Міністерство освіти і науки України
Кам’янець-Подільський національний університет імені Івана Огієнка
Фізико-математичний факультет
Кафедра комп’ютерних наук

**Кваліфікаційна робота магістра**

з теми: **«Методи та засоби управління паркуванням на основі мобільних технологій»**

Виконав: здобувач вищої освіти групи KN1-M24,
спеціальності 122 комп'ютерні науки
Пастушков Дмитро Сергійович
(прізвище та ім’я і по батькові здобувача вищої освіти)
Керівник: Філатов А. С., канд. техн. наук.
(прізвище та ініціали, науковий ступінь, учене звання)
Рецензент: Оптасюк С. В., канд. ф.-м. наук, доцент
(прізвище та ініціали, науковий ступінь, учене звання)

м. Кам’янець-Подільський – 2025 р.

**ЗМІСТ**

# ВСТУП

Сучасний світ переживає значне зростання кількості автомобілів. З кожним роком кількість транспортних засобів на дорогах збільшується, що обумовлено як загальним підвищенням рівня життя, так і доступністю автомобілів. У багатьох родинах сьогодні є не один, а два або більше автомобілів. Це створює нові виклики для міської інфраструктури, особливо у сфері організації паркування.

Проблема паркування стає дедалі гострішою у великих містах. Часто водії витрачають значний час на пошук вільного місця для паркування біля місця призначення. Це не лише створює незручності для автомобілістів, але й спричиняє серйозні екологічні наслідки. Під час таких пошуків автомобілі спалюють більше пального, що призводить до збільшення викидів шкідливих речовин в атмосферу, погіршуючи якість повітря. Крім того, невпорядкованість процесу паркування часто призводить до порушення правил дорожнього руху, створення заторів та зниження загальної безпеки на дорогах.

На жаль, у багатьох містах відсутні ефективні інструменти, які б дозволяли водіям заздалегідь дізнаватися про завантаженість паркувальних майданчиків. У більшості випадків водії змушені покладатися на удачу, їдучи до парковок “наосліп”, не знаючи, чи є там вільні місця. Такий підхід є неефективним і вимагає впровадження сучасних технологічних рішень.

У зв'язку з цим виникає необхідність створення інноваційної системи, яка б забезпечувала моніторинг стану паркувальних майданчиків у режимі реального часу. Метою цієї кваліфікаційної роботи є розробка такої системи, що надасть користувачам можливість оперативно отримувати інформацію про доступні паркомісця, сприятиме зниженню часу пошуку паркування та позитивно впливатиме на екологічну ситуацію.

Для досягнення поставленої мети було сформульовано такі задачі:

1. Розробити систему моніторингу стану паркувальних майданчиків, яка надасть інформацію про кількість вільних місць у режимі реального часу.

2. Створити інструменти для управління паркувальними майданчиками, зокрема адміністративну панель для власників та операторів парковок.

3. Розробити додаток для інспекторів з паркування, що дозволить ефективно контролювати дотримання правил паркування.

4. Реалізувати функцію бронювання паркомісця для користувачів, що забезпечить зручність та оптимізацію використання паркувальних ресурсів.

Розробка такої системи дозволить вирішити низку актуальних проблем у сфері паркування, забезпечивши комфорт і зручність для водіїв, а також позитивно вплине на екологічний стан міських територій. У процесі виконання цієї роботи будуть використані сучасні технології програмування, аналітики даних та IoT-рішення для створення інтегрованої платформи, яка відповідатиме потребам сучасного суспільства.

# РОЗДІЛ 1. Аналіз наявних інструментів та технологій для реалізації системи моніторингу паркувальних майданчиків

## 1.1 Огляд сучасних систем моніторингу парковок

На сьогоднішній день існує низка рішень для моніторингу та управління паркувальними майданчиками, які реалізовані за допомогою сучасних технологій. До них належать системи Smart Parking, мобільні додатки, а також інтегровані рішення, що дозволяють користувачам знаходити паркувальні місця та спрощують управління паркувальними майданчиками для адміністраторів.

Одним із популярних підходів є впровадження систем Smart Parking, які використовують датчики для відстеження зайнятості паркомісць у реальному часі. Ці датчики можуть бути вбудовані в дорожнє покриття, встановлені на стовпах або інтегровані у самі автомобілі. Інформація від датчиків передається на центральний сервер, а потім відображається у вигляді зручного інтерфейсу для користувачів.

Існує також велика кількість мобільних додатків для пошуку паркувальних місць, таких як ParkMobile, EasyPark, SpotHero, Parkopedia тощо. Вони надають користувачам можливість:

- Знаходити вільні паркомісця на карті.
- Отримувати інформацію про вартість паркування.
- Оплачувати паркування через додаток.
- Деякі сервіси навіть дозволяють бронювати паркомісця.

Такі рішення значно полегшують життя водіям, особливо у великих містах з високою щільністю транспортного потоку.

Попри очевидні переваги, сучасні системи моніторингу парковок мають низку недоліків та обмежень, які впливають на їх ефективність і масштабованість:

1. **Неповне покриття**: Багато існуючих рішень обмежені за географією або типом обслуговуваних паркувальних майданчиків. Наприклад, деякі мобільні додатки працюють лише в обраних містах чи регіонах.

2. **Висока вартість впровадження**: Інтеграція датчиків у кожне паркомісце потребує значних фінансових інвестицій. Це обмежує можливості масштабування таких систем, особливо у країнах із низьким рівнем доходів.

3. **Неточність даних**: Датчики можуть давати хибні результати через технічні несправності або зовнішні фактори, такі як погодні умови чи неправильне паркування автомобілів.

4. **Відсутність інтеграції між різними системами**: Багато платформ працюють незалежно одна від одної, що ускладнює об'єднання даних з різних джерел. Це створює незручності для користувачів, які змушені використовувати декілька додатків.

5. **Недостатня гнучкість у функціях**: Деякі системи пропонують лише базовий функціонал, наприклад, пошук вільного місця або оплату паркування. Водночас відсутні такі важливі можливості, як бронювання місць або інтеграція з іншими транспортними рішеннями.

6. **Екологічні наслідки**: Хоча ці системи спрямовані на зменшення викидів, пов'язаних із пошуком паркомісць, їхня обмежена доступність і неточність даних можуть не досягати бажаного екологічного ефекту.

Таким чином, сучасні системи моніторингу парковок мають потенціал для покращення, але потребують інноваційного підходу, щоб стати доступнішими, надійнішими та ефективнішими для користувачів. У рамках даного проєкту пропонується врахувати ці обмеження для створення універсальної системи, яка вирішить існуючі проблеми

## 1.2 Технологічні основи проєкту

Для реалізації проєкту обрано технології, які забезпечують необхідний функціонал та відповідають вимогам сучасної розробки. Вони дозволяють створити ефективну, масштабовану та зручну систему, що охоплює як серверну, так і клієнтську частину.

React використовується для створення адміністративної панелі, що є важливим компонентом системи. Цей фреймворкк є однією з найпопулярніших для розробки інтерфейсів користувача завдяки компонентному підходу. Це дозволяє будувати складні інтерфейси з незалежних елементів, які легко тестувати, підтримувати та оновлювати. Віртуальний DOM забезпечує високу продуктивність і швидке оновлення даних, що робить React ідеальним для завдань цього проєкту. Крім того, активна спільнота сприяє постійному розвитку інструментів та бібліотек, які значно спрощують процес розробки.

Для створення мобільних додатків обрано React Native, який дозволяє розробляти програми для IOS і Android з використанням єдиного коду. Це економить час і ресурси, оскільки не потрібно писати окремі додатки для кожної платформи. React Native забезпечує нативну продуктивність, дозволяючи додаткам виглядати і працювати як розроблені нативно. Крім того, його функція Hot Reloading прискорює розробку, дозволяючи розробникам миттєво бачити зміни в коді. Завдяки гнучкості цієї платформи, її легко адаптувати для виконання конкретних функцій мобільних додатків, включаючи моніторинг і бронювання паркомісць.

Node.js використовується як серверна платформа завдяки своїй високій продуктивності та здатності обробляти одночасно велику кількість запитів. Асинхронна модель роботи дозволяє створювати швидкі та масштабовані серверні рішення, що є ключовим для системи, яка потребує обробки великих обсягів даних. Node.js також сприяє спрощенню розробки, оскільки використовує JavaScript на серверній стороні, що забезпечує єдність

технологій у проєкті. Крім того, широка екосистема модулів і бібліотек значно полегшує реалізацію різноманітних функцій.

MongoDB було обрано як базу даних, адже вона забезпечує гнучке та масштабоване зберігання інформації у форматі JSON-подібних документів. Ця база даних особливо зручна для проєктів, де структура даних може змінюватися у процесі розробки, оскільки дозволяє уникнути складних міграцій. MongoDB оптимізована для роботи з великими обсягами даних і забезпечує швидкий доступ до них. У проєкті база даних використовується для зберігання інформації про паркомісця, бронювання та користувачів, а її інтеграція з Node.js через бібліотеки, такі як Mongoose, забезпечує зручність роботи та управління даними.

Home Assistant використовується у проєкті для інтеграції та управління фізичними пристроями, які забезпечують функціонування «розумної» частини системи. Home Assistant є сучасною платформою автоматизації, що підтримує сотні типів пристроїв і протоколів. Вона дозволяє централізовано керувати сенсорами та актуаторами, зокрема пристроями на основі протоколу Zigbee, який є одним із провідних стандартів у сфері IoT завдяки своїй енергоефективності та надійності.

У проєкті Home Assistant використовується як проміжна ланка між апаратними Zigbee-пристроями та серверною частиною системи. За його допомогою відбувається:

- зчитування показників датчиків (наприклад, сенсорів зайнятості паркомісця);
- управління виконавчими елементами при необхідності;
- обмін даними через API для подальшої обробки на сервері Node.js.

Завдяки широким можливостям інтеграції, Home Assistant забезпечує стабільну і безпечну взаємодію з апаратними компонентами. Це суттєво спрощує процес розробки та дозволяє зосередитися на логіці застосунку, уникаючи складностей прямої взаємодії з Zigbee-пристроями. Крім того,

платформа підтримує автоматизації, що можуть бути використані для створення додаткових сценаріїв роботи системи.

Загалом, обраний стек технологій – React, React Native, Node.js, MongoDB, Home Assistant та Zigbee – надає всі необхідні інструменти для реалізації функціоналу системи, роблячи її зручною для користувачів, швидкою у виконанні запитів і масштабованою для подальшого розвитку. Такий підхід гарантує високий рівень якості проєкту та його відповідність сучасним вимогам.

## 1.3 Огляд апаратних рішень для моніторингу та бронювання

Для реалізації системи моніторингу та бронювання паркомісць використано апаратні рішення на базі технології Zigbee, яка забезпечує надійний та енергоефективний зв'язок між пристроями. Також інтеграція цих пристроїв із хмарним сервісом дозволяє ефективно обробляти дані та забезпечує доступ до них у режимі реального часу.

