МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет радіоелектроніки

Центр післядипломної освіти
Кафедра Програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

другий (магістерський)
(рівень вищої освіти)

Дослідження алгоритмів оптимізації процесів керування предметами загального доступу

Виконав:
студент 2 курсу групи ІПЗздм-21-1

Трофуненко І. С.
(прізвище, ініціали)

Спеціальність	121 – Інженерія програмного забезпечення
Тип програми	Освітньо-наукова
Керівник	проф. Шубін І. Ю.

(посада, прізвище, ініціали)

Допускається до захисту

Зав. Кафедри _____ З.В. Дудар

2023 р.

Харківський національний університет радіоелектроніки

Факультет Центр післядипломної освіти

Кафедра Програмної інженерії

Рівень вищої освіти другий (магістерський)

Спеціальність 121 – Інженерія програмного забезпечення
(код і повна назва)

Тип програми освітньо-наукова програма

Освітня програма Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)

«____»_____2023 р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Трофуненку Іллі Сергійовичу
(прізвище, ім'я, по-батькові)

1. Тема роботи (проекту) Дослідження алгоритмів оптимізації процесів керування предметами загального доступу

затверджена наказом по університету від «03» квітня 2023 р. № 83 Стз

2. Термін подання студентом роботи до екзаменаційної комісії «12» травня 2023 р.

3. Вихідні дані до роботи *електронні ресурси за обраною тематикою, алгоритми позиціонування об'єктів на основі бездротових технологій, предмети загального користування, принципи розробки програмного забезпечення*

4. Перелік питань, що потрібно опрацювати в роботі *реферат, вступ, аналіз предметної галузі, формування вимог до програмної системи, аналіз алгоритмів фіксації переміщень об'єктів, виконання експериментального дослідження, висновки, перелік джерел посилань; опис прийнятих програмних рішень; висновки.*

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз предметної галузі	03 квітня 2023	*виконано*
2	Огляд існуючих методів	10 квітня 2023	*виконано*
3	Розробка алгоритмів, проєктування та розробка ПЗ	15 квітня 2023	*виконано*
4	Підготовка пояснювальної записки	20 квітня 2023	*виконано*
5	Спецчастина	28 квітня 2023	*виконано*
6	Підготовка презентації та доповіді	03 травня 2023	*виконано*
7	Попередній захист	05 травня 2023	*виконано*
8	Нормоконтроль, рецензування	07 травня 2023	*виконано*
9	Занесення роботи в електронний архів	08 травня 2023	*виконано*
10	Допуск до захисту в зав. кафедри	11 травня 2023	*виконано*

Дата видачі завдання «03» квітня 2023 р.

Студент  Трофуненко І. С.

Керівник роботи _____ проф. Ігор ШУБІН

РЕФЕРАТ / ABSTRACT

Кваліфікаційна робота магістра містить 92 с., 36 рис., 3 табл., 20 джерел.

АВТОМАТИЗАЦІЯ, АНАЛІТИКА, БАЗИ ДАНИХ, ОПТИМІЗАЦІЯ, ІНФОРМАЦІЙНА СИСТЕМА, АЛГОРИТМІЗАЦІЯ, КОМЕРЦІЙНА СТРУКТУРА, RFID, BLE МІТКИ, BIG DATA.

Об’єктом дослідження є процес аналізу методів та ефективних алгоритмів роботи з великими обсягами даних у високонавантажуваних системах, а також проектування інформаційної системи автоматизованого аналізу та обробки наборів даних, що включають інформацію про використання предметів загального користування у комерційних організаціях.

Метою роботи є дослідження та проєктування методів та алгоритмів оптимізації керування предметами загального доступу, а також проектування та розробка відповідної програмної системи.

Методами дослідження є: логічне, концептуальне, наукове моделювання, класифікація, емпіричний науковий метод, ER- та UML-моделювання предметної області,

В результаті виконано дослідження ефективності методів позиціонування об’єктів у приміщенні, розроблено гібридний метод позиціонування, проведено експериментальне дослідження, а також розроблено програмну систему для оптимізації процесів керування предметами загального доступу за рахунок моніторингу у реальному часі.

AUTOMATION, ANALYTICS, DATABASES, OPTIMIZATION, INFORMATION SYSTEM, ALGORITHMS, COMMERCIAL STRUCTURE, RFID, BLE TAGS, BIG DATA.

The object of the research is the process of analyzing methods and effective algorithms for working with Big Data in high-load systems, as well as designing an information system for automated analysis and processing of data sets that include information on the use of common-use items in commercial organizations.

The purpose of the work is to study and design methods and algorithms for optimizing the management of common access items, as well as designing and developing the corresponding software system.

Research methods include logical, conceptual, empirical scientific methods, scientific modeling, classification, ER and UML modeling of the subject area, use of object-oriented and component-oriented architectural styles

During the work, research was conducted on the effectiveness of methods for positioning objects in a room, a hybrid positioning method was developed, experimental research was accomplished, and a software system was developed for optimizing processes for managing publicly accessible items through real-time monitoring.

Умови публікації пояснювальної записки

Я, Трофуненко Ілля Сергійович
(прізвище, ім'я, по батькові)

студент групи ІПЗздм-21-1 здобувач вищої освіти на другому (магістерському) рівні

кафедра програмної інженерії,
(повна назва кафедри)

заявляю: моя кваліфікаційна робота на тему
Дослідження алгоритмів оптимізації процесів керування предметами загального доступу,
(назва роботи)

що буде представлена до ЕК для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElArKhNURE. Всі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлений (а) з діючим положенням «Про протидію академічному плагіату в ХНУРЕ», згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

BLE – Bluetooth low energy.

UWP – Ultra-wideband.

RFID – Radio-frequency identification.

GPS – Global positioning system.

VLC – Visible light communication.

RSSI – Received signal strength indicator.

PDR – Pedestrian dead reckoning.

ML – Machine learning.

RESP – Redis serialization protocol.

API – Application programming interface.

ВСТУП

З розвитком промисловості, сфер будівництва, надання користувацьких послуг, розвитку автоматизованих виробництв необхідність моніторингу даних про використання або переміщення певних об’єктів невпинно зростає. При цьому, експоненційно зростають обсяги даних, які необхідно обробляти та аналізувати. Збір та дослідження таких даних може помітно впливати на комерційні заклади, дозволяючи оптимізувати певні внутрішні процеси й механізми з використанням отриманих даних. Прикладом такого типу інформації можуть бути статистичні та аналітичні дані про користування спільними предметами, а також переміщення об’єктів в приміщенні. Для різних типів комерційних структур аналітична обробка таких даних може надавати певні переваги, наприклад: відстеження об’єктів шерінг-сервісів (сервіси шерінгу автомобілей, двоколісного транспорту та електротранспорту, офіс-шерінг сервіси), транспортних компаній різних типів, елементів будівельного процесу (інструменти, спеціалізована техніка), робототехніки. Отримання таких переваг може мати істотний вплив на прибуткову та організаційні складові комерційної структури.

Зважаючи на вищезазначену мету та потенційні переваги необхідно розробити програмну систему, що надасть можливість працювати з суб’єктами різного типу. При цьому, для максимізації доцільності впровадження такої системи, необхідно дослідити і розробити найбільш ефективні алгоритми та підходи до роботи з великими обсягами даних. Крім того, система має бути максимально зручною та доступною для первинного впровадження.

У науковій роботі необхідно дослідити принципи роботи з великими даними та алгоритми оптимізації Додатково, мають бути розглянуті такі питання як зберігання, обробка та аналіз даних. Ми також розглянемо різноманітні технології та інструменти, що доступні для роботи з великими даними, такі як розподілені системи, обчислення в хмарі та алгоритми машинного навчання.

Крім того, ми дослідимо найкращі практики роботи з великими даними, включаючи очищення даних, інтеграцію даних, візуалізацію даних та забезпечення

безпеки даних. Також мають бути розглянуті етичні аспекти, пов'язані з роботою з великими даними, такі як приватність, справедливість та упередження.

Для автоматизації процесу збору та отримання такої інформації необхідно розробити складну інформаційну систему, яка надасть можливості: визначати та відстежувати статуси та розташування певних об’єктів, створювати обмеження доступу до певних елементів, класифікуючи суб’єкти доступу за ролями, отримувати сповіщення про порушення політик безпеки щодо об’єктів загального користування адміністраторами організацій, автоматизації процесу фіксації користування об’єктом.

Описана вище система буде актуальною для таких типів сучасних комерційних структур:

- логістичні та транспортні компанії;
- виробничі підприємства;
- сервіси шерінгу;
- будівельні організації;
- магазини, супермаркети та гіпермаркети;
- торгівельно-розважальні та офісні центри.

Ця система є дуже корисним та зручним інструментом, який за рахунок ефективного алгоритмічного трекінгу та аналізу даних про переміщення людей у різних типах комерційних структур надасть можливість оптимізувати бізнес-процеси, підвищити прибутки, конверсії та покращити організацію робочого процесу на підприємстві чи у закладі.

Варто зазначити, що кількість аналогічних систем є дуже незначною, що робить їх вартість дуже великою. Розроблювана система матиме нижчу вартість початкового встановлення, налаштування та обслуговування, що дозволить поширити цю систему серед різних типів комерційних структур та зробить її доступною не тільки для великого, а й для малого та середнього типів бізнесу.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Аналіз предметної галузі

З розвитком загальної економічної системи, а також стрімким поширенням та вдосконаленням технологій виникає необхідність впровадження систем відстеження розташування та переміщення об’єктів комерційної організації. Такі системи дозволяють систематизувати та оптимізувати внутрішні процеси різних типів організацій, наприклад: процеси відстеження вантажів, контейнерів та техніки у логістичних компаніях, процеси контролю користування ресурсами спільного користування (сервіси спільного користування автомобільним, велосипедним або електричним транспортом), технологічні процеси виробничих організацій (рух сировини та готової продукції). Крім того, невпинне зростання вимог до точності та ефективності виконання логістичних та виробничих операцій вплинуло на необхідність розробки трекінгових систем.

Описані системи мають велику цінність для вищезазначених організацій, через те, що дозволяють зменшувати час на виробництво, переміщення та інвентаризацію продукції, вантажів, інструментів. Такі переваги неодмінно впливають на якість внутрішніх процесів та дозволяють збільшувати прибутковість компаній, що їх інтегрують.

Відповідно до теми роботи буде розроблена інформаційна система (ІС), яка автоматизує процес керування предметами загального користування у реальному часі для оптимізації роботи комерційних структур. Аналіз предметної області є важливим та необхідним етапом розробки будь-якої ІС. На цьому етапі виявляються інформаційні потреби всієї сукупності користувачів майбутньої системи.

Для ефективного визначення структури предметної області необхідно визначити основні об’єкти, процеси та користувачів.

Першим кроком необхідно визначити потенційних користувачів майбутньої системи: бізнес-користувачі, тобто комерційні структури; клієнти бізнес-користувача; співробітники бізнес-користувача.

Наступним кроком буде визначення основних об’єктів системи, серед них:

– клієнт бізнес-користувача;

– співробітник бізнес-користувача;

– бізнес-користувач;

– зчитувач, що фіксує переміщення і позначає певну зону закладу;

– станція-зчитувач, біля якої розташовуються предмети загального користування, та яка фіксує початок або закінчення користування предметом;

– зафіксоване переміщення особи у певний проміжок часу у певній зоні;

– клас клієнтів або співробітників бізнесу (використовується для налаштування рівнів доступу до певних об’єктів);

– заборонена зона для певного класу клієнтів або співробітників бізнесу.

Також позначимо основні процеси предметної області:

– відстеження користування певним об’єктом та його переміщення;

– перегляд та аналіз даних про користування та переміщення певних об’єктів;

– контроль користування певними предметами;

– оптимізація робочих процесів (диспетчеризація переміщення техніки, управління виробничими компонентами), за рахунок отриманих даних про переміщення;

– контроль місцезнаходження об’єктів в реальному часі;

– перегляд історії переміщень та взаємодій з певним об’єктом.

Таким чином, ми визначили ключові компоненти та процеси предметної галузі, а також описали передумови та доцільність створення системи що розробляється.

1.2 Аналіз ринку

Етап аналіз ринку є важливою частиною дослідження, оскільки він дозволяє оцінити ситуацію на ринку, визначити ключових гравців, їх стратегії та динаміку

змін. У цьому розділі будуть проаналізовані тенденції ринку, конкуренція, потенційні клієнти та їх потреби. Також будуть розглянуті сильні та слабкі сторони конкурентів та можливості розвитку продукту на ринку. Крім того, вищезазначений аналіз дозволяє визначити актуальність продукту, а також його загальну конкурентоспроможність відносно існуючих альтернатив на ринку.

В ході виконання аналізу конкурентів було знайдено декілька відповідних продуктів.

Першими в ході аналізу буде розглянуто системи позиціонування від компанії Sewio [1]. Зазначена компанія надає послуги з розробки та впровадження індивідуальних систем позиціонування в приміщенні, з використанням технології UWP (Ultra-Wideband). Наведемо приклад такої системи на рисунку 1.1.

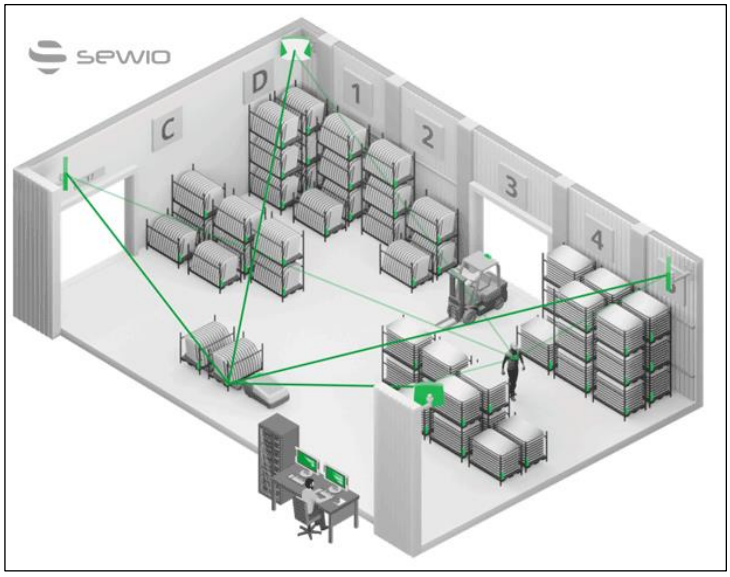

Рисунок 1.1 – Демонстраційне зображення впровадженої системи

Системи, які розробляються компанією, складаються зі зчитувачів, ретрансляторів (для посилення сигналу), міток (розташовуються на об’єктах зчитування), серверного обладнання, а також відповідного програмного забезпечення. Варто зазначити, що всі вказані компоненти поставляються виключно у комплекті і формуються під вимоги кожного замовника.

Дослідивши можливості описаного продукту визначимо його ключові недоліки, а саме: висока вартість впровадження системи, а також відсутність

прикладів застосування системи для користувачів, що не відносяться до великого бізнесу. Таким чином, даний продукт має велику кількість переваг (швидкість, технологічність, надійність), проте підходить тільки для обмеженої цільової аудиторії та має велику вартість початкового встановлення.

Наступним кроком розглянемо систему компанії Esri під назвою ArcGIS IPS [2]. Зазначена система дозволяє здійснювати моніторинг та навігацію територією визначеного приміщення за аналогією з технологією GPS . На рисунку 1.2 наведено зображення головної сторінки сервісу.

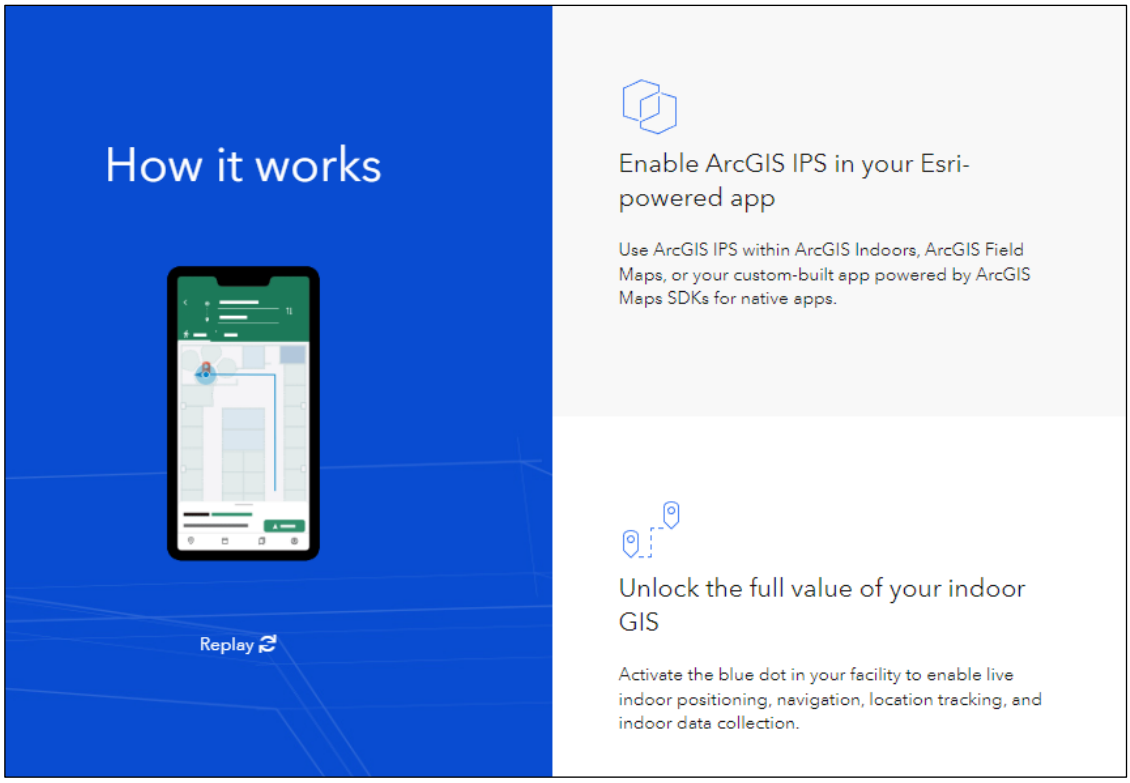

Рисунок 1.2 – Головна сторінка сервісу ArcGIS IPS

Проаналізувавши зазначений продукт було визначено, що система надає функціональність відстеження переміщень користувачів мобільних пристроїв територією приміщення, а також надає їм можливості навігації.

Звернемо увагу на те, що система має декілька суттєвих недоліків у порівнянні з системою, що розробляється. По-перше, для позиціонування він використовує технологію позиціонування на основі Wi-Fi сигналу, що вимагає повного і щільного покриття території приміщення сигналом бездротової мережі,

що може бути складною задачею, якщо територія велика. Крім того, технологія Wi-Fi, в силу обмежень протоколу має порівняно низьку точність позиціонування. По-друге, для побудови маршрутів та навігації, впровадження такої системи вимагає побудови мап місцевості та вимагає незмінності перешкод всередині приміщення.

Таким чином, можна зробити висновок, що зазначена система має вузьку спеціалізацію, а її вартість впровадження дуже висока, аналогічно до першого конкуренту.

Додатково необхідно зазначити, що ринок систем внутрішнього позиціонування невпинно зростає. Це підтверджує дослідження компанії «MarketsAndMarkets», яке позначає, що на момент написання дослідницької роботи ринок таких систем щорічно зростає на 22,5% [3]. Крім того, за прогнозом, зазначеним у дослідженні, ринок позиціонування зросте з 8,8 мільярдів доларів США у 2022, до 24 мільярдів доларів США у 2027 році.

Таким чином, можна зробити висновок, що система, яка розробляється, має продукти-конкуренти, проте вона має значні функціональні переваги, надає нові варіанти використання, є більш дешевою у початковій інтеграції, а також є більш універсальною та поширюється на багато моделей використання.

1.3 Виявлення проблем та актуалізація рішень

Наступним кроком необхідно визначити потенційні обмеження системи та проаналізувати доступні підходи до її реалізації.

Зазначимо, що на момент проведення дослідження існує велика кількість різноманітних технологій, що дозволяють вирішити поставлене завдання. Провівши попередній аналіз доступних методів реалізації описаної раніше задачі, було визначено наступні технології.

RFID (Radio-Frequency Identification) – ця технологія використовує радіочастотні хвилі для ідентифікації та відстеження об'єктів, таких як товари,

засоби транспорту та обладнання. Для цього на об'єкти закріплюються спеціальні мітки з радіочастотними елементами, які можуть бути зчитані спеціальними сканерами.

GPS (Global Positioning System) – ця технологія використовує супутникову навігацію для відстеження руху засобів транспорту та інших об'єктів. Для цього на об'єкти встановлюються спеціальні приймачі GPS, які передають дані про їх розташування.

Camera-based AI positioning system (система відстеження, що працює на основі камер та штучного інтелекту) – ця технологія використовується для аналізу великих обсягів даних про рух об'єктів та виявлення закономірностей. Застосування штучного інтелекту дозволяє прогнозувати та уникати проблем у логістичних та виробничих процесах.

Wi-Fi positioning system (система позиціонування на основі бездротового протоколу передачі даних Wi-Fi) – система, що працює з використанням RSSI (індикатор потужності отриманого сигналу) і MAC-адресу (контроль доступу до носіїв), програма може обчислити поточне місцезнаходження пристрою кінцевого користувача.

Bluetooth low energy beacons positioning system (система позиціонування на основі блютуз-передатчиків низького енергоспоживання) – радіочастотна технологія для бездротової комунікації, яка може використовуватися для виявлення та відстеження місцезнаходження людей, пристроїв, включаючи відстеження активів, навігацію в приміщенні, служби наближення тощо. Bluetooth – технологія, яка широко розповсюджена всередині приміщень і підтримується багатьма сучасними пристроями.

Проаналізувавши доступні опції для реалізації програмної системи було обрано BLE-мітки, в якості технологій початкової реалізації програмного продукту. Така технологія дозволяє створити максимально доступну та легку у впровадженні систему, а її технологічні можливості цілком задовольняють вимогам системи.

1.4 Постановка задачі

Метою кваліфікаційної роботи є дослідження та розробка ефективних методів аналізу та контролю переміщень об’єктів всередині приміщень організації за допомогою технологій, описаних у пункті 2.2.

В ході виконання роботи необхідно виконати наступні задачі:

– емпіричним методом дослідити ефективність існуючих алгоритмів відстеження переміщень об’єктів;

– розробити ефективні алгоритми фіксації переміщень об’єктів, а також генерації їх маршруту;

– дослідити технології та методи розробки й впровадження високонавантажених систем;

– дослідити підходи до вибору бази даних з урахуванням вимог до проєкту;

– розробити програмну систему, що складатиметься з 4 ключових компонентів:

а) серверна частина;

б) веб-частина;

в) мобільний додаток;

г) розумний пристрій.

2 ПЕРЕЛІК ВИМОГ ДО ПРОГРАМНОЇ СИСТЕМИ

2.1 Загальні відомості

Метою кваліфікаційної роботи є дослідження алгоритмів, а також розробка програмної системи для оптимізації процесів керування предметами загального доступу у комерційних структурах. В цьому розділі необхідно описати ключові вимоги до програмної системи. Зважаючи на те, що майбутня програмна система складатиметься з 4 програмних частин, вимоги наводяться до кожного модуля окремо.

