Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра системного аналізу та інформаційних технологій

Бакалаврська кваліфікаційна робота на тему:

«ІНФОРМАЦІЙНА СИСТЕМА ДЛЯ БРОНЮВАННЯ ТА ВИКОРИСТАННЯ СПОРТИВНОГО ІНВЕНТАРЮ В СПОРТЗАЛАХ»

Виконав: студент 4 курсу, групи 2ІСТ-21б
спеціальності 126 - «Інформаційні системи та технології»
_____Максим БАРАНОВСЬКИЙ
Керівник: Ph.D., ст. викл. каф. САІТ
_____Ольга ВОЙЦЕХОВСЬКА
«04» 06 2025 р.
Рецензент: к.т.н., доцент кафедри КН
_____Ігор АРСЕНЮК
«12» 06 2025 р.

Допущено до захисту
Завідувач кафедри САІТ
_____д.т.н., проф. Віталій МОКІН
«06» 06 2025 р.

Вінниця ВНТУ – 2025 рік

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра системного аналізу та інформаційних технологій
Рівень вищої освіти – перший (бакалаврський)
Галузь знань – 12 Інформаційні технології
Спеціальність 126 – «Інформаційні системи та технології»
Освітньо-професійна програма – Прикладні інформаційні технології

ЗАТВЕРДЖУЮ
Завідувач кафедри САІТ
д.т.н., проф. Віталій МОКІН
"28" 05 2025 року

ЗАВДАННЯ
НА БАКАЛАВРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
Барановському Максиму Дмитровичу

1. Тема роботи: «Інформаційна система для бронювання та використання спортивного інвентарю в спортзалах»
керівник роботи: Войцеховська О. О., Ph.D., ст. викл. каф. САІТ
затверджені наказом ВНТУ від «20» 05 2025 року № 97
2. Термін подання студентом роботи 31.05.25
3. Вихідні дані до роботи:
1) Дані про спортивний інвентарь;
2) Сценарії використання спортивного інвентарю;
3) Стандарти класифікації спортивного інвентарю.
4. Зміст текстової частини:
1) Аналіз предметної області;
2) Вибір оптимальних інформаційних технологій;
3) Розроблення інформаційної системи для бронювання та використання спортивного інвентарю.
5. Перелік ілюстративного матеріалу:
1) Архітектура інформаційної системи;
2) Блок-схема зв'язків програмного забезпечення;
3) UML-діаграми;
4) Блок-схема алгоритму роботи мобільного додатка;
5) Структурна схема функціонування мобільного додатку;
6) Інтерфейс додатку.

6. Консультанти розділів роботи:

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 28.03.25 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва та зміст етапу	Термін виконання		Примітка
		початок	закінчення	
1	Аналіз предметної області	01.04	15.04	вик
2	Огляд аналогів	15.04	30.04	вик
3	Вибір оптимальних інформаційних технологій	01.05	10.05	вик
4	Розроблення інформаційної системи для бронювання та використання спортивного інвентарю в спортзалах	10.05	20.05	вик
5	Оформлення матеріалів до захисту БКР	20.05	30.05	вик

Студент _____ Максим БАРАНОВСЬКИЙ

Керівник роботи _____ Ольга ВОЙЦЕХОВСЬКА

АНОТАЦІЯ

Бакалаврська кваліфікаційна робота складається з 86 сторінок формату А4, на яких є 43 рисунка, список використаних джерел містить 23 найменувань.

Метою цієї роботи є розроблення інформаційної системи для бронювання та використання спортивного інвентарю в спортзалах.

Розроблено інформаційну систему для бронювання та управління спортивним інвентарем, а саме мобільний додаток на основі таких сучасних веб-технологій, як: React, TypeScript, Vite і TailwindCSS. Описано архітектуру, реалізацію основних функцій для користувачів і адміністраторів, а також інтеграцію модуля штучного інтелекту для формування персоналізованих тренувальних рекомендацій. Розроблена інформаційна система автоматизує процеси бронювання, підвищує якість обслуговування клієнтів і забезпечує гнучке управління інвентарем для адміністрації спортивного закладу.

Ключові слова: інформаційна система, мобільний додаток, бронювання, спортивний інвентар, React, TypeScript, Vite, TailwindCSS, штучний інтелект.

ABSTRACT

The bachelor's thesis consists of 86 A4 pages, which contain 43 figures, and the list of sources used contains 23 names.

The aim of this work is to develop a modern information system for booking and using sports equipment in gyms.

An information system for booking and managing sports equipment has been developed, namely a mobile application based on such modern web technologies as: React, TypeScript, Vite and TailwindCSS. The architecture, implementation of basic functions for users and administrators, as well as the integration of an artificial intelligence module for generating personalized training recommendations have been described. Developed information system automates booking processes, improves the quality of customer service and provides flexible inventory management for the administration of a sports facility.