Технологія Zigbee є стандартом бездротового зв'язку, розробленим для пристроїв з низьким енергоспоживанням та обмеженими ресурсами. Основні характеристики пристроїв Zigbee включають:

- **Низьке енергоспоживання**: пристрої можуть працювати на батарейках протягом кількох років, що є важливим для паркувальних систем, де обслуговування має бути мінімальним.
- **Стабільний бездротовий зв'язок**: Zigbee працює на частоті 2.4 ГГц і підтримує передачу даних на відстані до 100 метрів, що забезпечує покриття великих територій.
- **Мережева структура**: технологія підтримує топології “зірка”, “дерево” та “сітка”, що робить її гнучкою для розгортання в різних умовах.

- **Широкий спектр пристроїв**: існує безліч компонентів, таких як комутатори, реле, датчики руху, що дозволяють створювати комплексні рішення для моніторингу та управління.

Комутатори та реле Zigbee використовуються для управління фізичними елементами системи, такими як шлагбауми чи освітлення. Вони можуть отримувати сигнали від центрального сервера або хмарного сервісу та виконувати необхідні дії в реальному часі.

Хмарний сервіс відіграє ключову роль у забезпеченні взаємодії між пристроями Zigbee та серверною частиною системи. Основні переваги використання хмарного сервісу:

- **Реальний час**: пристрої передають дані до хмари, де вони обробляються та відправляються на сервер у реальному часі. Це дозволяє миттєво реагувати на зміни в системі.
- **Масштабованість**: хмарна інфраструктура легко адаптується до збільшення кількості пристроїв та обсягу даних.
- **Безпека**: сучасні хмарні сервіси забезпечують високий рівень захисту даних завдяки шифруванню та використанню надійних протоколів.
- **Доступність**: інтеграція хмарного сервісу дозволяє користувачам та адміністраторам отримувати доступ до системи з будь-якого пристрою та місця.

У даному проєкті хмарний сервіс використовується для передачі даних від пристроїв Zigbee до бекенду, побудованого на Node.js. Це забезпечує синхронізацію даних між апаратною частиною та програмним забезпеченням, дозволяючи реалізувати функції моніторингу та бронювання.

Пристрої Zigbee вже широко використовуються у багатьох реальних проєктах, що підтверджує їх ефективність. Наприклад:

1. **Розумні будинки**: Zigbee застосовується для управління освітленням, клімат-контролем та безпековими системами, забезпечуючи автоматизацію та зручність для користувачів.

2. **Системи розумного паркування**: багато міст світу впровадили рішення на базі Zigbee для моніторингу паркувальних місць, що дозволяє зменшити час пошуку паркування та оптимізувати використання простору [7, 8, 20].

3. **Промислові додатки**: у логістичних центрах технологія використовується для моніторингу вантажів та управління процесами в реальному часі.

Застосування Zigbee у цих сферах демонструє її універсальність, надійність та ефективність. У рамках даного проєкту використання цих пристроїв дозволяє створити гнучку та масштабовану систему моніторингу та управління паркомісцями.

## 1.4 Аналіз потреб цільової аудиторії

Розробка системи моніторингу та бронювання паркомісць має враховувати різні потреби та очікування її основних користувачів [17, 20]. До них належать водії, адміністрація парковок та інспектори.

Для водіїв однією з ключових вимог є швидкість доступу до інформації про вільні паркомісця. Система повинна працювати в реальному часі, щоб мінімізувати час пошуку місця для паркування. Крім того, важливим є аспект зручності. Інтерфейс мобільного додатка має бути простим та інтуїтивно зрозумілим, що дозволить користувачам легко здійснювати пошук і бронювання місць. Доступність інформації також грає важливу роль, адже можливість отримати дані про стан паркування у будь-який час та з будь-якого пристрою значно підвищує ефективність використання системи, особливо у великих містах з високою завантаженістю паркувальних зон.

Адміністрація парковок та паркувальні інспектори мають свої специфічні очікування. Вони потребують інструментів для управління завантаженістю паркінгів, що дозволяє ефективніше планувати ресурси та

регулювати потік автомобілів. Окрім того, система повинна забезпечувати можливість моніторингу стану паркомісць у режимі реального часу, що дозволяє інспекторам оперативно виявляти порушення та реагувати на них. Адміністраторам також важливо мати доступ до аналітичних даних та статистики, які допомагають приймати обґрунтовані рішення щодо оптимізації роботи паркінгу.

Впровадження інтелектуальної системи моніторингу та бронювання паркомісць має значні екологічні та соціальні переваги. Одним із важливих екологічних аспектів є зменшення викидів шкідливих речовин у повітря завдяки скороченню часу, який водії витрачають на пошук паркомісця. Це також сприяє зниженню заторів, адже оптимізація процесу паркування зменшує завантаженість міських вулиць і покращує дорожню ситуацію. З соціальної точки зору, ефективне використання паркувальних зон допомагає створити більш упорядковане міське середовище, що позитивно впливає на життя громади в цілому. Крім того, спрощення процесу паркування підвищує рівень комфорту для водіїв та покращує їхній досвід користування міською інфраструктурою.

Загалом, аналіз потреб різних категорій користувачів і врахування екологічних та соціальних переваг дозволяє створити систему, яка не лише вирішує практичні завдання, але й робить внесок у покращення міського середовища та якості життя громадян.

# РОЗДІЛ 2. Архітектура та програмна реалізація системи розумного паркування

## 2.1. Загальна архітектура системи

Система розумного паркування являє собою комплексне рішення, що поєднує програмні, мережеві та апаратні компоненти, які взаємодіють між собою для здійснення повного циклу роботи: від відображення доступних паркомісць до керування реальним парковочним стопером через HomeAssistant. Архітектура системи побудована таким чином, щоб забезпечити високу надійність, безпеку, простоту розширення та можливість інтеграції з новими пристроями або службами без суттєвих змін у її основній логіці.

Центральне місце в архітектурі займає серверна частина, що відповідає за всю бізнес-логіку та взаємодію з базою даних. Саме сервер опрацьовує запити від адміністративної панелі та мобільного застосунку, перевіряє права доступу, виконує необхідні обчислення й керує процесами, пов’язаними з відкриттям або блокуванням паркомісця. Бекенд виступає єдиною точкою істини, гарантуючи узгодженість інформації між усіма клієнтськими інтерфейсами та апаратною частиною.

Важливою складовою системи є інтеграція з HomeAssistant – платформою для керування IoT-пристроями. Саме через HomeAssistant сервер отримує доступ до Zigbee-реле, яке фізично піднімає або опускає парковочний стопер. Такий підхід дозволяє значно спростити архітектуру системи, оскільки взаємодія з фізичним обладнанням відбувається через стандартизований API HomeAssistant, а не через низькорівневі протоколи чи спеціалізоване обладнання. Це також дає змогу швидко підключати нові пристрої або замінювати існуючі без перегляду основного програмного коду.

Клієнтська частина системи складається з двох окремих застосунків – веб-адміністративної панелі та мобільного застосунку. Адміністративна

панель призначена для керування інфраструктурою, конфігурацією системи та переглядом актуального стану паркомісць. Вона надає можливість редагувати тарифи, змінювати параметри паркомісць, додавати нові елементи й контролювати роботу обладнання в режимі реального часу. Панель створена з орієнтацією на працівників, які супроводжують систему, тому акцент у ній зроблено на інформативності та простоті керування.

Мобільний застосунок, у свою чергу, створений для кінцевих користувачів і дозволяє взаємодіяти з системою на рівні повсякденного використання: переглядати доступні місця на карті, вибирати автомобіль, відкривати або завершувати паркування та керувати особистим профілем. Завдяки інтеграції з геолокацією та картою користувач отримує інформацію про найближчі паркомісця та може легко орієнтуватися в просторі. Весь функціонал мобільного застосунку працює через бекенд, що гарантує коректність і синхронність даних.

Особливість архітектури полягає у чіткому розмежуванні ролей кожного компонента: бекенд відповідає за логіку і доступ до обладнання, HomeAssistant – за виконання фізичних команд, веб-панель – за адміністративний контроль, а мобільний застосунок – за взаємодію з користувачем. Такий поділ дозволяє системі бути модульною та легко доповнюваною. Наприклад, у майбутньому можна додати нові типи пристроїв (датчики, камери, автоматичні шлагбауми), нові клієнтські інтерфейси або навіть розподілити серверну частину на мікросервіси – і всі ці зміни природно впишуться в існуючу архітектуру.

Таким чином, загальна архітектура системи створює міцний фундамент, який поєднує програмні й апаратні рішення в єдину інфраструктуру. Вона забезпечує зручність використання для користувачів, гнучкість і масштабованість для розробників, а також стабільність і керованість для адміністраторів. Усі компоненти взаємодіють між собою через чіткі інтерфейси, що робить систему передбачуваною, надійною та готовою до подальшого розвитку.

## 2.2. Серверна частина (Backend API)

Серверна частина системи є центральним елементом програмної архітектури, оскільки саме тут зосереджена основна бізнес-логіка, механізми перевірки доступу, взаємодія з базою даних та зв'язок із HomeAssistant. Вона об'єднує всі клієнтські застосунки – веб-адміністративну панель та мобільний застосунок – в єдиний керований комплекс, що працює за узгодженими правилами та гарантує цілісність даних. Сервер побудовано на основі Node.js з використанням Express, що дозволило створити гнучке, продуктивне та розширюване API.

Архітектура бекенду організована за принципом чіткого розділення відповідальності. Маршрути відповідають лише за приймання запиту та виклик відповідного контролера. Контролери обробляють дані, перевіряють права доступу, здійснюють базову валідацію та викликають сервіси, які містять основну логіку. Такий підхід дозволяє уникнути дублювання коду, зробити його більш зрозумілим та простим у підтримці. Завдяки цьому архітектура легко розширюється: нові функціональні модулі можуть бути додані без впливу на вже існуючі частини системи.

Усі дані зберігаються в MongoDB, а доступ до них здійснюється за допомогою Mongoose. Моделі описують структуру колекцій та забезпечують валідацію на рівні шеми. До головних моделей системи належать користувачі, автомобілі, паркомісця, тарифи та сесії паркування. Для кожної структури визначені обов'язкові поля, типи даних і правила валідації, що дозволяє гарантувати цілісність інформації та мінімізувати ризик втрати або пошкодження даних. Особливу роль відіграє модель сесії паркування, оскільки вона фіксує момент початку й завершення паркування, обчислює вартість та зберігає історію взаємодії користувача з системою.

Окрему увагу приділено механізму автентифікації. Сервер використовує токени доступу, які надсилаються клієнтськими застосунками після входу. Кожен захищений маршрут перевіряє наявність і валідність токена, що

унеможливлює несанкціонований доступ до внутрішніх частин системи. Адміністративні функції також захищені правами доступу, що дозволяє розмежувати можливості між різними типами користувачів [3, 4].

Важливою складовою серверної частини є обробка логіки, пов’язаної з паркомісцями. Сервер відповідає за створення нових паркомісць, їх редагування та оновлення статусу. Під час взаємодії з мобільним застосунком або адміністративною панеллю бекенд виконує перевірку умов, наприклад, чи не зайняте місце, чи зазначений автомобіль належить користувачу, чи не існує активної сесії, і лише після цього формує відповідну дію. Такий підхід забезпечує стійкість системи до помилок та некоректних дій [8, 20].

