Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра інформатики факультету інформатики

Архітектура системи захисту рухомих об'єктів

Текстова частина до курсової роботи
за спеціальністю «Комп’ютерні науки» - 122

Керівник курсової роботи
доцент
Гороховський С.С.

_____

(Підпис)
“___” _____ 2022 року

Виконав студент КН-МП1
Алєксєєв А.В.
“___” _____ 2022 року

Київ 2022

ЗМІСТ

АНОТАЦІЯ

Робота розглядає проблему захисту значної частини рухомих транспортних засобів, що потребують охорони – велосипедів, самокатів, гіроскутерів, скейтбордів, автомобілів (за потреби у системі захисту). Наведено порівняння та основні особливості вже створених систем захисту, виділено переваги та виявлено недоліки.

В рамках виконання роботи, спроектовано архітектуру системи захисту транспортних засобів, розроблено бекенд та фронтент частину.

Оскільки засоби пересування мають різну форму, функціонал та розміри, отримана система є універсальною та гнучкою для використання. Розроблена система може використовуватися для будь-якого виду велосипедів, самокатів (електро чи звичайних), а також інших засобів, які набули популярності на сучасному ринку.

Система дає змогу відслідковувати стан об'єкту захисту, знаходячись на будь-якій відстані від нього, за наявності підключення до мережі. Крім того застосунок має можливість повідомляти про будь-яку спробу пограбування.

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ

Охоронна система – автоматизований комплекс, що забезпечує захист майна.

GPS – набір засобів, які надають можливість визначати місцезнаходження об'єкта у просторі. Положення об'єкта визначається за допомогою GPS-приймача, що обробляє сигнали супутників [1].

GSM - міжнародний стандарт для мобільного цифрового стільникового зв'язку з розділенням каналу за принципом TDMA та високим рівнем безпеки за рахунок шифрування з відкритим ключем, створеним Європейським інститутом стандартизації електрозв'язку (ETSI) [2].

Клієнт - компонент системи, наділений можливістю надсилати запити на сервер, завдяки встановленому протоколу передачі даних.

Сервер - програма, яка надає послуги іншим програмам, що називають клієнтами. Зв'язок клієнт/сервер здійснюється за допомогою передачі повідомлень, використовуючи певний протокол зв'язку, та використовує алгоритми кодування запитів клієнта і відповідей сервера.

Геркон - електромеханічний пристрій, в основі роботи якого лежить перемикач, керований магнітним полем.

Акселерометр – засіб що дає змогу вираховувати швидкість об’єкта, визначати рівень вібрації або нахилу.

Тампер – пристрій, що надає можливість повідомити про відкриття/закриття корпусу об’єкту.

ІДЕ – програмний комплекс, що надає можливість розробки програмного забезпечення.

Калібрування – сукупність дій, які покращують точність вимірювального пристрою, зменшуючи його систематичну похибку. Це задає точність вимірювання пристрою.

Проксі сервер – програма, що надає можливість клієнтам робити запити до мережевих сервісів, використовуючи посередника, з метою звхисту від мережевих атак, кешування, стиснення даних або їх зміна, тощо.

Міграція БД – процес керування змін схеми реляційної бази даних з метою її оновлення чи повернення до певної версії.

DNS-сервер – програма, що дає змогу відповідати на DNS-запити, використовуючи певний протокол.

ВСТУП

Теперішній світ технологій зазнає покращень щодня. Цей прогрес не оминає і область захисту як інформації, так і майна. При цьому надійність, стабільність та рівень захисту є головними критеріями оцінки захисних систем. Щодо захисту майна – існує безліч прикладів для будь-якої сфери життя: будинків, машин, робототехніки. Але на цьому ринку доволі важко знайти повноцінну систему, що буде охороняти менш цінні з фінансової точки зору техніки – бюджетних повсякденних транспортних засобів, таких як велосипеди, самокати, гіроскутери, тощо. Через це користувачі не мають змоги залишати без нагляду ці засоби, що зменшує попит на користування цими об’єктами.

Відсутність повноцінних систем не означає те, що систем захисту велосипедів не має взагалі. Це означає те, що існуючі системи не задовільняють користувача своїм рівнем захисту, особливо звертаючи увагу на простоту викрадення чи зламу таких транспортних засобів.

Зручність у використанні є ще одним основним фактором для користувачів охоронних систем. Окрім невизначеності з надійністю захисту, системи для рухомих об’єктів у своїй більшості не мають мобільних застосунків, а частина з них не мають можливості давати фідбек користувачу – наприклад інформувати про тривоги, відображати стан системи в цілому, ставити чи знімати девайс з охорони. Це викликає ще ряд незручностей, до яких сучасні користувачі не звикли.

Ще одним важливим критерієм є можливість встановлювати стороннє втручання до системи зі змогою повідомлення про такі випадки користувача. До таких випадків належать спроби злому захисту, вимикання чи заглушення датчиків, що відображають поточний стан охорони. Окрім наявності перевірки таких кейсів, система не має втрачати у швидкості роботи. Оскільки на рухомих об’єктах не має можливості встановити дротову систему, цей критерій є також дуже важливим, адже передачу даних у мережі або через радіопротоколи можна контролювати.

При дослідженні охоронних систем на ринку рухомих об'єктів, можна виявити багато слабких місць, що можна обійти. Це може бути незручність у експлуатації, низький рівень захисту, простота злому. Технології, які використовуються у цих системах є застарілими, часто передбачуваними і не надають надійність, яку гарантують системи у сфері захисту нерухомості. Отже головними вимогами для розробки є: висока надійність та наявність мобільного додатку, що буде відображати будь-які зміни системи та крім того, надавати змогу повноцінного керування.

Створення системи захисту рухомих об'єктів з використанням сучасних технологій є досить актуальною проблемою у сфері захист. Наприклад, аналізуючи вулиці Києва та інших промислових або економічних центрів України, охорону велосипеда використовує дуже незначна частка користувачів [4]. Це пов’язано з непопулярністю таких систем, їхньою ненадійністю та відсутності керування через телефон. Більшість користувачів велосипедами звертається до досить застарілих методів – велозамків. Але це рішення дуже ненадійне, бо злом такої охорони є повсякденним для досвічених грабіжників. Зовсім плачевною є стан ринку захисту інших об'єктів – таких як самокати чи гіроскутери, скейти. Ці об'єкти користувачі звикли взагалі не залишати без нагляду, адже складно звичними методами захистити такі засоби і бути впевненими у їх надійності. Але це проблема лише ринку захисних систем, адже можливості для захисту таких об'єктів не менші, ніж для велосипедів.

Під час проектування, слід врахувати, що систему з наявним GPS не можливо ввести в оману – вона має декілька рівнів захисту та надає змогу миттєво попереджувати користувача про загрозу. Оскільки більшість транспортних засобів, для яких буде використовуватися розробка, не мають змоги фіксуватися у просторі (ланцюгом з замком, наприклад), необхідно використати засоби відлякування грабіжників та привертання оточуючих – бузери чи сирени.

Використання GSM модуля сприяє зручності у керуванні через мобільний додаток.

У цій роботі буде звернено увагу на реалізацію тієї частини системи, що має відношенняя до спеціальності комп'ютерих наук – серверної та клієнтської частини. Реалізація Embedded частини не входить до цілей роботи, а буде застосовуватися лише для проектування системи.

Об’єктом дослідження є сфера охорони рухомих об’єктів.

Предметом дослідження є автоматизовані системи захисту бюджетних транспортних засобів з використанням мобільного додатку.

Мета дослідження: створення на основі існуючих систем власної, що матиме зручніше керування та більшу надійність.

Задачі дослідження:

1. Аналіз сфери охорони рухомих об’єктів. Порівняння аналогів між собою та виділення їх переваг і недоліків. Визначення актуальності дослідження.
2. Визначення технологій для розробки мобільного додатку та бекенд частини.
3. Проектування архітектури системи, структури бази даних для серверу.
4. Розробка апаратної та програмної частини системи, а саме: сервер та мобільний додаток за спроектованою структурою.

1. АНАЛІЗ ІСНУЮЧИХ СИСТЕМ

1.1. Аналіз існуючих систем

Для розуміння ситуації на ринку систем захисту, перед розробкою було проведено пошук та аналіз вже створених аналогів для порівняння між собою. Це дає змогу встановити чіткі критерії для проектування моделі та розробки системи. Порівняння систем між собою чітко визначає засоби, необхідні для використання у проекті, надає змогу виділити технології та механізми, що мають переважати у кожній з частин реалізації системи.

Було проаналізовано переваги та недоліки наявних на теперішній час аналогів. Критерії для оцінки та порівняння систем:

- особливості функціоналу;
- взаємодія з користувачем;
- надійність;
- швидкість роботи.

До уваги були взяті системи:

1. Ajax Systems
2. Системи, розроблені компанією «MCC Ukraine»
3. Системи захисту самокатів, велосипедів, мотоциклів «Wialon»
4. Велосигналізація «Спартак»
5. Сигналізація «Bicycle Alarm ДУ»

Через незасвоєність ринку захисту рухомих об’єктів, більша частина, конкурентних виробників, як не дивно, не мають власного сайту, а приклади їхніх систем можна знайти лише на сервісах продажу.

Проаналізуємо кожну систему, що потрапила під порівняння.

Ajax Systems не мають сигналізації для рухомих об’єктів, але є гарним прикладом реалізації бездротового захисту. Наразі Ajax виступають самою надійною бездротовою системою захисту нерухомості у Європі [6]. Система працює на власному радіопротоколі, і передає дані через Ethernet або GSM. Мінімалістичний дизайн, велика кількість документації та використання міжнародних стандартів у області охорони спрощують як користування

системою, так і її керування. Власний радіопротокол Jeweler розробник виставляє як гарантований захист від злому[6]. У цій системі дуже складно знайти хоча б один недолік. Проте ця система не дає змогу захисту рухомих об’єктів, хоча концепція системи може бути застосована у цій сфері. Захист транспортних засобів відрізняється від захисту нерухомості тільки відсутністю пультової охорони, що не є необхідністю. Важливим фактором цієї системи є автономність роботи. Декілька років без зміни батарейок – еталон, якого варто притримуватися під час розробки (хоч для транспортних засобів достатньо роботи 24 годин, за умови наявності можливості підзарядки акумуляторів).

Рис. 1.1. Система захисту Ajax Systems

Система «Magnum Bike» від розробника «MCC Ukraine» вже стосується транспортних засобів (у цьому випадку - велосипедів) і маж достатньо широкий функціональний ряд. Наявність взаємодії з користувачем виділяє цю систему і виділяє її як гарне рішення серед аналогів. Аналізуючи систему можна виділити такі переваги:

1. наявність мобільного додатку та брелка;
2. наявність декількох режимів захисту – охорона (звукова індикація на тривоги), трекінг (відслідковування координат велосипеда), маяк (періодичний моніторинг місцезнаходження велосипеду, для зменшення енерговитрат).