2.2 Вимоги до серверної частини системи

До серверної частини програмної системи наводяться такі вимоги:

– можливість створення облікового запису компанії;

– можливість реєстрації терміналів зчитування та налаштування зон спостереження;

– можливість реєстрації анонімних клієнтів для бізнес-користувача;

– можливість реєстрації іменних та анонімних карток з мітками для відстежування користування предметами загального доступу;

– можливість обробки даних фіксації відвідувань певної зони;

– можливість додавання предметів з обмеженим доступом;

– можливість визначення осіб або груп осіб, для яких доступ до предмету обмежений;

– можливість отримання кількісної статистики за усіма зчитувачами за різні проміжки часу;

– можливість отримання кількісної статистики за певними предметами за різні проміжки часу;

– можливість отримання детальної історії використання певного предмету;

– можливість отримання статистики про використання недоступних предметів;

– можливість перегляду історії переміщень певного об’єкту чи предмету загального користування територією приміщення.

Додатково, до серверної частини висуваються вимоги захисту користувацьких даних, наприклад, паролів та інших чутливих даних користувачів.

2.3 Вимоги для клієнтської веб-частини системи

До клієнтської веб-частини висуваються наступні вимоги:

– можливість створення облікового запису;

– можливість додавання пристроїв відстежування та налаштування зон відстежування;

– можливість реєстрації об’єктів загального користування компанії;

– можливість реєстрації співробітників та клієнтів компанії;

– можливість перегляду статистики про використання певних об’єктів чи предметів загального користування;

– можливість перегляду розгорнутої статистики за певними зчитувачами, об’єктами;

– можливість перегляду кількісної статистики використання предметів;

– можливості перегляду та оновлення профілю бізнес-користувача;

– можливість додавання предметів з обмеженим доступом;

– можливість обмеження доступу до певних предметів для певних клієнтів або співробітників (або для їх груп);

– можливість перегляду історії переміщень певного об’єкту чи предмету загального користування.

Також, у клієнтській веб-частині необхідно передбачити та реалізувати можливості локалізації, а саме підтримка української та англійської мов

клієнтським інтерфейсом, а також можливості подальшого додавання мов локалізації.

2.4 Вимоги до мобільної частини системи

До мобільної частини висуваються наступні вимоги:

– можливості перегляду і оновлення профілю користувача;

– можливості перегляду статистики останніх переміщень предметів;

– можливості перегляду кількісної статистики за певними предметами;

– можливості перегляду заборонених використань предмету;

– можливість отримання push-повідомлень про заборонені використання чи переміщення предметів.

Аналогічно до клієнтської мобільної частини, необхідно передбачити можливості локалізації користувацького інтерфейсу, тобто реалізувати підтримку української та англійської мов інтерфейсом, а також подбати про додавання нових мов локалізації у майбутньому.

2.5 Вимоги до IoT-пристрою

До IoT-пристрою наводяться наступні вимоги:

– можливість встановлення та реєстрації пристроїв та зон спостереження;

– можливості створення анонімних та іменних міток;

– можливості збору та отримання інформації у реальному часі про переміщення об’єктів за допомогою зчитувачів, які будуть відстежувати наявність міток у своєму радіусі дії.

3 АНАЛІЗ АЛГОРИТМІВ ФІКСАЦІЇ ПЕРЕМІЩЕННЯ ОБ'ЄКТІВ

3.1 Аналіз існуючих підходів

Позиціонування в приміщенні за допомогою маяків Bluetooth Low Energy (BLE) привернуло значну увагу після випуску протоколу BLE. Було докладено низку зусиль для покращення продуктивності внутрішнього позиціонування на основі BLE. Однак деякі дослідження приділяють увагу позиціонуванню в приміщенні на основі BLE у щільному середовищі Bluetooth, де розповсюдження сигналів BLE стає складнішим.

До цього часу система позиціонування на основі GPS успішно вирішувала проблеми локалізації та навігації на відкритому повітрі. Однак люди проводять більше 80-90% свого часу в приміщенні. За оцінками, до 2019 року ринок внутрішнього позиціонування зріс до 4,4 мільярда доларів США з високим попитом у сфері охорони здоров'я, роздрібної торгівлі, гостинності, подорожей та інших секторах [4].

GPS, який здійснив революцію в локалізації поза приміщеннями, виявився неефективним для внутрішнього середовища через відсутність покриття сигналом [5]. Таким чином, було застосовано декілька альтернативних підходів для систем позиціонування всередині приміщень: системи позиціонування на основі Wi-Fi [6], системи позиціонування на основі Bluetooth Low Energy (BLE) [7], системи позиціонування на основі радіочастотної ідентифікації (RFID), Ultra-wide Band – системи позиціонування на основі широкосмугового діапазону (UWB) [8], системи позиціонування на основі комунікації у видимому світлі (VLC) [9], системи позиціонування за допомогою датчиків. Ці методи базуються на різних джерелах інформації, таких як технології бездротового зв'язку або вимірювання датчиків.

Однак у кожного з них є недоліки, такі як низька точність, ненадійність, висока складність або висока вартість обладнання. Наприклад, рішення на основі Wi-Fi мають обмеження через обмежену кількість точок доступу і їх негнучкість у розгортанні, що призводить до низької точності від 5 до 15 м. У випадку рішень на основі RFID, він має найкращу точність серед усіх технологій (похибка менше 0,1

м) і не потребує батареї протягом усього терміну служби, але малий радіус дії (менше 1 м) і широке та дороге встановлення великих обсягів читачів обмежує його використання. Хоча рішення на основі UWB мають хорошу точність (похибка менше 0,3 м) і радіус дії до 150 м, високе енергоспоживання та висока вартість є їх головними недоліками.

Комбінована система позиціонування RFID і BLE була представлена для підвищення точності позиціонування в приміщенні з низькою щільністю розгортання. Гібридний метод позиціонування в приміщенні для позиціонування активів за допомогою BLE та RFID досяг 90% точності в межах 1,21 м [10]. BLE також інтегровано з PDR для досягнення точності позиціонування менше 1 м [10].

3.2 Трилатеральний метод

RSSI (Received Signal Strength Indication) – показник рівня отриманого сигналу.

Точніша оцінка відстані між мобільним пристроєм і маяком BLE є ключем до обчислення позиції за допомогою алгоритму триангуляції, який ґрунтується на низькій варіації RSSI. Тому перед застосуванням алгоритму триангуляції необхідно згладити сигнал RSSI. Фільтр рухомого середнього використовувався для згладжування отриманих сигналів RSSI наступним чином:

$$smooth\_RSSI_N = 0.1 * RSSI_N + 0.9 * RSSI_{N-1}$$

Таким чином, отримали можливість обчислювати значення згладжених отриманих сигналів RSSI. Побудуємо діаграму порівняння згладжених та оригінальних кривих RSSI.

Сигнали RSSI, отримані в реальному часі, і згладжені сигнали RSSI показані на рисунку 3.1.

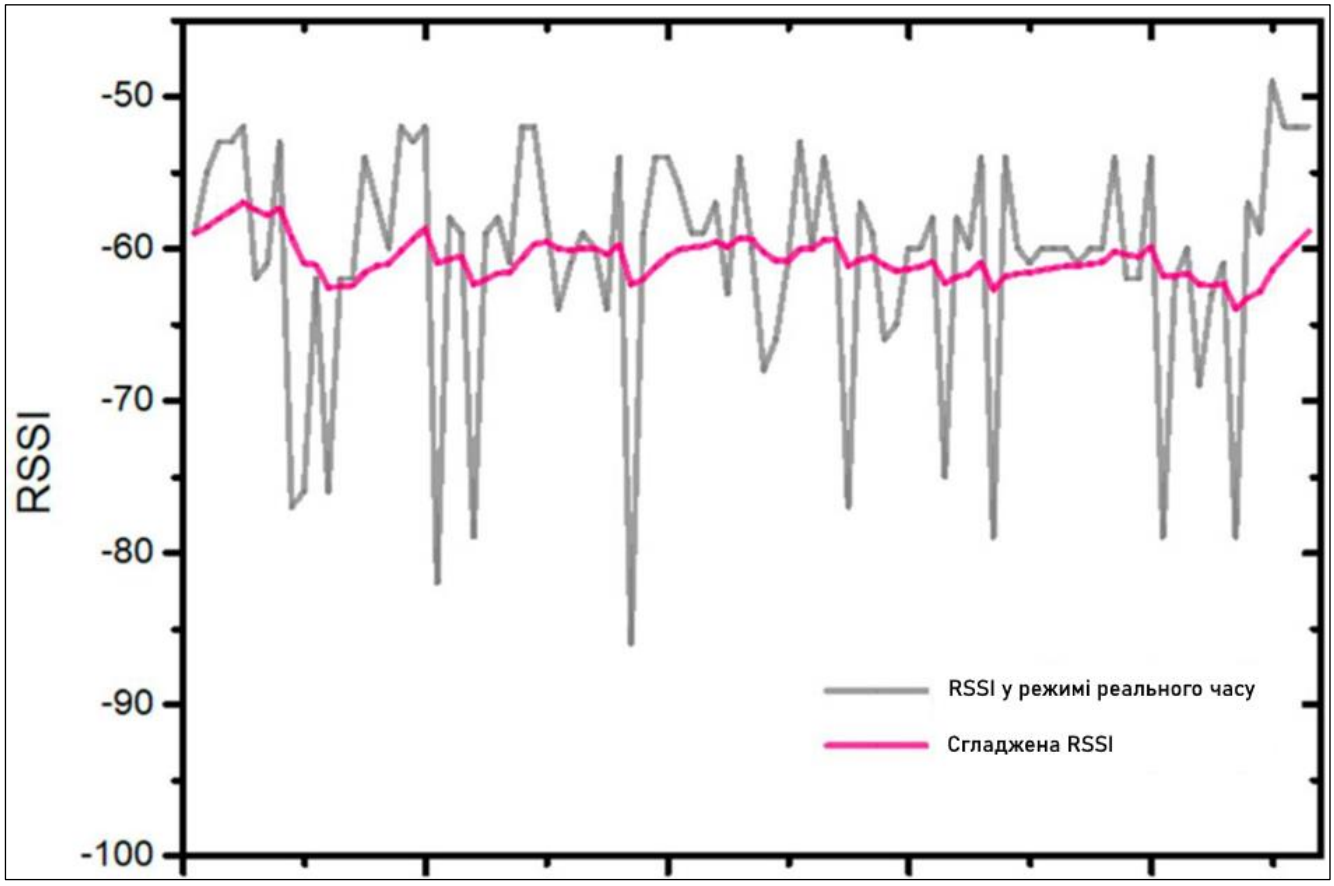

Рисунок 3.1 – Криві згладженої RSSI та RSSI у реальному часі

Сигнали RSSI в реальному часі сильно змінювалися на постійній відстані 1,0 м від маяка BLE через щільне середовище Bluetooth. Виявилося, що фільтр рухомого середнього ефективно зменшив коливання RSSI. У результаті фільтр рухомого середнього було застосовано для всіх наступних фільтрацій RSSI у цій роботі. Таким чином, вдалося значно поліпшити середньоквадратичне відхилення для вимірюваних значень RSSI. Проте варто зазначити, що фільтр згладжування може мати незначний рівень похибки у складних експериментальних умовах, наприклад за наявності низького рівню сигналу або високої інтерференції, викликаної великою кількістю сигналів Bluetooth у мережі середовища виконання.

Значення RSSI зменшується зі збільшенням відстані від маяка BLE, і навколишнє середовище впливатиме на поширення сигналу BLE, де на RSSI

сильно впливають явища багатопроменевого поширення та завмирання. Найбільш використовуваною моделлю для моделювання зв'язку між виміряним RSSI та відповідними відстанями є модель втрат на шляху логарифмічної відстані, як показано нижче:

$$RSSI(d) = RSSI(d_0) - 10n * \log_{10}\frac{d}{d_0} + X_\sigma,$$

де $RSSI(d_0)$ є контрольним значенням RSSI, виміряним на відстані d_0.

Параметр n є показником ступеня втрат на шляху, що вказує швидкість збільшення втрат на шляху, пов'язаних з відстанню. d представляє фактичну відстань до маяка, тоді як RSSI(d) є виміряним значенням RSSI на відстані d. X_σ являє собою випадкову величину Гауса з нульовим середнім, спричинену згасанням тіні [11].

У цій роботі d_0 було встановлено на 1 м, а $RSSI(d_0)$ було виміряним значенням RSSI на відстані 1 м до маяка BLE. Оскільки експериментальне середовище не мало великих перешкод, згасання тіні не очікувалося, і X_σ було встановлено на нуль. За допомогою виміряних значень RSSI та попередньо відомих відстаней показник ступеня втрат на шляху n можна обчислити за таким рівнянням:

$$n = \frac{RSSI(d_0) - RSSI(d)}{10 * \log_{10} 10}$$

Щоб отримати модель розповсюдження BLE, значення RSSI трьох маяків BLE були виміряні на різних відстанях відповідно в одному середовищі. На кожній відстані проводився 2-хвилинний збір RSSI з кожним маяком BLE.

На рисунку 3.2 показано виміряні значення RSSI та теоретичні значення RSSI з моделі розповсюдження сигналу BLE щодо еталонної відстані. Середнє значення n = 1,64 було використано для подальших вимірювань.

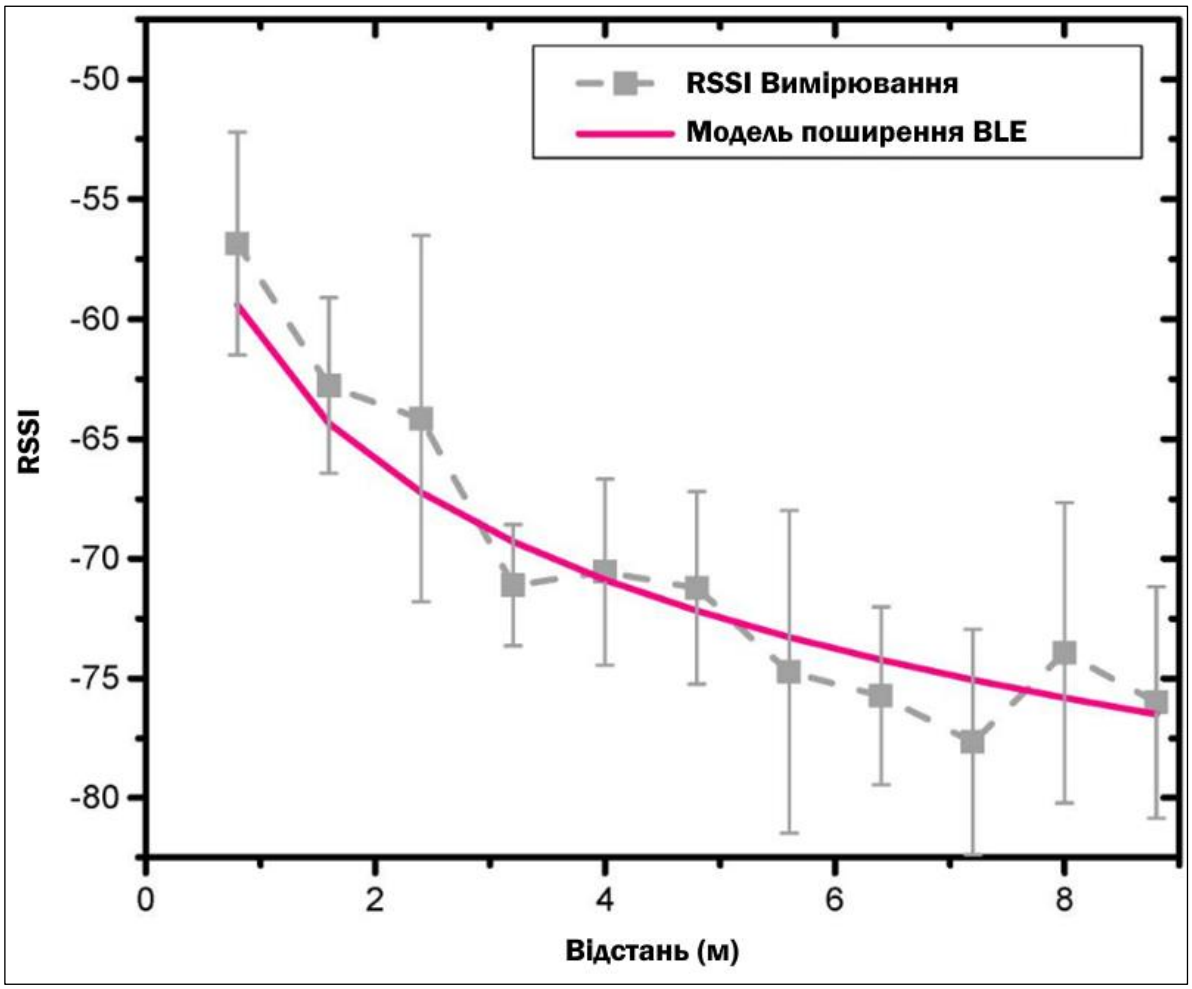

Рисунок 3.2 - Значення вимірювання RSSI та значення моделі поширення BLE

У таблиці 3.1 наведено експоненту втрат на шляху n трьох маяків BLE у щільному середовищі Bluetooth.

Таблиця 3.1 - Обчислення експоненти шляху втрат n

	BLE-1	BLE-2	BLE-3	Average
n	1,76	1,47	1,70	1,64

Після отримання значень RSSI попередньо відомих маяків BLE можна розрахувати відстані до маяків за допомогою моделі розповсюдження BLE. Потім методом триангуляційного алгоритму розраховували положення приймача RSSI. У цьому методі були потрібні принаймні три еталонних маяка BLE із заздалегідь відомими позиціями. У цьому методі кожен опорний маяк утворює навколо себе коло з радіусом відстані до приймача теоретично, а положення невідомого приймача відповідає перетину цих трьох кіл [12]. Відстані на рівнині між і-м відомим радіомаяком BLE і невідомим приймачем можна виразити наступним рівнянням:

$$d_i^2 = (x - x_i)^2 + (y - y_i)^2,$$

$$A\vec{x} = b,$$

$$A = \begin{bmatrix} 2(x_1 - x_n) & 2(y_1 - y_n) \\ \vdots & \vdots \\ 2(x_{n-1} - x_n) & 2(y_{n-1} - y_n) \end{bmatrix},$$

$$\vec{x} = \begin{bmatrix} x \\ y \end{bmatrix},$$

$$b = \begin{bmatrix} x_1^2 - x_n^2 + y_1^2 - y_n^2 + d_n^2 - d_1^2 \\ \vdots \\ x_{n-1}^2 - x_n^2 + y_{n-1}^2 - y_n^2 + d_{n-1}^2 - d_1^2 \end{bmatrix}.$$

Таким чином, рівняння можна розв’язати як задачу найменших квадратів наступним чином:

$$\vec{x} = (A^T A)^{-1}(A^T b).$$

В результаті проведених операцій, отримали загальні рівняння обчислення позиції об’єкту в радіусі масиву маяків за методом триангуляції.

3.3 PDR метод

PDR (Pedestrian dead reckoning) – це техніка відносного позиціонування, яка поширює позицію на основі розрахункового напрямку ходьби та відстані кожного кроку на двовимірній площині. Для PDR існує є три важливі завдання: виявлення кроку, оцінка довжини кроку та визначення курсу.

Згідно з методом PDR, акселерометр, магнітний датчик і гіроскоп, вбудовані в смартфон, будуть об'єднані в інерціальний вимірювальний блок для визначення положення мобільного терміналу за такою формулою:

$$\begin{cases} x_{i+1} = x_i + sl * \cos\alpha_i \,, \\ y_{i+1} = y_i + sl * \sin\alpha_i \end{cases}$$

де *(x, y)* – координати положення;

sl – довжина кроку;

α – кут руху.

Таким чином, алгоритм PDR в основному складається з визначення ходи, довжини кроку та напрямку руху. Крім того, у цій роботі довжина кроку одного об'єкту була прийнята рівною 0,70 м.

Акселерометр, вбудований у термінал, може виявити появу кроку. І вертикальний, і горизонтальний сигнали графіка акселерометра показують кількість зроблених кроків. В ході аналізу алгоритму було проведено вимірювання прискорення, а на основі отриманих даних було побудовано діаграму середнього прискорення алгоритму. На рисунку 3.3 показана форма хвилі середнього прискорення, зафіксованого протягом 20 кроків, де середнє прискорення являє собою середньоквадратичне значення сили прискорення вздовж осей x, y та z.

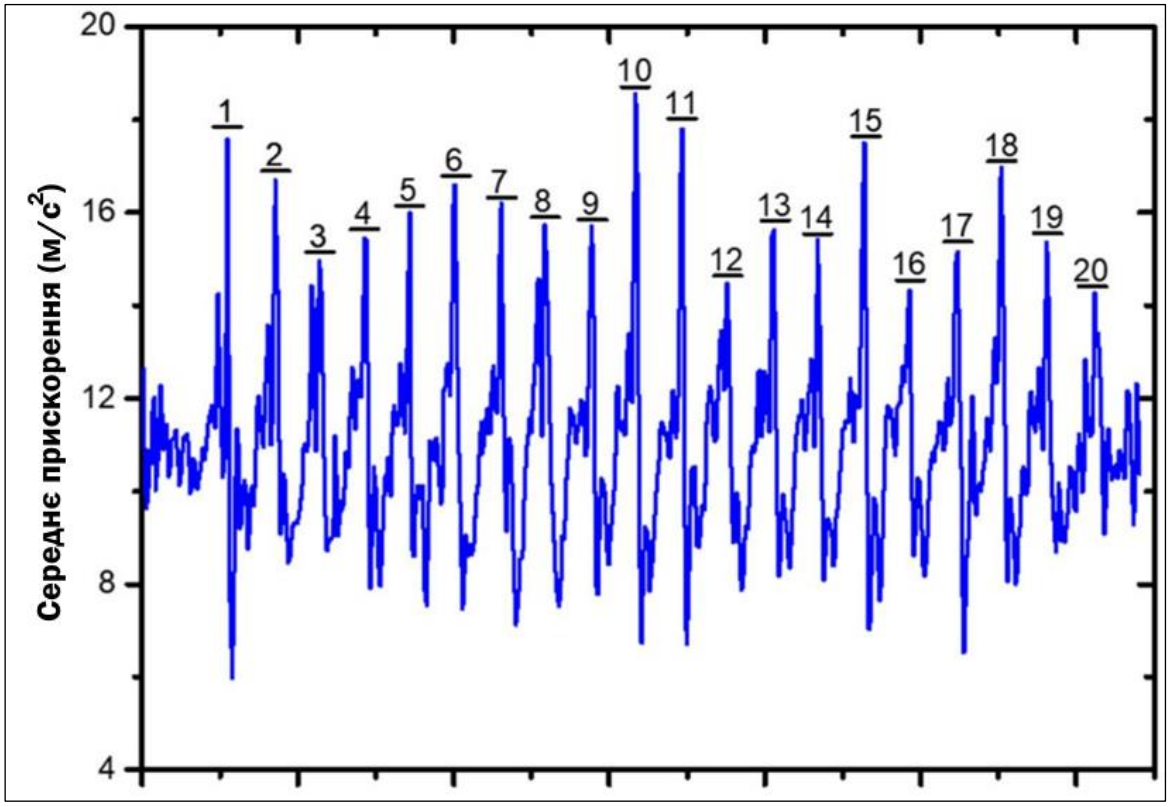

Рисунок 3.3 – Середнє прискорення 20 кроків

Алгоритм виявлення може бути застосований до графіка акселерометра та може бути використано кілька методів ідентифікації. Один із найпростіших способів – визначити сходинку через високий пік. Однак через численні піки та варіацію амплітуди вимірювань високий пік недостатньо надійний. Як альтернатива, вдосконалений метод використовує той факт, що за високим піком завжди слідує низький пік під час зробленого кроку. Таким чином, високий і низький поріг були встановлені для розпізнавання піків, а крок зараховувався на високому піку, за яким слідував низький пік. Тим не менш, ненавмисний рух мобільного телефону може призвести до неправильного підрахунку кроків. Для вирішення проблеми запропоновано метод ідентифікації кроків, який характеризується часом тривалості.