Особливо важливою частиною бекенду є механізм інтеграції з HomeAssistant, який відповідає за реальне виконання команд щодо відкриття або блокування парковочного стопера. Після того як користувач або адміністратор ініціює дію, сервер виконує всі необхідні логічні перевірки, зберігає інформацію в базі даних і лише після цього надсилає команду до HomeAssistant. Потім бекенд отримує відповідь про зміну стану реле та оновлює дані про сесію паркування. Така двонаправлена взаємодія забезпечує синхронність фізичного стану пристрою зі станом, відображеним у системі.

Сервер також виконує функцію центру розрахунку вартості паркування. При завершенні сесії програма визначає її тривалість та відповідний тариф, після чого розраховує суму до списання. Цей процес реалізовано на рівні сервісів і дозволяє легко змінювати логіку обчислення вартості або додавати нові тарифні моделі.

Архітектурна гнучкість бекенду, використання сучасних інструментів та чітке розділення відповідальностей роблять серверну частину надійною основою всієї системи. Вона здатна обробляти великий обсяг запитів, забезпечувати безпеку користувачів та гарантувати коректну взаємодію між програмними та апаратними компонентами. Такий підхід створює міцний фундамент для подальшого розширення функціональності та інтеграції нових можливостей без значної перебудови архітектури.

## 2.3. Система керування пристроями через HomeAssistant

HomeAssistant є ключовим елементом архітектури системи розумного паркування, оскільки забезпечує стабільне та уніфіковане керування фізичними пристроями, які блокують або відкривають конкретне паркомісце. Через нього реалізована інтеграція з Zigbee-реле та іншими компонентами IoT-інфраструктури, що дозволяє відокремити апаратну частину від програмної та значно спрощує розробку і масштабування системи.

У рамках цього проєкту використовується Zigbee-реле з чотирма каналами, кожен з яких відповідає за певну дію парковочного механізму. Після підключення реле до Zigbee-шлюзу воно автоматично розпізнається в HomeAssistant і стає доступним як одна або декілька сутностей (entities). HomeAssistant бере на себе весь процес керування пристроєм: встановлення з’єднання, підтримку Zigbee-мережі, відстеження стану реле та виконання команд у реальному часі. Саме завдяки цьому бекенд не взаємодіє з пристроєм безпосередньо і не залежить від нюансів конкретної Zigbee-реалізації.

Комунікація між серверною частиною системи та HomeAssistant здійснюється через його офіційний REST API. Бекенд надсилає HTTP-запити для активації потрібного каналу реле та отримання актуального стану пристрою. Запит може містити команду увімкнення, вимкнення або перемикання каналу, залежно від логіки керування конкретним парковочним стопером. У відповідь HomeAssistant повертає оновлені дані про стан пристрою, що дозволяє серверу переконатися у коректному виконанні операції та, у разі необхідності, повідомити клієнтський застосунок про результат.

Використання HomeAssistant замість прямої інтеграції з Zigbee-реле має низку важливих переваг. По-перше, це підвищує стабільність роботи системи, оскільки вся взаємодія з пристроями покладається на платформу, спеціально розроблену для роботи з IoT-обладнанням. HomeAssistant вже має вбудовані механізми повторних спроб, обробки збоїв, відновлення зв’язку та логування.

Це дозволяє серверу працювати значно простіше: достатньо передати команду, не переймаючись низькорівневими деталями протоколів.

По-друге, HomeAssistant надає можливість легко змінювати фізичні пристрої без змін у бекенді чи мобільному застосунку. Якщо у майбутньому буде використано інше реле, інший Zigbee-шлюз або навіть інший протокол (наприклад, Wi-Fi, Z-Wave чи Matter), HomeAssistant приховає від системи всі відмінності. Це робить архітектуру гнучкою і придатною для масштабування.

По-третє, централізоване управління через HomeAssistant суттєво підвищує безпеку. Фізичні пристрої не мають прямого доступу до мережі, у якій працюють користувацькі застосунки, і не потребують відкритих портів або власних API. Усі операції виконуються через HomeAssistant, який має власні механізми аутентифікації, ключі доступу та журналювання подій.

Взаємодія бекенду з HomeAssistant є інтегральною частиною системи. Після отримання запиту на відкриття або блокування паркомісця бекенд перевіряє права користувача, валідність операції, стан активної сесії та лише після цього надсилає команду до HomeAssistant. У відповідь система отримує підтвердження виконання дії, що дозволяє фіксувати подію в базі даних і коректно відображати зміни в адміністративній панелі та мобільному застосунку.

Таким чином, HomeAssistant забезпечує основу фізичного контролю над паркомісцями, виконуючи роль проміжного рівня між серверною частиною та реальним обладнанням. Саме завдяки йому система може працювати стабільно, безпечно та універсально, а також підтримувати широкі можливості для майбутнього розширення IoT-функціоналу.

## 2.4. Веб-адміністративна панель

Веб-адміністративна панель є інструментом для керування всією системою розумного паркування та забезпечує доступ до функцій, необхідних для контролю інфраструктури, налаштування тарифів, моніторингу

паркомісць і перегляду актуальної інформації. Вона створена з використанням React і TypeScript, що забезпечує високу надійність коду, модульність та передбачуваність роботи інтерфейсу. Підхід компонентної архітектури дозволяє ефективно організувати логіку та повторно використовувати елементи інтерфейсу в різних частинах системи.

Однією з ключових особливостей адміністративної панелі є робота з картою, яка дозволяє зручно переглядати всі паркомісця в їх реальному розташуванні. Карта використовується як основний спосіб представлення інформації про стан паркувальних зон, а також як засіб швидкої навігації між ними. Адміністратор може відкривати деталі кожного паркомісця, переглядати його координати, стан, прив’язаний тариф та виконувати інші дії, що значно пришвидшує роботу з великими об’єктами.

У системі також реалізована можливість додавання і редагування паркомісць. Інтерфейс дозволяє обирати точку безпосередньо на карті, після чого дані координати автоматично завантажуються у форму створення елемента. Це спрощує процес адміністративного налаштування та мінімізує ризики помилок при створенні нових зон. Схожа логіка застосовується й до інших елементів панелі: тарифи, HTML-вміст і користувачі мають власні інтерфейси, що дозволяють швидко змінювати параметри або додавати нові значення.

Для роботи з даними адміністративна панель використовує запити до бекенду через спеціальні сервісні модулі. Кожен з них відповідає за певний напрямок взаємодії: тарифи, користувачі, паркомісця або HTML-вміст. Панель не містить зайвої логіки і виступає лише інтерфейсом до можливостей бекенду, що робить її простою в обслуговуванні та розширенні. У випадку зміни API достатньо оновити лише відповідний сервісний модуль.

Особливу роль у роботі панелі відіграє система валідації форм, яка забезпечує коректність даних перед надсиланням на сервер. Валідація виконується з використанням React Hook Form у поєднанні з Yup, що дозволяє легко описувати правила для кожного поля та швидко відображати помилки

користувачу. Це важливо для стабільності системи, адже некоректні дані можуть впливати на роботу мобільного застосунку, бекенду та HomeAssistant.

З точки зору користувацького досвіду інтерфейс панелі побудований з застосуванням Tailwind CSS, завдяки чому система має сучасний вигляд та адаптивну верстку. Tailwind допомагає легко налаштовувати стиль окремих компонентів, не створюючи при цьому надлишкових CSS-файлів. Простота стилізації пришвидшує розробку нових елементів та робить інтерфейс візуально узгодженим.

Адміністративна панель також передбачає систему авторизації для обмеження доступу до керування паркувальною інфраструктурою. Після входу користувач отримує доступ до основних розділів панелі, де може переглядати та змінювати відповідні дані. Це забезпечує захист від несанкціонованого доступу та розмежує можливості між адміністратором і звичайними користувачами мобільного застосунку.

У комплексі веб-адміністративна панель виконує роль центру управління системою. Вона надає інструменти для щоденної роботи, швидкого конфігурування та моніторингу паркомісць і дозволяє розширювати функціонал без суттєвих змін у бекенді чи мобільному застосунку. Завдяки чітко структурованому інтерфейсу та інтеграції з бекендом усі операції виконуються швидко, безпечно та з мінімальною кількістю помилок.

## 2.5. Мобільний застосунок

Мобільний застосунок є ключовим інструментом взаємодії кінцевого користувача з системою розумного паркування. Він забезпечує доступ до всієї необхідної функціональності: від процесу реєстрації та входу до керування автомобілями, перегляду карти, відкриття паркомісця та контролю активної сесії паркування. Уся робота застосунку побудована на основі React Native та Expo, що забезпечує можливість кросплатформенного розвитку й однакову поведінку на пристроях Android та iOS.

Застосунок використовує сучасний підхід до навігації, який поєднує стекові екрани для етапів авторизації та нижнє меню для основних функцій після входу. Спочатку користувач проходить процес верифікації номера телефона, підтвердження SMS-коду та встановлення PIN-коду. Цей етап реалізовано через низку послідовних екранів, що керуються React Navigation. Після успішної авторизації користувач отримує доступ до головного інтерфейсу, де наведені основні можливості застосунку.

Одним з найбільш важливих компонентів мобільного застосунку є карта, яка дозволяє переглядати розташування доступних паркомісць у реальному часі. Вона реалізована на основі React Native Maps, що забезпечує плавну роботу та підтримку геолокації. Після отримання дозволу застосунок визначає місцезнаходження користувача та відображає його на карті, а також завантажує список паркомісць через бекенд. Завдяки цьому користувач може оцінити, які місця доступні поруч, а також натиснути на конкретну точку, щоб отримати інформацію про тариф, відстань та можливість відкриття паркомісця.

Логіка роботи з автомобілями користувача реалізована через окремий екран, де можна переглядати список зареєстрованих авто, додавати нові та редагувати існуючі. Ця частина інтерфейсу використовує модальні вікна та нижні штори, завдяки чому користувач отримує інтуїтивно зрозумілий і сучасний досвід взаємодії. Валідація форм виконується через react-hook-form, що забезпечує швидкий зворотний зв’язок у разі помилок і робить процес введення даних більш надійним.

Коли користувач обирає паркомісце та автомобіль, застосунок надсилає запит до бекенду на відкриття стопера. Далі бекенд взаємодіє з HomeAssistant, який керує Zigbee-реле, і у випадку успішного виконання команда відображається в інтерфейсі мобільного застосунку. Користувач може бачити статус активної парковки, час, який минув, та іншу важливу інформацію. Усі оновлення відбуваються в режимі реального часу через періодичні запити до сервера, що дозволяє підтримувати актуальність даних.

Окрему роль у застосунку відіграє контекст глобального стану, який зберігає інформацію про авторизованого користувача, список авто, активну парковку та інші дані. Такий підхід дозволяє уникати дублювання логіки та забезпечує доступ до спільних даних у різних компонентах інтерфейсу. Крім того, застосунок використовує локальне сховище для зберігання токена та інформації, необхідної для автоматичного входу під час наступного запуску.

Екран профілю надає можливість переглядати інформацію про користувача, поповнювати баланс та виходити з системи. Процедура поповнення реалізована через окреме модальне вікно, що відкривається знизу екрана, забезпечуючи зручний і зрозумілий інтерфейс. Увесь процес повністю інтегрований з бекендом, що гарантує коректність транзакцій та контроль стану рахунку.