Щодо недоліків – вони також є. Серед них слід зазначити:

1. малу автономність (менше доби) – неприйнятно для тривалого користування;
2. неможливість керування системою – мобільний додаток лтше відображає стан системи, що для сучасного користувача замало;
3. Великий розмір самої системи – не підійде для усіх велосипедів, не кажучи вже про використання для інших транспортних засобів.

Велосигналізація «Спартак» на противагу попередній системі має дуже малий розмір, що вже надає змогу встановити цю систему на самокати, гіроскутери, тощо. Крім того, система надає можливість керування: 2 режими – «під охороною», та «знято з охорони». Розробник виділяє ще одну значну перевагу системи – більше 48 годин автономності у режимі під охороною що в цілому достатньо для такої системи. Але можна виділити такі недоліки – відсутність мобільного додатку (керування лише за допомогою кнопки), не має захисту від злому – якщо систему помітити, то зняти її не скласть багато зусиль. Надійність системи також бажає чекати на покращення.

Сигналізація «Bicycle Alarm ДУ» є одним з найдорожчих продуктів на ринку захисту велосипедів. Має привабливий дизайн та зручне керування. Крім того малі розміри та непомітність у встановленні, що у таких спосіб вирішує проблему зовнішнього втручання. До того ж, має декілька різних реалізацій, що надає змогу підібрати систему під власні потреби. Проте, має ряд недоліків, серед яких можна виділити низьку автономність (розробник сам заявляє про це, рекомендуючи постійно заряджати акумулятори), а відсутність мобільного додатку тільки збільшує цю проблему, що зменшує зручність у користуванні.

На ринку захисту самокатів можна виділити систему від компанії «Wialon». Мають декілька прикладів систем з різним функціональним вмістом. Головними перевагами цих систем слід виділити: наявність GPS трекера, що дозволяє відслідковувати місцеположення об'єкту захисту, компактний розмір, непомітність встановлення (встановулюється як ліхтарик, або ж взагалі міні-трекер не досяжний людському зору), висока автономність (800 мА*ч з низьким рівнем споживання електроенергії забезпечує автономність протягом

48 годин), декілька режимів роботи з різним енергоспоживанням, наявність зворотнього зв’язку з користувачем. Щодо зворотнього зв’язку – реалізація цього механізму є досить примітивною – відправлення повідомлення на ваш номер телефону з статусом системи та її місцеположенням при тривогах. Головними недоліками такої системи є відсутність мобільного додатку, незручність у керуванні, спричинена реалізацією цього механізму через SMS повідомлення. Крім того у розробника «Wialon» наявні системи захисту для мотоциклів. Ці системи мають більший функціонал, надійніші механізми захисту та покращений зворотній зв’язок. Але є значний недолік – системи захисту мотоциклів розробник зробим набагато більшим за розмірами та енергоспоживанням, що унеможливлює її встановлення на інші транспортні засоби.

Аналізуючи описані системи, можна зробити такі висновки. По-перше, аналогів системи, що може бути встановлена на будь-які транспортні засоби майже немає на ринку (можна виділити лише продукцію компанії «Wialon»). Більшість розглянутих систем захисту рухомих об’єктів не мають мобільного додатку (виключенням є «Magnum Bike»), а керування виконується застарілими та не зручними способами.

Модуль передачі даних через модем майже не використовується у аналогах, хоча без наявності мобільного додатку, він не є обов’язковою складовою. Жодна з проаналізованих систем не має навіть мінімальної документації, а пояснення нюансів у використанні слід шукати серед форумів охоронних систем, що значно ускладнює експлуатацію системи користувачем.

З розглянутих систем лише «Magnum Bike» має достатній механізм захисту від зовнішнього втручання («Bicycle Alarm ДУ» як було зазначено, лише маскує систему, але іншого захисту від злому не передбачено). Інші системи не захищають користувача від злому зовсім – систему не складно демонтувати, або зруйнувати, що не надасть змогу користувачеві дізнатися про викрадення або злам.

Усі описані переваги та недоліки будуть мати місце під час розробки власного проекту.

1.2. Протоколи зв’язку та автентифікація

Протоколи зв’язку – це встановлений розробником механізм передачі даних від одного пристрою до іншого і навпаки.

Розглянемо такі протоколи:

1. Мережеві протоколи
2. Протоколи, основані на передачу через дроти
3. Радіопротоколи

Системи захисту рухомих об’єктів можуть використовувати різні механізми. Використання мережевого протоколу веде до значного збільшення споживання електроенергії системою, але є найшвидшим рішенням. Системи, що працюють лише на цьому протоколі, найбільше підходять під концепцію смарт будинків, адже можуть передавати великі вибірки даних. Проблемою використання цього протоколу також може буди надійність захисту інформації. Для впевненості у надійності системи необхідно використовувати алгоритми шифрування даних, які бувають як схильні до злому, так і навпаки незламними.

Задля реалізації цього механізму, необхідно реалізувати сервер, на який будуть приходити зашифровані пакети даних. У свою чергу, сервер розшифровуватиме ці пакети та передає користувачеві у читабельному вигляді.

Майже усі повноцінні (насамперед це означає наявність мобільного додатку) системи охорони використовують мережеві протоколи передачі даних, бо він дозволяє передавати дані з будь-якої точки світу на інший континент, тощо[4].

Дротова передача даних застосовується у переважній більшості систем, усі компоненти якої під’єднуються через дроти. Передача даних дуже швидка, надійний захист від втручання ззовні реалізувати дуже просто. Протокол можна вважати достатньо надійним. Але він має один великий недолік – вибірка переданих даних замала.

Через один канал є змога передавати лише один біт інформації, що ускладнює проектування системи, збільшує кількість каналів, а це веде до отримання незручностей у налаштуванні та користуванні системою. Для рухомих об’єктів використання цього протоколу неможливе, адже система знаходиться у русі та не може прив’язуватися до одного місця.

Радіпротоколи – найбільш просунутий спосіб передачі у сфері охорони. Має багато нюансів, таких як вибір частоти передачі даних у радіоефірі, протокол шифрування інформації, складнощі у реалізації захисту від зовнішнього втручання. Такі протоколи з відкритим кодом легко обходяться досвідченими злодіями, тому для надійної системи необхідно використовувати комерційні протоколи з закритим кодом. Прикладом реалізації протоколу є jwl, розроблений «Ajax Systems». Як зазначалося, плюсами цього протоколу є швидкість передачі даних, неможливість зламу, та комерційний механізм шифрування даних від розробника[6].

Зовнішнє втручання у системах з використанням радіопротоколу можливе завдяки заглушенню частот роботи, або відправки підлаштованих пакетів, які система може хибно вважати за очікувані. Для запобігання таких способів зламу, достатньо реалзіувати на стороні протоколу перевірку на глушіння частот (аналізуючи радіоефір) та підрахунок контрольної суми під час передачі даних, та інші способи – такі як нумерація пакетів, створення механізмів автентифікації пристроїв та багато інших. Будь-яка спроба втручання у систему має своєчасно відображатися у мобільному додатку.

Радіопротокол суттєво зменшує споживання системи, адже передача даних відбувається за короткий проміжок часу з певним інтервалом (фреймом). Коли необхідно здійснити передачу даних, передавач (датчик) переходить у режим відправки пакету (увесь інший час датчик споживає мінімальну частку електроенергії) на декілька мілісекунд, і після отримання відповіді про успішну доставку пакету знов так би мовити засинає до наступного фрейму. Для цього механізму складно виділити хоча б один недолік, лише можна зазначити, що

він складний у реалізації, адже у ньому взаємодіють усі механізми між собою, і похибка у роботі одного механізму веде до некоректної роботи усієї системи.

Щодо складності реалізації, система має містити антену на приймачі та передавачі задля взаємодії між собою. Дальність роботи таких систем обмежена відстанню до 1-2км. Отже за умови, що приймачем є мобільний додаток, а передавачем система, встановлена на рухомий об’єкт, користувач не може відійти далі від транспортного засобу більш ніж на встановлену антенами відстань, що є значним мінусом.

Оскільки розробка має реалізувати можливість будь-якого користувача під’єднатися до будь-якої системи, наявної у нього, обов’язково необхідна реалізація механізмів автентифікації між приймачем та передавачем.

Для систем з радіпротоколом, проблема автентифікації вирішена беспосередньо у реалізації самого протоколу. Зазвичай передавач запам’ятовує приймач по встановленому індивідуальному номеру, та передає дані з використанням цього айді. Приймач же фільтрує усі запити, вибираючи серед них лише адресовані саме йому. Далі він перевіряє чи належить передавач до пристроїв, що знаходяться під його керуванням. Якщо належить – то дані оброблюються, та передаються користувачу, а якщо не належить – то скоріш за все ці дані будуть проігноровані (у деяких випадках користувачу може бути повідомлено про спроби зламу системи).

Передача даних користувачу з будь-якої системи не можлива без попередньої автентифікації та синхронізації. Синхронізація реалізовується за допомогою персонального ідентифікатора користувача, та персонального ідентифікатора системи.

Для цієї реалізації система має мати базу даних. Тобто при передачі даних через, наприклад, мережевий протокол, буде передаватися ідентифікатор системи. На сервері цей ідентифікатор буде прив’язаний до певного користувача. Додавання цього взаємозв’язку реалізується при реєстрації системи користувачем у мобільному додатку. Тоді будь-який запит буде мати свого адресата.

Аналізуючи розглянуті протоколи, можна зробити висновок, що найкращим вибором для реалізації системи є мережевий протокол, адже він гарантує доступ до системи з будь-якої точки світу. При embedded розробці безпосередньо фізичної складової системи (плати та програмування датчиків) слід звернути увагу на можливість використання радіопротоколу задля мінімізації споживання струму і покращенням автономності системи.

1.3. Концепція системи та постановка задачі

Отже у пункті 1.1 проаналізовано переваги та недоліки існуючих систем, і на основі отриманих результатів з'являється концепція системи, що буде вдосконалювати переваги аналогів та мінімізувати їхні недоліки. Система має бути бездротовою, передача даних на сервер – завдяки мережевим протоколам, з наявним шифруванням даних. Усі спроби зламу мають надходити користувачу своєчасно з детальним поясненням тривоги.

Тож, вимогою до реалізації сервера є наявність захисту від спроб втручання– DoS та інших атак.

Про будь-яку зміну поточного стану системи, користувач має дізнаватися миттєво, отримуючи інформацію у мобільний застосунок. Під поняттям зміни стану розуміється безпосереднє втручання зловмисників до транспортного засобу – спроба зрушити об'єкт з місця, нахилити його, чи від'єднати охоронну систему від об'єкту захисту.

Крім цього важливою складовою є можливість відслідковувати механічні пошкодження (наприклад несправність тамперу чи вихід зі строю акселерометра), інформувати про розряд батареї. Якщо хоча б один з наведених пунктів буде відсутній, система стане уразливою до втручання ззовні і може не відпрацювати під час тривог.