У нашому методі виявлення кроків було ідентифіковано високі піки та низькі піки графіка прискорення, і крок було виявлено, коли всі дві умови були виконані. По-перше, значення різниці високого піку та сусіднього низького піку має перевищувати встановлений поріг. По-друге, час між високим піком і сусіднім

низьким піком має становити від 150 до 400 мс. Щоб знайти найкращий поріг, були проведені вимірювання кроків у різних порогах. Користувач пройшов 20 кроків у кожному тесті, а результати наведено в таблиці 3.2. Виявилося, що чим нижчим був поріг, тим чутливішим був метод підрахунку кроків. коли поріг становив 3, рівень помилок був найнижчим.

Таблиця 3.2 – Вимірювання кроків з різними пороговими значеннями

Threshold	1	1,5	2	2,5	3	3,5	4
Count	41	41	34	25	22	15	8

Датчик орієнтації використовувався для визначення напрямку руху тестового терміналу. Щоб визначити точність курсу, було проведено вісім різних тестів напрямку курсу. У кожному напрямку було зафіксовано 1000 разових вимірювань. Результати випробувань наведено в таблиці 3.3.

Таблиця 3.3 – Значення вимірювань напрямку

Бажаний напрямок	0/360	45	90	135	180	225	270	315
Середнє	-1,25	-44,86	-90,82	-135,38	178,77	224,99	270,86	315,53
Стандартне відхилення	1,22	1,30	1,97	1,43	0,90	1,29	1,22	1,36

Результати показали, що середня похибка становить менше 2°. Можлива причина: об’єкт перемістився вздовж осі x, і очікувалося, що значення x збільшиться на 0,70 м, а значення y не змінилося. Якщо вимірювання курсу виміряно неправильно на 2°, похибка відстані складатиме:

$$x_{i+1} - x_i = 0.70 * \cos(2°) = 0.69999 \text{ м},$$

$$y_{i+1} - y_i = 0.70 * \cos(2°) = 0.00389 \text{ м},$$

$$\varepsilon = \sqrt{(0.70 - 0.70 * \sin(2°))^2 + (0.70 - 0.70 * \cos 2°)^2} = 0.00389 \text{ м}$$

Розрахунки демонструють висновок, що відхилення на 2° призводить лише до похибки відстані менше 1 см, що є прийнятним для подальшого обчислення.

3.4 Розробка гібридного методу

Наступним кроком є розробка методу гібридного внутрішнього позиціонування в поєднанні з алгоритмом трилатерації та PDR обчисленням. Метод гібридного позиціонування зосереджувався на подоланні високої варіації RSSI та тривалого інтервалу часу збирання BLE, що є результатом щільного середовища Bluetooth, і використовував переваги алгоритму трилатерації та PDR обчислення, але уникав можливих недоліків, як показано на рисунку 3.4. Як згадувалося раніше, система позиціонування в приміщенні була розділена на дві фази позиціонування, як показано на рисунку 3.5. Протягом короткого інтервалу для визначення місцезнаходження мобільного пристрою використовувався розрахунок мертвих місць. Однак протягом тривалого інтервалу використовувався алгоритм триангуляції, щоб виправити відхилення мертвого рахунку, спричинене накопичувальними помилками зондування. Щоб досягти цього, фільтр Калмана використовувався як центр злиття для об’єднання двох отриманих позицій з двох алгоритмів.

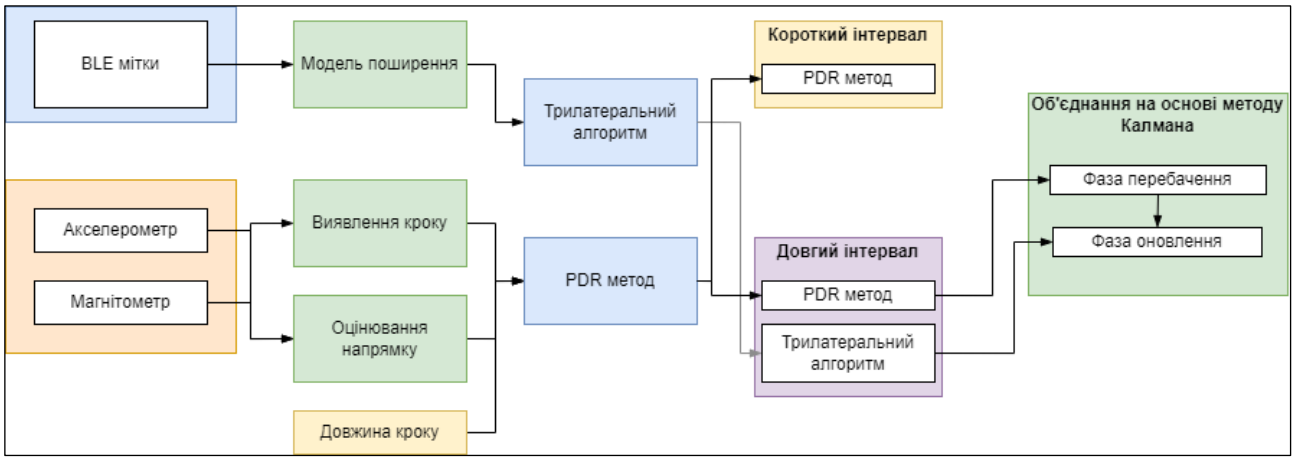

Рисунок 3.4 – Діаграма гібридного алгоритму позиціонування

Наведемо також демонстраційну схему роботи алгоритму, що базується на поєднанні аналізу коротких та довгих інтервалів.

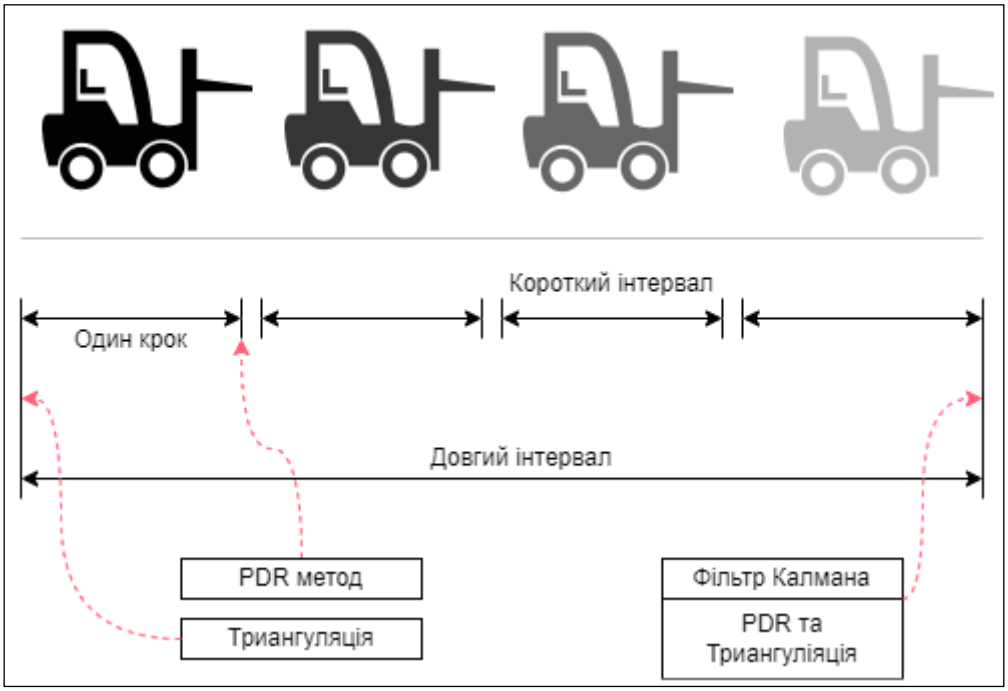

Рисунок 3.5 – Демонстраційне представлення двофазного позиціонування

Таким чином, ми отримали первинне уявлення про принцип роботи гібридного алгоритму, який досліджується у науковій роботі.

Як згадувалося раніше, багато небажаних і відволікаючих сигналів Bluetooth у середовищі впливатимуть на сканування бажаних і специфічних сигналів BLE у щільному середовищ. Крім того, для одного збору сигналу BLE потрібен середній інтервал часу 6,652 с, що означає, що частота одного сигналу маяка BLE, зібраного телефоном, становила 0,15 Гц. У цій внутрішній системі позиціонування було вісім маяків BLE, розподілених у стелі, що означає, що частота одного з восьми сигналів маяків, зібраних телефоном, становила 1,2 Гц. Алгоритм трилатерації брав принаймні три сигнали BLE, щоб обчислити позицію цілі, і знадобилося б принаймні 2,5 с, щоб зібрати достатню кількість сигналів BLE, які подають алгоритм трилатерації, щоб отримати позицію. Однак протягом інтервалу 2,5 с один об’єкт, можливо, переміщається на велику відстань, і результат алгоритму трилатерації втрачає своєчасність, що призводить до неправильної оцінки

позиціонування. Таким чином, розрахунок мертвої точки сприятиме оцінці позиціонування в реальному часі. PDR метод забезпечує високоточне визначення положення на початку вимірювань. Незважаючи на це, ефективність розрахунку безперервно падає через накопичені помилки вимірювання. Отже, необхідно розділити систему позиціонування в приміщенні на дві фази позиціонування в щільному середовищі Bluetooth. Протягом короткого інтервалу для визначення місцезнаходження мобільного пристрою використовувався PDR алгоритм. Протягом тривалого інтервалу використовувався алгоритм триангуляції, щоб виправити відхилення PDR, спричинене накопичувальними помилками фіксацій. Щоб отримати достатню кількість сигналів BLE для алгоритму трилатерації, довгий інтервал було встановлено на 3 с.

Під час фази довгих інтервалів був обраний фільтр Калмана для інтеграції алгоритму трилатерації та методу PDR [13]. У фільтрі Калмана модель стану отримується методом мертвого рахунку, а рівняння стану можна виразити так:

$$\hat{x}_i = (F * \hat{x}_{k-1}) + B * \sum_{i=1}^{N} u_i,$$

де вектор стану $\hat{x}_k = (x, y)^T$ визначає координати цілі від PDR розрахунку;

F і B – одиничні матриці;

N – загальна кількість підрахованих кроків протягом фази довгого інтервалу.

Тоді, так як $u_i = sl * (\cos\alpha_i, \sin\alpha_i)^T$ є зміною координати, отриманою з і-го кроку. У цьому випадку sl – довжина кроку, а α_i – напрямок руху. Таким чином, зміна координат в результаті роботи PDR методу протягом фази довгого інтервалу може бути виражена як:

$$u_k = \sum_{i=1}^{N} sl * \begin{bmatrix} \cos\alpha_i \\ \sin\alpha_i \end{bmatrix}.$$

Модель вимірювання отримана методом трилатерації. Модель вимірювання може бути виражена як:

$$\hat{z}_k = H\hat{x}_k,$$

де $\hat{z}_k = (x_k, y_k)^T$ визначає координату цілі з трилатераційного позиціонування;

Н є ідентичною матрицею.

Коли часовий інтервал досягає 3 с, фільтр Калмана оновлює положення за допомогою прогнозу та фази оновлення. На етапі прогнозування позицію можна передбачити наступним чином:

$$\hat{x}_{\bar{k}} = \hat{x}_{k-1} + \sum_{i=1}^{N} sl * \begin{bmatrix} \cos\alpha_i \\ \sin\alpha_i \end{bmatrix},$$

$$P_{\bar{k}} = FP_{k-1}F^T + Q,$$

де Q – коваріація шуму від PDR методу.

На етапі оновлення позиції з обох методів можна об’єднати таким чином:

$$K_k = P_{\bar{k}} * H^T(HP_{\bar{k}}H^T + R)^{-1},$$

$$\hat{x}_k = \hat{x}_{\bar{k}} + K_k(\hat{z}_k - H\hat{x}_{\bar{k}}),$$

$$P_k = (I - K_tH)P_{\bar{k}},$$

де K – підсилення Калмана;

I – матриця ідентичності;

R – коваріація шуму від методу позиціонування трилатерації;

$\hat{x}_k$ – об’єднане розташування.

Було експериментально перевірено різні значення для R і Q, і найкращі результати були отримані при використанні R = 4 і Q = 0,1.

3.5 Виконання експериментального дослідження

Щоб оцінити продуктивність запропонованого алгоритму, ми провели експерименти всередині приміщення, схему якого зображено на рисунку 3.6. Розмір площі 5,6 × 8,8 м². На площі було 15 Bluetooth пристроїв, які безперервно транслювали свої сигнали, для імітації щільного середовища Bluetooth, яке максимально наближено до реального випадку використання системи. Вісім терміналів BLE було розміщено в квадратній зоні, зображеній на рисунку 3.6. Кожен з маяків розміщувався в стелі, на висоті 3 м. Для виконання експерименту було встановлено маршрут, який необхідно було подолати об'єкту з Bluetooth-пристроєм (для тестування використовувався смартфон). Відстань між двома маяками становила від 4 до 6 м, так що площа була повністю покрита сигналами BLE.

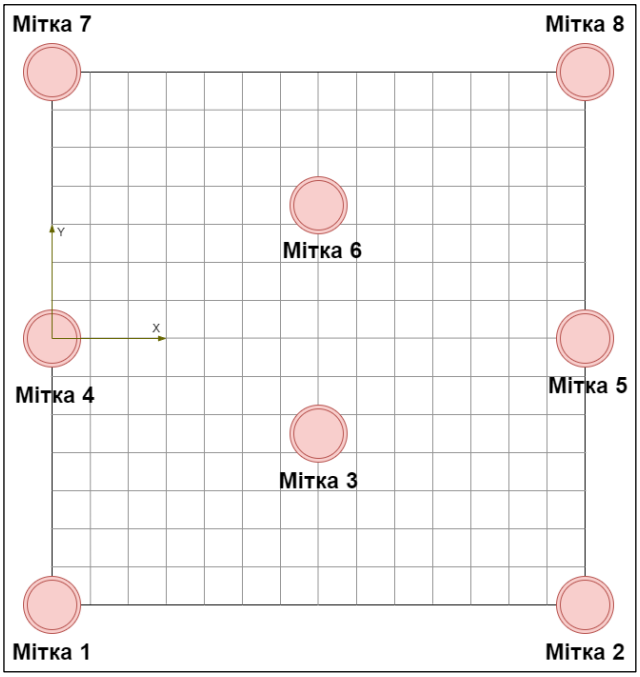

Рисунок 3.6 – Схема приміщення експериментального дослідження

Оскільки застосовувався метод двовимірної трилатерації, вертикальна відстань Δh між радіомаяком і мобільним пристроєм була потрібна для отримання горизонтальної відстані dh між маяком і мобільним пристроєм.

Отже, горизонтальна відстань dh в термінах виміряної відстані dm і вертикальної відстані Δh між радіомаяком і мобільним визначається як:

$$d_h = \sqrt{d_m^2 - \Delta h^2}.$$

Наступним кроком оцінимо продуктивність PDR методу, трилатерального та гібридного методу. Варто зазначити, що PDR метод вимагає наявність координат початкового положення. У кожному тесті об’єкт (людина), який мав смартфон, продовжував рухатися по визначеному шляху довжиною 28,8 м у зоні, охопленій модулями BLE. В експерименті було перевірено ефективність позиціонування трьох методів у щільному середовищі Bluetooth.

В експерименті з використанням PDR методу магнітний датчик і акселерометр смартфона використовувалися для позиціонування. PDR метод повинен отримати свою початкову координату на початку позиції, оскільки він не може самостійно обчислити початкову координату. Розрахунковий шлях і фактичний пройдений шлях обчисленого PDR показані на малюнку 3.7.

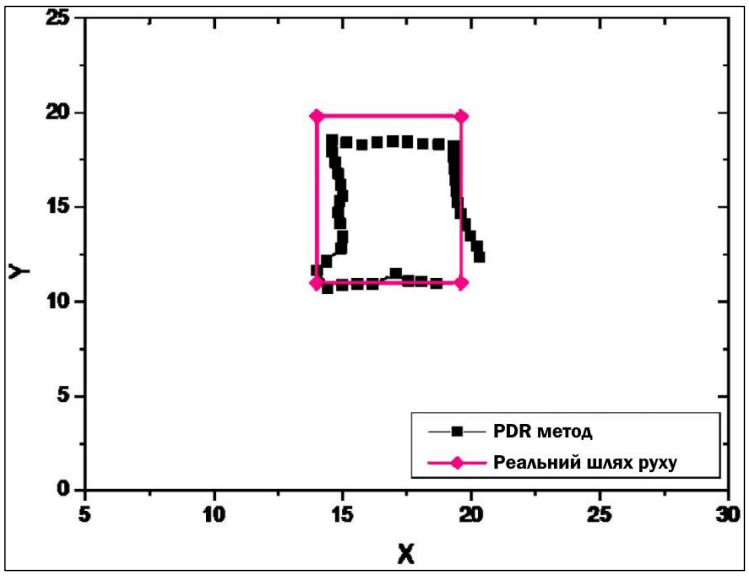

Рисунок 3.7 – Результат роботи PDR методу

Розрахунковий шлях за методом PDR був подібним до фактичного пройденого шляху. Оскільки метод мертвого розрахунку вимагає початкової

координати для обчислення шляху, сукупна похибка збільшувалася зі збільшенням оціненого шляху, і більша похибка з’являлася в кінцевій координаті місця та фактичній координаті.

Наступним етапом є тестування трилатераційного методу. Зміна інтенсивності сигналу Bluetooth стає більшою, а інтервал сканування модулів BLE стає довшим у щільному середовищі Bluetooth. Це мало великий вплив на точність позиціонування методу трилатерації. Розрахунковий шлях і фактичний пройдений шлях трилатерації показано на рисунку 3.8.

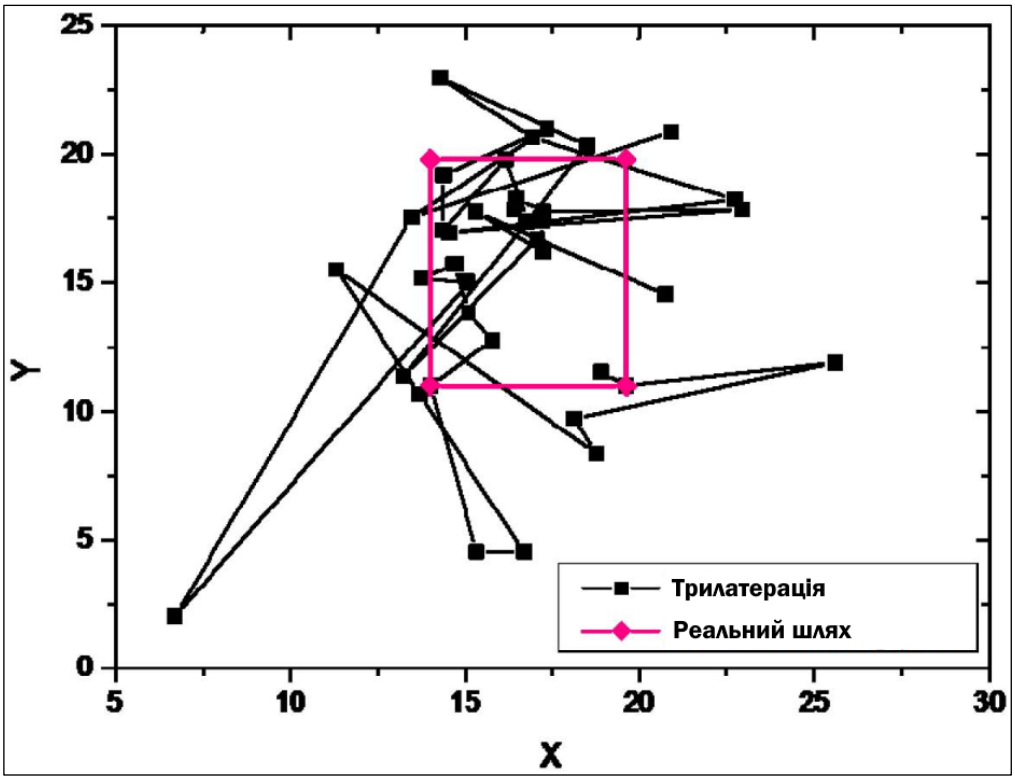

Рисунок 3.8 – Результат роботи трилатераційного методу

Розрахункова траєкторія експерименту з трилатерацією сильно відрізнялася від фактичної пройденої траєкторії. Позиціонування BLE коливається внаслідок швидкого загасання або ефекту багатопроменевості в приміщенні. Через швидке згасання та багатопроменевість у середовищі експерименту метод трилатерації не точно визначив фактичне розташування. Отриманий шлях експерименту трилатерації дуже відрізнявся від фактичного шляху. Результати експерименту

показали, що методом трилатерації, заснованому на Bluetooth, важко самостійно завершити позиціонування в приміщенні з меншою похибкою.

Наступним кроком виконаємо тестування гібридного методу. Гібридний метод має переваги PDR методу та методу трилатерації. Він може не тільки гарантувати, що оцінений шлях має невелику похибку з фактичним пройденим шляхом, але також може постійно компенсувати оцінений шлях, щоб запобігти великій сукупній похибці. Оцінений шлях і фактичний пройдений шлях гібридного методу показано на рисунку 3.9.