Keywords: information system, mobile application, booking, sports equipment, React, TypeScript, Vite, TailwindCSS, artificial intelligence.

ЗМІСТ

ВСТУП

Актуальність теми. У сучасному спортивному середовищі ефективне управління ресурсами спортзалів є ключовим фактором для забезпечення оптимального досвіду тренувань. За останніми даними, більшість відвідувачів спортзалів стикаються з проблемами доступу до потрібного інвентарю, що негативно позначається на ефективності їх тренувань. Відсутність зручних систем бронювання спортивного обладнання призводить до незадоволеності клієнтів, неефективного використання ресурсів та зниження рентабельності спортзалів.

Сучасні веб-технології пропонують широкі можливості для створення інтерактивних систем, які здатні суттєво покращити процеси координації та розподілу спортивного інвентарю. Впровадження цифрових систем управління спортзалами стає ключовим кроком для оптимізації обліку та підвищення продуктивності використання ресурсів. Тому актуальною є задача по створенню інформаційної системи для бронювання та використання спортивного інвентарю в спортзалах, що поєднує зручність користувацького інтерфейсу з передовими технологіями, такими як штучний інтелект та геолокаційні сервіси.

Мета і задачі дослідження. Метою дослідження є розроблення інформаційної системи для бронювання та використання спортивного інвентарю в спортзалах.

Для досягнення поставленої мети необхідно розв'язати такі задачі:

- провести аналіз предметної області та існуючих рішень для бронювання спортивного інвентарю;
- здійснити вибір оптимальних інформаційних технологій для розробки інформаційної системи;
- здійснити вибір оптимальної IT-інфраструктури;
- розробити інформаційну систему для бронювання та використання спортивного інвентарю в спортзалах у вигляді мобільного додатку;

– провести тестування додатку для оцінки його ефективності та надійності.

Об'єктом дослідження є процес розроблення інформаційної системи для бронювання та використання спортивного інвентарю в спортзалах.

Предметом дослідження є методи та програмні засоби розроблення інформаційної системи для бронювання та використання спортивного інвентарю в спортзалах.

Публікації. За результатами даної роботи опубліковано тези доповідей на тему: «Інформаційна система для бронювання та використання спортивного інвентарю в спортзалах» на Міжнародній науково-практичній інтернет-конференції «Молодь в науці: дослідження, проблеми, перспективи» (Вінниця, 2024-2025 рр.) [1].

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз предметної області

Сфера фізичної культури та спорту в сучасному суспільстві набуває все більшого значення, оскільки здоровий спосіб життя, регулярна фізична активність і відвідування спортзалів стають невід'ємною частиною повсякденності багатьох людей. Зростання кількості спортивних закладів, фітнес-центрів і тренажерних залів супроводжується підвищенням вимог до якості обслуговування, ефективності використання матеріально-технічної бази та впровадження інноваційних рішень для оптимізації внутрішніх процесів. Згідно з аналітичним оглядом Gymnago, перехід на цифрові системи управління спортзалами у 2025 році став необхідністю, що дозволяє покращити облік ресурсів та підвищити ефективність їх використання [2].

Однією з найактуальніших проблем для адміністрацій спортзалів та їхніх клієнтів є питання раціонального розподілу та використання спортивного інвентарю. Згідно з аналітичним звітом Міністерства молоді та спорту України, у 2025 році понад 65% відвідувачів спортивних закладів зазначили труднощі з доступністю необхідного інвентарю, що впливає на якість тренувального процесу [3]. У багатьох закладах досі використовуються застарілі підходи до обліку обладнання, наприклад, паперові журнали чи усна домовленість між відвідувачами. Це призводить до низки негативних наслідків: виникнення черг, конфліктів за популярне обладнання, нераціонального використання ресурсів, а також до зниження рівня задоволеності клієнтів. Згідно з дослідженням Core Health & Fitness про глобальні фітнес-тренди 2025 року, нестача доступного інвентарю залишається однією з головних проблем, з якими стикаються відвідувачі спортзалів [4].

Відсутність автоматизованих систем бронювання ускладнює роботу персоналу, оскільки адміністратори змушені вручну контролювати наявність обладнання, вести облік його використання та вирішувати спірні ситуації між

клієнтами. Це не лише збільшує навантаження на працівників, а й підвищує ризик помилок, втрати або псування інвентарю. Крім того, без сучасних інформаційних рішень неможливо ефективно аналізувати статистику завантаженості обладнання, планувати закупівлі чи обслуговування, а також прогнозувати пікові періоди використання.