Вимога до автономності системи – знаходитись у робочому стані не менш ніж тиждень. Бажано мати можливість роботи і більші проміжки часу, але це не має впливати на розмір фізичної частини системи, адже вона матиме можливість встановлюватися на маленькі транспортні засоби. Крім того необхідно додати можливість зручної підзарядки акумуляторів.

У курсовій роботі метою є розробка системи захисту рухомих об’єктів. До цієї системи входять: мобільний застосунок для керування та зворотного зв’язку, сервер для обробки подій та безпосередньо запрограмована плата з GSM модулем, що фізично встановлюється на велосипед. Як вже було зазначено, у цій роботі буде наведено реалізацію серверу та мобільного додатку.

Реалізуючи серверну частину та мобільний застосунок, я намагався дотримуватись вищеописаних критеріїв та наблизити роботу системи до ідеалу.

Вимоги до розробки: швидкодія роботи, реалізація захисту від зовнішнього втручання у систему на стороні серверу та мобільного додатку.

Висновки до розділу

1. Проаналізовано системи зі схожим функціоналом. Перша з наведених, не має прикладів охорони транспортних засобів, але показує якою має бути система захисту в ідеалі. Друга наведена система працює без підзарядки 2 дні, що замало для користувача, має незручне керування і встановлюється лише на велосипед. Третя система має можливість встановлення на різні транспортні засоби але не має мобільного додатку та реалізує застарілий метод зворотнього зв’язку. Інші системи взагалі не мають ані зворотнього зв’язку, ані захисту від зовнішнього втручання.

2. У розділі проведено аналіз аналогів та сформовано концепцію для розробки нової системи.

3. Проведено аналіз трьох протоколів передачі даних, описано їхні переваги та недоліки, зроблено висновки щодо вибору протоколу для розробки.

2. ОПИС ТЕХНОЛОГІЙ, ЩО ВИКОРИСТОВУЮТЬСЯ ДЛЯ РОЗРОБКИ СИСТЕМИ

2.1. Принципова модель системи

В цьому розділі представлено опис технологій , що використовувались для розробки серверної частини та клієнтської частини мобільного додатку.

Серверна частина була зроблена на мові програмування Java з використанням середовища IntelliJ IDEA Community Edition 2022.1.1. База даних розроблена з використанням СУБД MySQL у середовищі MYSQL Workbench. Деплой серверної частини зроблено на хмарній платформі Heroku, а проксі-сервер написано на NodeJS за допомогою середовища IntelliJ IDEA. Мобільний застосунок реалізовано за допомогою технологій React, react-router, фреймворк фреймворк Electron та Chakra UI.

Для розуміння функціоналу системи треба розробити модель системи, адже сервер та мобільний додаток відображають інформацію, яка буде отримана на стороні фізичної частини – плати, реалізація якої не входить у мету роботи.

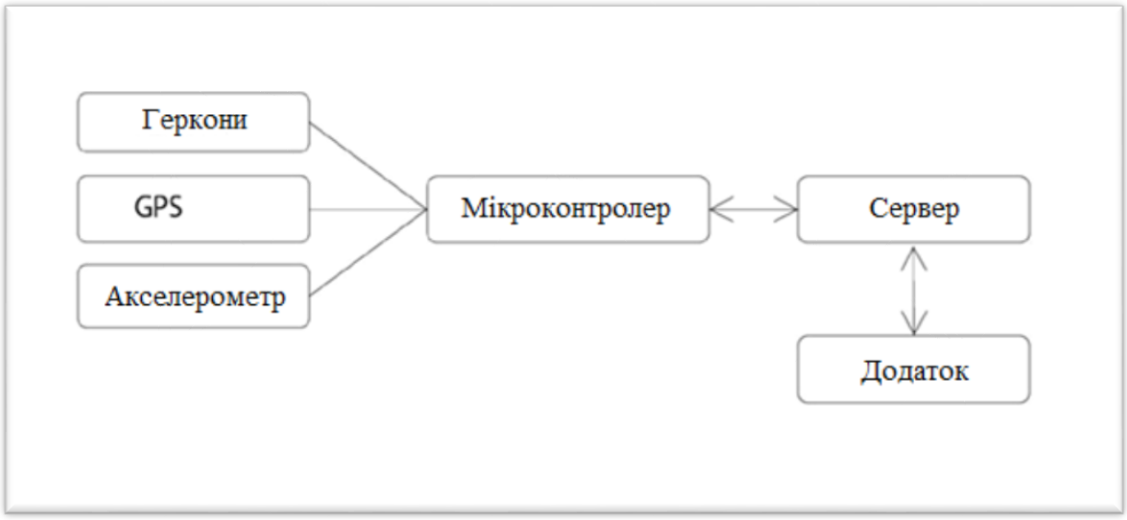

Рис. 2.1. Модель системи

Запрограмований мікроконтролер використовує GSM модуль для передачі даних на сервер, а також має антену і змогу передавати дані по

радіопротоколу (для взаємодії з бездротовими елементами – тамперами, акселерометром, герконами, тощо).

Передача на сервер реалізована через мережевий протокол. Сервер оброблює отримані від мікроконтролера дані, та оновлює таблицю у базі даних. Клієнт з певним інтервалом моніторить зміни у базі та відображає їх у відповідних меню додатку.

2.2. Інформаційне забезпечення системи

Серверна частина буде розроблена на мові програмування Java з використанням Spring Boot. Java – об’єктно орієнтована мова програмування заснована на класах. Має невеликі залежності реалізації, що надає гнучкості і можливості роботи на будь-якій операційній системі. Джава є статично типізованою і безпечною мовою, що надає перевагу саме цій мові у порівнянні її з прямим конкурентом – NodeJS. Крім того ця мова надає зручні можливості у реалізації multithreading, а через це оптимально розподіляє навантаження ЦП. Надійне керування пам’яттю та паралельне накопичення також є плюсами розробки на Java. Крім того Java відома тим, що має дуже стабільні інтерфейси, завдяки своїй перевіреній часом структурі. Через це програми на Java мають високу стабільність роботи, і зміни у API не викликає жодних проблем. Крім того, Java має велику документацію, що значно спрощує розробку. Крім того, сервер мусить працювати з великими обсягами даних і Java з цією проблемою гарно справляється.

Spring boot спрощує створення автономних програм промислового рівня на основі Spring, які можна «просто запустити». Він дозволяє простими способами створювати веб додатки, не забираючи у розробника багато зусиль на налаштування та написання коду. Spring Boot має ряд особливостей – він має можливість керувати залежностями, автоматичну конфігурацію, вбудовані контейнери сервлетів.

База даних серверу буде розроблена на MySQL у СУБД MySQL Workbench. MySQL був розроблений компанією «ТсХ» для підвищення швидкодії обробки великих баз даних. Ця система керування базами даних

(СКБД) з відкритим кодом була створена як альтернатива комерційним системам. MySQL з самого початку була дуже схожою на mSQL, проте з часом вона все розширювалася і зараз MySQL — одна з найпоширеніших систем керування базами даних. Вона використовується, в першу чергу, для створення динамічних веб-сторінок, оскільки має чудову підтримку з боку різноманітних мов програмування.

Основною вимогою для розробки мобільного додатку є можливість запускати його на будь-якій операційній системі. Для реалізації було обрано JavaScript бібліотеку для створення користувацьких інтерфейсів React з використанням react-router та Chackra UI.

React - JavaScript бібліотека з відкритим кодом, що надає змогу написання інтерфейсів користувача та покликана вирішувати проблему динамічного оновлення вебсторінки, яка виникає при розробці односторінкових застосунків. React надає можливість розробникам створювати застосунки, що використовують дані, які можуть змінюватися у реальному часі, не роблячи при цьому перезавантаження сторінки. Головна мета розробки полягає у тому, що додаток має високу швидкодію, простий у використанні та реалізації, масштабований. React обробляє тільки користувацький інтерфейс у застосунках.

Використання фреймворку Electron є сучасним рішенням для надання можливості додатку працювати на будь-якій операційній системі. За допомогою цієї технології розроблені такі додатки як Atom, VS Code, Discord, які є популярними не тільки через кросплатформенність, а і через швидкодію та зручність у викоистанні.

Висновки до розділу:

У цьому розділі було розглянуто технології, якими користувались для створення системи.

Клієнтська частина була розроблена на React з використанням фрейморку Electron. Реалізована для використання на будь-якій операційній системі.

Середовище розробки – IntelliJ IDEA, що є найзручнішою IDE для розробки на обраній мові програмування.

Серверна частина реалізована на Java – однією з найпопулярніших платформ для реалізації мережевих застосунків. Дозволяє оброблювати велику кількість запитів одночасно. IDE для написання серверу – IntelliJ IDEA. IDE має зручний інтерфейс та надає можливість рефакторингу коду.

3. ОПИС ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1. Архітектура системи

Система складається з декількох підсистем.

Рис. 3.1. Структура системи

Embedded підсистема, полягає у роботі мікроконтролера. Він обробляє дані, отримані з модулів, керує цими модулями та виконує взаємодію з сервером. Ця частина системи отримує інформацію, яку датчики передають по радіопротоколу – стан герконів, акселерометра та тамперів. Кожна зміна у стані будь-якого модуля є приводом миттєво формувати зашифрований пакет даних для відправки на сервер.

Крім того, мікроконтролер виконує функцію калібрування датчику акселерометра – цю інформацію передавати на сервер не потрібно, адже калібрування виконується лише один раз після виходу у режим роботи «під охороною». Після процесу калібрування, мікроконтролер моніторить усі зміни у положеннях акселерометра до тих пір, поки користувач не переведе систему у режим «знято з під охорони».

Крім того, незалежно від зміни стану та режиму роботи, мікроконтролер раз у виставлений у додатку час шле умовні «пінги» на сервер – реалізація механізму захисту від зовнішнього втручання на стороні мікроконтролера. Завдяки цим статусам, користувач завжди знає, що система на зв’язку і з нею усе в порядку. GSM модуль веде обробку даних на сервері, і відповідає за

передачу мікроконтролеру даних від користувача. Це забезпечує керування бездротовою системою.

Підсистема серверу відповідає за функціонал обробки даних від користувача та мікроконтролера, аналізу та заповнення таблиць бази даних. Крім того через сервер проходить процеси авторизації та регестрації, додавання систем, їх видалення та модифікація. Передача даних від користувача до мікроконтролера відбувається також завдяки серверу – зміна налаштувань у додатку оновлює інформацію у базі даних, яку мікроконтролер отримує через API запит. Структура БД буде розглянута пізніше.

Клієнт (мобільний додаток) взаємодіє із сервером через API запити – з заданим користувачем інтервалом додаток надсилає запит, що повертає інформацію про системи поточного користувача. При будь-яких змінах у базі, додаток відобразить ці зміни у встановлених меню додатку, яких всього 3 – список усіх систем користувача, інформація про обрану систему та панель пуш-повідомлень.

У меню інформації про систему реалізовано можливість ставити та знімати систему з охорони, а також є можливість змінювати налаштування обраної системи – період опитування на стороні мікроконтролера, вмикання чи вимикання різних датчиків.