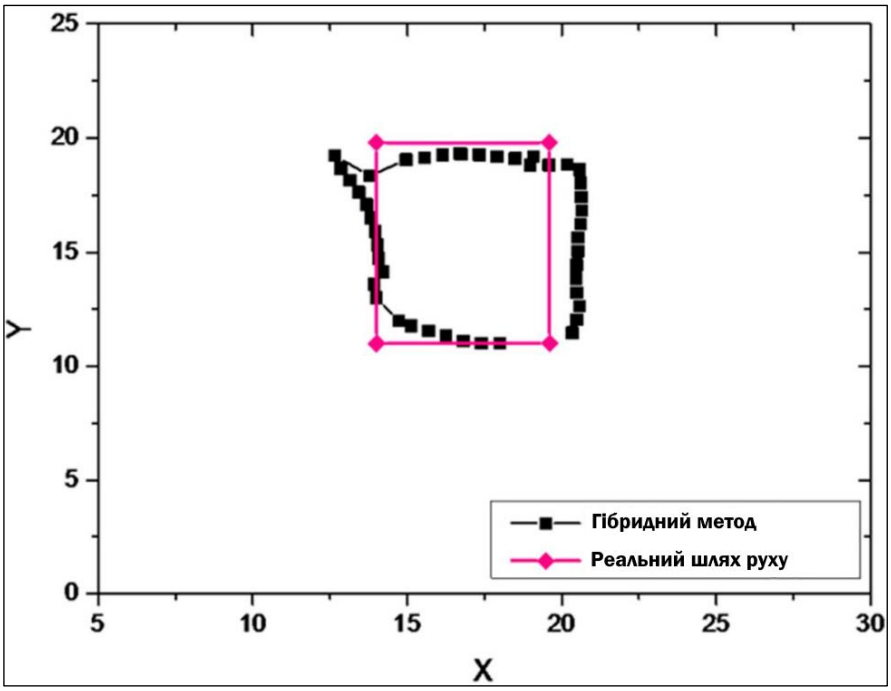

Рисунок 3.9 – Результат роботи гібридного методу

Оцінений шлях гібридного методу в основному збігався з фактичним пройденим шляхом. Згідно з передумовою оцінки шляху, розрахованого методом PDR, гібридний метод на основі Калмана ефективно зменшує оцінену помилку шляху та досягає кращих результатів позиціонування.

Згідно з результатами експерименту, розрахунковий шлях PDR методу схожий на фактичний пройдений шлях, і може реалізувати позиціонування в приміщенні без додаткового обладнання. Однак для визначення місцезнаходження

потрібні початкові координати, а його кумулятивна похибка зростає разом із довжиною шляху ходьби.

Метод трилатерації потребує апаратної підтримки модуля BLE. Крім того, індивідуальний метод трилатерації легко піддається впливу щільного середовища Bluetooth і має великий вплив на його точність позиціонування. Тому для отримання точніших координат позиціонування потрібно обчислювати більшу кількість даних.

Гібридний метод поєднує в собі переваги PDR методу та методу трилатерації. Оцінений шлях гібридного методу в основному такий самий, як фактичний пройдений шлях. Гібридний метод не тільки має переваги плавної оцінки шляху, але також має переваги корекції шляху розташування в реальному часі, що мінімізує помилку позиціонування.

Додатково зробимо висновок відносно перспектив та можливостей розвитку отриманих результатів дослідження. Наступним кроком є інтеграція рішень на основі ML (machine learning) та AI (artificial intelligence), що дозволили б суттєво збільшити точність позиціонування, а також швидкість виконання обчислень за рахунок використання даних попередніх фіксацій. Крім того, штучний інтелект дозволив би істотно покращити результати обчислень та точність фіксацій у складному середовищі, наприклад, при низькій якості покриття або за умов високої інтерференції сигналів інших пристроїв, що знаходиться всередині приміщення.

4 АРХІТЕКТУРА ТА ПРОЄКТУВАННЯ ПРОГРАМНОЇ СИСТЕМИ

4.1 UML проєктування

Невід’ємною частиною процесу розробки комплексної програмної системи є етап проєктування. На цьому етапі визначаються вимоги до майбутньої системи, в тому числі вимоги першого випуску та перспективи подальшого розвитку продукту.

Проєктування дозволяє заздалегідь продумати архітектуру майбутньої системи, знизивши вартість її безпосередньої розробки та кількість потенційних слабких місць етапу імплементації. Крім того, проєктування надає можливість заздалегідь сформувати будову кожного окремого модуля комплексної програмної системи та зробити вибір технологій розробки.

Для проєктування програмної системи у кваліфікаційній роботі використовуватиметься UML-моделювання, яке є загальноприйнятим та широко поширеним стандартом проєктування програмного забезпечення.

UML-моделювання (Unified Modeling Language) - це стандартний мовний інструментарій для візуального опису і проектування складних систем. Він дозволяє представляти різні аспекти системи в зрозумілій, демонстраційній формі за допомогою діаграм і спеціальних символів, що дозволяє створювати абстрактні моделі системи, що їх легко зрозуміти та змінювати. UML є універсальною специфікацією для проектування програмного забезпечення, яка дозволяє розробляти високоякісні і надійні системи та забезпечує зручний інтерфейс для спілкування між розробниками та замовниками проектів [14].

Для формування загального розуміння про варіанти використання системи що проєктується, побудуємо Use Case діаграму (діаграма варіантів використання). Зважаючи на те, що модуль серверної частини охоплює більшість вимог до функціональності майбутнього продукту, визначимо основні сценарії використання системи та відобразимо їх на UML-діаграмі.

Розроблену діаграму варіантів використання програмної системи продемонструємо на рисунку 4.1.

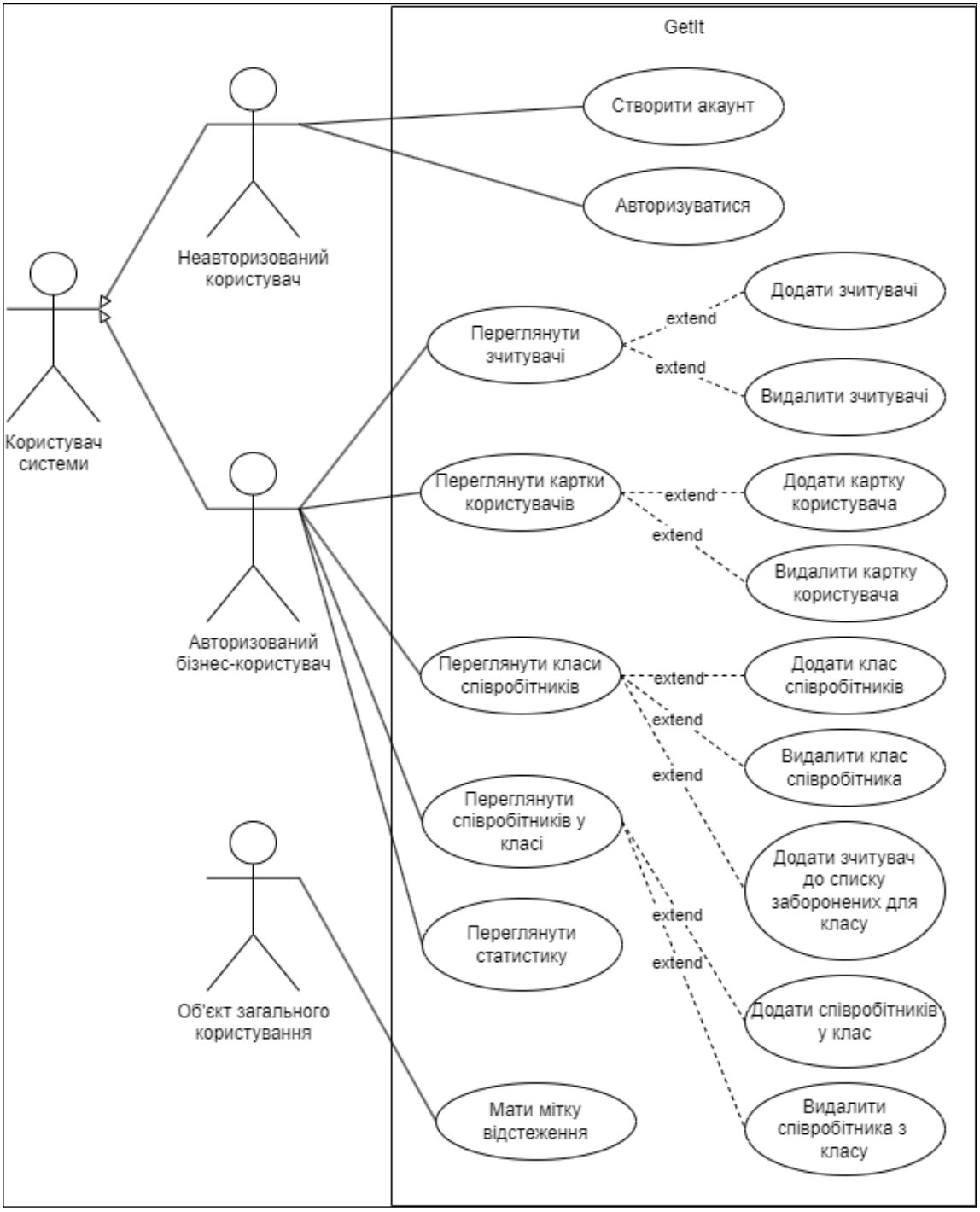

Рисунок 4.1 – Діаграма прецедентів

Представлена діаграма складається з наступних елементів: актори (особи, що взаємодіють з системою), варіанти використання, різні типи зв’язків між компонентів (асоціація, узагальнення та розширення). Зв’язки типу «Асоціація» позначаються цільною прямою лінією та відображають взаємодію між акторами та доступними для них варіантами використання. «Узагальнення» об’єднують акторів

за принципом розширення. У свою чергу «Розширення» показують додаткові можливі варіанти використання.

Важливим кроком проєктування програмної системи є формування уявлення про принципи взаємодії модулів системи з іншими компонентами. Найбільш оптимальним способом ілюстрації в даному випадку є діаграма компонентів.

Розробимо діаграму та покажемо її на рисунку 4.2.

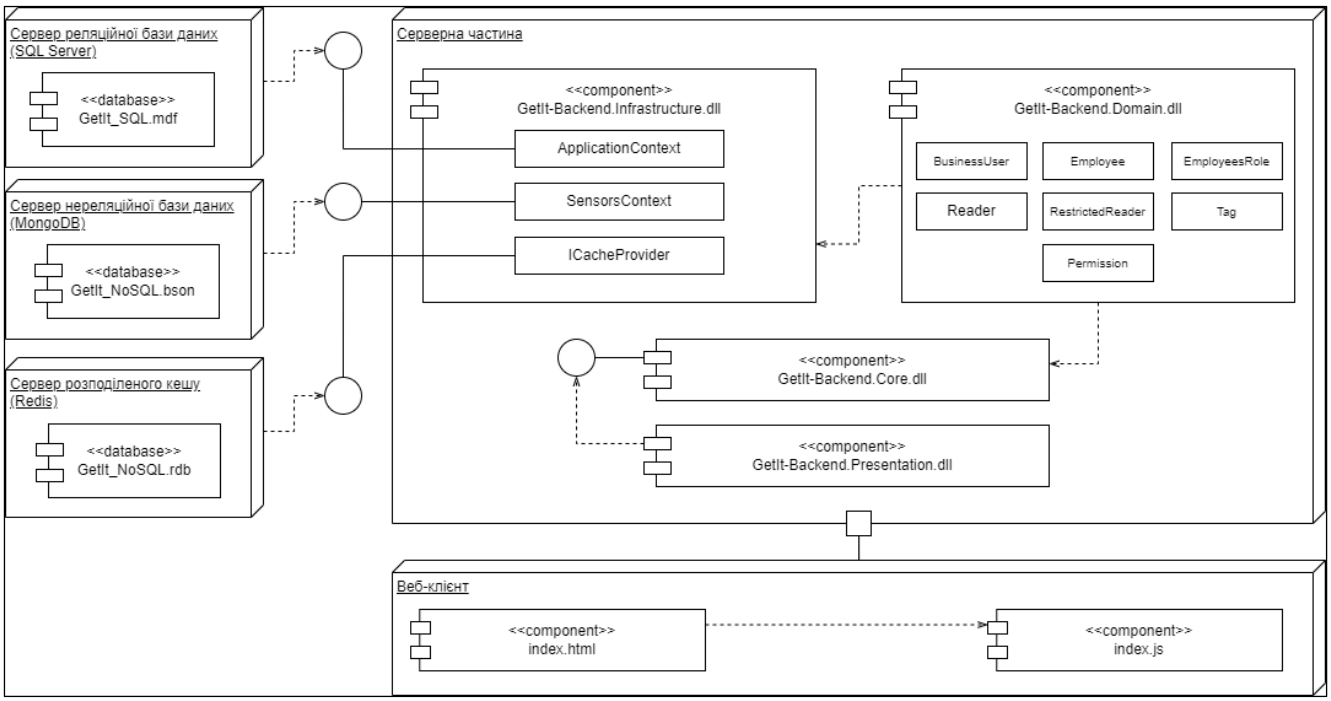

Рисунок 4.2 – Діаграма компонентів

На наведеній діаграмі позначено два модулі: серверну частину та клієнтську веб-частину. Зазначимо, що серверна сторона в свою чергу поділяється на 3 субкомпоненти: сервер реляційної бази даних, сервер нереляційної бази даних (документо-орієнтованої) та сервер розподіленого кешування.

Рішення щодо побудови архітектури серверної частини було прийняте на основі досліджень щодо найкращих практик розробки архітектури програмного забезпечення [15]. Така архітектура покликана закласти фундамент системи, що здатна швидко і з мінімальними затратами ресурсів масштабуватися, для обслуговування більшої кількості користувачів та значно більшої кількості даних.

Дані між сховищами розподілятимуться наступним чином:

– у реляційній базі даних зберігатимуться структуровані і обмежені в кількості дані: про користувачів, дані що стосуються внутрішньої діяльності системи, дані про транзакції;

– у нереляційній базі даних зберігатимуться дані про фіксації та переміщення;

– у розподіленому кеші будуть зберігатися найбільш часто використовані дані, що дозволить значно прискорити роботу системи.

Завдяки вище описаним рішенням щодо побудови структури сховищ даних програмна система буде здатна працювати з великою кількістю користувачів та даних, а її можливості масштабування дозволять оптимізувати роботу системи в майбутньому.

Центральним елементом діаграми є серверна частина програмної системи. Вона виконує функцію основного обчислювального компоненту, який взаємодіє з іншими модулями за протоколом HTTPS. Серверна частина обов’язково повинна мати багатошарову архітектуру, що сприятиме можливостям подальшої підтримки, обслуговування та розширення системи. Спираючись на принципи побудови багатошарової архітектури наведені у [16], позначимо на діаграмі внутрішній устрій серверного застосунку.

Серверна частина та бази даних здійснюють обмін даними за протоколом TCP/IP, а з розподіленим кешем сервер взаємодіє за протоколом RESP (Redis serialization protocol). Фізично бази даних та серверний застосунок мають бути розташовані на різних обчислювальних вузлах, що також позитивно вплине на можливості обслуговування системи та покращить рівень її відмовостійкості.

Наступним кроком варто розглянути структуру клієнтської веб-частини більш детально, для чого пропонується побудувати діаграму станів.

Діаграма станів є важливим інструментом для моделювання поведінки програмного додатку. Вона дозволяє детально розібратися в життєвому циклі додатку та проаналізувати його стани. На діаграмі використовуються різні елементи, такі як стани, зв’язки між станами та сторожові умови. Отриману діаграму позначимо на рисунку 4.3.

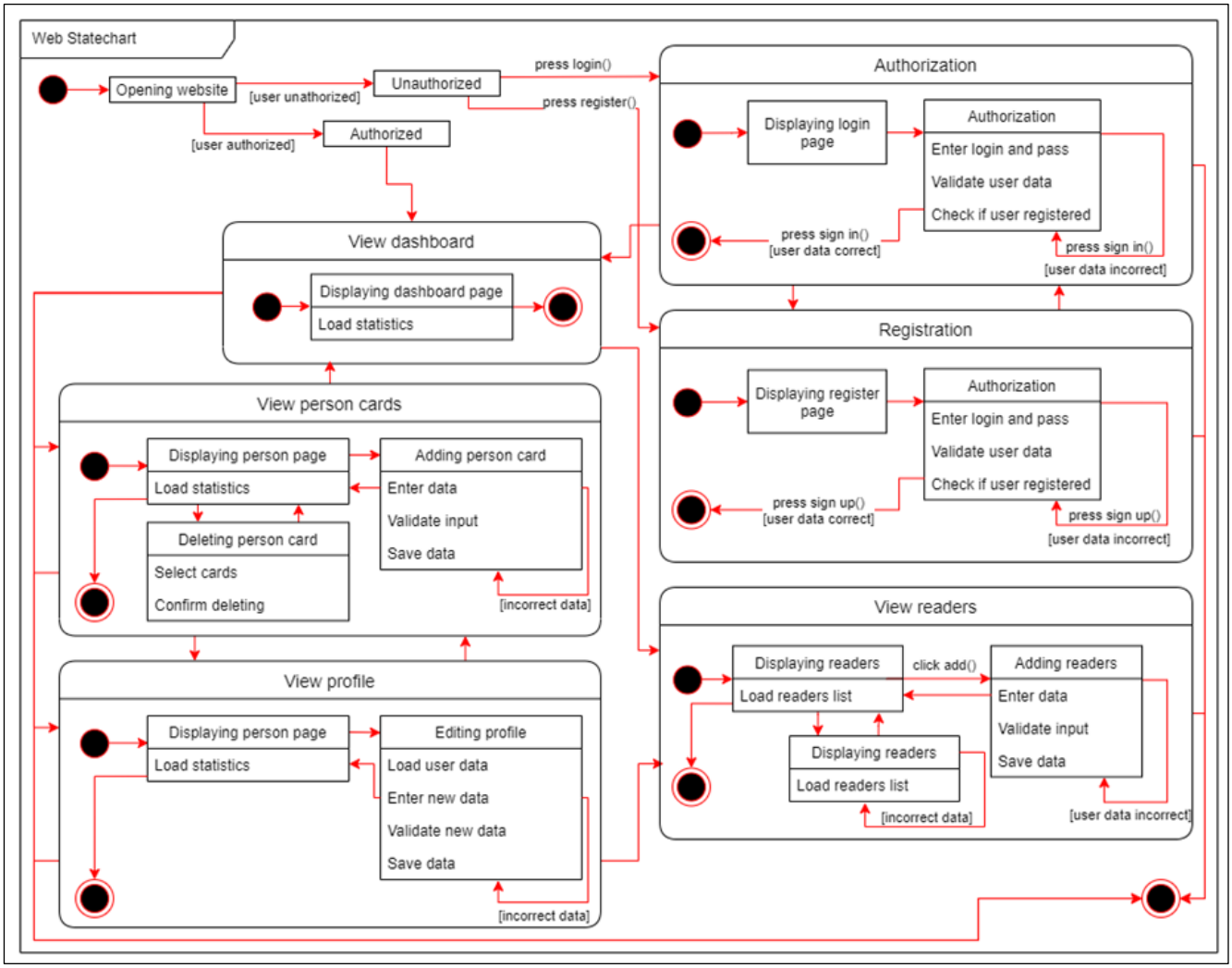

Рисунок 4.3 – Діаграма станів веб-частини системи

Отримана діаграма наочно демонструє всі можливі способи навігації та надає детальну інформацію про можливі дії на різних сторінках, включаючи процес авторизації, валідації і т. д.

Ця діаграма станів дозволяє відобразити всі можливі стани системи, їх параметри та можливі переходи між ними. Крім того, ця діаграма допомагає проілюструвати всі доступні користувачеві дії на конкретній сторінці додатку.

4.2 Проєктування загальної архітектури програмної системи

Згідно до вхідних даних кваліфікаційної роботи, а також вимог сформованих у пункті 2, необхідно спроєктувати та розробити комплексну програмну систему,

що складається з 4 модулів: серверної частини, клієнтської веб-частини, мобільної частини, IoT-пристроїв.

Для побудови загального уявлення про програмну систему та її архітектуру необхідно розробити діаграму розгортання. Зобразимо результат проєктування на рисунку 4.4.

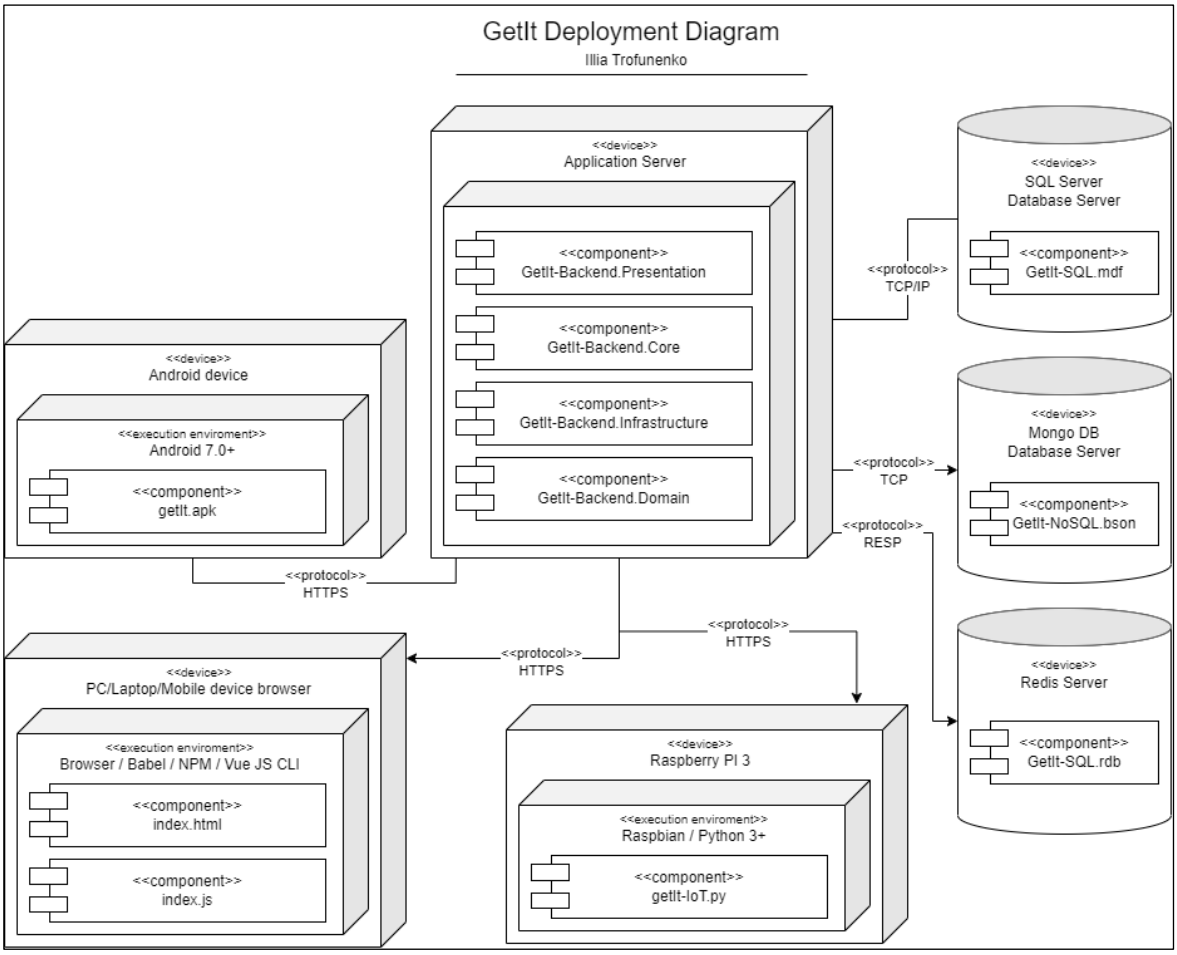

Рисунок 4.4 – Діаграма розгортання системи (високорівнева архітектура системи)

Наведена вище діаграма, детально описує принципи взаємодії компонентів системи. На діаграмі позначені обчислювальні одиниці, на яких будуть розгорнуті розроблені програмні застосунки, а також умовний опис їх фізичної структури. Взаємодії між модулями позначені зв’язками, на кожному з яких визначено протокол обміну даними.