З розвитком цифрових технологій та поширенням веб-сервісів з'явилася можливість впровадження інформаційних систем, які дозволяють автоматизувати процеси бронювання, обліку та моніторингу спортивного інвентарю. Такі системи забезпечують прозорість розподілу ресурсів, дозволяють уникати дублювання бронювань, зменшують кількість конфліктних ситуацій та підвищують ефективність використання обладнання. Впровадження цифрових рішень для управління спортивним обладнанням дозволяє автоматизувати процеси бронювання та моніторингу, що значно підвищує ефективність використання ресурсів [5].

Інформаційна система для бронювання спортивного інвентарю повинна вирішувати низку завдань, серед яких:

- забезпечення зручного доступу користувачів до каталогу обладнання з можливістю фільтрації за категоріями, станом чи популярністю;
- відображення актуальної інформації про доступність інвентарю;
- можливість бронювання обладнання на визначений час із автоматичним підтвердженням чи відмовою;
- сповіщення користувачів про статус бронювання, нагадування про час використання;
- ведення історії використання інвентарю для кожного користувача;
- аналітика завантаженості та популярності обладнання для адміністрації.

Особливу актуальність набуває інтеграція таких систем із сучасними технологіями, зокрема штучним інтелектом, який може пропонувати персоналізовані рекомендації щодо вибору інвентарю чи тренувальних програм, а також прогнозувати пікові навантаження на обладнання. Додатково,

використання геолокаційних сервісів дозволяє відображати розташування інвентарю у залі, що спрощує його пошук для нових відвідувачів.

Варто зазначити, що впровадження інформаційних систем у спортивній сфері відповідає сучасним тенденціям цифрової трансформації та розвитку сервісної економіки. Це дозволяє не лише оптимізувати внутрішні процеси, а й підвищити рівень задоволеності клієнтів, що є ключовим чинником успіху будь-якого спортивного закладу. Використання спеціалізованого програмного забезпечення для управління бронюваннями, наприклад, у Fitness Studio Software, дозволяє фітнес-центрам покращити комунікацію з клієнтами, зменшити кількість пропущених занять та оптимізувати розподіл ресурсів [6]. Крім того, цифрові системи дають змогу збирати та аналізувати великі обсяги даних про використання обладнання, що відкриває нові можливості для стратегічного планування, маркетингу та розвитку бізнесу.

Таким чином, аналіз предметної області підтверджує актуальність розробки та впровадження інформаційної системи для бронювання та використання спортивного інвентарю. Це рішення дозволяє вирішити низку організаційних, технічних та сервісних проблем, підвищити ефективність роботи спортзалу та забезпечити якісний сервіс для його відвідувачів.

1.2 Огляд аналогів

З метою створення сучасної та зручної інформаційної системи для бронювання спортивного інвентарю доцільно проаналізувати існуючі рішення, які вже представлені на ринку. Аналіз аналогів дозволяє не лише перейняти найкращі практики, а й уникнути типових недоліків, що зустрічаються у конкурентів. Серед найбільш відомих систем, які частково або повністю вирішують подібні завдання, можна виділити такі продукти:

- Pitchbooking;
- Wodify;
- Glofox.

Розглянемо кожен з них детально.

Pitchbooking [7] – це платформа, яка спеціалізується на управлінні бронюванням спортивних об’єктів, зокрема майданчиків, залів та інвентарю. Система дозволяє користувачам у реальному часі переглядати доступність обладнання, здійснювати бронювання на зручний час, а адміністраторам – ефективно контролювати розклад та завантаженість ресурсів. Однією з переваг Pitchbooking є простий та інтуїтивно зрозумілий інтерфейс, що робить процес бронювання максимально швидким і зручним навіть для нових користувачів. Крім того, система підтримує автоматичні сповіщення про майбутні бронювання та зміни у розкладі, що підвищує рівень сервісу. Проте, функціонал Pitchbooking більше орієнтований на бронювання цілих об’єктів (залів, майданчиків), а не окремих одиниць інвентарю, що може бути обмеженням для спортзалів із великою кількістю різноманітного обладнання. На рисунку 1.1 зображено інтерфейс додатку «Pitchbooking».