У меню зі списком систем відображено усі додані користувачем системи. Якщо у користувача є велосипед та самокат, він може встановити на кожен з засобів охоронну систему та керувати цими системами у одному додатку.

Панель пуш повідомлень містить інформацію про усі зміни, що сталися з поточною системою – постановка/зняття з охорони, тривоги, відновлення тривоги (повернення стану датчика до очікуваного). У цьому меню зазначено дату та час кожного повідомлення для зручності у користуванні.

3.2. Структура бази даних

База даних складається з 5 таблиць – user, user_role, role, systems, flyway_schema_history.

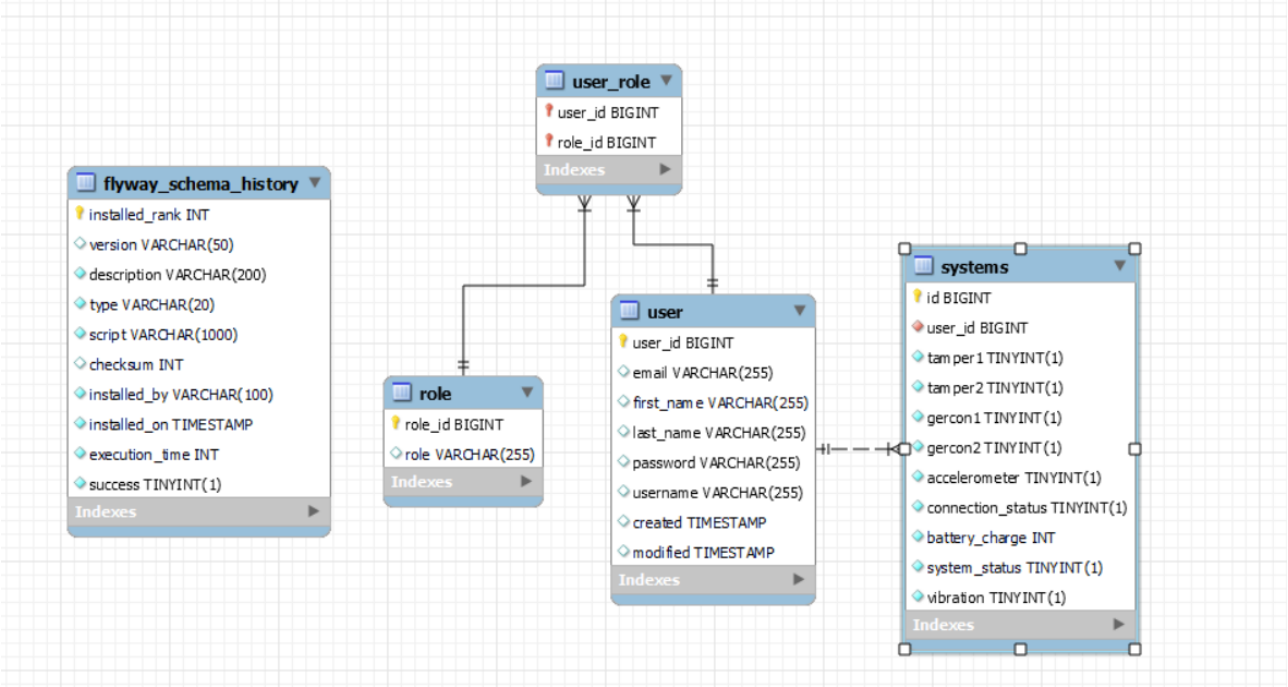

Рис. 3.2. Структура бази даних

Таблиця role містить інформацію про роль користувача – адміністратор чи звичайний користувач. У кінцевому вигляді системи, адміністратор може змінювати налаштування, ставити та знімати систему з охорони, а звичайний користувач лише може переглядати стан системи у реальному часі.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
role_id	BIGINT	☑	☑	☐	☐	☐	☐	☑	☐	
role	VARCHAR(255)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
		☐	☐	☐	☐	☐	☐	☐	☐	

Рис 3.3. Типи даних та Default значення полів у таблиці role

Таблиця має 2 поля – role та role_id та 2 записи – роль адміністратора та роль користувача.

Таблиця user містить усю інформацію про користувачів, зареєстрованих у системі.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
user_id	BIGINT	☑	☑	☐	☐	☐	☐	☑	☐	
email	VARCHAR(255)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
first_name	VARCHAR(255)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
last_name	VARCHAR(255)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
password	VARCHAR(255)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
username	VARCHAR(255)	☐	☐	☑	☐	☐	☐	☐	☐	NULL
created	TIMESTAMP	☐	☐	☐	☐	☐	☐	☐	☐	CURRENT_TIMESTAMP
modified	TIMESTAMP	☐	☐	☐	☐	☐	☐	☐	☐	CURRENT_TIMESTAMP ON...
		☐	☐	☐	☐	☐	☐	☐	☐	

Рис 3.4. Типи даних та Default значення полів у таблиці user

Ця таблиця містить інформацію про пошту, прізвище та ім'я, пароль у хешованому вигляді, юзернейм користувача, а також час створення і зміни кожного запису. Має зв'язок з таблицею user_role – поле user_id є primary key.

Таблиця user_role містить інформацію про роль кожного користувача. Вона пов’язана з таблицями user та role.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
user_id	BIGINT	☑	☑	☐	☐	☐	☐	☐	☐	
role_id	BIGINT	☑	☑	☐	☐	☐	☐	☐	☐	
		☐	☐	☐	☐	☐	☐	☐	☐	

Рис 3.5. Типи даних та Default значення полів у таблиці user_role

Містить 2 поля – user_id та role_id, що ставлять у відповідність кожному користувачу права доступу до системи.

Таблиця systems містить список усіх доданих користувачами систем. Вона має зв’язок з таблицею user, аби давати змогу знати, якому користувачу належить дана система.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
user_id	BIGINT	☐	☑	☐	☐	☐	☐	☐	☐	
tamper1	TINYINT(1)	☐	☑	☐	☐	☐	☐	☐	☐	
tamper2	TINYINT(1)	☐	☑	☐	☐	☐	☐	☐	☐	
gercon1	TINYINT(1)	☐	☑	☐	☐	☐	☐	☐	☐	
gercon2	TINYINT(1)	☐	☑	☐	☐	☐	☐	☐	☐	
accelerometer	TINYINT(1)	☐	☑	☐	☐	☐	☐	☐	☐	
connection_status	TINYINT(1)	☐	☑	☐	☐	☐	☐	☐	☐	
battery_charge	INT	☐	☑	☐	☐	☐	☐	☐	☐	
system_status	TINYINT(1)	☐	☑	☐	☐	☐	☐	☐	☐	
vibration	TINYINT(1)	☐	☑	☐	☐	☐	☐	☐	☐	
frame	INT	☐	☑	☐	☐	☐	☐	☐	☐	'10'

Рис. 3.6. Типи даних та Default значення полів у таблиці systems

У цій таблиці зберігається інформація про стан кожного датчика у системі – тамперів, герконів, акселерометра, вібромоторчика. Також тут міститься інформація про поточний статус заряду батареї у відсотках, статус

системи – під охороною чи знято з охорони, статус про зв'язок системи з сервером (зв'язок є/зв'язок втрачено), період опитування для мікроконтролера (встановлюється користувачем у додатку).

Таблиця flyway_schema_history - таблиця, згенерована фреймворком Flyway, яка зберігає інформацію про міграцію БД.

3.3. Функції та основні дії у системи

Система має три складові – мікроконтролер, сервер та клієнт. Ці частини взаємодіють між собою. Далі будуть наведені основні дії у системі.

На етапі початку роботі із системою, користувач має увійти у додаток. При вводі даних пошти та паролю, додаток робить API запит логіну на сервер, перевіряючи наявність користувача та відповідності паролю, після чого сервер повертає jwt токен у разі успішного виконання запиту, або помилку неуспішного статусу виконання. Якщо запит успішний, користувачу відкривається головне меню додатку. Якщо ж дані акаунту введено невірно, користувачу буде повідомлено, що введено некоректні дані та він залишиться у меню логіну. Якщо облікового запису не існує, користувач має можливість зареєструватися. Після вводу даних про себе, додаток відправить API запит на додавання нового користувача до БД, та поверне результат виконання запиту. Після реєстації відкриється меню логіну, де дані необхідно буде ввести знову для отримання jwt токену.

У головному меню користувач бачить усі системи, прив’язані до його акаунту та має можливість додати нову за необхідності. При додаванні системи необхідно ввести її ID – у кожної системи це значення унікальне. При виконанні API запиту на додавання системи, сервер перевірить чи належить ця система іншому користувачу, і якщо ні – поверне успішний результат виконання запиту та система з’явиться у головному меню. Якщо система вже належить іншому користувачу, сервер поверне неуспішний результат запиту та система не буде додана користувачу.

При видаленні користувача, усі записи з ним будуть видалені, але якщо до нього прив’язано хоча б одну систему – видалення не можливе. Для

успішного видалення користувача, треба видалити усі системи у додатку і тільки після цього запит на видалення буде успішним.

У меню системи користувач може змінювати стан системи та налаштування. У таких випадках буде сформовано запит на зміну запису у таблиці systems. Якщо запит відповідає усім обмеженням, встановленим у БД, статус виконання буде успішним і зміни будуть відображені у додатку.

Для постійного оновлення інформації у додатку, авторизований клієнт має змогу отримати інформацію про усі додані до акаунту системи та інформацію про себе як користувача, використовуючи GET запит. Якщо у користувача ще не маж систем, буде повернено лише інформацію про юзера.

Висновки до розділу

У даному розділі було наведено архітектуру системи з усіма підсистемами, описано функціонал кожної з них. Система має складатися з трьох частин, що пов’язані між собою. Мікроконтролер аналізує дані усіх модулів на платі залежно від режиму роботи та передає оновлені дані на сервер з вказаним користувачем інтервалом. Сервер дає змогу відправляти запити користувачу та мікроконтролеру для отримання та оновлення потрібної інформації. Сервер містить базу даних, яку постійно оновлює згідно отриманих даних від користувача або мікроконтролера. Мобільний додаток виводить актуальну інформацію про користувача, його системи, відображую зміну стану кожної системи у поточному часі, надає змогу керувати системою.

База даних складається з 5 таблиц. У таблиці «user» міститься інформація про кожного користувача та приписані системи. Таблиця «system» містить інформацію про стан кожного датчика, наявного у фізичній частині (тампери, геркони, тощо). Сервер реалізує можливість зміни та отримання інформації з бази даних.

Наведено механізми роботи системи, взаємозв’язок користувача, серверу та мікроконтролера. Детально розібрано особливості роботи серверу. Наведено опис роботи мобільного додатку, показано можливі запити до сервера від клієнта.

Для кожного запиту наведено можливі результати виконання, які виникають при певних діях користувача.

4. РОЗРОБКА СИСТЕМИ

4.1. Розробка серверу

Серверна частина є мозком системи. Через неї користувач може отримувати необхідну йому інформацію та вносити зміни у роботу системи за необхідності.