Центральним вузлом загальної програмної системи є серверна частина, яка виконує ключові обчислення та втілює бізнес-логіку системи. Важливо зазначити,

що завдяки багатошаровій архітектурі серверного рішення, існує можливість розподілення екземплярів застосунків на декілька фізичних вузлів, що дозволить підвищити пропускну здатність системи та знизити затримки під час роботи. Проєктуванням архітектури серверної частини передбачено можливість масштабування системи у випадку, якщо реальні обсяги даних та кількість користувачів перевищать розрахункові.

При побудові архітектури серверної частини важливим критерієм було визначення модулів як слабко зв'язаних одиниць, що також надасть додаткові можливості обслуговування системи. Таким чином, система може бути розширена та модифікована в майбутньому з урахуванням потреб користувачів.

Наступним важливим елементом є сервери баз даних. Відповідно до архітектури, описаної у пункті 4.1, система матиме декілька серверів баз даних, які фізично будуть розташовані на окремих серверах, що надасть вже раніше описані переваги, а також підвищить рівень надійності, безпеки та відмовостійкості. Програмні модулі, які працюватимуть з базами даних будуть побудовані на абстракціях, що за необхідності дозволить швидко змінити СКБД у майбутньому.

Останнім ключовим компонентом загальної архітектури системи є фізичні IoT-пристрої. Такі пристрої поділятимуться на 2 типи: мітки та термінали, які фіксуватимуть сигнали міток, за допомогою чого сервер зможе здійснювати позиціонування об'єктів. Задля оптимізації, пристрої окрім збору даних з датчиків будуть обробляти проміжні результаті обчислень, наприклад сила сигналу та потенційну відстань від зчитувача.

Додатково, у серверній частині буде виділено компонент, який здійснюватиме фінальну обробку та надсилання даних до основного серверу за протоколом HTTPS, або напряму – у базу даних, в залежності від сценарію обробки. В якості технологій для міток буде використано BLE та RFID підходи, для різних сценаріїв використання. Взаємодія між основною серверною частиною та сервером IoT-пристроїв буде також побудована на абстракціях, що дозволить їх замінювати або розширювати іншими технологіями розумних девайсів у майбутньому.

Для реалізації та забезпечення максимального доступного та ефективного клієнтського досвіду використовуватимуться веб- та мобільна частини, що дозволять отримувати та надсилати дані до сервера, спираючись на запити та дії користувача. Для реалізації цієї взаємодії використовується зашифрований протокол HTTPS.

Підводячи підсумок щодо розробленої діаграми архітектури, маємо зазначити, що ключовим вузлом системи є серверний модуль, який здійснює обчислення та виконує бізнес-логіку, взаємодіючи з іншими підсистемами за різними типами протоколів.

4.3 Проєктування архітектури систем зберігання даних

Важливим кроком проєктування програмної системи є розробка архітектури систем зберігання даних. Для виконання цієї задачі необхідно прийняти рішення щодо обрання технологій та підходів до баз даних, з якими взаємодіятиме система.

Зважаючи на вимоги до системи, а саме: масштабованість, відмовостійкість, швидкість та ефективність, здатність зберігати великі обсяги дані, необхідно побудувати архітектуру, що задовільнить описаним параметрам.

Користуючись рішеннями, прийнятими у пункті 4.1 спроєктуємо архітектуру систем зберігання даних.

Для проєктування реляційної бази даних скористаємося методологією логічного проєктування, та використаємо ER-модель (модель «сутність-зв’язок»).

Модель Entity-Relationship (ER-модель) – це методологія описування структури даних, яка використовує графічні символи для представлення сутностей, атрибутів та зв'язків між ними. ER-модель використовується для проектування баз даних і допомагає управляти складністю бази даних, полегшує розуміння взаємозв'язків між даними та дозволяє перевірити правильність структури даних перед її реалізацією.

У ER-моделі сутність - це об'єкт або поняття, що може бути ідентифікованим

та описаним даними. Атрибут - це характеристика сутності, яка містить певну інформацію про сутність. Зв'язок - це взаємозв'язок між двома або більше сутностями, який описує, як одна сутність пов'язана з іншою.

ER-модель дозволяє розглядати дані як набір сутностей та зв'язків між ними, що дозволяє розуміти відносини між даними та їх структуру. Вона також дозволяє визначати правила цілісності даних, що допомагає забезпечити, що дані введені в базу даних будуть коректні та надійні.

ER-модель є важливим інструментом для проектування баз даних, оскільки дозволяє розглядати дані з точки зору взаємозв'язків між ними, що дозволяє зробити оптимальний вибір для збереження даних та забезпечення їх ефективного використання.

Використовуючи дані, отримані на етапі аналізу предметної області, а також припущення зроблені на етапі проєктування побудуємо ER-діаграму реляційної бази даних.

Проаналізуємо та визначимо сутності у системі, згідно предметної області розроблюваної системи: компанія (бізнес-користувач), приміщення, мітка, клас міток, зчитувач, зона приміщення, дозвіл.

Наступним кроком визначимо типи зв’язків між сутностями. У нашому випадку використовуватимемо зв’язки типу «1:Б» (1 до багатьох) та «Б:Б» (багато до багатьох). Зауважимо, що другий тип визначається як два зв’язки «1:Б» та проміжної таблиці.

На основі отриманих сутностей визначимо їх атрибути та побудуємо ER-діаграму (див. рис. 4.5).

Як було зауважено раніше, для реалізації зв’язків «багато до багатьох» існує необхідність створення проміжних сутностей.

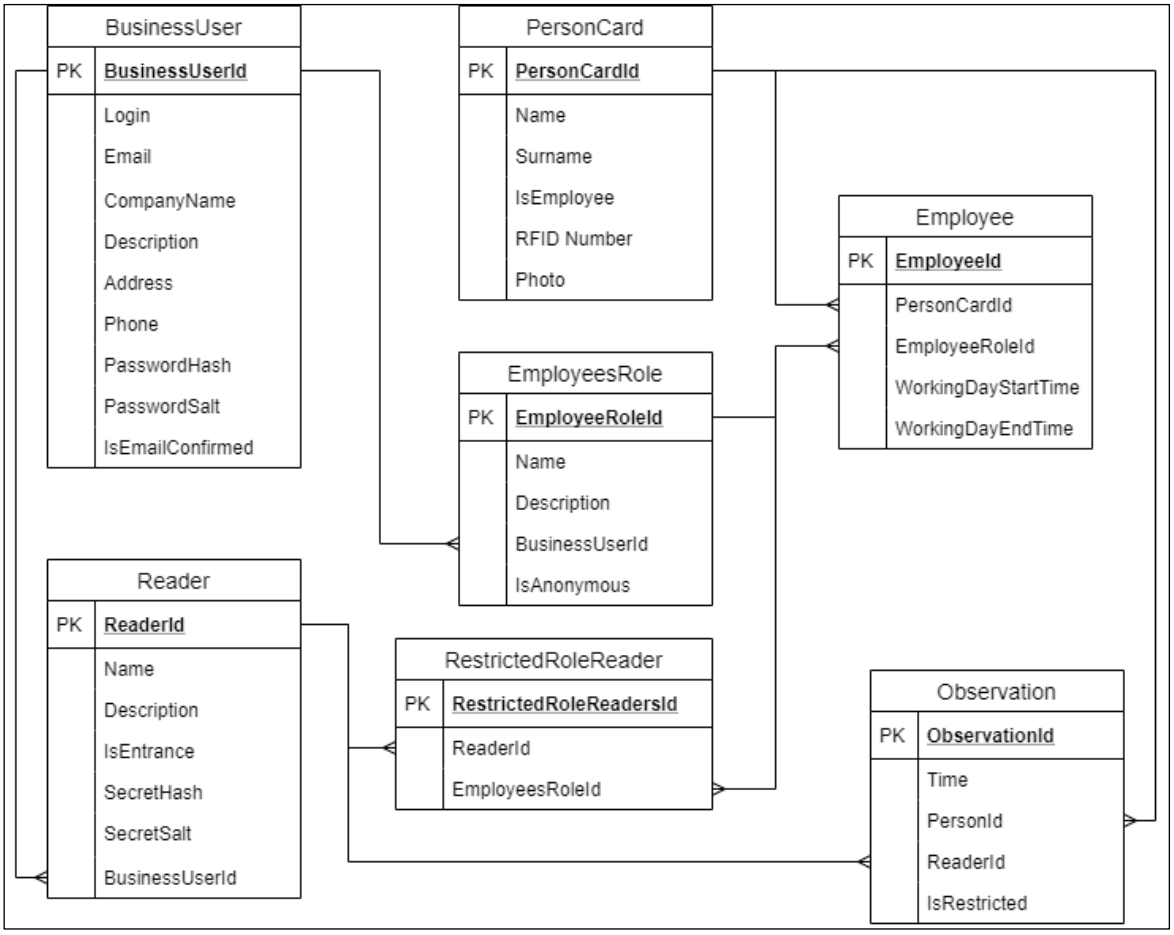

Рисунок 4.5 – ER-діаграма логічної моделі бази даних системи

Зазначимо, що наведена діаграма бази даних відповідає третій нормальній формі, що включає умови виконання першої, другої та третьої нормальних форм, а саме:

– перша нормальна форма (1НФ) вимагає, щоб кожен атрибут таблиці мав тільки одне значення та значення кожного атрибута були атомарні (недільні). Інакше кажучи, 1НФ забезпечує, щоб жоден атрибут не містив повторювану групу значень;

– друга нормальна форма (2НФ) вимагає, щоб кожний неключовий атрибут таблиці пов'язувався тільки з одним ключем (первинним або складним). Іншими словами, 2НФ виключає наявність залежностей між атрибутами, що не пов'язані з ключем;

– третя нормальна форма (3НФ) вимагає, щоб кожен неключовий атрибут таблиці пов'язувався тільки з первинним ключем, а не з іншими неключовими

атрибутами. Іншими словами, 3НФ забезпечує, щоб в таблиці не було транзитивних залежностей між атрибутами.

4.4 Проєктування UI та UX клієнтських частин системи

Під час проектування майбутньої системи було створено інтерактивні мокапи для веб- та мобільних частин системи з метою спрощення процесу розробки.

При розробці мокапів інтерфейсу системи було застосовано принципи створення зручного, зрозумілого та функціонального інтерфейсу, зокрема, евристичні принципи Якоба Нільсена [17].

У якості базового підходу до розробки графічних інтерфейсів використовувався Design Thinking Process framework (процес дизайн-мислення). Дизайн-мислення є творчим підходом до вирішення проблем, який широко використовується в дизайні продуктів та послуг. Це процес, орієнтований на користувача, який фокусується на розумінні потреб користувачів і створенні рішень, які задовольняють ці потреби. Рамка процесу дизайн-мислення зазвичай складається з наступних п'яти етапів: співпереживання, визначення, генерація ідей, прототипування, тестування.

Розроблені прототипи дозволяють натискати та навігуватися між сторінками майбутньої системи. Для забезпечення єдиного стилю графічного інтерфейсу було створено кольорову схему проекту та набір дизайн-ресурсів, таких як іконки та піктограми.

Для демонстрації розробленого дизайну клієнтської веб-частини наведемо деякі приклади сторінок. На рисунку 4.6 зображено вигляд сторінки авторизації.

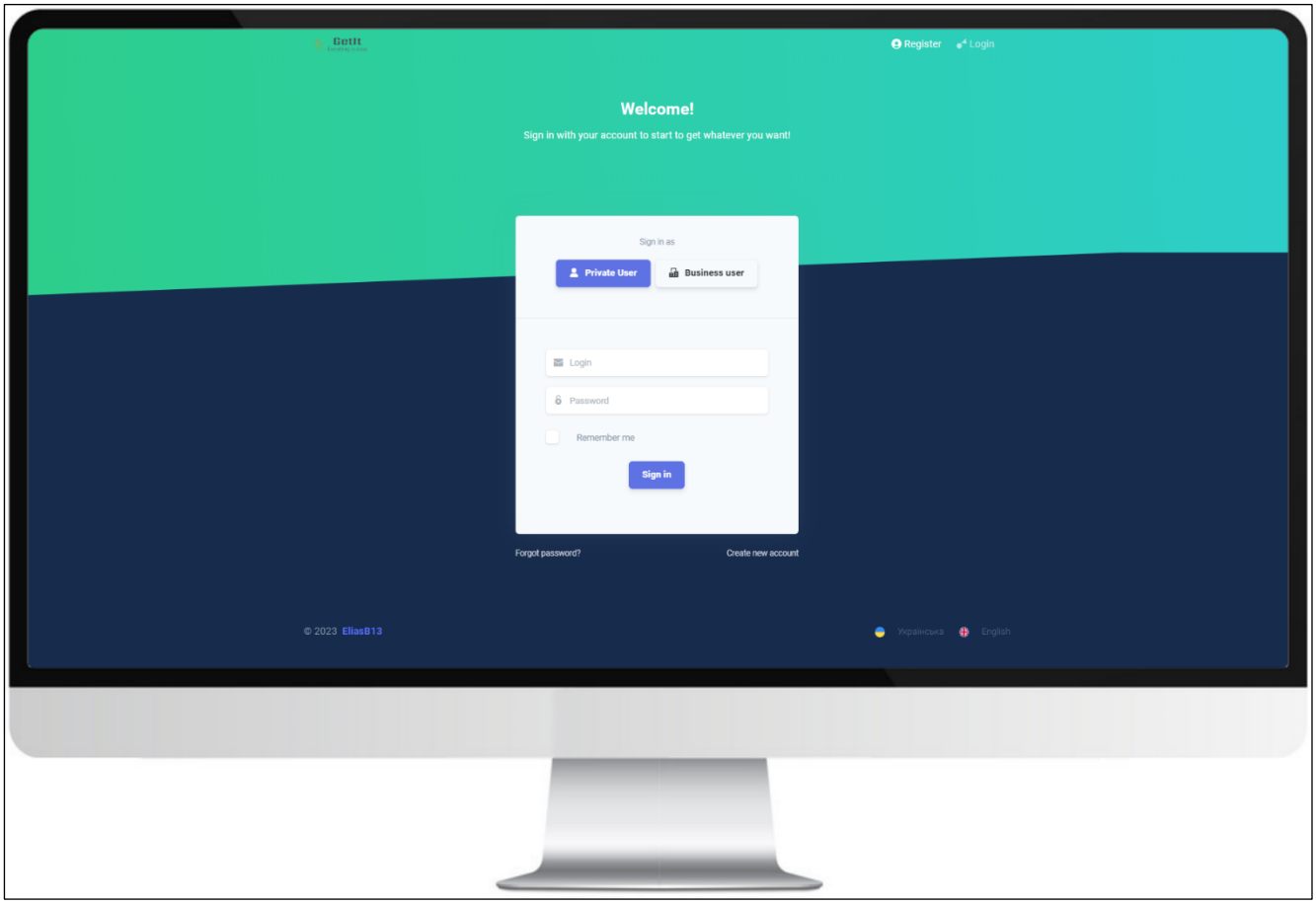

Рисунок 4.6 – Сторінка авторизації

Наступною наведемо приклад сторінки адмін-панелі компанії, що містить статистику користування системою. На цьому екрані користувач може отримати різні види статистики, а саме: статистика за зареєстрованими та незареєстрованими користувачами, статистика по кількості використаних предметів загального доступу, статистика фіксацій по різних зонах, статистика за певними мітками та зчитувачами, історія переміщень певних об’єктів.

Для представлення статистичних даних було використано різноманітні елементи графічного інтерфейсу, такі як лінійні та стовпчикові діаграми, графіки, картки окремих показників з відсотковими показниками зміни метрик, а також додаткові сторінки з детальною статистикою за певними мітками та зчитувачами. Зазначені елементи є адаптивними та змінюють свій розмір та зовнішній вигляд відповідно до поточного розміру дисплею пристрою.

Зображення мокапу описаної сторінки додатку зі статистикою наведемо на рисунку 4.7.

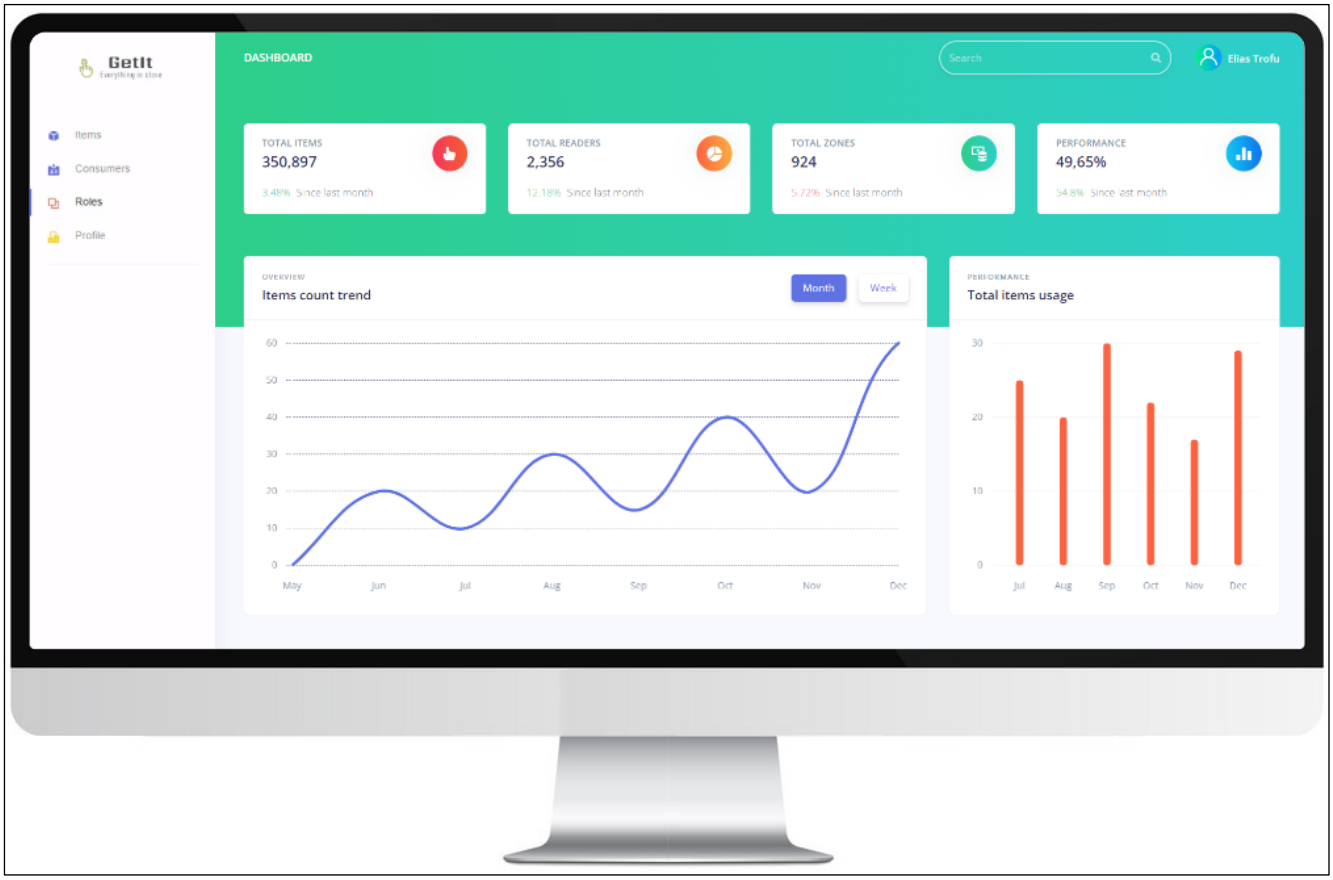

Рисунок 4.7 – Панель статистики компанії

Наступною продемонструємо вигляд сторінки керування предметами загального користування, де користувач має переглядати, додавати та видаляти предмети, за якими відбувається відстеження. Крім того, користувач має змогу перейти на сторінку певного предмету, де може побачити історію його використань, а також, за наявності, мапу історії переміщень певного предмета за проміжок часу. Для заздалегідь налаштованих об’єктів є можливість відстежувати переміщення у реальному часі.

Наведемо також макет дизайну сторінки керування користувацьким профілем (див. рис. 4.8).

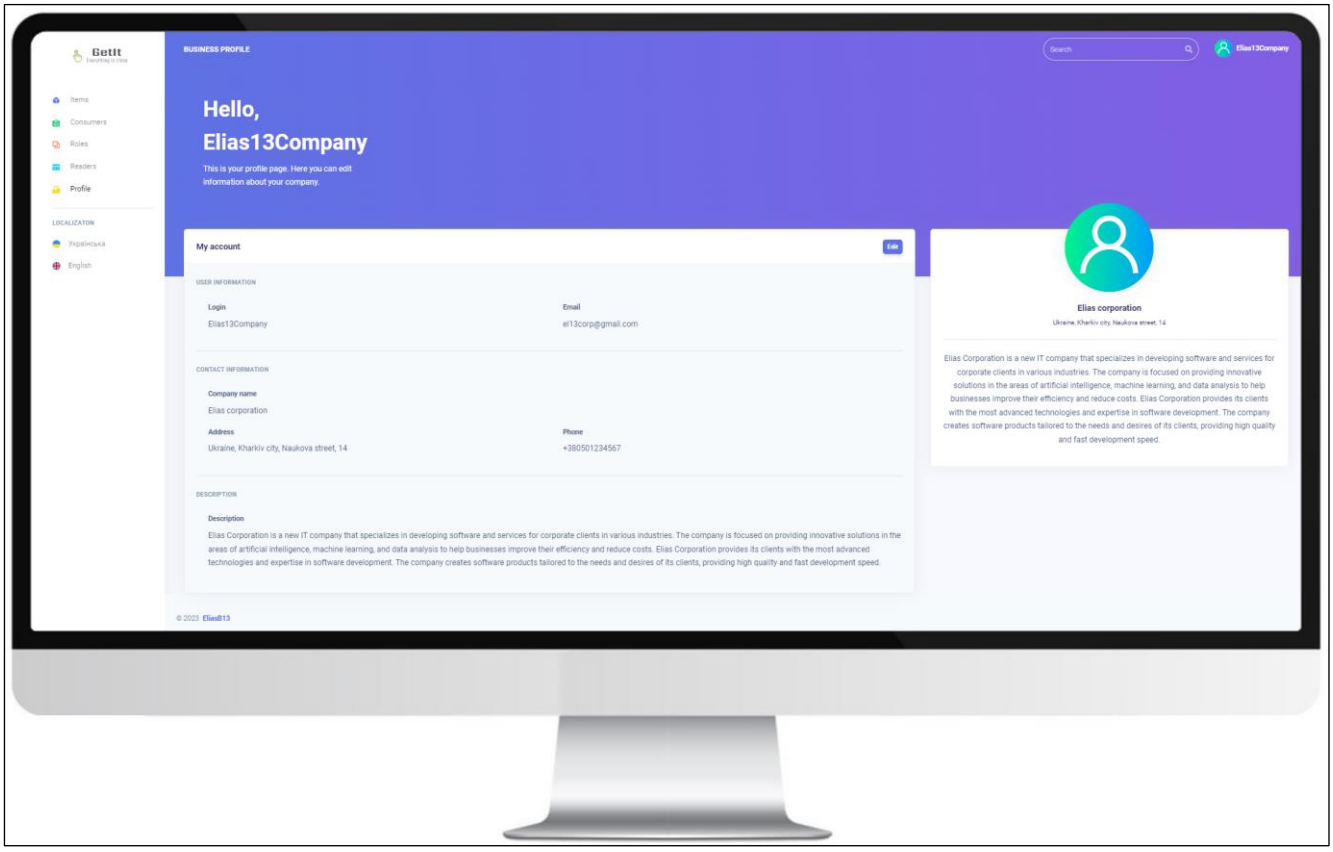

Рисунок 4.8 – Сторінка керування користувацьким профілем

Зауважимо, що інтерфейс зазначених сторінок побудовано з урахуванням зазначених на початку розділу евристик та принципів побудови користувацького інтерфейсу (кнопки та елементи змінюють зовнішній вигляд при наведенні та натисканні, тривалі операції позначаються елементами, що відображають процес виконання, для багатьох операцій імплементовано як одиничне так і множинне виконання.