У цілому мобільний застосунок виконує роль основного інструменту, через який користувач взаємодіє з паркувальною системою. Він поєднує карту, керування авто, авторизацію та контроль активних сесій у єдиний зручний інтерфейс. Завдяки чіткій структурі, використанню сучасних технологій та інтеграції з бекендом і HomeAssistant застосунок забезпечує стабільну, передбачувану та комфортну роботу для користувачів.

## 2.6. Висновки до розділу

У цьому розділі було розглянуто повну програмну архітектуру системи розумного паркування, що охоплює серверну частину, веб-адміністративну панель, мобільний застосунок та підсистему керування пристроями через HomeAssistant. Разом ці компоненти формують комплексне рішення, яке поєднує програмні та апаратні засоби в єдину інтегровану інфраструктуру.

Серверна частина забезпечує виконання всієї бізнес-логіки, централізоване управління даними та обробку запитів від клієнтських застосунків. Вона виступає ядром системи, що координує взаємодію між користувачами, адміністраторами та фізичними пристроями. Окреме значення

має інтеграція з HomeAssistant, яка дозволяє перенести роботу з IoT-пристроями у спеціалізоване середовище, мінімізуючи складність та підвищуючи стабільність системи.

Веб-адміністративна панель забезпечує ефективне керування паркомісцями, тарифами та іншими параметрами системи. Вона надає адміністраторам необхідні інструменти для моніторингу роботи інфраструктури та швидкого внесення змін. Завдяки використанню сучасних веб-технологій інтерфейс залишається зручним, гнучким і придатним до розширення.

Мобільний застосунок виконує роль інтерфейсу для кінцевих користувачів, забезпечуючи доступ до карти, управління автомобілями та виконання операцій із паркування. Його функціонал тісно інтегрований з бекендом і фізичною інфраструктурою, що дозволяє користувачам взаємодіяти зі системою у зручний спосіб та в реальному часі. Завдяки продуманій архітектурі він забезпечує стабільний та інтуїтивно зрозумілий користувацький досвід.

Загалом система демонструє цілісність підходу до реалізації проєкту, де кожен компонент виконує чітко визначену роль, а разом вони створюють ефективне, масштабоване та надійне рішення для автоматизації процесів паркування. Інтеграція програмних і апаратних складових через HomeAssistant відкриває можливість подальшого розвитку системи та підключення нових пристроїв без значних змін у логіці роботи. Така архітектура забезпечує основу для розширення системи, підвищення її функціональності та адаптації до майбутніх потреб.

## РОЗДІЛ 3. Демонстрація роботи системи

### 3.1 Загальний опис сценарію роботи користувача

Робота мобільного застосунку побудована таким чином, щоб користувач міг виконувати всі дії, пов’язані з паркуванням, максимально просто, інтуїтивно та швидко. Основна ідея полягає в тому, що весь функціонал системи подано у вигляді послідовності логічних кроків: користувач реєструється, додає свій автомобіль, поповнює баланс, переглядає доступні паркомісця на карті, виконує бронювання і, зрештою, здійснює паркування. Така структура дозволяє повністю автоматизувати процес взаємодії з паркувальною інфраструктурою, уникаючи контакту з фізичними пристроями та спрощуючи роботу користувача до кількох натискань на екрані смартфона.

Сценарій передбачає, що кожен користувач починає роботу із створення облікового запису. Реєстрація виконується за номером телефону, а підтвердження здійснюється через SMS-код, що дозволяє забезпечити автентичність користувача. Після цього застосунок пропонує встановити PIN-код, який буде використовуватися для подальшої авторизації без повторних SMS-повідомлень. Отримані дані надсилаються до серверної частини, де відбувається збереження інформації та створення нового профілю.

Після успішного входу користувач переходить до налаштування власного автопарку. Мобільний застосунок дозволяє зберігати декілька автомобілів, що робить систему зручною для сімей або користувачів, які володіють кількома транспортними засобами. Кожен автомобіль можна додати, відредагувати або видалити, а всі зміни відразу синхронізуються з бекендом.

Наступним етапом є керування балансом. Оскільки система передбачає оплату за фактичний час паркування, користувач може поповнити рахунок у будь-який момент. Сума зберігається на стороні сервера, а мобільний застосунок відображає її в режимі реального часу. Поповнення балансу є

важливою частиною сценарію, адже саме з цього етапу користувач отримує можливість активно взаємодіяти з паркомісцями.

Після налаштування профілю користувач переходить до роботи з картою. На ній відображаються всі доступні паркомісця, їхній статус, тариф та розташування. Карта дозволяє швидко знайти найближче місце та оцінити, яке з них найзручніше для паркування. Обравши конкретне паркомісце, користувач може його забронювати. Бронювання закріплює місце за користувачем, і система блокує можливість резервування для інших.

Прибувши на місце, користувач підтверджує свою присутність через кнопку «I’m on place». У цей момент мобільний застосунок надсилає команду на сервер, який, у свою чергу, передає її до HomeAssistant для опускання паркостопера. Після цього починається активна сесія паркування. Користувач бачить інформацію про тривалість, тариф і автомобіль, з яким він паркується. Система автоматично веде облік часу та забезпечує точний розрахунок вартості.

Після завершення паркування користувач натискає кнопку завершення. Сервер завершує сесію, обчислює кінцеву суму та списує її з балансу. Через певний час HomeAssistant автоматично підіймає паркостопер, а місце знову стає доступним для інших.

Таким чином, мобільний застосунок формує повноцінний цикл взаємодії користувача з системою – від першого входу до завершення парковки. Архітектура дозволяє виконувати всі операції у зручному та інтуїтивному інтерфейсі, приховуючи складні технічні процеси взаємодії між бекендом, базою даних та HomeAssistant.

### 3.2. Реєстрація нового користувача у мобільному застосунку

Процес реєстрації у мобільному застосунку складається з кількох послідовних кроків, які забезпечують автентифікацію користувача та збір базової інформації, необхідної для подальшої роботи із системою. Реєстрація

реалізована таким чином, щоб залишатися простою для користувача, але водночас гарантувати безпечний доступ до особистих даних та функцій керування паркомісцями.

Процес починається з введення номеру телефону. На першому екрані користувачу необхідно вказати свій номер, після чого мобільний застосунок надсилає його на бекенд. Сервер перевіряє, чи існує вже такий користувач у системі. Якщо номер не знайдено, створюється новий тимчасовий запис, і користувач переходить до наступного кроку реєстрації.

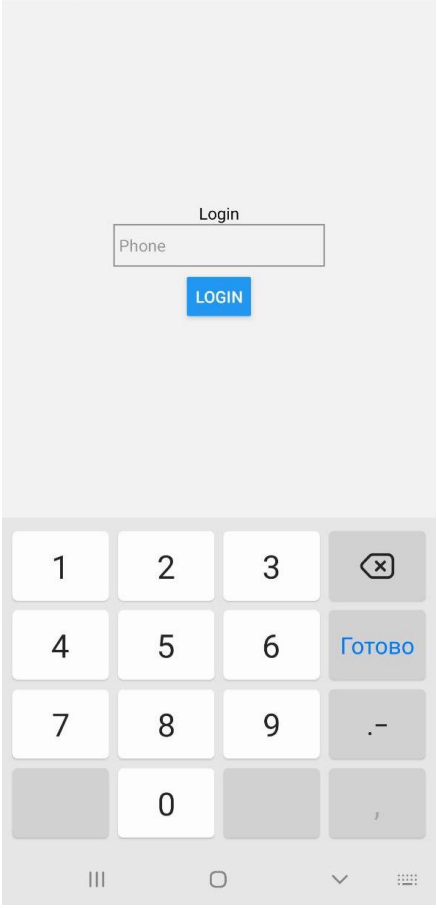


Рисунок 3.1 – Екран логіну

На другому екрані відображається форма підтвердження телефонного номера. На введений користувачем номер надсилається SMS-код, який необхідно ввести у відповідне поле. Після успішної верифікації бекенд оновлює статус користувача, позначаючи номер підтвердженим. Це дозволяє

запобігти створенню облікових записів із некоректними або недоступними номерами телефону.

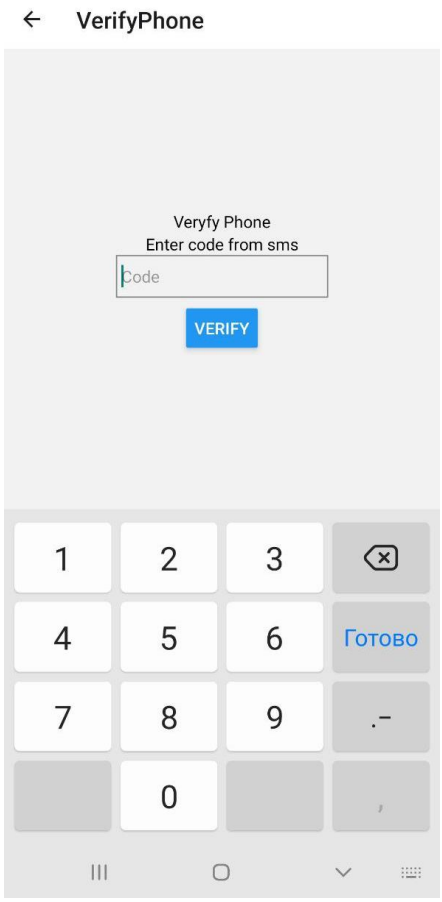


Рисунок 3.2 – Екран верифікації номера телефону

Далі користувач переходить до етапу введення додаткової інформації. На цьому екрані необхідно вказати ім’я та хоча б один автомобіль, оскільки система працює виключно з транспортними засобами, прив'язаними до конкретного користувача. До завершення заповнення всіх полів кнопка переходу до наступного кроку залишається неактивною. Такий підхід унеможливлює реєстрацію неповного профілю та гарантує коректність подальшої взаємодії із сервісом.

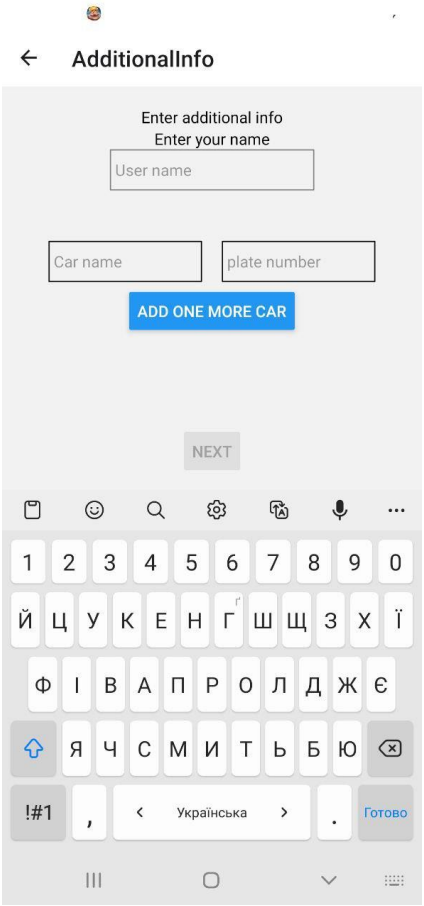


Рисунок 3.3 – Екран вводу інформації про користувача

Після введення ім’я та даних про автомобіль кнопка переходу стає доступною, що дозволяє продовжити процес реєстрації. На цьому етапі інформація синхронізується з бекендом, де створюється повноцінний профіль користувача разом із його автопарком.