Сервер розроблено на Java. Деплой зроблено на хмарній платформі Heroku. Це найкраще бюджетне рішення, але за наявності фінансування, хост краще змінити. Оскільки heroku підтримує лише ClearDB MySQL, задля успішного деплою необхідно було привести кодування таблиць до utf-8 формату.

Для обмеження доступу до серверу та покращення надійності його роботи, використовується проксі сервер, створений на платформі NodeJS.

Приклад реалізованих GET та POST запитів до сервера буде наведено нижче, використовуючи Postman.

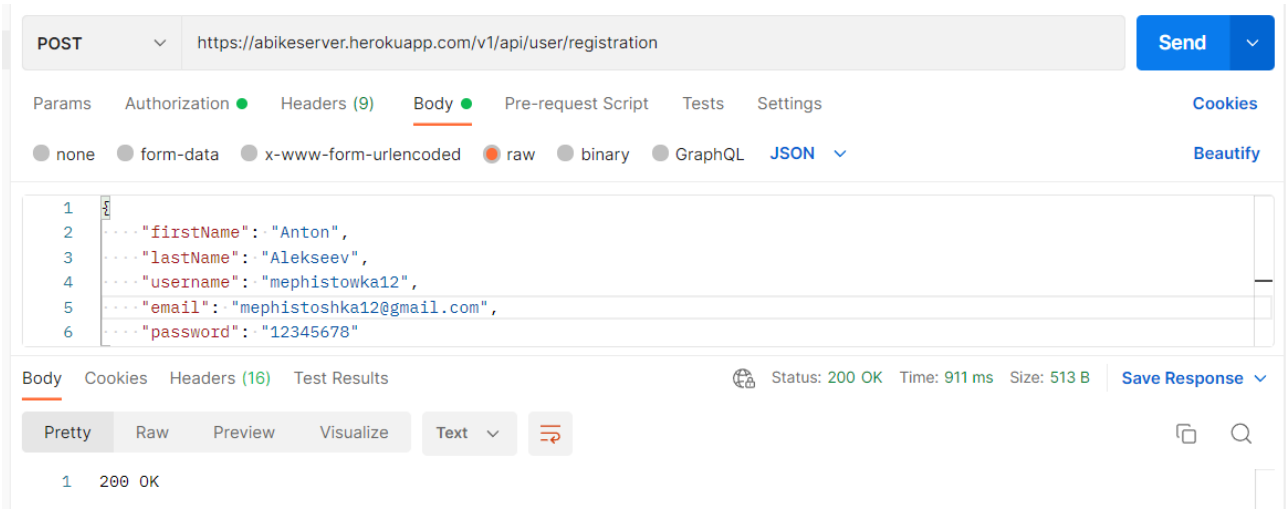

Рис. 4.1. Успішний запит на створення користувача у системі

Якщо користувач з вказаним юзернеймом або поштою вже існує, то запит буде не успішним, а сервер поверне Bad_request з описом помилки, що інформує про неправильні введені дані.

Слід зауважити, що це єдиний можливий запит, який можливо виконати без авторизації у системі.

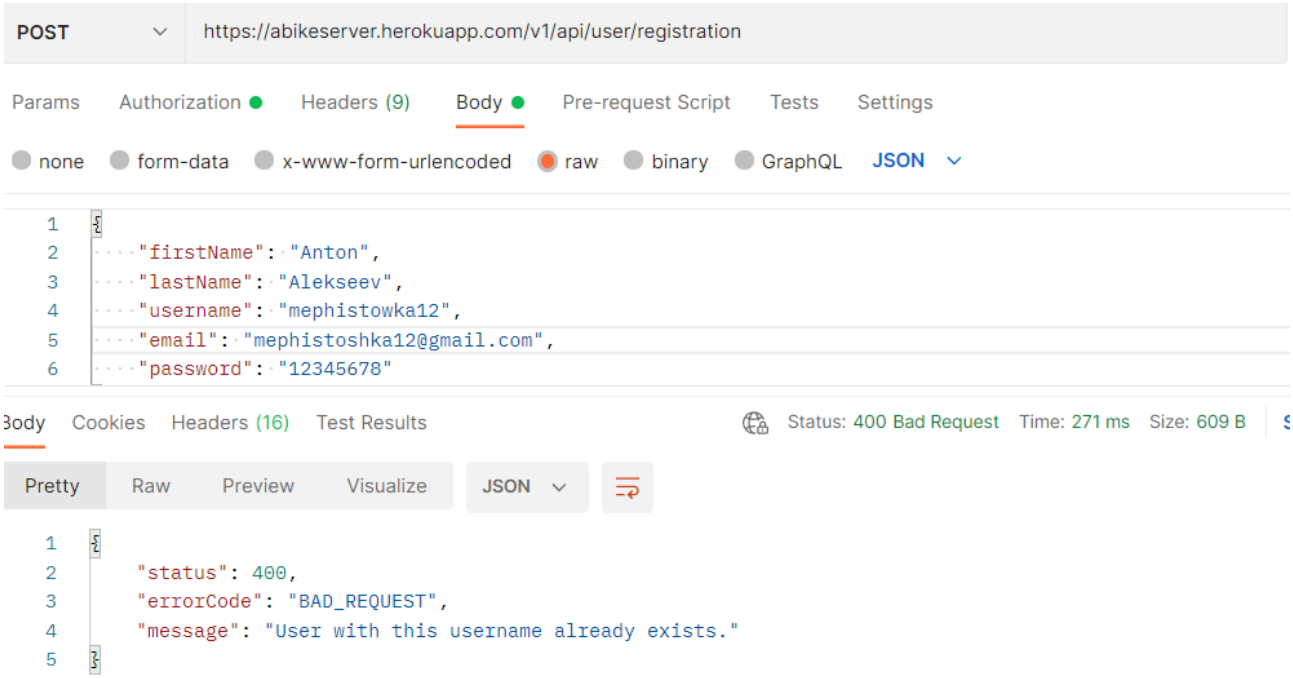

Рис 4.2. Запит реєстрації з вже існуючим юзернеймом

Запит логіну у систему повертає jwt токен, якщо запит був успішним. Увійти у систему можна вказавши юзернейм або пошту на вибір користувача та пароль від акаунту.

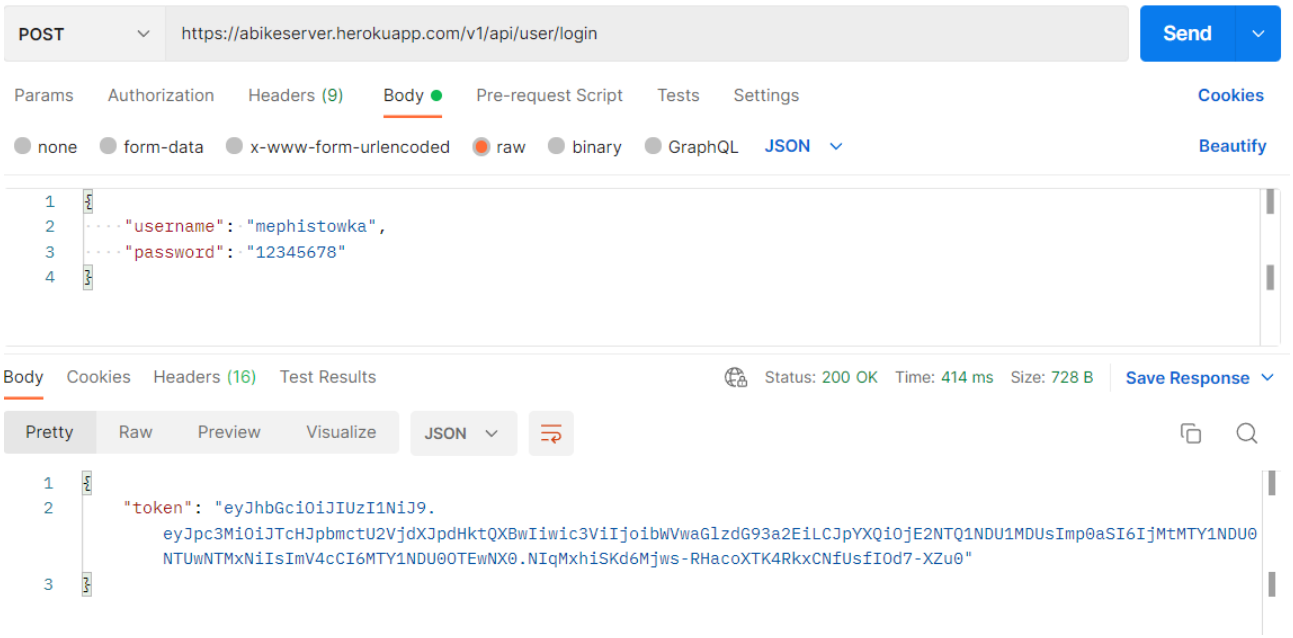

Рис. 4.3. Успішний запит логіну у систему

Якщо дані були введено не правильно, логін не відбудеться, а замість токену буде повернено номер помилки, що вказує на помилку у введених даних.

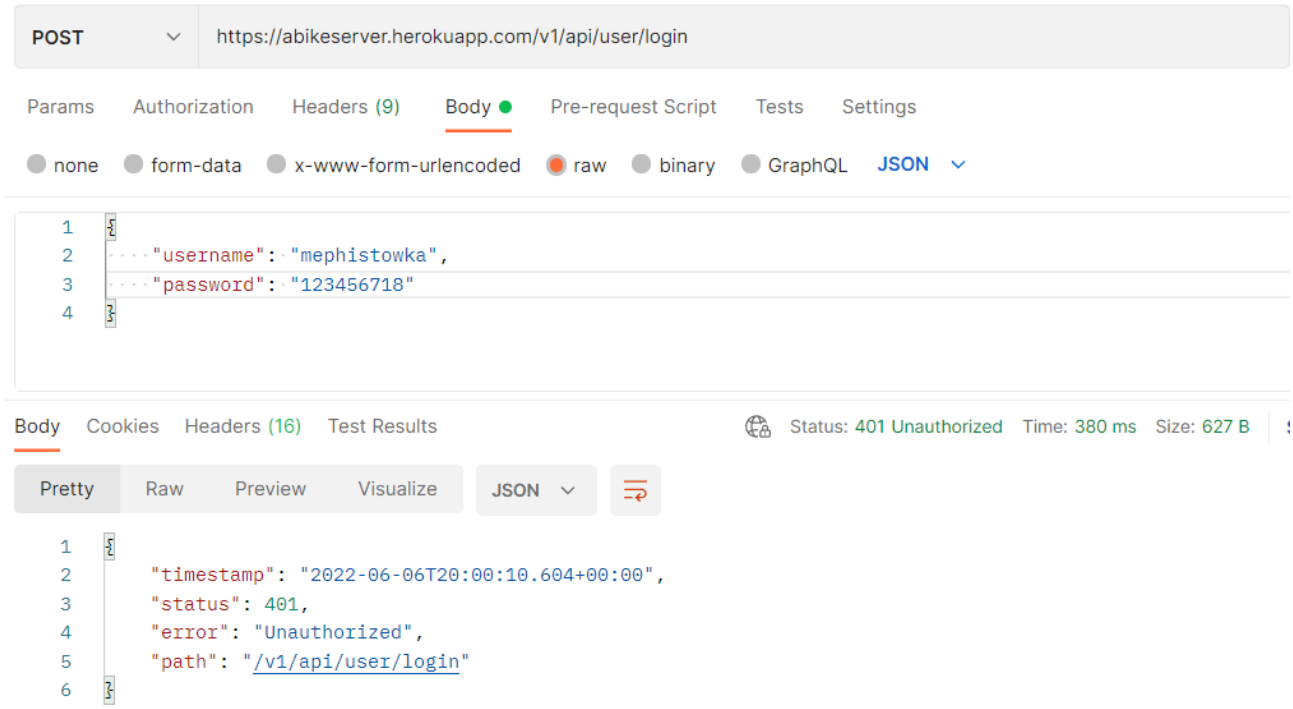

Рис. 4.4. Неуспішний запит логіну у систему

Аналогічно з логіном, створено запит logout та deleteUser, що відповідно виходять з системи та видаляють користувача.