5 ОПИС ПРИЙНЯТИХ ПРОГРАМНИХ РІШЕНЬ

5.1 Загальні відомості

В ході виконання кваліфікаційної роботи, на основі даних отриманих під час етапів аналізу ринку та проєктування, з використанням технологій програмного забезпечення, описаних у вступі, було розроблено комплексну програмну систему для автоматизації процесу керування предметами загального доступу. Обрані для розробки технології є актуальними і сучасними, та відповідають вимогам гнучкості розробки, надійності, працездатності та наявністю підтримки.

5.2 Опис фізичної моделі бази даних

В ході реалізації системи зберігання даних, відповідно до рішень прийнятих у розділі 4, було побудовано архітектуру що складається з наступних компонентів: реляційна база даних для структурованих і обмежених в кількості даних, нереляційна база даних для неструктурованих даних, кількість яких умовно необмежена, та розподілений кеш для оптимізації процесу роботи з даними в системі.

Для реалізації реляційної бази даних було обрано СКБД Microsoft SQL Server. Зазначена СКБД є однією з найбільш популярних реляційних систем управління базами даних, завдяки високій продуктивності, можливостям масштабування, структурованості, механізмам підтримки цілісності даних, надійності та безпеки.

У якості нереляційної бази даних було обрано сучасне рішення під назвою MongoDB, яке є прикладом документо-орієнтованої NoSQL СКБД. Така система дозволяє значно швидше працювати з великими обсягами даних, за рахунок відсутності зв’язності між таблицями, а також відсутності необхідності поєднання даних. Крім того, MongoDB надає помітні можливості горизонтального масштабування, реплікації та розподіленості.

Для пришвидшення загальної роботи з даними у системі для операцій читання було обрано сервіс кешування Redis. Він дозволяє значно швидше отримувати доступ до даних, які найчастіше використовуються, що позитивне вплине на загальну продуктивність системи та ефективність її роботи.

Зважаючи на дані, отримані на етапі проєктування, побудуємо фізичну схему бази даних, дотримуючись вимог цілісності даних. Наступним кроком, згенеруємо діаграму розробленого сховища даних, для демонстрації її фізичної моделі.

В результаті розробки отримано діаграму фізичної моделі реляційної бази даних зображену на рисунку 5.1.

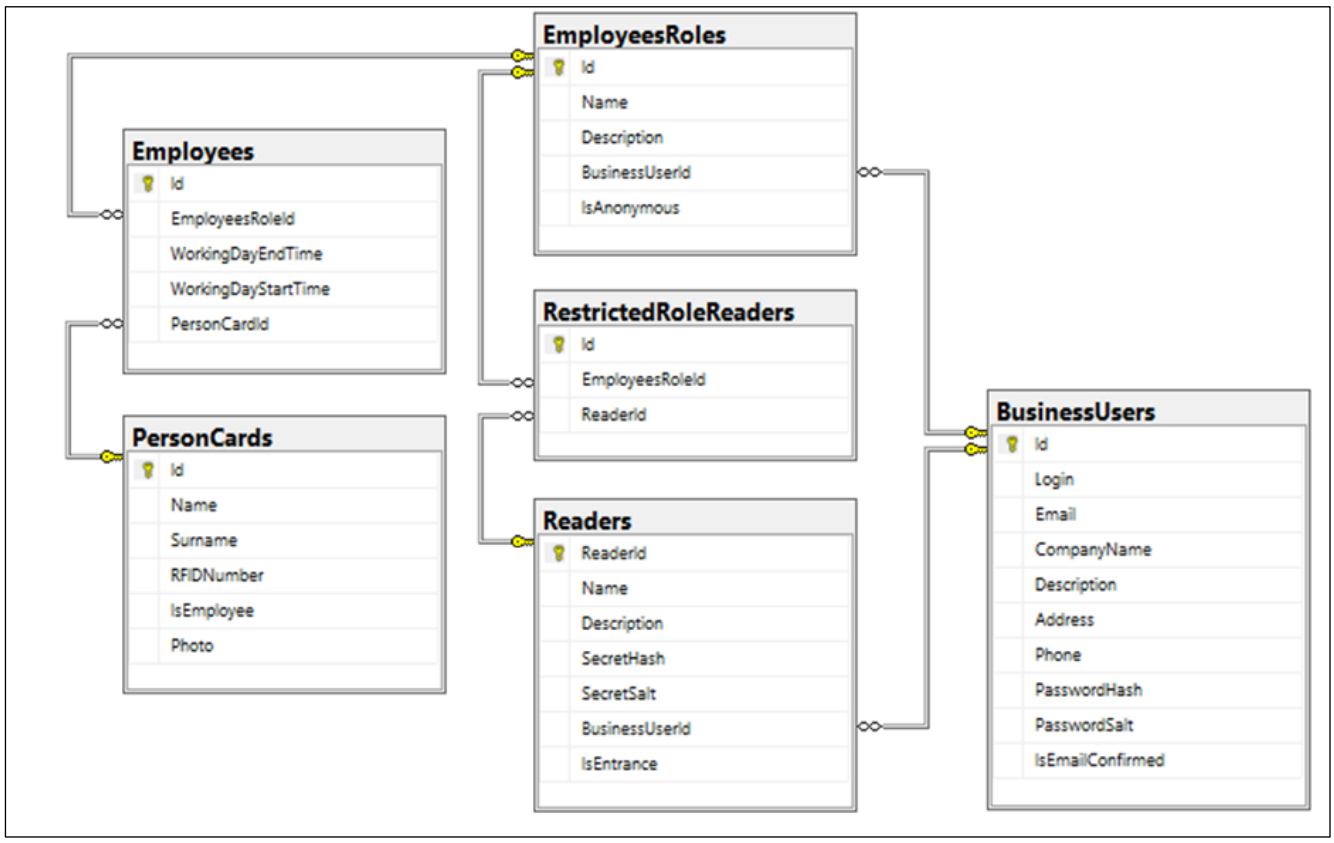

Рисунок 5.1 – Фізична модель розробленої реляційної бази даних

Зазначимо, що робота з системами керування базами даних на програмному рівні виконувалася за допомогою абстракцій та окремих контекстів виконання для кожного типу сховища. Для побудови моделей використовувався програмний підхід «Code first», який дозволяє будувати фізичну модель на основі доменних моделей всередині застосунку.

5.3 Опис серверної частини додатку

Користуючись рішеннями, прийнятими на етапі проєктування у розділі 4.1 та 4.2, виконаємо реалізацію серверної частини програмної системи. У якості технологій реалізації обрано Microsoft ASP.NET 7, який надає широкий інструментарій для розробки систем подібного типу, забезпечуючи високий рівень надійності та безпеки, та закладаючи помірний рівень масштабованості.

5.3.1 Опис програмного рішення

В ході виконання даного розділу було розроблене серверне програмне рішення, що має багатошарову архітектуру. Рішення поділяється на наступні шари: інфраструктура, бізнес-логіка, представлення.

Зазначене програмне рішення було побудовано з використанням найкращих практик розробки серверних рішень на базі платформи ASP.NET 7 з використанням мови програмування C#. В процесі розробки використовувалися принципи SOLID, DRY, KISS та YAGNI, що позначають набори принципів програмування, які значно покращують можливості подальшої підтримки програмного коду.

Інфраструктурний шар відповідає за забезпечення взаємодії між додатком та іншими системами, такими як бази даних, сервери, сервіси, мережі тощо. У цьому шарі розміщені всі компоненти, що відповідають за забезпечення безпеки додатку, авторизацію та аутентифікацію користувачів, а також всі інші компоненти, необхідні для забезпечення роботи застосунку.

Шар бізнес-логіки відповідає за реалізацію логіки додатку, яка визначає, як він повинен працювати. У цьому шарі розміщені всі компоненти, що відповідають за обробку даних та обчислення, виконання бізнес-логіки, валідацію даних, обробку помилок та інші компоненти, що відповідають за роботу додатку.

Шар представлення відповідає за відображення даних користувачеві. У цьому шарі розміщені всі компоненти, що відповідають за відображення

інтерфейсу користувача, обробку взаємодії з користувачем, а також за забезпечення взаємодії між користувачем та додатком. У контексті серверного додатку, шар представлення містить програмний інтерфейс рішення, з яким будуть працювати інші модулі системи, а також документацію, наприклад OpenAPI Swagger.

У якості прикладу наведемо частину коду серверного рішення, що реалізує гібридний алгоритм, описаний у розділі 3.4.

```
private double Distance(RssiValue[] unknown, AdjustedFingerprinting
    data)
{
  double distance = 0.0;

  for (int i = 0; i < unknown.Length; ++i)
  {
    distance += (unknown[i].Rssi - data.RssiValueAndGateway[i].Rssi)
        * (unknown[i].Rssi - data.RssiValueAndGateway[i].Rssi);
  }

  return Math.Sqrt(distance);
}

public static List<AdjustedFingerprinting> GetFingerprintings(int
    environmentId)
{
  if (!fingerprintings.ContainsKey(environmentId))
  {
    string json = Get(string.Format($"{GET_FINGERINTING_COMMAND} -env
        {environmentId}"));
    List<Fingerprinting> listOfFingerprinting =
        JsonConvert.DeserializeObject<List<Fingerprinting>>(json);

    List<AdjustedFingerprinting> adjustedFingerprintings = new
        List<AdjustedFingerprinting>();
    fingerprintings.Add(environmentId, adjustedFingerprintings);

    AdjustedFingerprinting adjustedFingerprinting = new
        AdjustedFingerprinting();
    int firstGateway = listOfFingerprinting[0].GatewayId;
    RssiValue rssiValue = new RssiValue();

    for (int i = 0; i < listOfFingerprinting.Count; i++)
    {
      if (listOfFingerprinting[i].GatewayId == firstGateway)
      {
        adjustedFingerprinting = new AdjustedFingerprinting()
        {
          Coordinates = GetPoint(environmentId,
          listOfFingerprinting[i].Xaxis,
          listOfFingerprinting[i].Yaxis) };
```

Продовження лістингу коду

```
            adjustedFingerprintings.Add(adjustedFingerprinting);
          }

          rssiValue = new RssiValue()
          {
            GatewayId = listOfFingerprinting[i].GatewayId,
            Rssi = (listOfFingerprinting[i].Rssi)
          };

          adjustedFingerprinting.RssiValueAndGateway.Add(rssiValue);
        }
      }

      return fingerprintings[environmentId];
    }
```

Неможливо не відзначити той факт, що для реалізації принципів абстракції (з метою зменшення зв'язності коду) в програмному рішенні використовується IoC (Inversion of Control) шаблон, який реалізується за допомогою DI-контейнеру (Dependency Injection), що дозволяє автоматизувати процес керування залежностями у програмному рішенні. Нижче наведемо приклад ін'єкції залежності у конструкторі контролеру застосунку:

```
private readonly IEmployeesService employeesService;

public BusinessUsersController(IEmployeesService employeesService)
{
      this.employeesService = employeesService;
}
```

Безпосередньо програмний інтерфейс серверного рішення побудований на основі підходу під назвою REST API (Representational State Transfer Application Programming Interface – програмний інтерфейс передачі стану представлення). Такий шаблон використовується для створення веб-сервісів, що дозволяє взаємодіяти з додатками та ресурсами через мережу Інтернет за протоколами HTTP та HTTPS.

REST API реалізується на базі наступних принципів:

– застосунок використовує клієнт-серверну архітектуру, де клієнт і сервер взаємодіють за допомогою стандартних HTTP запитів і відповідей;

– застосунок не зберігає жодного стану між запитами клієнта. Кожен запит клієнта повинен містити всю необхідну інформацію, яка потрібна для обробки запиту сервером;

– застосунок підтримує кешування, де результати запитів можуть зберігатися на клієнтській або проміжній стороні для подальшого використання;

– застосунок має одну єдину точку доступу, яка дозволяє клієнтам взаємодіяти з додатком через стандартний інтерфейс;

– застосунок використовує стандартизований набір HTTP запитів (GET, POST, PUT, DELETE) і форматів даних (наприклад, JSON або XML) для взаємодії між клієнтом та сервером;

– застосунок підтримує рівні абстракції, де кожен рівень може бути реалізований незалежно від інших, що дозволяє розподіляти складність системи та забезпечувати більшу масштабованість.

Таким чином, розроблена програмна система побудована за шаблоном REST API та відповідає усім вищезазначеним умовам й принципам.

Для авторизації у проєкті використовується шаблон "JWT Authorization", який передбачає використання JSON Web Token. При цьому клієнт надсилає запит з даними для авторизації, а сервер відправляє токен, який додається до авторизованих запитів. Клієнт повинен зберігати токен у локальному сховищі для доступу до авторизованих частин інтерфейсу серверної частини. Кожен токен має обмежений час дії. Для безпеки, персональні дані користувачів не зберігаються у серверній частині або у базі даних. Замість цього використовується принцип порівняння хеш-сум, які зберігаються у зашифрованому вигляді у базі даних.

5.3.2 Демонстрація роботи серверної частини

У якості специфікації програмного інтерфейсу було обрано програмне забезпечення OpenAPI Swagger. Таке ПЗ є стандартизованим методом документації рішень на основі шаблону REST API. Swagger - це набір інструментів для

створення, опису та візуалізації REST API. Він допомагає описати структуру API, доступні ресурси, операції, параметри, відповіді, коди помилок та інші важливі деталі. Swagger дозволяє створити документацію API, яка буде доступна для розробників та користувачів, і допомагає знизити час на розробку та налагодження API

На рисунку 5.2 наведено зображення роботи інтерфейсу Swagger серверної частини.

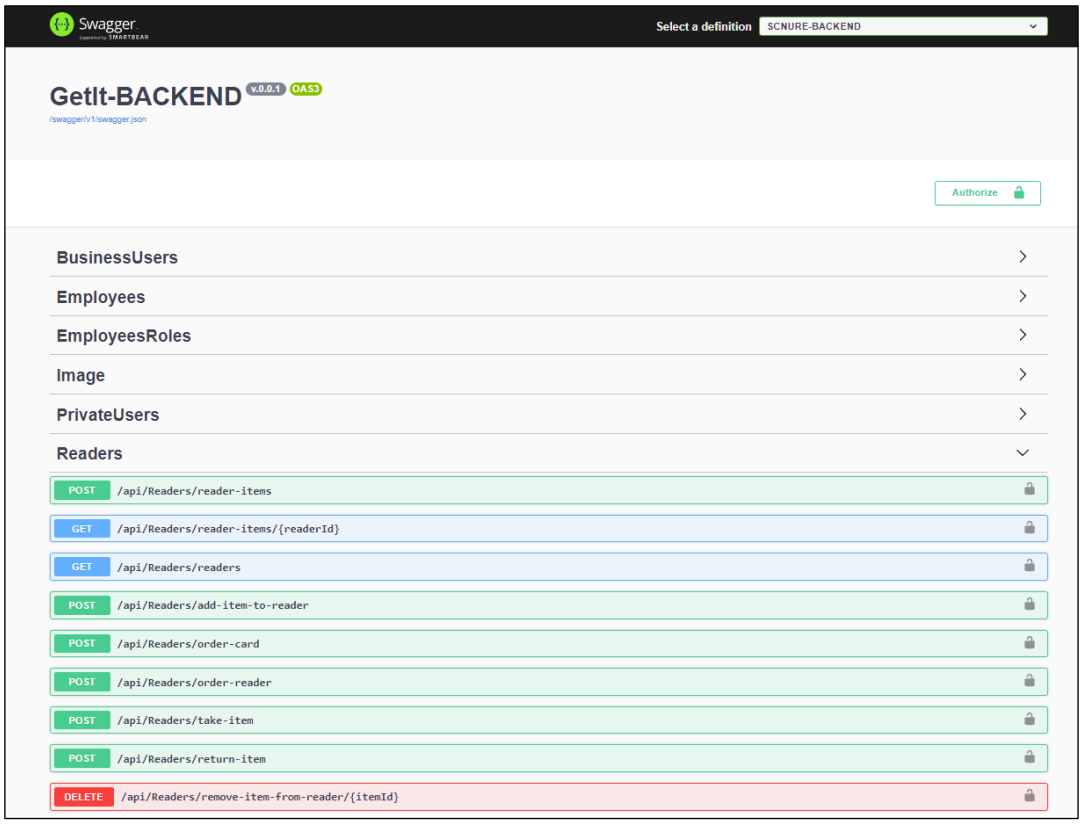

Рисунок 5.2 – Демонстрація інтерфейсу Swagger серверної частини

Наступним кроком перевіримо результати роботи застосунку, виконавши деякі з доступних запитів.

Наведемо також зображення приклау виконання запиту отримання повної статистики компанії за визначеними параметрами, наприклад проміжок часу. У відповідь на відправку запиту отримаємо повідомлення у форматі JSON, яке зможуть використовувати клієнтські частини програмного рішення (веб- та мобільна).

Для демонстрації роботи серверного рішення наведемо результат виклику кінцевої точки отримання статистичних даних (див. рис. 5.3).

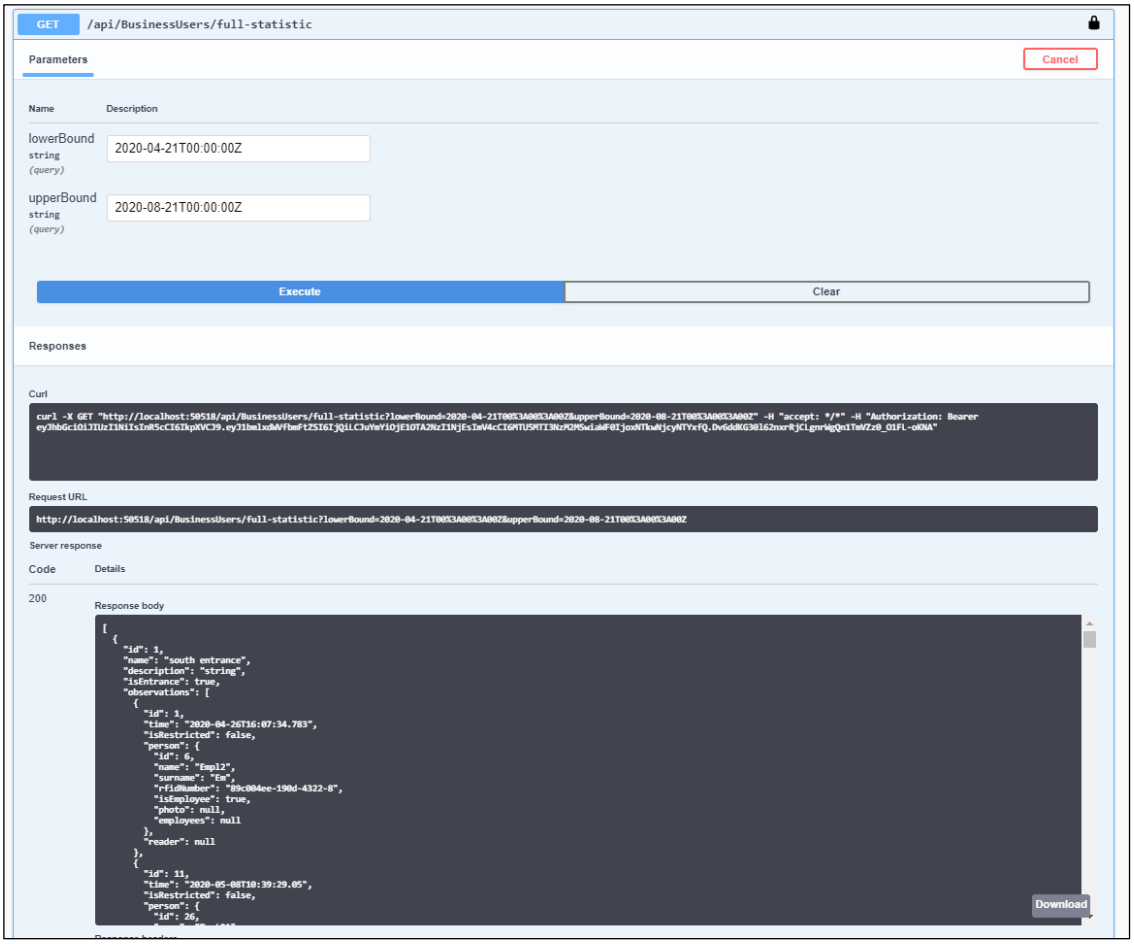

Рисунок 5.3 – Демонстрація виклику кінцевої точки серверного рішення

Таким чином, в ході виконання роботи було розроблено серверне програмне забезпечення, яке має доступний інтерфейс та ефективну документацію, на базі якого можна побудувати повноцінну комплексну систему.

5.4 Опис клієнтської веб-частини

Користуючись рішеннями прийнятими на етапі проєктування у розділі 4.1 та 4.4 виконаємо реалізацію клієнтської веб-частини програмної системи. У якості

технологій реалізації обрано Vue JS, що є провідним сучасним фреймворком розробки веб-застосунків.

5.4.1 Опис програмного рішення

Наведемо на рисунку 5.4 структуру програмного рішення веб-застосунку. При побудові архітектури клієнтського додатку було використано компонентно-орієнтований шаблон, при якому структура поділяється на окремі самостійні компоненти, які можуть використовуватися багаторазово.

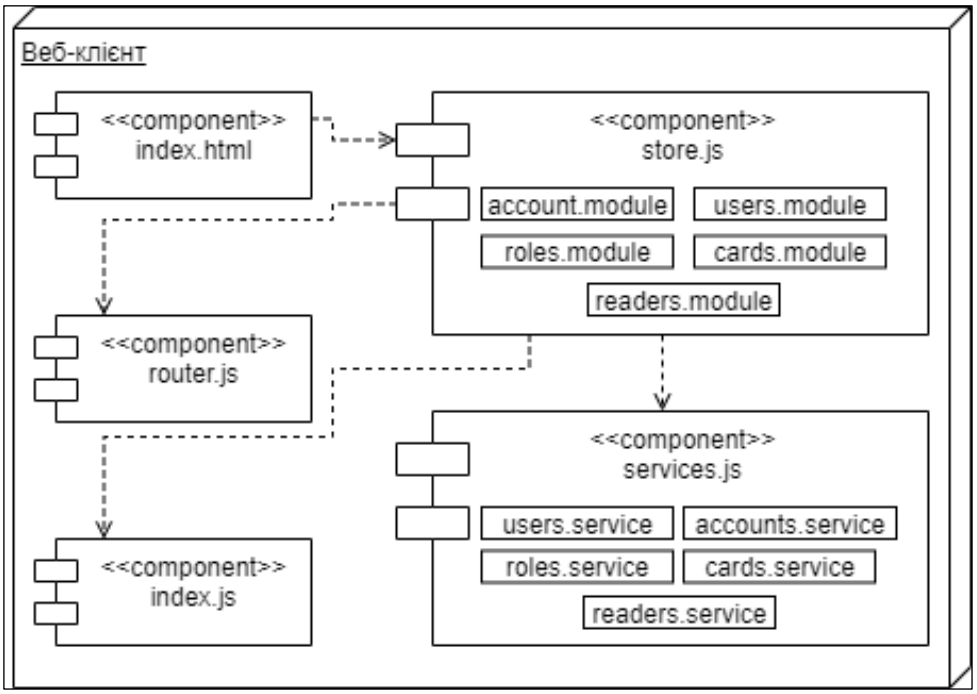

Рисунок 5.4 – Діаграма компонентів

Компонентно-орієнтована архітектура дозволяє значно полегшити процес розробки рішення, а також сприяє підвищенню загальної якості застосунку.