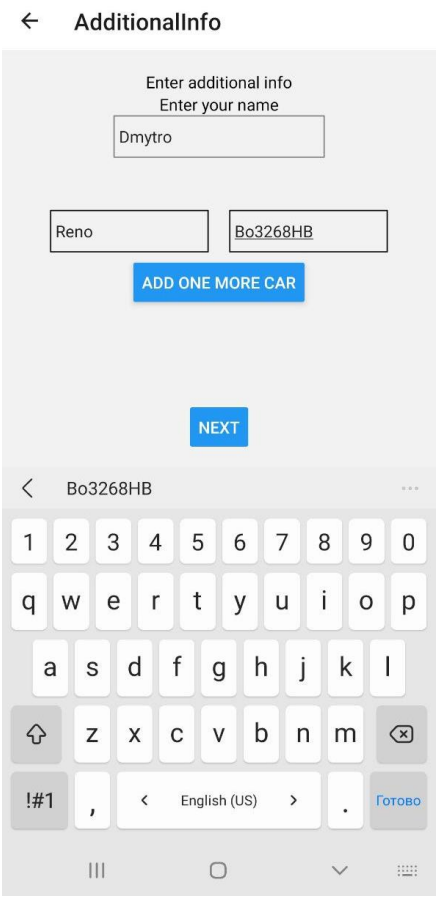


Рисунок 3.4 – Заповнена інформація про користувача

Наступним кроком є створення PIN-коду, який буде використовуватись для швидкої авторизації у додатку без необхідності повторного отримання SMS. Користувач вводить свій PIN-код на спеціальному екрані, після чого переходить до етапу його підтвердження. Цей механізм підсилює безпеку облікового запису та дозволяє залишатися в системі, не проходячи повну реєстрацію щоразу при повторному вході.

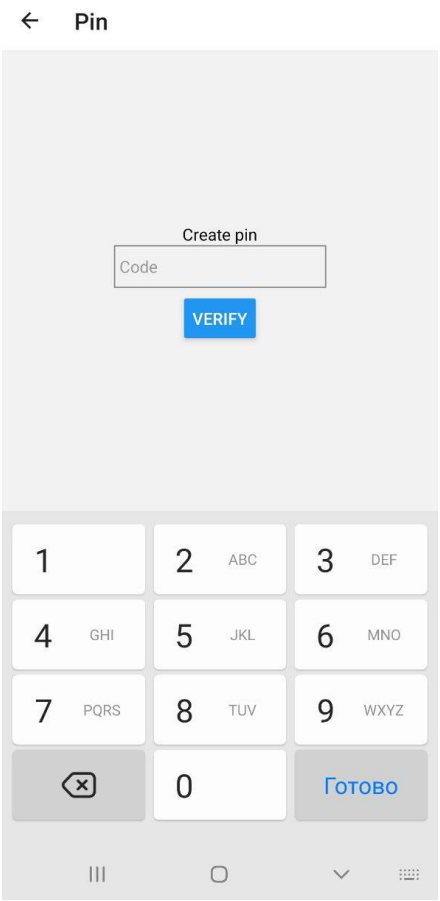


Рисунок 3.5 – Екран створення PIN-коду

Після повторного введення PIN-коду застосунок надсилає його на сервер для збереження. Якщо обидва введені значення збігаються, бекенд фіксує PIN як спосіб майбутньої авторизації користувача. У разі успіху користувача автоматично перенаправляє до головного екрану – карти доступних паркомісць.

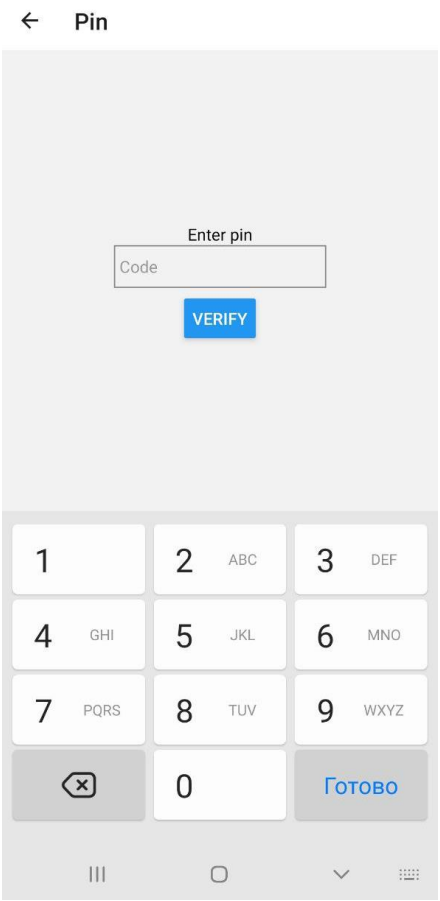


Рисунок 3.6 – Екран вводу PIN-коду

Після завершення всіх етапів реєстрації користувач вперше потрапляє на головний екран програми – карту. З цього моменту він може переглядати доступні місця, виконувати бронювання, керувати профілем та автомобілями. Реєстрація завершується повноцінною активацією облікового запису та підготовкою до використання всього функціоналу системи.

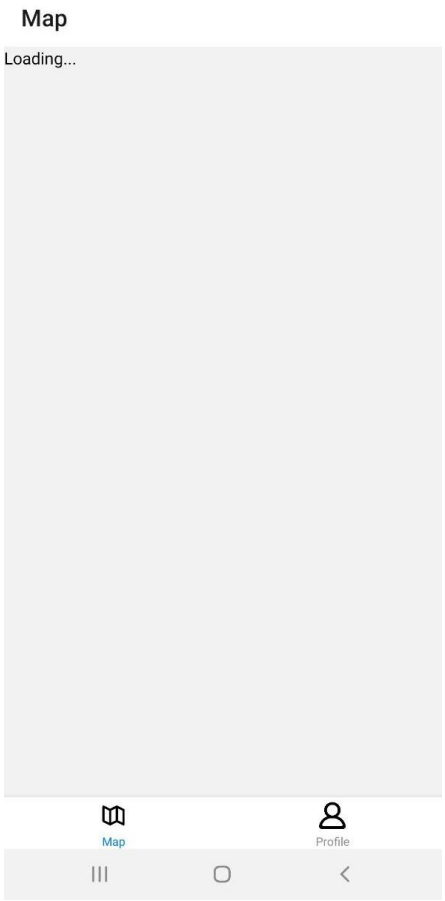


Рисунок 3.7 – Екран з картою

## 3.3. Авторизація користувача

Авторизація в мобільному застосунку побудована таким чином, щоб забезпечити зручний та швидкий доступ до системи без зайвих повторних дій з боку користувача. На відміну від процесу реєстрації, який виконується лише один раз, авторизація може відбуватися у різних сценаріях залежно від того, чи збережені дані користувача на пристрої, та чи є вони актуальними.

Після першого успішного входу застосунок зберігає необхідну інформацію локально на пристрої користувача. До цих даних належать ідентифікаційні параметри, отримані від бекенду, а також строк їхньої дії. Якщо збережені дані ще актуальні, користувача не просять повторно вводити PIN-код або номер телефону. У такому разі застосунок відкривається одразу на головному екрані — карті паркомісць. Це робить використання сервісу швидким і зручним під час щоденної експлуатації.

Якщо ж термін дії локальних даних минув (для цього застосунок використовує фіксований період у 24 години), користувачеві необхідно повторно пройти коротку процедуру авторизації. У такому випадку після запуску застосунку він бачить перший екран — поле для введення номеру телефону. Застосунок надсилає цей номер на бекенд, де виконується перевірка. Оскільки акаунт уже існує, сервер не створює нового користувача, а повертає відповідь про необхідність ввести PIN-код, створений під час первинної реєстрації.

Після цього користувач переходить на екран введення PIN-коду.

Введений PIN-код надсилається на сервер, де він перевіряється на відповідність збереженому для цього облікового запису. Якщо PIN вказано правильно, користувач отримує нові валідні авторизаційні дані, які знову зберігаються на пристрої з новим строком дії у 24 години. Після цього застосунок автоматично перенаправляє користувача на карту.

Якщо ж PIN введено неправильно, користувач бачить повідомлення про помилку і може повторити спробу. Така логіка забезпечує належний рівень захисту персональних даних і водночас не створює зайвих незручностей під час щоденного користування застосунком.

Таким чином, авторизація у мобільному застосунку поєднує два важливі аспекти: зручність для користувача та безпечне зберігання даних. Завдяки локальному кешуванню інформації застосунок відкривається миттєво, а процедура повторного входу потребує мінімуму дій і використовується лише тоді, коли це дійсно необхідно.

### 3.4. Налаштування автомобілів користувача

Автомобілі є центральним елементом у роботі системи, оскільки всі операції з паркуванням прив’язуються до конкретної машини. Під час реєстрації користувач уже вводить інформацію про свій перший автомобіль, адже без цього створити обліковий запис неможливо. Однак у реальному житті

користувач може мати декілька автомобілів або змінювати їх з часом, тому в застосунку передбачено окремий інтерфейс для керування автопарком.

Доступ до керування автомобілями знаходиться на вкладці **Profile**, яку можна відкрити через нижню навігаційну панель. На цьому екрані користувач бачить загальну інформацію свого профілю та пункт **Cars**, що веде до списку доданих транспортних засобів.

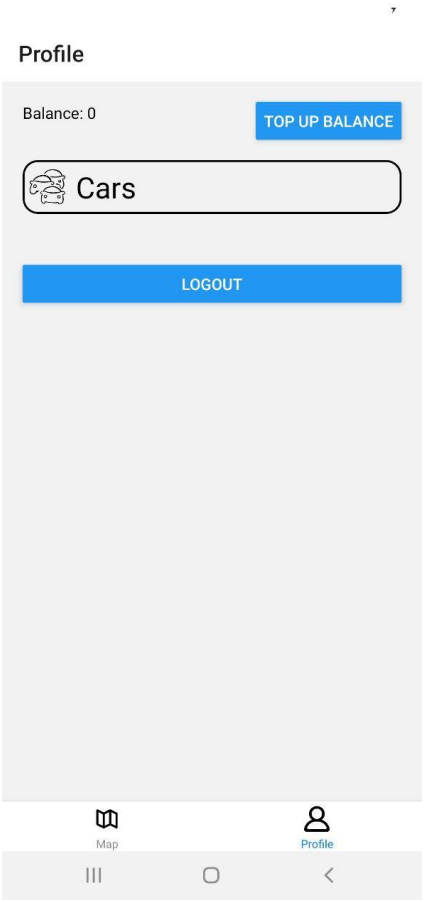


Рисунок 3.8 – Екран профілю

Після переходу до розділу Cars користувачу відкривається список автомобілів, доданих раніше. Кожен автомобіль відображається у вигляді окремого блоку, де видно його назву та номерний знак. Тут же розміщена кнопка для видалення автомобіля. Інтерфейс відображений максимально просто, щоб користувач одразу бачив актуальні дані про свої машини та міг швидко приймати необхідні дії.

