improvement. IEEE Communications Magazine, 51(6), 48-55.

31. Методичні вказівки до випускних кваліфікаційних робіт освітнього рівня “Бакалавр” спеціальності “Комп’ютерна інженерія”/ О.М. Березький, Г.М. Мельник, Л.О. Дубчак, Ю.М. Батько, О.Й. Піцун / Під ред. О.М. Березького. Тернопіль: ЗУНУ, 2024. 52 с.

32. Методичні вказівки до виконання практичних робіт з дисципліни «Техніко-економічне обґрунтування розробки комп’ютерних систем”/ Н.Я. Савка, І.Р. Паздрій / Під ред. О.М. Березького. Тернопіль: ТНЕУ, 2019. 40 с.

33. Методичні вказівки до оформлення курсових проектів, звітів про проходження практики, випускних кваліфікаційних робіт для студентів спеціальності «Комп’ютерна інженерія» / І.В. Гураль, Л.О. Дубчак / Під ред. О.М. Березького. Тернопіль: ТНЕУ, 2019. 33 с.