Аналізуючи наведений рисунок також визначимо, що основними структурними елементами додатку є: точка входу у додаток – HTML-компонент index.html, початковий файл Javascript коду – index.js, роутер – router.js, складений компонент сховище – store.js, складений компонент сервісів – services.js. На діаграмі також позначено спосіб та послідовність взаємодії компонентів. Розглянемо деякі з наведених елементів.

Компонент сховище реалізоване за допомогою Vuex. Vuex - це бібліотека управління станом для додатків Vue.js. Вона дозволяє зберігати стан додатку в централізованому місці, яке називається Store (сховище), та забезпечує стандартний спосіб зміни стану за допомогою мутацій.

Компонент сервісів представляє собою фасад всіх сервісів, які містяться у програмному рішенні. Кожен сервіс є набором Javascript-функцій, які інкапсулюють певну частину функціональності веб-застосунку. Здебільшого, сервіси виконують отримують вхідні параметри (наприклад, користувацьке введення), формують на їх основі певні запити за HTTPS протоколом, та отримують відповіді. Компоненти інтерфейсу спілкуються з компонентами сервісів за допомогою сховища, що дозволяє уніфікувати потоки переміщення даних всередині застосунку. Компонент роутеру – це інструмент для навігації між сторінками або розділами вашої веб-додатку. Він дає змогу організувати структуру додатку і забезпечити його навігаційну логіку.

Для поліпшення процесу розробки та повторного використання програмного коду важливо мати структурований початковий код додатку. Одним із способів досягнення цього є побудова діаграми пакетів. Наведемо отриману діаграму пакетів, для демонстрації принципів структурування програмного коду веб-застосунку (див. рис. 5.5).

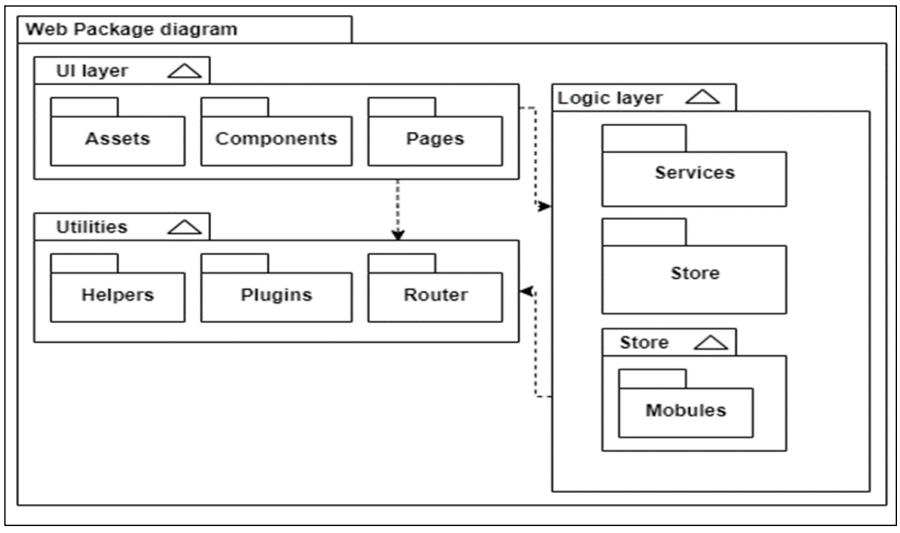

Рисунок 5.5 – Діаграма пакетів

У діаграмі основними складовими є пакети, які слугують для групування елементів програмного коду на більш високому рівні організації. Зв'язки залежностей між пакетами вказують, що зміна елемента в одному пакеті може вплинути на елементи в залежному пакеті.

Ця діаграма дозволяє отримати детальну структуру організації елементів програмного рішення в модулі, спрощує процес його розробки та покращує якість вихідного коду.

5.4.2 Демонстрація роботи веб-частини

Щоб продемонструвати роботи веб-застосунку наведемо знімки екранних форм деяких сторінок. На рисунку 5.6 показано зовнішній вигляд сторінки авторизації.

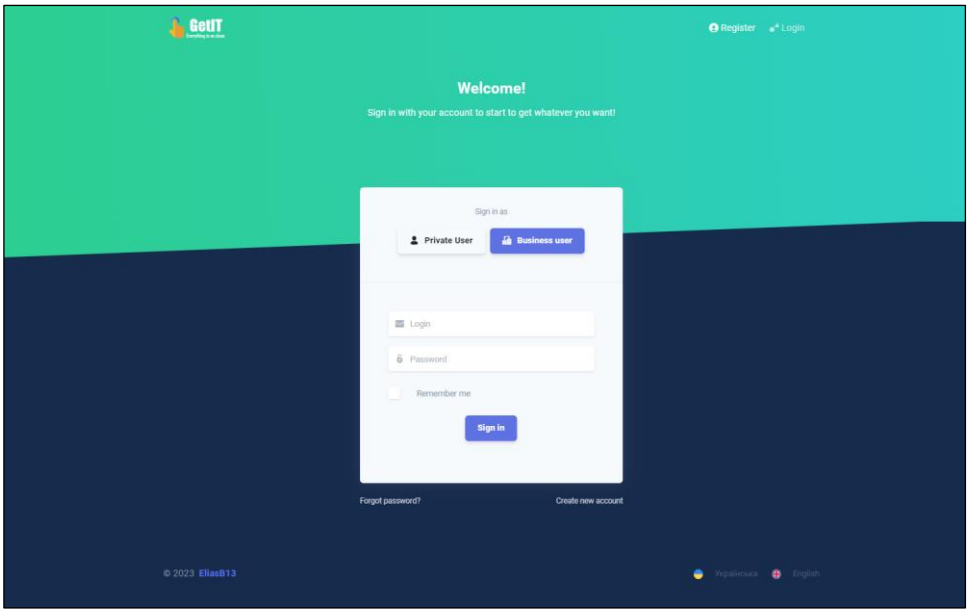

Рисунок 5.6 – Сторінка авторизації сервісу

Наступним кроком наведемо вигляд сторінки панелі адміністратора, на якій є різні типи статистичних даних компанії за певними параметрами. На цій сторінці користувач може переглядати, додавати та видаляти предмети, за якими відбувається відстеження. Крім того, користувач має змогу перейти на сторінку

певного предмету, де може побачити історію його використань, а також, за наявності, мапу історії переміщень певного предмета за проміжок часу. Для заздалегідь налаштованих об’єктів є можливість відстежувати переміщення у реальному часі.

На рисунку 5.7 показано вигляд розробленої сторінки панелі адміністратора веб-застосунку.

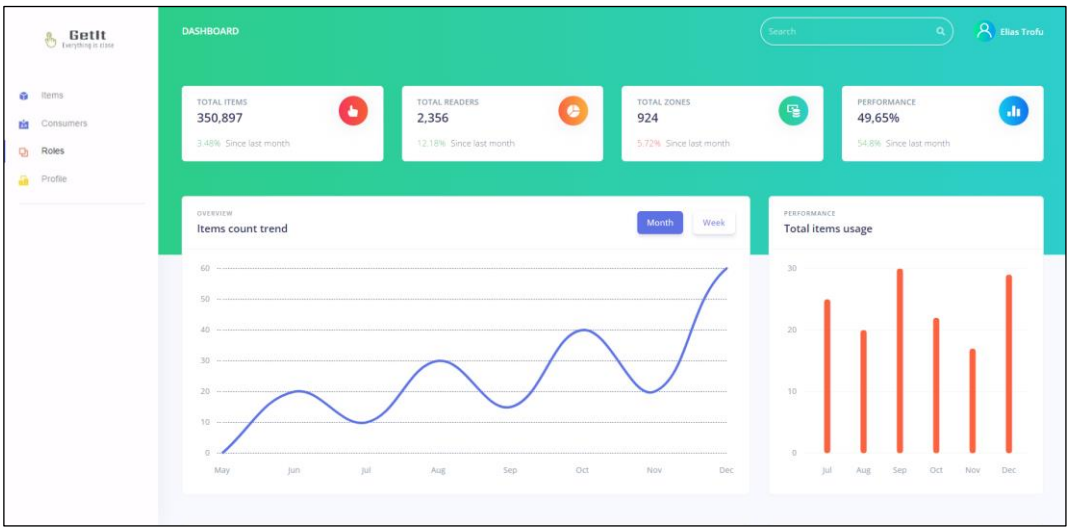

Рисунок 5.7 – Сторінка статистики

Також наведемо результат розробки сторінки керування користувачами компанії (див. рис. 5.8). На цій сторінці можна додавати, редагувати, видаляти користувачів компанії та їх класів.

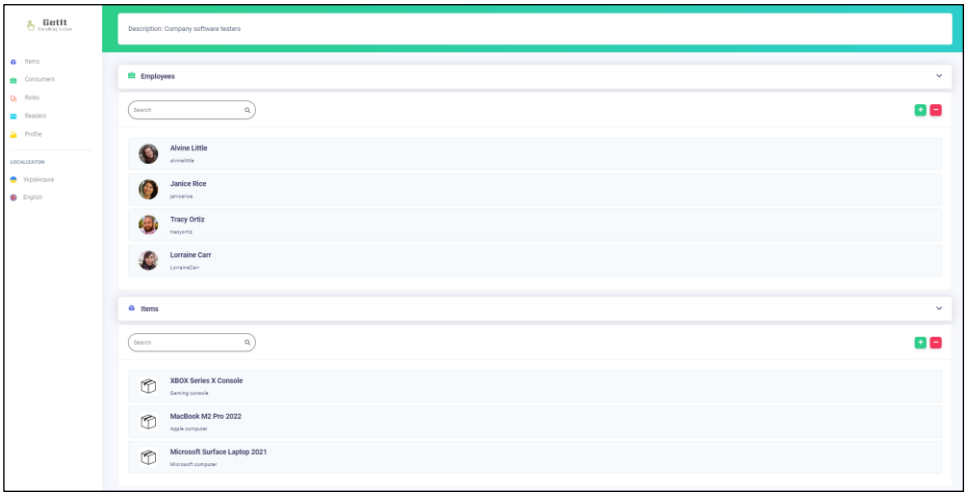

Рисунок 5.8 – Сторінка клієнтів та співробітників бізнесу

Таким чином, було розроблено клієнтську веб-частину, яка виконуватиме роль головного користувацького інтерфейсу у програмній системі, що розробляється.

5.5 Опис мобільної частини системи

Користуючись рішеннями прийнятими на етапі проєктування у розділі 4.1 та 4.4 виконаємо реалізацію клієнтської мобільної частини програмної системи. У якості технологій реалізації обрано мову програмування Kotlin та ОС Android 7.0+, що є сучасним стандартом розробки мобільних додатків.

5.5.1 Опис програмного рішення

В ході розробки мобільного програмного рішення було створене програмне забезпечення з використанням компонентно-орієнтованої архітектури та патерну MVVM.

MVVM (Model-View-ViewModel) - це патерн проектування програмного забезпечення, який дозволяє розділити логіку бізнес-логіки від логіки відображення і керування даними.

MVVM складається з трьох основних компонентів: моделі (Model), представлення (View) та моделі представлення (ViewModel). Модель представляє собою даний додаток, який ми працюємо з ним. Представлення є графічним інтерфейсом, який відображає дані, які ми взаємодіємо з ними. Модель представлення виступає в якості посередника між моделлю та представленням, вона обробляє дані з моделі та перетворює їх на форму, зручну для представлення в інтерфейсі користувача.

Наведемо приклад взаємодії серверного та мобільного застосунку на рисунку 5.9.

Визначимо основні структурні складові додатку: набір ресурсів, компоненти-обробники графічного інтерфейсу (activities), служби, основний компонент конфігурації та моделі, що відображають елементи предметної області, а також моделі запитів та відповідей серверної частини.

Ресурси додатку – це набір компонентів, які не містять основної логіки додатку. Зазвичай, це зображення, кольори, рядкові константи та XML-макети сторінок додатку.

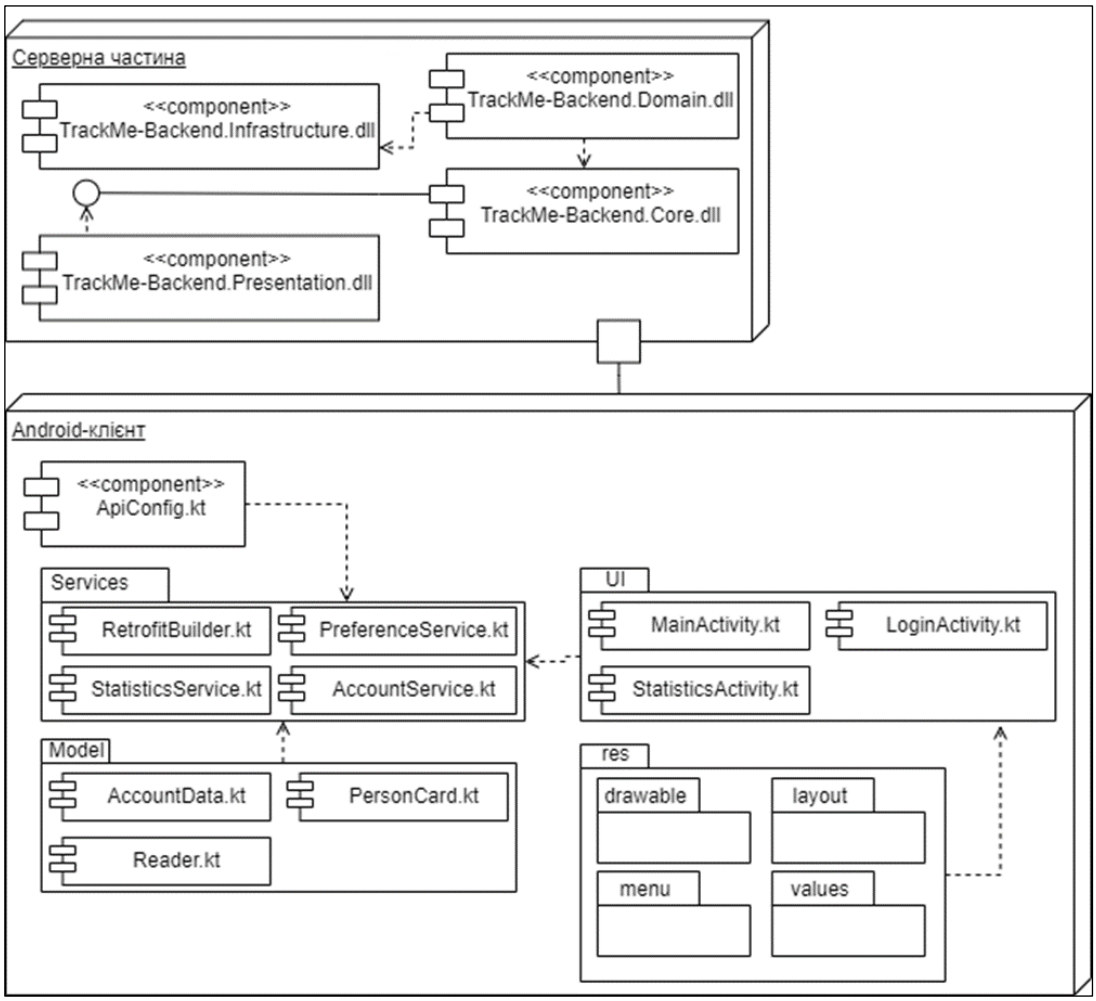

Рисунок 5.9 – Діаграма компонентів

Компоненти-обробники графічного інтерфейсу (activities) є написаними мовою Kotlin компонентами, які визначають екрани додатку, які містять елементи графічного інтерфейсу та фрагменти, які можуть бути замінені під час виконання додатку. Ці обробники пов'язуються з XML-макетами, в яких визначається

розмітка сторінок додатку.

Компонент конфігурації визначає статичний допоміжний об'єкт, який вказує адресу та параметри підключення до серверної частини в одному місці.

Служби визначають компоненти, які містять основну логіку додатку та взаємодію з серверною частиною за вказаними інтерфейсами. Для виконання HTTP-запитів та обробки відповідей використовується бібліотека Retrofit, яка дозволяє зручним та ефективним способом створювати інтерфейси, які генерують HTTP-запити.

З метою структуризації уявлення про початковий код додатку, а також про архітектуру розробленого рішення використаємо діаграму пакетів.

Така діаграма дозволяє зобразити принципи організації програмного коду (див. рис. 5.10).

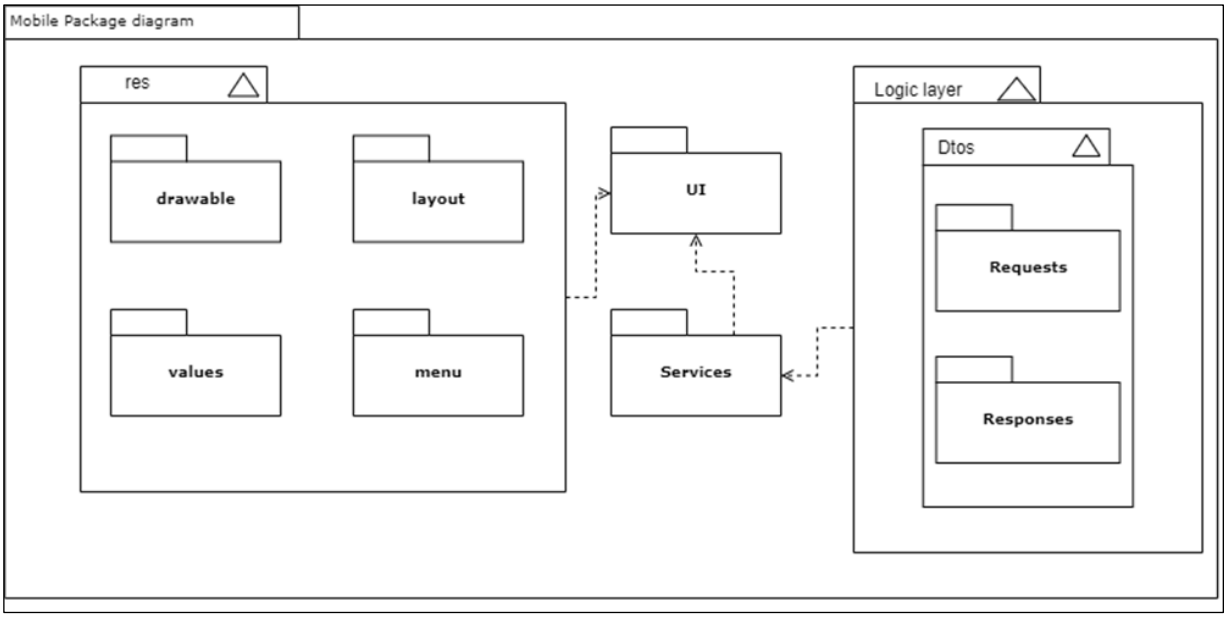

Рисунок 5.10 – Діаграма пакетів

Діаграма містить основні елементи, такі як пакети, які виконують роль елементів групування і дозволяють об’єднати більш низькорівневі елементи вищого рівня (на рівні структури). Пакети взаємодіють між собою за допомогою зв'язків залежності, що вказують, що зміна одного елементу призводить до зміни іншого, який залежить від нього.

5.5.2 Демонстрація роботи мобільної частини

Для тестування мобільної частини виконаємо компіляцію та запустимо режим налагодження на Android-пристрої, з метою демонстрації роботи застосунку. Наведемо наступні елементи: сторінку авторизації, загальної статистики, докладної статистики за певним зчитувачем (див. рис. 5.11).

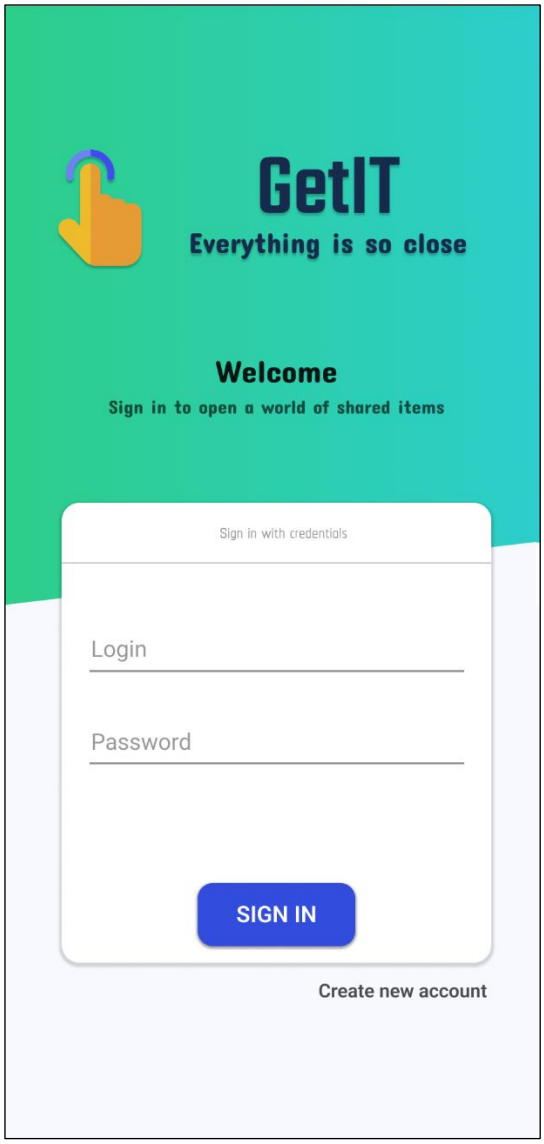

Рисунок 5.11 – Сторінка авторизації у мобільному застосунку

Зауважимо, що при розробці графічного інтерфейсу виконувалося чітке дотримання евристичних правил, з метою зробити користувацький досвід максимально ефективним та інтуїтивним.

Наступною зобразимо сторінку історії використання певного предмету на рисунку 5.12.