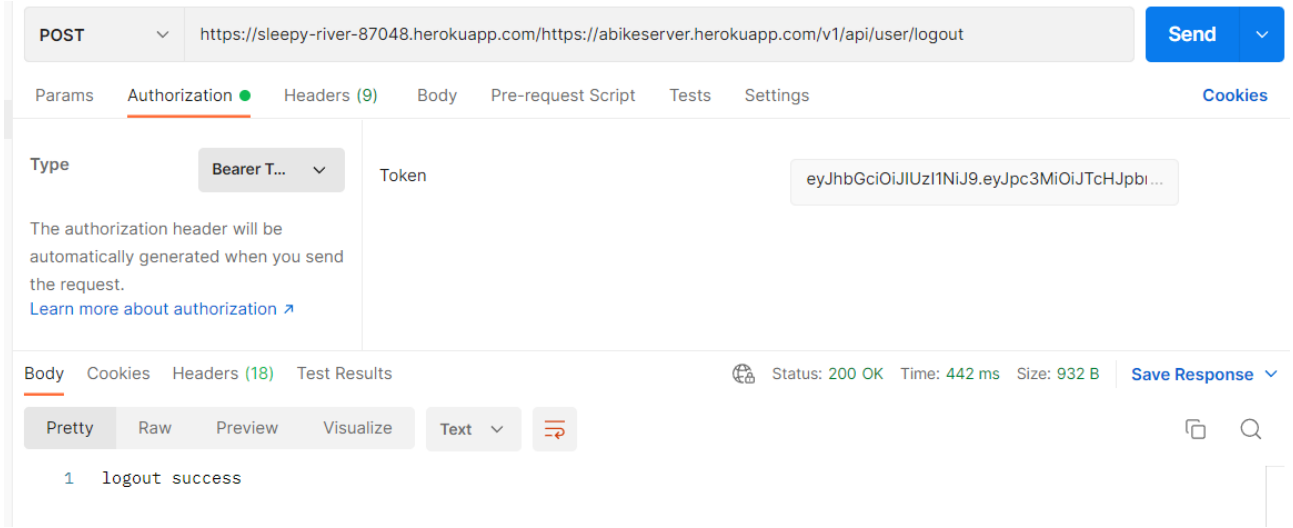

Рис. 4.5. Запит виходу із системи

Також реалізовано запит, що дозволяє дізнатися, чи є користувач адміністратором, чи ні. Якщо користувач має привілегії «Client», то наступний запит поверне Client.

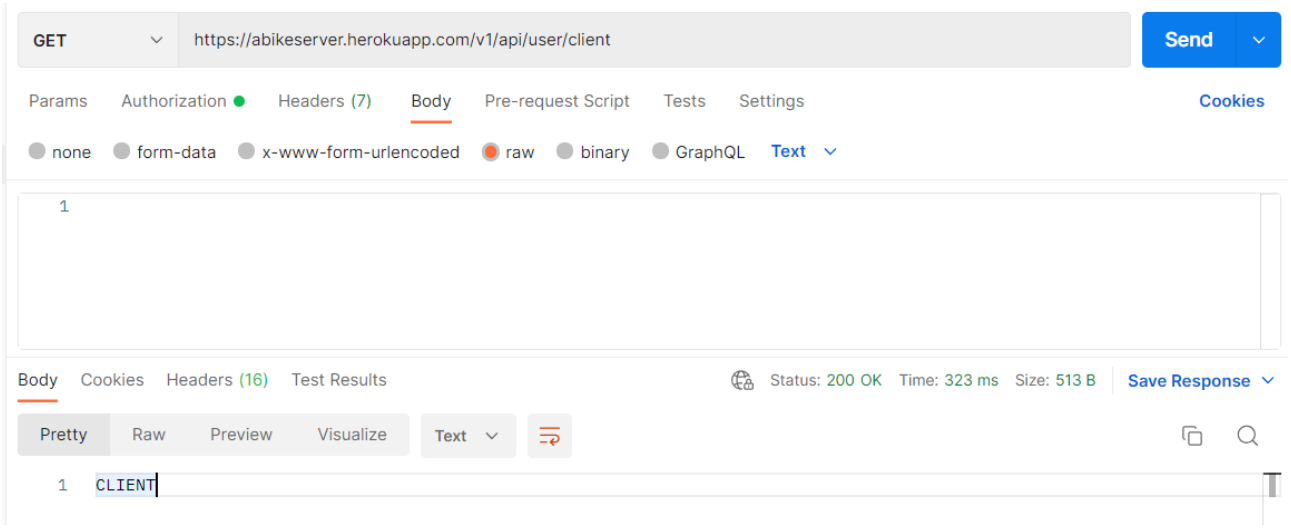

Рис. 4.6. Запит для отримання привілегій користувача

Важливою можливістю сервера є змога отримувати інформацію про усі системи користувача, та відповідно інформацію про самого клієнта. Цей функціонал дає змогу оновлювати поля у мобільному додатку у реальному часі.

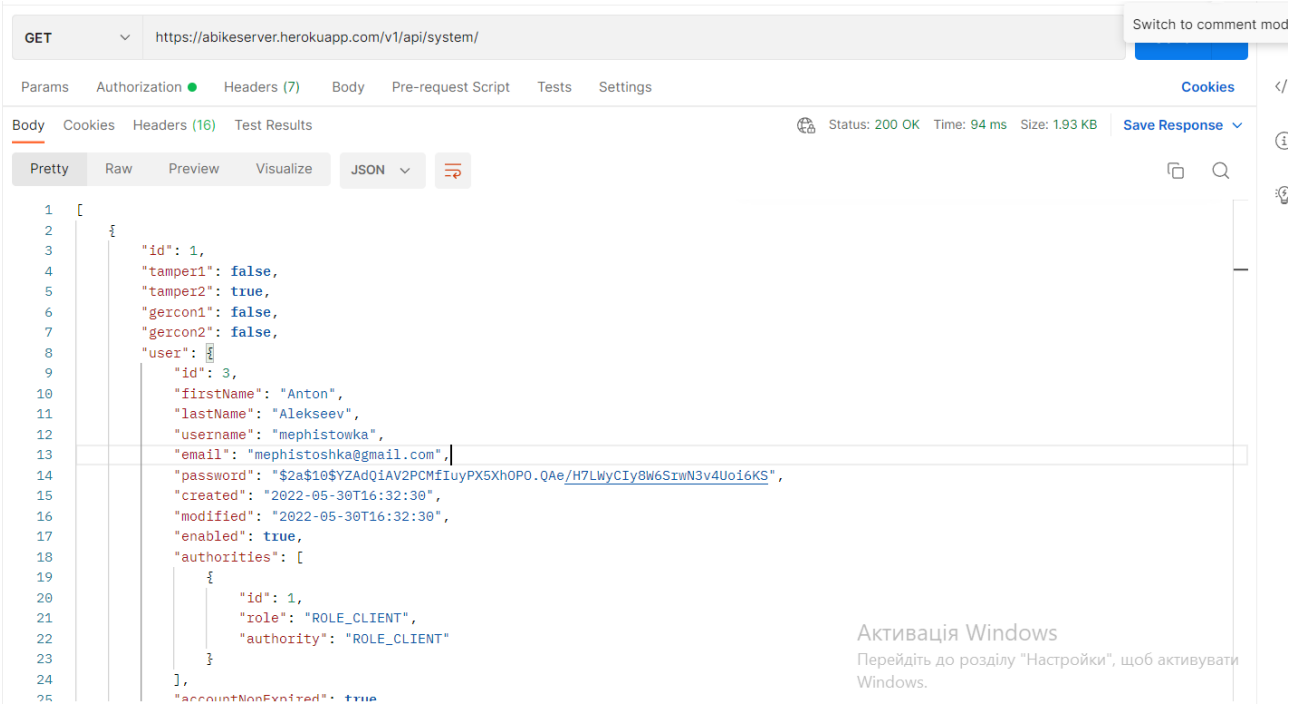

Рис. 4.7. Запит для отримання інформації про системи та користувача

Окрім цього реалізовано можливість змінювати дані про систему (цей функціонал може використовувати як мікроконтролер, так і користувач). Завдяки цьому можна ставити чи знімати систему з охорони, змінювати інформацію про поточний стан кожного датчика у системі.

Такий запит зробить зміни у базі даних та поверне інформацію про поточну систему, у якій відбулися зміни та користувача (аналогічно з GET запитом).