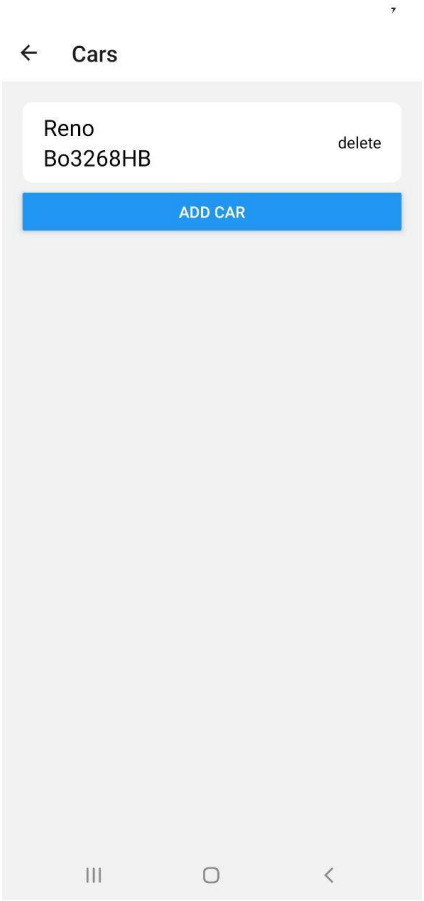


Рисунок 3.9 – Екран налаштування автомобілів

У разі потреби користувач може додати новий автомобіль. Для цього достатньо натиснути кнопку **ADD CAR**, після чого під списком автомобілів відкривається форма для введення назви машини та номерного знака. Обидва поля є обов'язковими, адже саме ці дані використовуються як в інтерфейсі, так і під час надсилання команд до бекенду – зокрема при бронюванні та початку паркування.

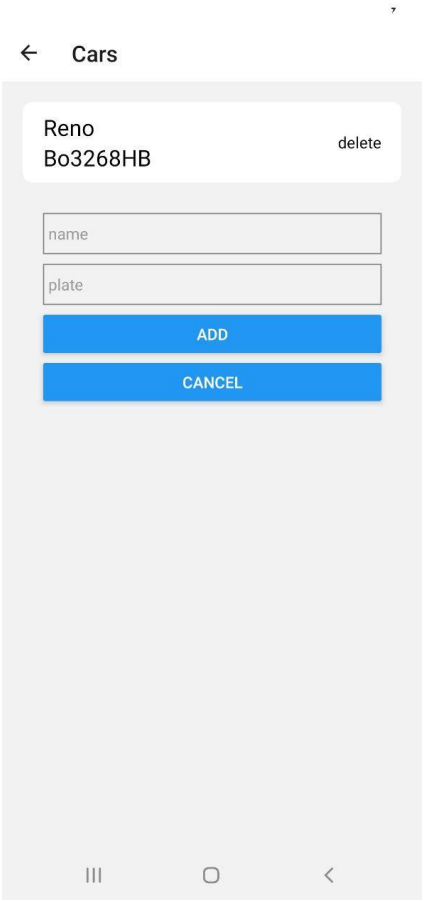


Рисунок 3.10 – Екран створення автомобіля

Після введення інформації користувач натискає кнопку **ADD**, і застосунок надсилає новий автомобіль на бекенд, де він зберігається у профілі користувача. Дані оновлюються миттєво, а новий автомобіль одразу з’являється у списку. Якщо ж користувач передумав або помилився, він може скористатися кнопкою **CANCEL**, щоб закрити форму без збереження змін.

Окрім додавання, користувач може видалити будь-який автомобіль зі списку. Це може знадобитися у випадках продажу авто, зміни номерного знака або іншої актуалізації даних. Натискання кнопки **delete** біля відповідного автомобіля запускає запит до бекенду, після чого автомобіль видаляється з профілю, і відображення списку оновлюється. Системою не накладається обмеження на мінімальну кількість автомобілів після реєстрації: після створення облікового запису користувач має право залишити навіть лише один авто або додати декілька.

Логіка керування транспортними засобами вибудувана так, щоб бути прозорою і гнучкою. Всі операції – додавання, видалення та редагування – негайно синхронізуються з бекендом. Це забезпечує коректну роботу всієї системи при бронюванні та старті паркування, оскільки сервер завжди отримує актуальний перелік автомобілів користувача.

У результаті користувач має повний контроль над своїм автопарком без складних налаштувань або зайвих дій. Застосунок дозволяє зручно підтримувати профіль у актуальному стані, забезпечуючи повну відповідність особистих даних реальній ситуації.

### 3.5. Керування балансом користувача

Фінансова модель роботи системи побудована навколо механізму віртуального балансу користувача. Такий підхід був обраний через те, що вартість паркування розраховується не погодинно, як у більшості аналогічних сервісів, а похвилинно та навіть посекундно. Час перебування автомобіля на паркомісці часто складно передбачити наперед, і традиційна модель із фіксованою оплатою за годину створює незручності: користувач змушений переплачувати за невикористаний час, що зменшує задоволеність сервісом.

У запропонованій системі оплати користувач сплачує тільки за фактичний час, протягом якого він дійсно користувався паркомісцем. Для реалізації цього підходу застосунок не знімає кошти напряму з картки під час завершення паркування, а працює через попереднє поповнення особистого віртуального балансу. Кошти, які користувач вносить при поповненні, надходять одразу на баланс сервісу, а йому у застосунку нараховується еквівалентна сума внутрішньої валюти. З цієї суми потім відбувається списання вартості парковки.

Такий підхід є вигідним для обох сторін: користувач отримує гнучку й чесну систему оплати, а сервіс – можливість акумулювати кошти наперед. Це

також зменшує транзакційне навантаження під час кожної сесії паркування та спрощує роботу системи.

Керування балансом здійснюється у вкладці **Profile**, де у верхній частині екрана відображається поточний стан рахунку. Поруч знаходиться кнопка **TOP UP BALANCE**, що відкриває вікно поповнення.

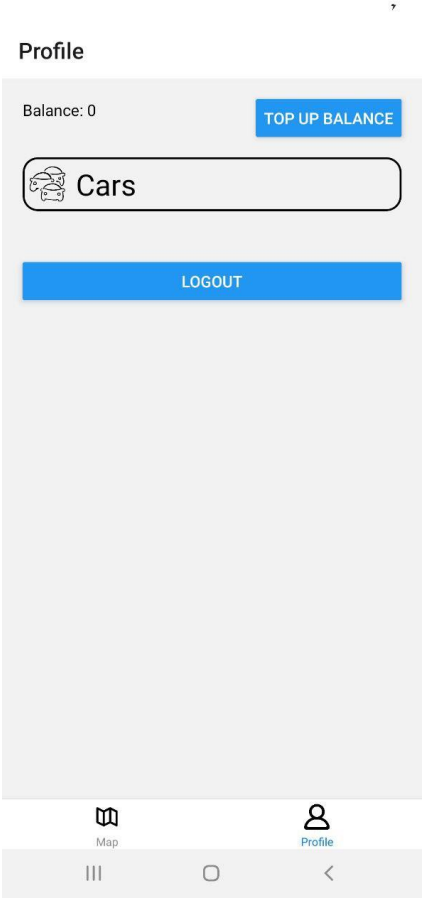


Рисунок 3.11 – Екран з балансом

Після натискання кнопки відкривається модальне вікно, де користувач може ввести бажану суму для поповнення. Застосунок виконує базову валідацію введених даних, не дозволяючи надсилати порожні або некоректні значення.

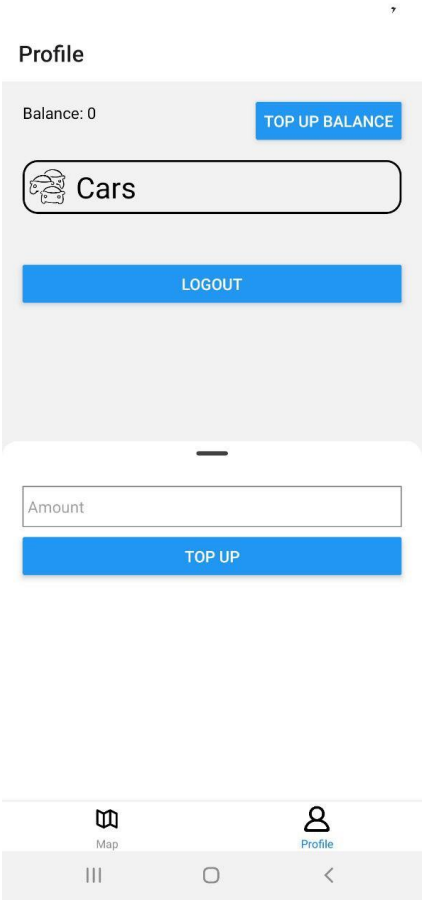


Рисунок 3.12 – Модальне вінко поповнення балансу

Після введення суми користувач підтверджує дію, і запит надсилається на бекенд. На даному етапі платіжна система ще не інтегрована, тому використовується тестова заглушка. Вона імітує успішне поповнення та одразу збільшує віртуальний баланс користувача на зазначену суму. Завдяки цьому механізму можлива повноцінна демонстрація роботи системи без реальних транзакцій.

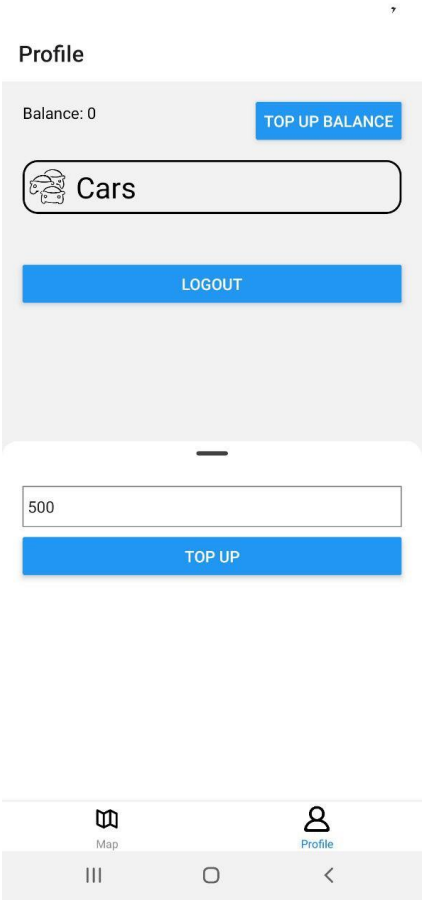


Рисунок 3.13 – Модальне вінко поповнення балансу з введеною сумою

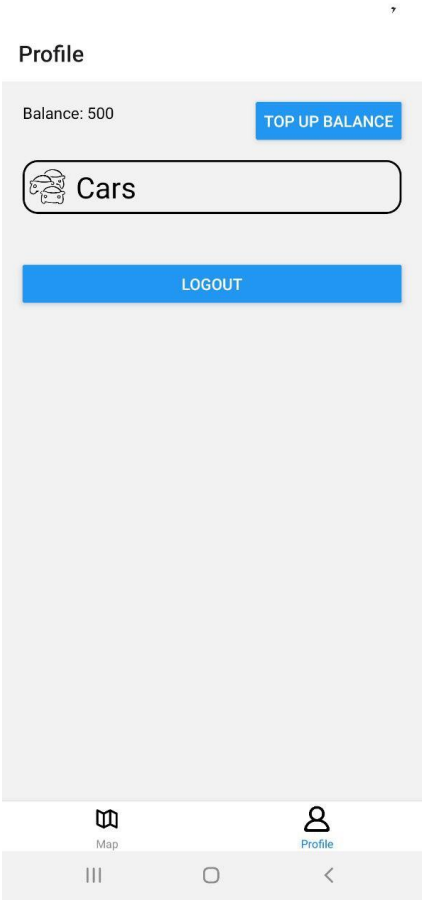


Рисунок 3.14 – Екран балансу після поповнення

Поповнений баланс дозволяє користувачу почати нову сесію паркування, але система додатково захищена логікою мінімального порогу. Під час натискання кнопки початку паркування бекенд перевіряє, чи достатньо коштів на рахунку для старту сесії. Мінімальною умовою є наявність суми, що покриває вартість однієї години паркування. Це дозволяє уникнути ситуацій, коли користувач одразу опиняється у мінусі через надто малий залишок.