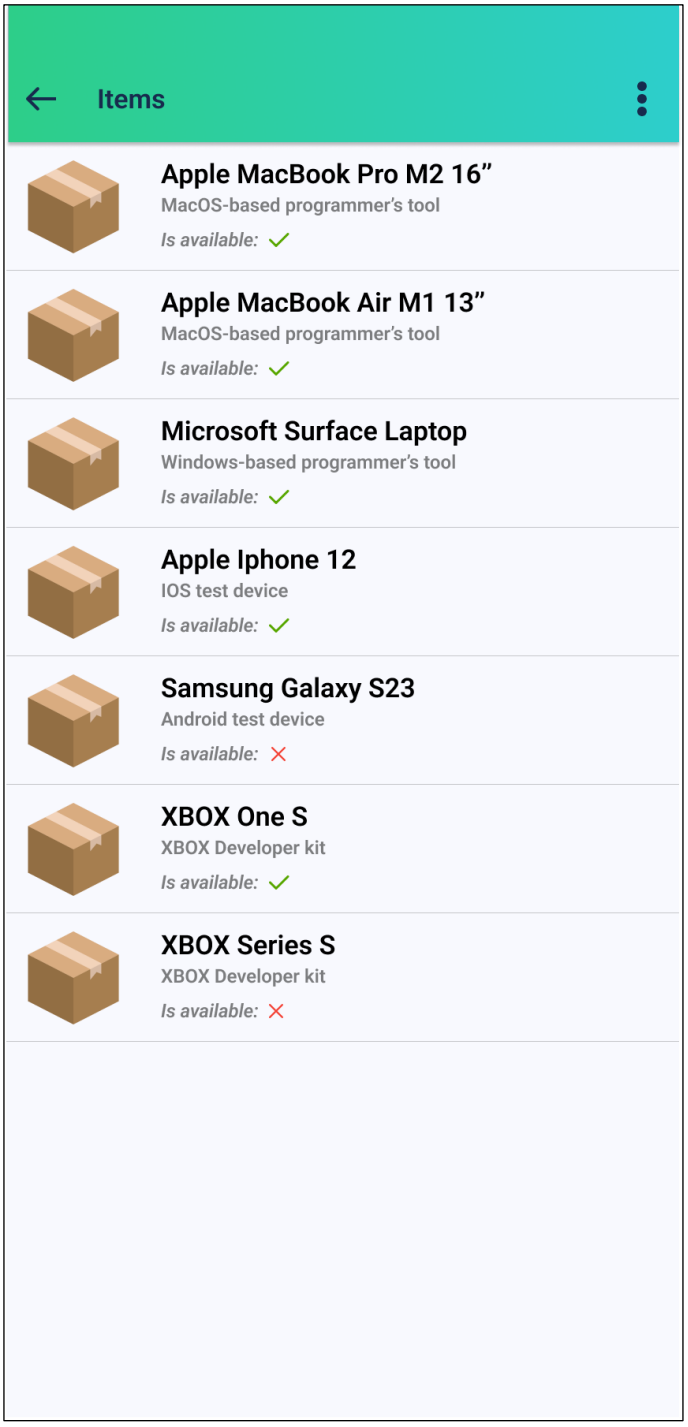

Рисунок 5.12 – Сторінка загальної статистики

Зазначена сторінка дозволяє компанії переглядати загальну статистику по предметах або зонах зчитування, а також переходити до детальної статистики за певним зчитувачем, міткою, або предметом загального користування.

На рисунку 5.13 наведемо сторінку детальної статистики користування спільним предметом.

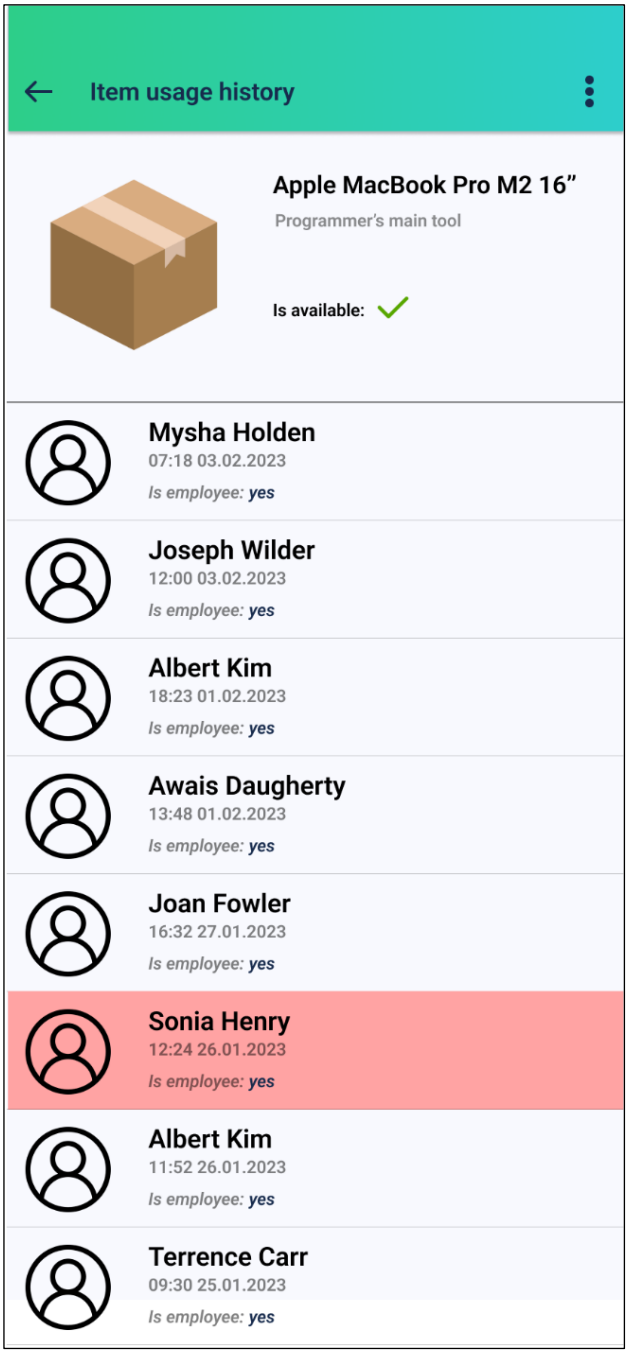

Рисунок 5.13 – Сторінка статистики певного предмета

Аналізуючи представлену інформацію, можна дізнатись яка особа та в який момент часу користувалась обраним предметом. Крім того, для заздалегідь налаштованих зчитувачів можна переглядати історію переміщень предмету загального користування.

5.6 Опис IoT-застосунку

Користуючись рішеннями прийнятими на етапі проєктування у розділі 4, виконаємо реалізацію клієнтської веб-частини програмної системи. У якості технологій розумного пристрою обрано BLE-мітки з терміналами на базі Raspberry Pi 3. Зазначимо, що система також дозволяє використання міток RFID, які можуть бути використані у якості більш дешевої версії продукту. Конфігурації на базі RFID міток матимуть нижчу вартість, проте й поступатимуться у швидкодії та точності системам на базі BLE

5.6.1 Опис програмного рішення

В ході реалізації IoT-частини системи було розроблено додаток терміналу зчитувача з використанням мови програмування Python.

На рисунку 5.14 зображено діаграму пакетів застосунку.

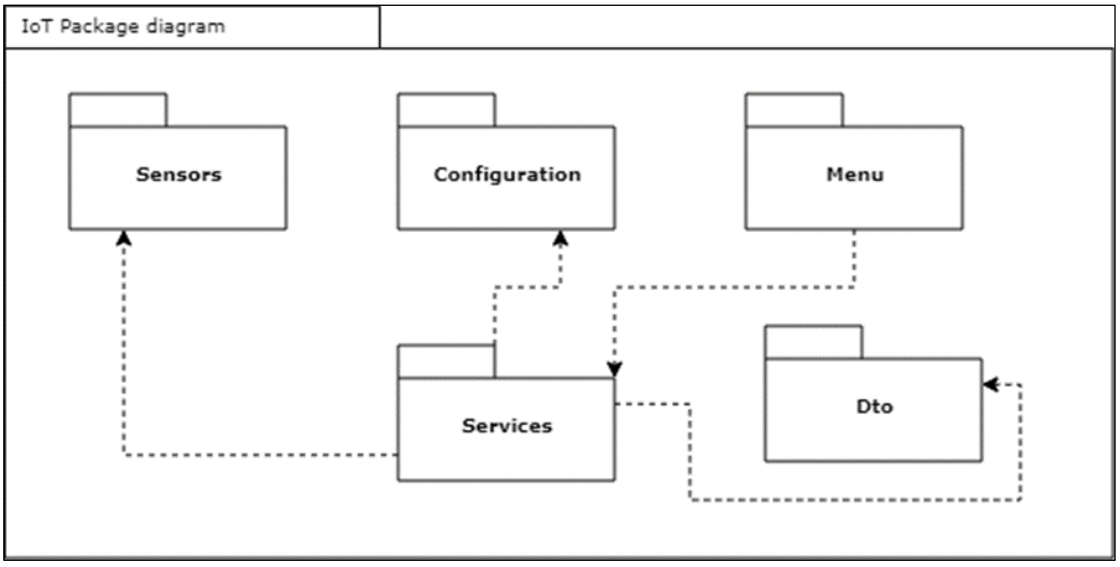

Рисунок 5.14 – Пакетна структура рішення розумного пристрою

За допомогою рисунку 5.14 можна виявити головні структурні компоненти додатку, такі як елементи, які забезпечують зчитування даних з сенсорів, елементи

конфігурації, елементи меню, служби та моделі запитів та відповідей серверної частини.

Діаграма взаємодії на рисунку 5.15 подробиць описує принцип взаємодії користувача з IoT-додатком та додатку з сервером. Важливо відзначити, що дії на діаграмі виконуються послідовно протягом життєвого циклу, що чітко пояснює принцип роботи додатку.

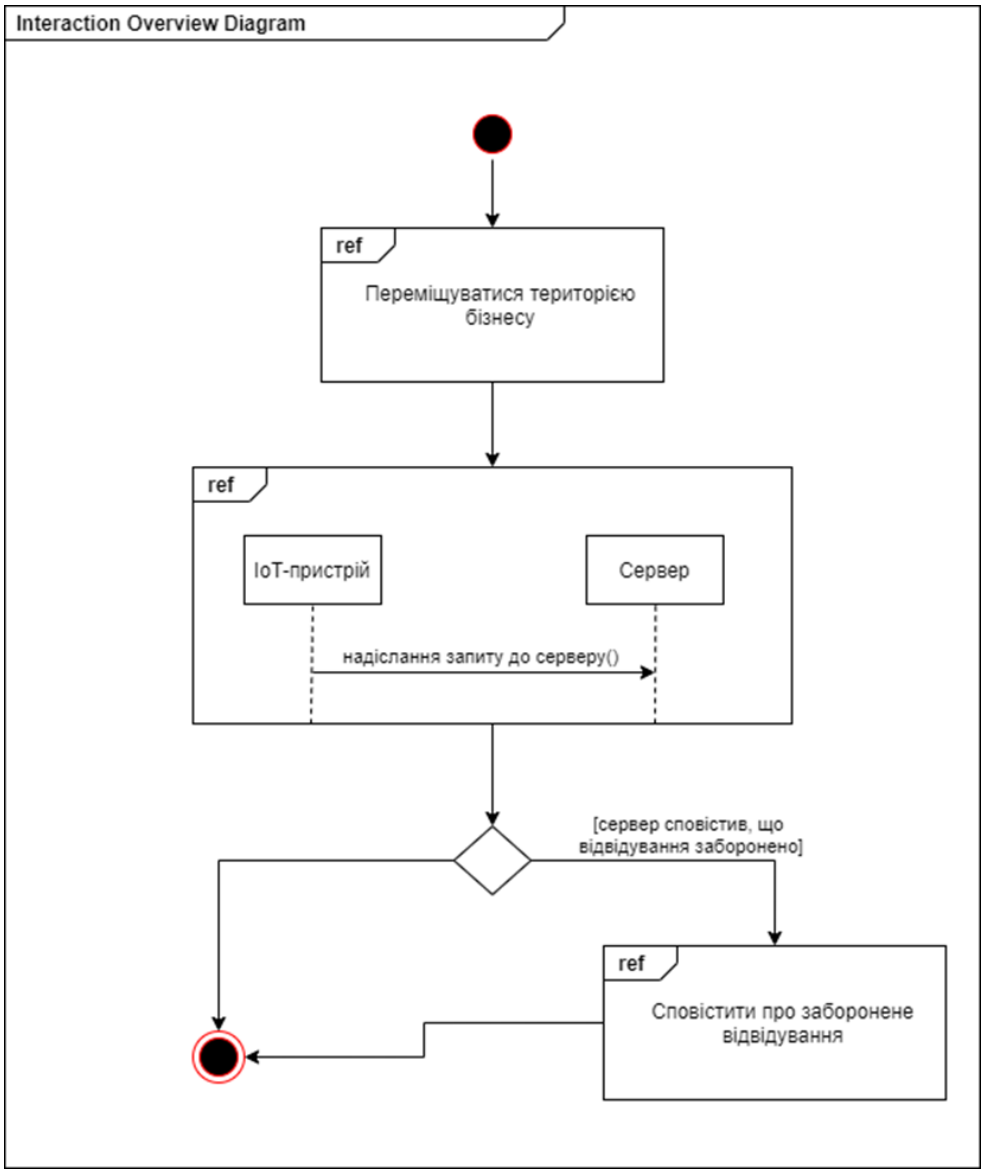

Рисунок 5.15 – Діаграма огляду взаємодії

Необхідно відзначити, що дії, зображені на діаграмі, є послідовними та виконуються протягом життєвого циклу додатку. Це дозволяє більш чітко зрозуміти принцип роботи програмного застосунку.

5.6.2 Демонстрація роботи IoT-застосунку

Наведемо приклад процесу ініціалізації розумного пристрою (термінал-зчитувач BLE-міток) та зобразимо його на рисунку 5.16

```
BLE Smart Device Software v. 1.0.1.1. Illia Trofunenko
--------------------------------------------------
[INIT] Device launch requested...
[INIT] Initializing sensor interface...
[INIT] Testing sensor availability...
[INIT] Testing sensor proximity...


[INIT] Initialization. Please wait...
[INIT] Initialization. Please wait...
[INIT] Initialization. Please wait...

[INIT] Checking server availability...
Configuration...

Smart Device terminal is ready to gather data about BLE beacons in radius!
```

Рисунок 5.16 – Екран ініціалізації терміналу

По завершенню ініціалізації пристрій готовий до фіксації сигналів BLE-міток.

Нагадаємо, що термінали, завдяки можливостям розумного пристрою Raspberry PI можуть здійснювати первинну обробку отриманих значень з датчиків, після чого надсилають інформацію на сервер. До первинної обробки відноситься обчислення рівню сигналу згладженого RSSI, алгоритм обчислення якого описаний у розділі 3.2.1, а також можуть робити обчислення координат місцезнаходження мітки відносно території приміщення.

Наступним кроком, перевіримо роботу розумного пристрою за допомогою міток.

Після завершення ініціалізації девайсу, він має сповістити про готовність до роботи. На цьому етапі можна вносити мітки і фіксувати їх переміщення.

Наведемо приклад консольного інтерфейсу розумного пристрою для покращення розуміння даних, які ним записуються та обчислюються (див. рис. 5.17).

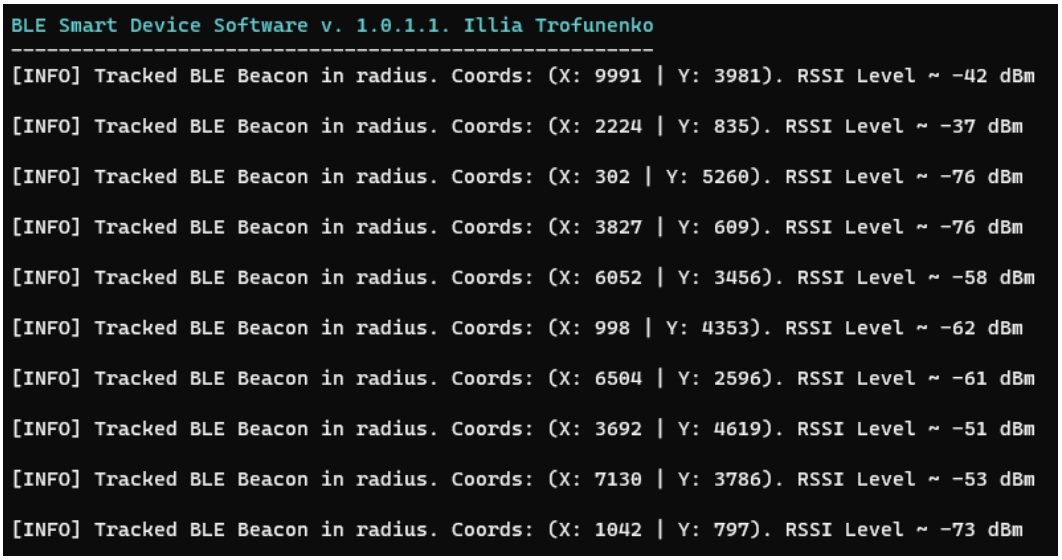

Рисунок 5.17 – Екран фіксації відвідування

Таким чином, було спроєктовано та розроблено програмно-апаратне рішення для вирішення задач позиціонування та відстеження предметів загального користування.

Підсумовуючи, можемо зробити висновки стосовно всієї розробленої системи. Ключова перевага отриманого комплексного програмного рішення це його ефективність та швидкодія, яких було досягнуто завдяки дослідженням проведеним у розділі 3, а також прийнятим рішенням щодо принципів та шаблонів проєктування програмного забезпечення з розділу 4. Крім того, система має додаткові переваги, а саме: високий рівень надійності та безпеки, розподілене кешування, істотні можливості для подальшого масштабування, які були закладені в кожний модуль системи, що розробляється. Стосовно перспектив розвитку такої системи необхідно відзначити необхідність подальшої інтеграції моделей штучного інтелекту у серверну частину, що дозволило б отримати додаткові оптимізації, а також можливості прогнозування шляху переміщення об’єкту.

ВИСНОВКИ

В результаті виконання кваліфікаційно роботи було проведено ряд досліджень та аналіз отриманих результатів з метою проєктування та розробки програмної системи автоматизації процесу керування предметами загального користування у комерційних структурах. Така система дозволяє різним типам бізнесу (наприклад, логістичним та транспортним компаніям, шерінг-сервісам, будівельним компаніям, торгівельним та офісним центрам, виробництвам тощо) отримувати актуальну та розгорнуту статистику про переміщення визначених об’єктів та предметів загального користування. Використання такої системи може значно підвищити прибутковість та покращити безпекові аспекти різних типів комерційних структур за рахунок аналізу статистики, отриманої за допомогою розроблюваної системи. Значними перевагами реалізованої системи перед її потенційними конкурентами є: нижча вартість впровадження та обслуговування, широкий спектр структур, які будуть зацікавлені у використанні цієї системи, а також підтримка веб- та мобільної версії сервісу.

В ході виконання роботи було проведено аналіз предметної області, в процесі якого було розглянуто основні методи автоматизації процесів фіксації даних про моделі переміщення об’єктів. Було проведено аналіз існуючих конкурентів, а також аналіз ринку та виявлено їх основні можливості та недоліки. На підставі аналізу предметної області було виконано постановку задачі та виконано експериментальне дослідження з метою порівняння ефективності існуючих алгоритмів позиціонування, а також розроблено гібридний алгоритм, що довів свою ефективність та певні переваги у порівнянні з існуючими підходами. Описаний алгоритм було використано у процесі реалізації програмної системи з метою максимальної оптимізації та підвищення ефективності процесу автоматизації переміщень об’єктів та предметів загального керування.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Система позиціонування в приміщенні – Sewio Indoor Location System. URL: https://www.sewio.net/indoor-location-tracking-and-positioning/ (дата звернення: 26.01.2023).

2. Система позиціонування на основі GPS – ArcGIS IPS. URL: https://www.esri.com/en-us/arcgis/products/arcgis-ips/overview (дата звернення: 27.01.2023).

3. MarketsAndMarkets Market global forecast to 2027. Дата оновлення: 03.02.2023. URL: https://www.marketsandmarkets.com/Market-Reports/indoor-location-market-989.html (дата звернення: 07.02.2023).

4. IndoorAtlas Global Survey Shows Indoor Positioning Explosive Growth and Demand for Geomagneticas Deployments Scale. URL: http://www.indooratlas.com/wp-content/uploads/2016/09/A-2016-Global-Research-Report-On-The-Indoor-Positioning-Market.pdf (дата звернення 01.03.2023)

5. Goswami S. Indoor Location Technologies. Springer New York, NY. 92 с.

6. Evennou F., Marx F. Advanced integration of WiFi and inertial navigation systems for indoor mobile positioning. Eurasip J. Appl. Signal Process, 2006. С. 2-9.

7. Faragher R., Harle R. An analysis of the accuracy of bluetooth low energy for indoor positioning applications; Proceedings of the 27th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS+'14); Tampa, FL, USA. 8–12 September 2014.

8. Ting S., Kwok S.K., Tsang A.H., Ho G.T. The study on using passive RFID tags for indoor positioning. Int. J. Eng. Bus. Manag. 2011. С. 2-15.

9. Luo J., Fan L., Li H. Indoor Positioning Systems Based on Visible Light Communication: State of the Art. IEEE Commun. Surv. Tutor. 2017. С. 287-293.

10. Kao C.H., Hsiao R.S., Chen T.X., Chen P.S., Pan M.J. A hybrid indoor positioning for asset tracking using Bluetooth low energy and Wi-Fi; Proceedings of the 2017 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW); Taipei, Taiwan. 12–14 June 2017.

11. Robesaat J., Zhang P., Abdelaal M., Theel O. An Improved BLE Indoor Localization with Kalman-Based Fusion: An Experimental Study. Sensors. 2017.

12. Samama N. Indoor Positioning: Technologies and Performance (IEEE Press). Wiley-IEEE Press; 1st edition (June 21, 2019). C. 125-181.

13. Charles K., Chen G. Kalman Filtering. Springer Cham, 29.03.2017. 247 c.

14. Fowler M. UML Distilled: A Brief Guide to the Standard Object Modeling Language. Addison-Wesley Professional, 15.09.2003 – 3rd edition. 208 c.

15. Dehghani Z., Ford N., Richards. M. Sadalage P. Software Architecture: The Hard Parts: Modern Trade-Off Analyses for Distributed Architectures. O'Reilly Media; 30.11.2021 – 1st edition. 345 c

16. Baptista G., Abbruzzese F. Software Architecture with C# 10 and .NET 6: Develop software solutions using microservices, DevOps, EF Core, and design patterns for Azure. Packt Publishing, 15.03.2022 – 3rd edition. 714 c.

17. Nielsen J. Usability Engineering. Morgan Kaufmann, 23.09.1993 – 1st edition. 384 c.

18. Schulz A., Schulz B. Perfect Scale. Detail, 18.12.2015 - 2nd edition. 144 c.

19. Teorey T., Lightstone S., Nadeau T. Database Modeling and Design. – Elsevier, 2006. – 296 P. – ISBN 978-0-12-685352-0.

20. Meier A., Kaufmann M. SQL & NoSQL Databases: Models, Languages, Consistency Options and Architectures for Big Data Management. – Springer Vieweg, 2019. – 248 P. – ISBN 978-3658245481.

21. Chetverikov G., Puzik O., Vechirska I. Multiple-valued structures of intellectual systems //Proceedings of the with Internations Computer Sciences and Information Technologies (CSIT). 2016, 7589907. -pp. 204-207