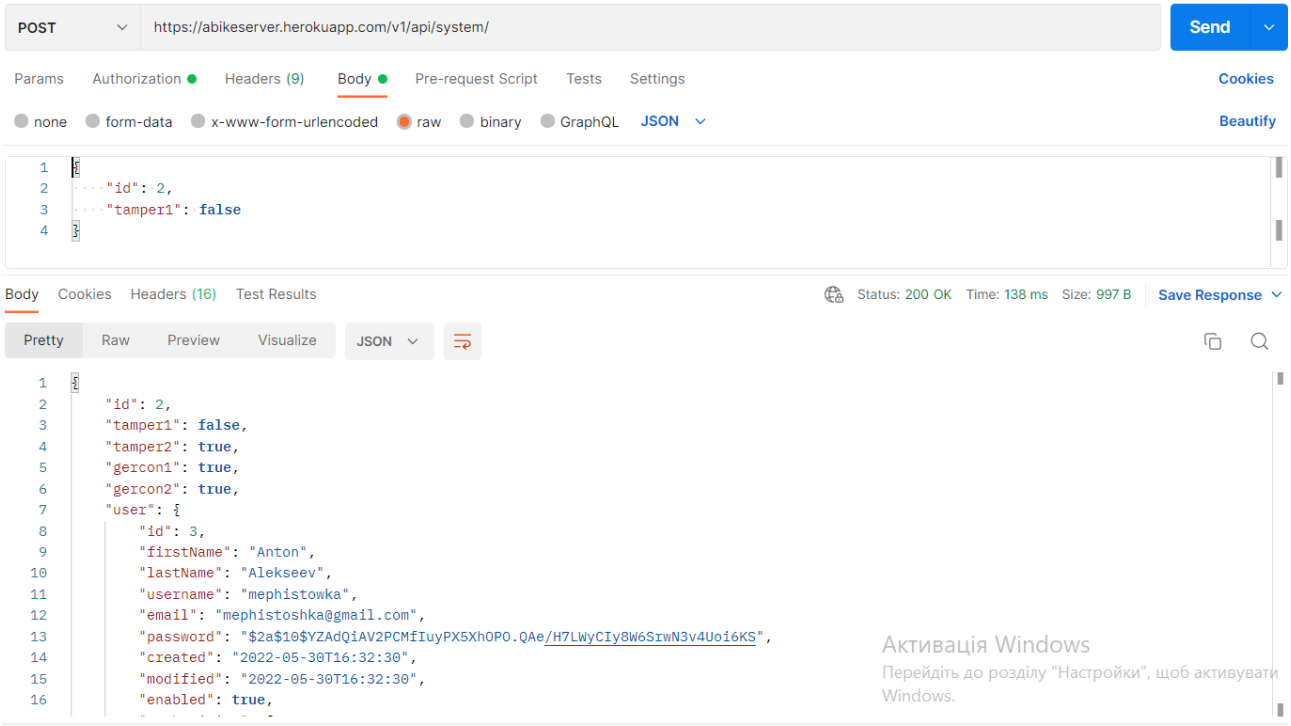

Рис. 4.7. Запит для зміни інформації у базі даних

Реалізація на Java. Запити реалізовано за допомогою Spring boot. Код проекту можна подивитись у відкритому репозиторії на GitHub[7].

4.2. Вибір серед готових рішень додатку

Прототипом було обрано додаток Ajax Systems – зручний у використанні, має приємний дизайн та просте керування. Він ідеально підходить до створеної у попередньому розділі структури.

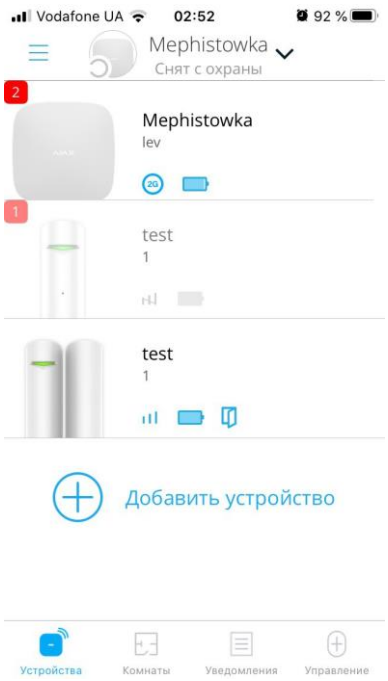

Рис. 4.8. Головне меню застосунку AJAX Systems

Меню пуш повідомлень також є гарним прикладом реалізації. Це меню є інформативним, без зайвої інформації, містить дату кожного повідомлення та візуально легко та швидко сприймається.

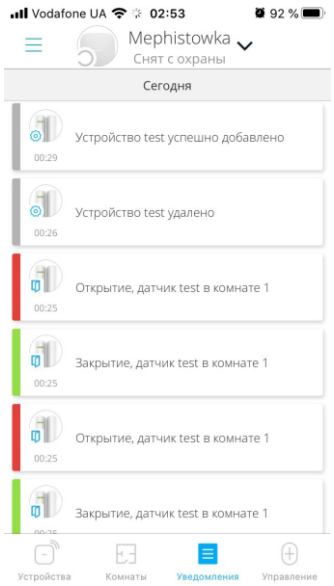

Рис. 4.9. Меню push повідомлень додатку AJAX Systems

4.3. Створення додатку

Розроблено демо версію додатку за допомогою React та Electron. Додаток може бути запущений на будь-якій операційній системі – Android, IOS, Windows, Mac OS, Linux.

При запуску додатку користувачу надається можливість увійти у систему.

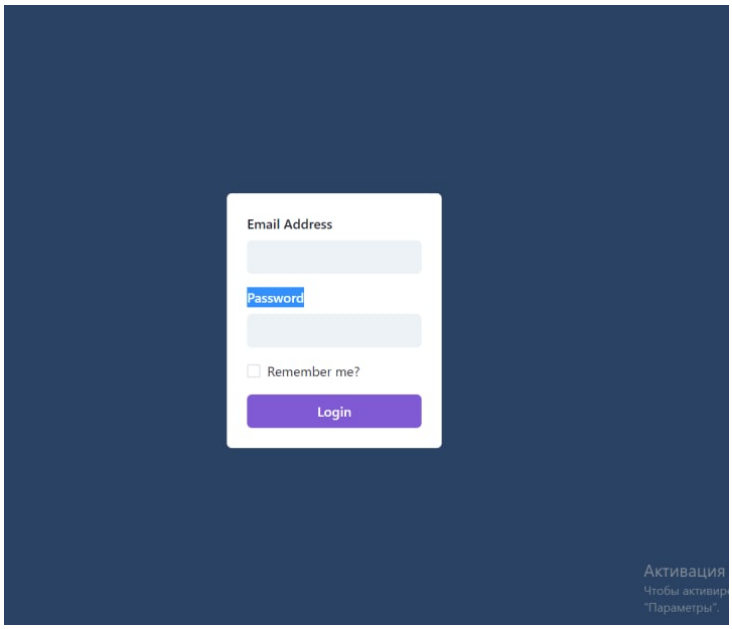

Рис 4.10. Меню логіну у додатку

Для логіну необхідно ввести свою пошту та пароль. При успішному запиті, користувачу відкривається головне меню додатку.

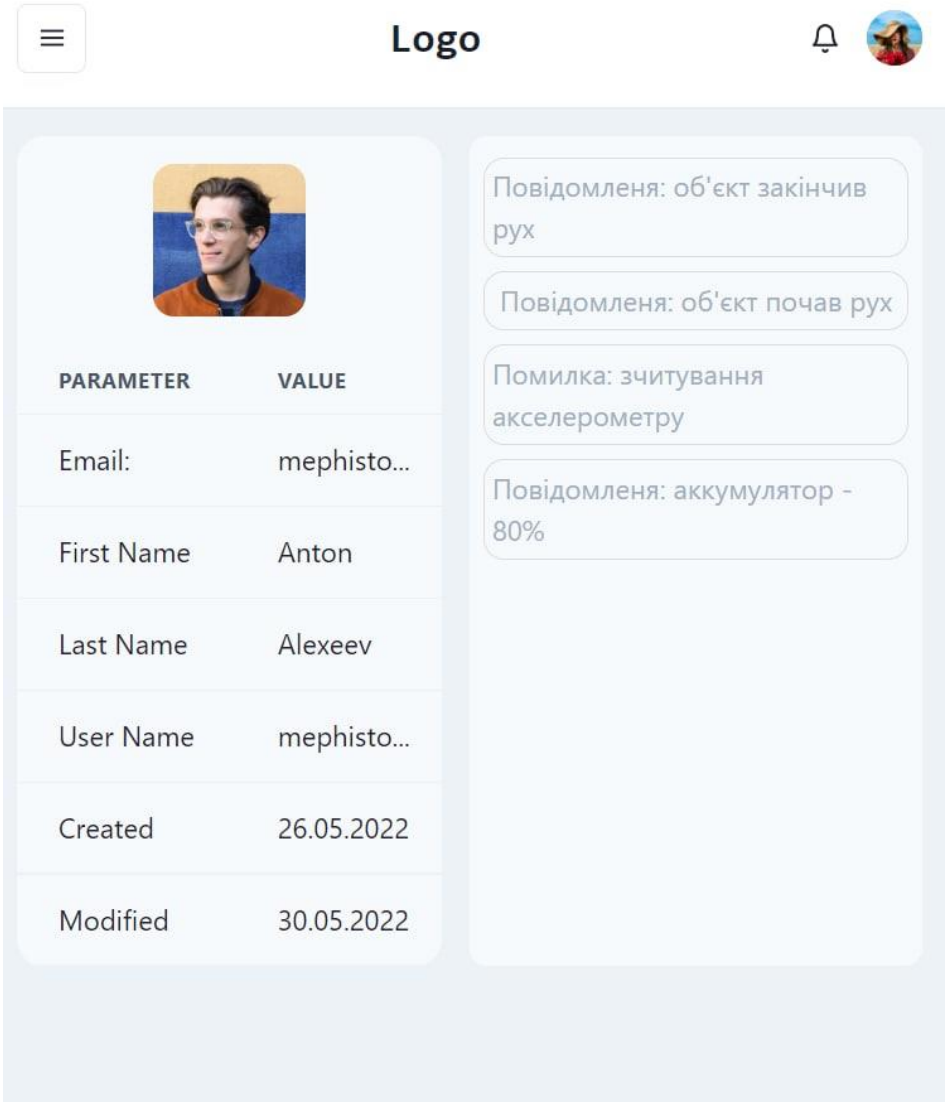

Рис. 4.11. Головне меню додатку

У головному меню, у лівій частині міститься інформація про користувача, а у правій – повідомлення про зміни стану усіх систем, що доступні юзеру. Зліва зверху знаходиться кнопка, що відкриває підменю з усіма наявними системами. У цьому підменю реалізовано можливість додати нову систему.

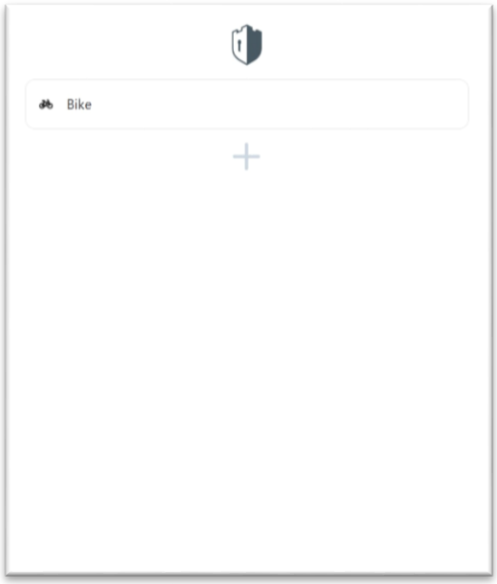

Рис 4.12. Підменю з системами

При натисканні на кнопку «плюс», з’являється вікно додавання системи. У цьому меню необхідно ввести унікальний ідентифікатор системи, її ім’я та опис за необхідності.