Однак система зберігає гнучкість і під час самої сесії. Якщо користувач помилився у прогнозі часу або затримався на паркомісці, баланс може тимчасово перейти у негативне значення. Наприклад, користувач міг поповнити баланс на суму, достатню для однієї години, але простояти півтори. У такому випадку завершення паркування призведе до від’ємного балансу, наприклад: –50 грн. Це нормальна ситуація, яка передбачена логікою роботи сервісу.

У разі від’ємного або недостатнього балансу система блокує можливість почати нову сесію паркування. Щоб знову отримати доступ до функції бронювання та паркування, користувач повинен повністю погасити мінус і залишити на рахунку хоча б мінімальний залишок, необхідний для старту.

Таким чином, механізм віртуального балансу поєднує гнучкість, прозорість і зручність. Він дозволяє користувачам платити тільки за реальний час перебування на паркомісці, а сервісу – стабільно працювати й надавати чесний тариф без прихованих переплат. У поєднанні з простим інтерфейсом поповнення баланс стає інтуїтивною й важливою частиною загального користувацького досвіду.

### 3.6. Бронювання місця та процес паркування

Екран із картою є центральним елементом взаємодії користувача з системою, оскільки саме тут відбувається вибір парковки, перегляд її параметрів, бронювання місця та запуск процесу паркування. На карті відображаються всі доступні парковки, позначені червоними маркерами. Після натискання на будь-який маркер у нижній частині екрана з’являється інформаційне вікно, у якому зазначено адресу парковки, кількість паркомісць, кількість доступних та вартість стоянки.

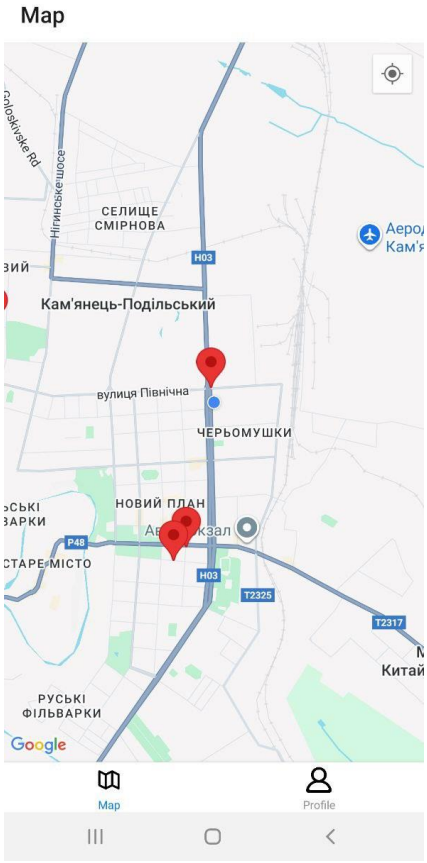


Рисунок 3.15 – Екран з картою

Однією з важливих частин логіки цього екрану є визначення фактичної відстані від користувача до обраної парковки. Додаток отримує поточні координати пристрою та порівнює їх із координатами парковки. Якщо користувач знаходиться на значній відстані, додаток забороняє початок паркування: кнопка **PARK** стає недоступною та відображається попередження про те, що необхідно наблизитися до парковки. Це виключає можливість зловживання системою, коли паркування запускається на відстані й користувач займає місце, якого фактично не використовує. У такому випадку доступною залишається лише кнопка **BOOK PLACE**, а також додатковий функціонал побудови маршруту до парковки через Google Maps. Натиснувши іконку навігації, користувач автоматично переходить у Google Maps із проложеним шляхом від його поточного місця до обраної парковки.

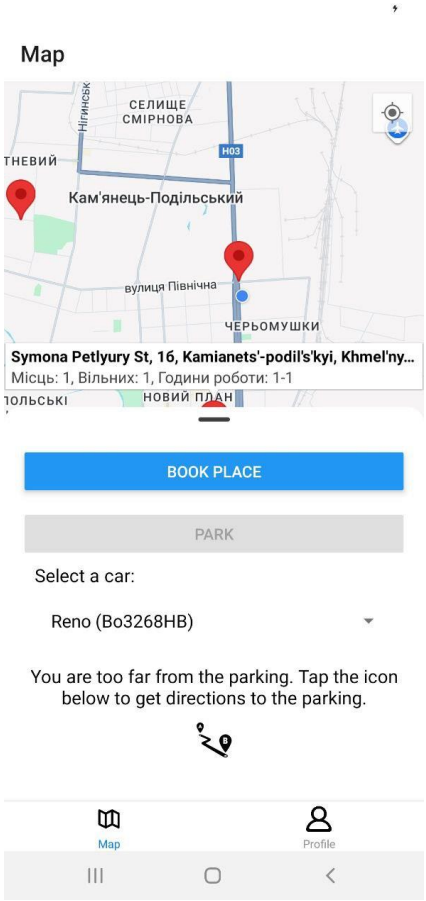


Рисунок 3.16 – Попередження про віддаленість від парковки

Бронювання місця є ключовою функцією додатку для парковок, обладнаних електричними паркостоперами. Такі паркостопери постійно перебувають у піднятому стані, блокуючи місце від сторонніх автомобілів. Коли користувач натискає кнопку **BOOK PLACE**, додаток надсилає запит до бекенду, де відбувається пошук доступного вільного стопера. У разі успішного бронювання інформація про стопер закріплюється за користувачем, і він отримує гарантію, що приїхавши на парковку, обов’язково знайде вільне місце.

Особливо важливо, що процес паркування починається автоматично у момент бронювання. Такий підхід продиктований бізнес-логікою: заброньоване місце фактично стає зайнятим, оскільки інші водії вже не можуть ним користуватися. Тому одразу після бронювання запускається лічильник часу та починається нарахування вартості відповідно до тарифу. Таким чином,

користувач оплачує не лише період фактичного перебування авто на місці, а й час, протягом якого місце було зарезервоване й недоступне іншим.

Після прибуття на парковку, користувач повинен підійти до свого стопера та натиснути кнопку **I'M ON PLACE**. Це сигналізує системі, що користувач фізично прибув до паркомісця. Бекенд у відповідь надсилає команду в HomeAssistant, який керує Zigbee-реле, відповідальним за конкретний паркостопер. Система отримує команду опустити стопер, після чого користувач може заїхати на місце та залишити автомобіль.

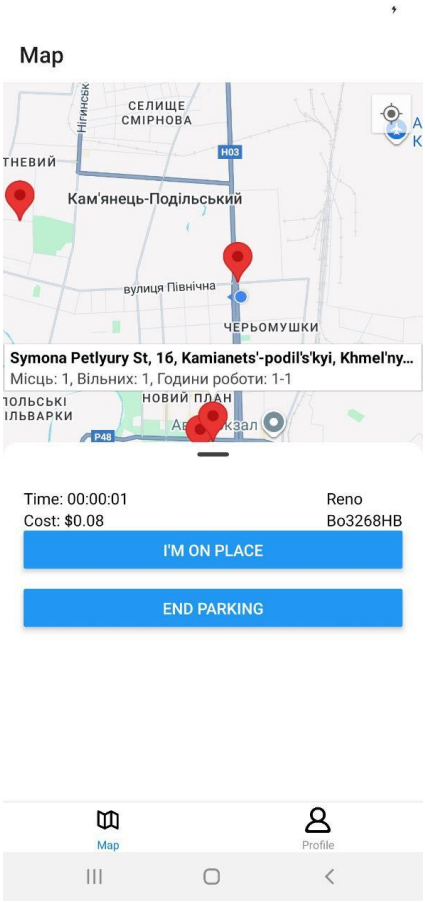


Рисунок 3.17 – Забронюване місце

У випадку, якщо бронювання не потрібно, і користувач знаходиться безпосередньо біля парковки, він може розпочати паркування вручну. Для цього потрібно вибрати зі списку один зі своїх автомобілів і натиснути кнопку **PARK**. Після запуску паркування додаток переходить у режим відображення таймера, де у реальному часі показується тривалість стоянки та поточна сума,

яка буде списана із балансу. Сума оновлюється щосекунди згідно з тарифом, а інформація про автомобіль теж відображається на екрані.

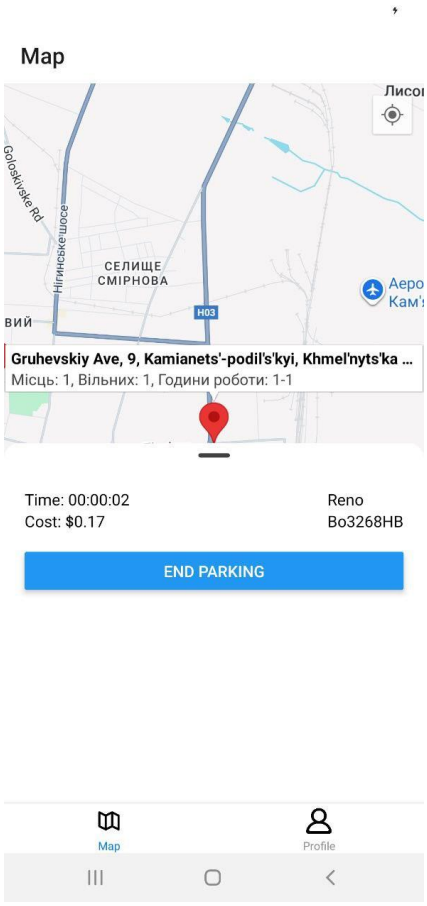


Рисунок 3.18 – Процес звичайного паркування

Поки триває паркування, користувач повністю контролює процес: він бачить, як змінюється час і вартість, а також може завершити паркування у будь-який момент за допомогою кнопки **END PARKING**. Коли користувач завершує паркування, бекенд зупиняє таймер, обчислює загальний час стоянки та остаточну вартість, після чого списує її з балансу користувача. Якщо на місці використовується паркостопер, то після завершення паркування система через HomeAssistant автоматично підіймає стопер через кілька секунд після виходу авто, звільняючи місце для інших водіїв.