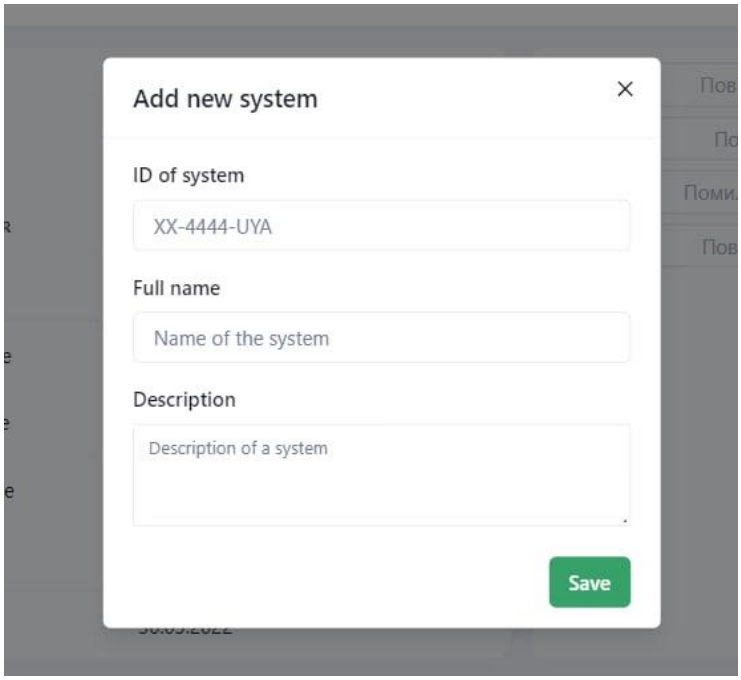

Рис. 4.13. Вікно додавання нової системи

При цьому можливіть додати ще одну систему не зникає – кнопка «плюс» зміщується під усі приписані до облікового запису системи.

Після успішного додавання, система з’явиться у підменю з системами користувача. Якщо натиснути на саму систему, буде відображене меню цієї системи з пуш повідомленнями від неї.

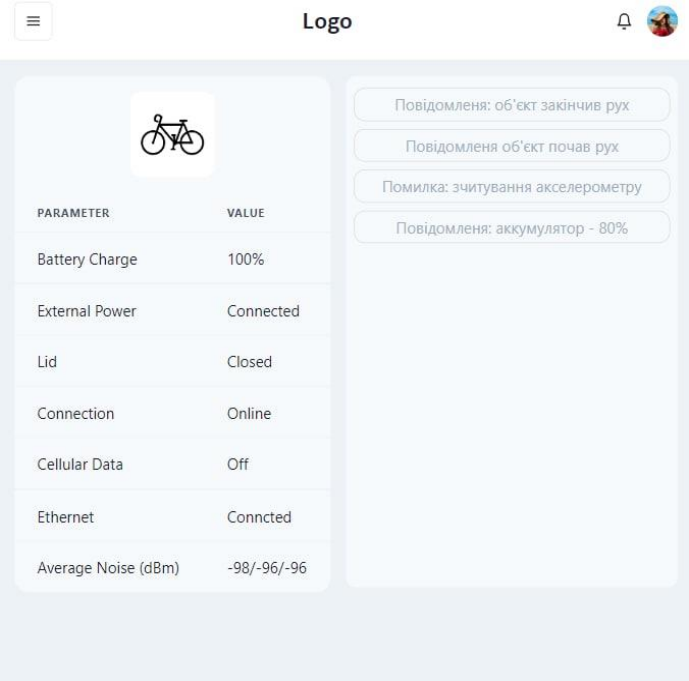

Рис. 4.14. Меню однієї з систем

У цьому меню можна відслідковувати стан системи – якщо відкрита тривожна зона, користувачу це буде відображено.

4.4. Керівництво користувачу

Керівництво призначене відобразити користувачу принципи роботи системи.

Для повноцінної роботи системи необхідно мати запрограмований пристрій (плата з мікроконтролером у пластиковому корпусі, далі – просто пристрій) та додаток на телефоні.

У додатку необхідно створити обліковий запис. Після першого запуску, додаток запропонує це зробити. Треба ввести пошту, юзернейм, ім’я, прізвище та пароль. Після цього необхідно залогінитись у систему.

Кожен пристрій має свій індивідуальний номер, що ідентифікує його. Жодні два пристрої не можуть мати однакові ідентифікатори – якщо таке трапилося, то одна з систем є підробкою та не взаємодіє з додатком.

Після встановлення системи на транспортний засіб, необхідно приписати її до облікового запису у додатку. Для цього треба натиснути кнопку «плюс» у меню зі списком систем користувача. З'явиться меню реєстрації пристрою, де необхідно ввести унікальний ідентифікатор пристрою, а також ввести ім'я системи, що буде відображатися у лівому меню.

Після цього необхідно натиснути кнопку «зареєструвати» та увімкнути пристрій кнопкою на корпусі. Якщо ідентифікаційний номер введено правильно, то запит виконається успішно і система з'явиться у додатку. При реєстрації переконайтеся, що на мобільному пристрої є інтернет.

Коли система з'явиться у додатку, можна перейти до інформації про цю систему, а також мати змогу ставити систему під охорону.

Після приписування система знаходиться у режимі «знято під охороною» та не буде відпрацьовувати жодну з тривог. Для того, аби система почала захищати велосипед, треба перевести її у режим «під охороною».

Управління режимами відбувається у меню «Керування», де можна змінити стан системи на протилежний. Всього є два стани – «під охороною» та «знято з охорони». Кожна зміна режиму буде виведена у панелі пуш-повідомлень.

Після постановки на охорону, система відображає кожну зміну тривожних зон: якщо велосипед зрушено з місця, прийде тривога по акселерометру, при вібрації – прийде пуш про вібрацію, аналогічно тривогам по геркону та зовнішнім тамперам. На кожну тривогу буде включено звукову індикацію на 2 хвилини.

Кожна тривога може бути приглушена (тобто вимкнена звукова індикація) у додатку – в головному меню під час тривоги з'являється кнопка «вимкнути звук». Ця кнопка не зникне навіть після натискання на неї. Аби вона пропала необхідно повернути систему у нормальне положення, тобто відновити

усі тривожні зони (якщо тривога по відкриттю тампера, то необхідно закрити його).

Деякі тривоги не даватимуть звукової індикації у будь якому стані системи. До таких тривог відноситься розряджена батарея та ввімкнена індикація.

Тривога по батареї означає, що необхідно підзарядити систему. В середньому, споживання системи вистачає на працю системи без зарядки не менше ніж на шість місяців.

Тривога по ввімкненій індикації по суті не є чимось критичним, хоча аналогічно іншим тривогам підсвічується червоним кольором. Ця тривога означає, що на поточний момент часу ввімкнена індикація-підсвічення (це може бути або індикація повороту чи гальмування, або підсвічення нічного режиму). У додатку вона відображається для того, аби користувач не забував вимикати її після користування велосипедом, адже індикація споживає чималий струм, хоча і має власні акумулятори.

Крім того система має ряд налаштувань, що можна змінити. Кожна приписана система у головному меню має кнопку налаштувань. При натисканні на неї користувачу відкриється панель, де можна змінити період опитування мікроконтролера, що варіюється від 10 секунд до 1 хвилини, з кроком у десять секунд. Чим менше значення цього налаштування, тим стійкіша система до зламу – користувач швидше дізнається, що його система перестала відповідати, якщо відбудеться будь-яке механічне втручання, що пошкодить функціонал системи. При цьому зменшення цього періоду збільшує споживання системи. Рекомендовано встановити період опитування системи у 20 секунд. Цього часу достатньо для реагування на моменти, коли система виведена з ладу, а споживання буде у 1.5 разів менше ніж при періоді у 10 секунд. Більше за 20 секунд ставити небезпечно, адже за цей час грабіжники здатні вкрасти велосипед та зникнути з поля зору.

Висновки до розділу

У цьому розділі було розроблено серверну частину системи та реалізовано демо версію мобільного додатку для керування та моніторингу.

Наведено основні запити до сервера, що будуть використовуватись додатком та мікроконтролером. Детально описано кожен запит, наведено деякі нюанси, які можуть виникнути під час обробки запитів.

Мобільний додаток є кросплатформеним, та складається з трьох частин – список систем користувача, інформація про систему та пуш повідомлення.

Було написано керівництво користувачу.

ВИСНОВКИ

1. Під час виконання роботи було розглянуто аналоги охоронних систем на ринку. Проаналізовано їхні переваги та недоліки. Зроблено розбір систем за такими критеріями: функціональні особливості, зручність у використанні та керуванні, надійність, швидкість роботи та автономність. Проведений аналіз дав змогу створити вдосконалену систему, що має переваги над усіма наведеними у роботі аналогами. Зроблено детальний розбір різних протоколів передачі даних. Поставлено задачу курсової роботи – розробити модель системи та реалізувати фронтент та бекенд частину.

2. Розроблено принципову модель системи. Зроблено вибір технологій для розробки мобільного додатку та серверу.

3. Спроектовано архітектуру системи та всіх її підсистем, наведено опис кожної з них. Розроблено базу даних з детальним описом усіх таблиць і полей, а також роль кожної з них у системі. Описано механізми роботи системи

4. Розроблено сервер за допомогою мови програмування Java. Створено демо версію мобільного застосунку, використовуючи React та фреймворк Electron. Також розроблено проксі сервер на NodeJS, наведено основні його функції. На основі отриманої системи, створено керівництво для користувача, де детально описано як користуватися системою.

У результаті виконання курсової роботи було розроблено серверну частину та мобільний додаток для системи захисту рухомих об’єктів.

В майбутньому систему можна покращити. По-перше, додати більше тривожних зон та інформації про систему у серверну частину розробки, зробити фонові повідомлення у додатку для тривог та додати звукові сповіщення для полегшення отримання інформації про зміну стану системи.

ПЕРЕЛІК ПОСИЛАНЬ

1. Guochang Xu; Yan Xu. GPS. Theory, Algorithms and Applications. Вид. 3-є. NY: Springer Berlin Heidelberg, 2016. 488с.
2. Макаров С.Б., Певцов Н.В., Попов Е.А., Сиверс М.А. Телекоммуникационные технологии: Введение в технологии GSM, Москва: М.: Академия, 2006. 256с.
3. Охоронна GSM-сигналізація для будинку : веб-сайт. URL: http://ddn.radioliga.com (дата звернення: 02.04.2020)
4. GSM сигналізація: переваги та недоліки : веб-сайт. URL: https://ohorona-kyiv.com (дата звернення: 02.04.2020)
5. Як забезпечити комплексну охорону об’єкта? : веб-сайт. URL: https://klaster.ua (дата звернення: 03.04.2020)
6. When Security is Art : веб-сайт. URL: https://ajax.systems (дата звернення: 20.04.2020)
7. Репозиторій проекту серверу системи: https://github.com/Mephistowka/bike_server
8. About Node.js : веб-сайт. URL: https://nodejs.org/ (дата звернення: 30.05.2020)
9. The drive to develop : веб-сайт. URL: https://www.jetbrains.com (дата звернення: 20.05.2020)
10. What is Proteus Visual Designer : веб-сайт. URL: https://www.labcenter.com/ (дата звернення: 15.05.2020)
11. Brian W. Kernighan and Dennis M. Ritchie The C Programming Language, 2nd ed., Prentice Hall, 1988. 27