Після завершення операції користувач повертається до стандартного вигляду карти, де знову може переглядати інші парковки, бронювати місця або запускати нову сесію паркування.

Уся логіка взаємодії – від визначення місцезнаходження користувача до управління фізичним обладнанням на парковці – реалізована таким чином, щоб процес був максимально автоматизованим, прозорим та зручним. Система забезпечує користувачу впевненість у тому, що заброньоване місце буде чекати на нього, а розрахунок вартості завжди точний і базується на фактичному часі користування. Завдяки інтеграції з HomeAssistant та електричними паркостоперами досягається повна синхронізація цифрової логіки додатку з фізичною інфраструктурою парковок [8, 9, 20].

## 3.7 Висновки до розділу

Розділ, присвячений демонстрації роботи мобільного застосунку, показує, що розроблена система паркування є не просто інтерфейсом для користувача, а повністю продуманою та інтегрованою екосистемою, у якій кожен елемент має визначену роль і логічно взаємодіє з іншими компонентами. Застосунок забезпечує користувачеві послідовний і логічный сценарій роботи – від створення облікового запису до завершення паркування – і робить це в максимально інтуїтивній формі.

Початкові етапи використання системи – реєстрація та авторизація – реалізовано таким чином, щоб звести кількість дій для користувача до мінімуму. SMS-підтвердження гарантує достовірність номера телефону, а PIN-код – безпечний доступ до облікового запису. Застосунок самостійно керує строком дії авторизації, зберігаючи дані локально, що забезпечує швидкий вхід у систему без зайвих перевірок. Така модель суттєво спрощує використання сервісу та робить його доступним навіть для користувачів, які не мають досвіду роботи зі складними застосунками.

Налаштування автомобілів користувача демонструє гнучкість системи та її здатність адаптуватися під реальні потреби. Можливість додавати необмежену кількість автомобілів, редагувати їхні дані або видаляти за потреби дозволяє формувати профіль, який відповідає актуальній ситуації.

Цей функціонал є важливою частиною системи, адже всі операції паркування прив’язуються саме до конкретного автомобіля.

Система внутрішнього балансу забезпечує унікальну модель оплати, де користувач платить тільки за фактично використаний час. На відміну від традиційних парковок, де встановлена мінімальна погодинна оплата, розроблена система дозволяє нараховувати вартість похвилинно або навіть посекундно. Завдяки цьому користувач не переплачує за невикористаний час, а сервіс отримує гнучку й чесну схему розрахунків. Важливо, що баланс працює разом із механізмами безпеки: він блокує можливість почати нову сесію при недостатній кількості коштів та запобігає накопиченню боргів.

На карті реалізовано ключову логіку взаємодії між користувачем, бекендом та фізичним обладнанням парковки. Система враховує місцезнаходження користувача і забезпечує динамічну зміну доступних дій залежно від того, чи знаходиться користувач поруч із паркомісцем. Можливість побудувати маршрут через Google Maps спрощує навігацію та робить сервіс ще зручнішим.

Особливо важливою частиною системи є функціонал бронювання. Він вирішує відразу кілька проблем, поширених на традиційних паркувальних майданчиках: невпевненість у наявності вільного місця, необхідність чекати, поки хтось виїде, або ризик знайти всі місця зайнятими після довгого пошуку. У розробленій системі користувач отримує можливість гарантовано закріпити місце, а через інтеграцію з HomeAssistant працює автоматичне управління електричними паркостоперами, які фізично захищають місце від сторонніх автомобілів. Такий рівень автоматизації не лише підвищує комфорт користувача, а й повністю синхронізує цифрову та фізичну частини сервісу.

Процес паркування працює у реальному часі, а користувач має повний контроль над своєю сесією. Він бачить точну вартість кожної секунди стоянки, може завершити паркування у будь-який момент, а система безпомилково обчислює остаточну суму та вивільняє місце для наступного користувача. Важливо, що навіть у випадку бронювання або затримки дороги система

поводиться передбачувано – час починає рахуватися автоматично, як тільки місце зарезервовано.

У своїй сукупності всі ці функції створюють мобільний застосунок, що є важливим елементом загальної інфраструктури «розумної» парковки. Він виконує одночасно роль інтерфейсу для користувача, механізму управління паркувальними пристроями та інструмента фінансової взаємодії із сервісом. Завдяки цьому користувач отримує сервіс, що забезпечує зручність, передбачуваність та повну автоматизацію усіх процесів, а сама система демонструє гнучкість, масштабованість та високий потенціал для подальшого розвитку.

# ВИСНОВКИ

У ході виконання кваліфікаційної роботи було здійснено повний цикл розробки інтерактивної системи для керування процесами паркування, яка включає три основні компоненти: серверну частину, веб-адміністративну панель та мобільний застосунок для кінцевих користувачів. Усі складові були детально проаналізовані, спроєктовані, розроблені та інтегровані між собою таким чином, щоб сформувати єдину функціональну екосистему, здатну забезпечити ефективну взаємодію з паркувальною інфраструктурою в реальному часі.

На початковому етапі було проведено аналіз існуючих рішень, визначено ключові проблеми, властиві традиційним системам паркування, та сформульовано вимоги до майбутньої системи [3,6,8,17,20]. Особлива увага приділялася питанням автоматизації, зручності користування, прозорості фінансових взаємодій та інтеграції з фізичними пристроями. Це дозволило визначити чітку ціль роботи – створити комплексну систему, здатну оптимізувати процеси паркування та забезпечити користувачам доступ до сучасних сервісів.

У межах реалізації поставленої задачі було створено серверну частину, яка відповідає за бізнес-логіку, обробку даних, взаємодію з базою даних, управління станами паркомісць і синхронізацію роботи між всіма клієнтськими компонентами [3, 4, 5, 13]. Сервер забезпечує стабільний доступ до даних, виконує розрахунок вартості паркування в режимі реального часу, контролює бронювання та здійснює перевірки балансу користувача. Особлива роль належить інтеграції з HomeAssistant – платформою, що дозволяє керувати електронними паркостоперами через Zigbee-реле [6, 7, 14]. Завдяки цьому система отримала можливість фізично контролювати доступ до паркомісць і працювати в повністю автоматизованому режимі.

Для адміністраторів було розроблено зручну веб-панель, яка дозволяє керувати користувачами, транспортними засобами, тарифами, парковками, робочими годинами та станом обладнання. Ця панель забезпечує прозорість і

повний контроль над усіма процесами системи, дозволяє швидко вносити зміни та відстежувати динаміку роботи сервісу. Таким чином, вона виконує роль центру управління всією інфраструктурою паркування.

Мобільний застосунок став ключовим інструментом взаємодії кінцевих користувачів із системою. Він охоплює повний набір функцій: реєстрацію, авторизацію, керування транспортними засобами, поповнення балансу, перегляд доступних парковок на карті, бронювання місця, початок і завершення паркування. Застосунок працює у тісній синхронізації з бекендом, відображаючи стан системи в реальному часі. Особливо важливими є функції автоматичного управління паркостоперами, можливість побудови маршруту через Google Maps, а також унікальна модель оплати – похвилинне або посекундне нарахування вартості. Завдяки цьому користувач платить лише за фактично використаний час стоянки, що робить сервіс максимально справедливим та конкурентоспроможним.

Окремо варто зазначити, що всі компоненти були протестовані як окремо, так і у складі інтегрованої системи. Були перевірені сценарії реєстрації та авторизації, бронювання місця, взаємодії з HomeAssistant, обробки фінансових операцій, роботи карти парковок, синхронізації між бекендом та клієнтськими застосунками. Тестування підтвердило коректність реалізації, стабільність роботи системи та адекватність її поведінки у різних режимах користування.

У межах даної роботи вдалося успішно досягти всіх поставлених цілей. Створена система демонструє високий рівень автоматизації, зручність використання та гнучкість у налаштуванні. Вона поєднує програмні та апаратні рішення, забезпечуючи повний цикл роботи з паркувальною інфраструктурою – від бронювання до завершення паркування з автоматичним підняттям паркостопера. Це дозволяє суттєво оптимізувати процеси, покращити користувацький досвід та створити новий підхід до організації парковок.

Важливим результатом є також масштабованість системи. Завдяки модульній архітектурі можна легко додавати нові парковки, тарифні моделі, типи обладнання чи додаткові сервіси. Це робить систему придатною для подальшого розширення та впровадження у реальних умовах міської інфраструктури.

Таким чином, виконана кваліфікаційна робота не лише підтвердила можливість створення комплексної автоматизованої системи паркування, але й продемонструвала перспективність її використання у майбутньому. Результати розробки можуть стати основою для подальших досліджень, оптимізації алгоритмів, розширення функціоналу та реального впровадження у сучасних «розумних» містах.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ


1. Flanagan D. JavaScript: The Definitive Guide. 7th ed. O'Reilly Media, 2020. 706 p.
2. Banks A., Porcello E. Learning React: Functional Web Development with React and Redux. 2nd ed. O'Reilly Media, 2020. 350 p.
3. Hammad M. Full-Stack Web Development with React and NodeJS. Packt Publishing, 2023. 482 p.
4. Cantelon M., Harter M., Holowaychuk T., Rajlich N. Node.js in Action. 2nd ed. Manning Publications, 2017. 375 p.
5. Chodorow K. MongoDB: The Definitive Guide. 3rd ed. O'Reilly Media, 2019. 410 p.
6. Portnoy M. Home Assistant for Beginners & Smart Home Automation. Independently published, 2023. 214 p.
7. Smith J. Understanding Zigbee: The Low-Power Wireless Technology for the Internet of Things. 2nd ed. TechPress, 2021. 198 p.
8. Ray T. IoT Integration: A Developer's Guide to Building Connected Systems. Packt Publishing, 2022. 322 p.
9. Google Maps Platform Documentation. URL: https://developers.google.com/maps/documentation (дата звернення: 10.11.2025).
10. React Native Documentation. URL: https://reactnative.dev/docs (дата звернення: 10.12.2025).
11. HomeAssistant Official Documentation. URL: https://www.home-assistant.io/docs (дата звернення: 15.11.2025).
12. MongoDB Manual. URL: https://www.mongodb.com/docs/manual/ (дата звернення: 02.10.2025).

13. Express.js API Reference. URL: https://expressjs.com/ (дата звернення: 02.10.2025).

14. Zigbee2MQTT Documentation. URL: https://www.zigbee2mqtt.io/ (дата звернення: 10.10.2025).

15. González J. Mastering Full-Stack Development with JavaScript. Springer, 2021. 289 p.

16. Heitkotter H., Majchrzak T. Cross-Platform Mobile App Development Frameworks: A Comparative Study. ACM Computing Surveys, 2017. Vol. 49(4). pp. 1–28.

17. Burke S. Designing User Interfaces for Mobile Applications. MIT Press, 2022. 256 p.

18. Albahari J. C# and .NET Pocket Reference: Modern Backend Development Patterns. O’Reilly Media, 2021. 180 p.

19. Singh A. Building Scalable Backend Systems with REST and Node.js. Nova Science Publishers, 2022. 300 p.

20. Newton R. Modern Smart Parking Systems: Architecture, Automation and IoT Integration. IEEE Smart Cities Journal, 2023. №12. pp. 45–57.