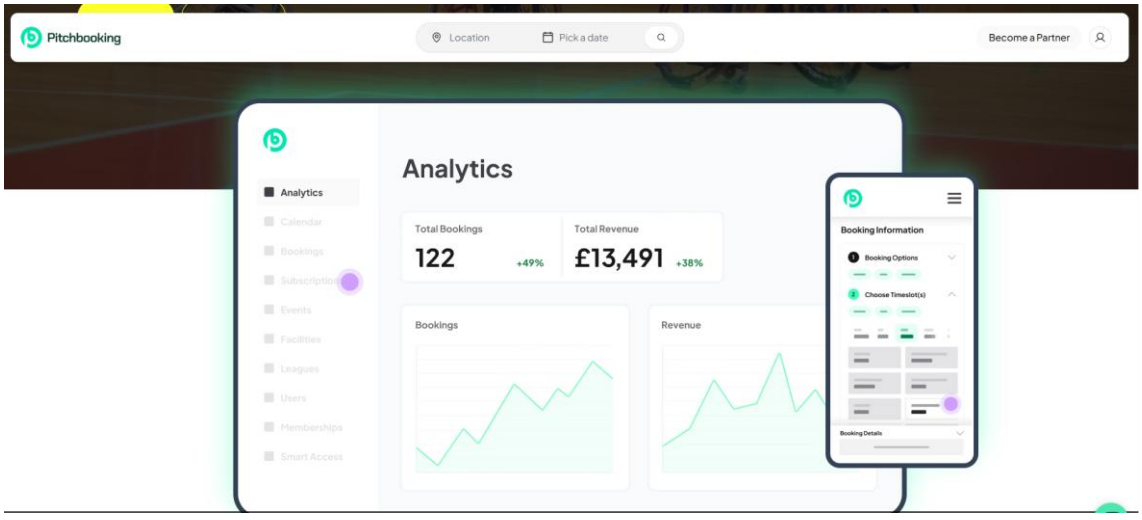

Рисунок 1.1 – Інтерфейс додатку «Pitchbooking»

Wodify [8] – це комплексна система для управління фітнес-клубами, яка включає модулі для бронювання занять, обліку клієнтів, ведення фінансової звітності та аналітики. Однією з особливостей Wodify (рис. 1.2) є можливість інтеграції з мобільними додатками, що дозволяє користувачам бронювати

тренування та обладнання зі смартфона. Система також надає адміністраторам інструменти для моніторингу завантаженості залу, аналізу популярності різних видів інвентарю та автоматичного формування розкладу. До переваг Wodify можна віднести гнучкість налаштувань, багатофункціональність та наявність аналітичних звітів. Водночас, для невеликих спортзалів система може бути надто складною та дорогою у впровадженні, а її інтерфейс – перевантаженим для звичайних користувачів.

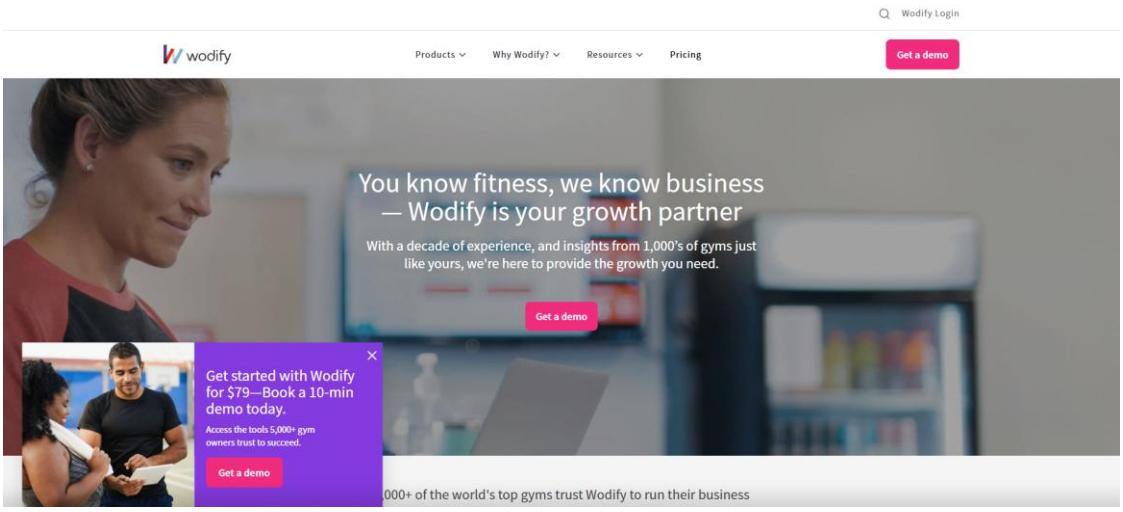

Рисунок 1.2 – Інтерфейс додатку «Wodify»

Glofox [9] – ще один популярний продукт, орієнтований на автоматизацію роботи фітнес-центрів та спортивних клубів. Glofox (рис. 1.3) дозволяє організовувати розклад занять, вести облік клієнтів, здійснювати онлайн-бронювання обладнання та тренувань. Система має сучасний мобільний додаток, що забезпечує зручний доступ до функцій бронювання у будь-який час. Серед переваг Glofox– простота використання, можливість інтеграції з платіжними системами та підтримка push-сповіщень. Однак, як і у випадку з Pitchbooking, основний акцент зроблено на бронюванні занять або тренажерних залів, а не окремих одиниць інвентарю. Це може створювати певні обмеження для закладів, де важливо контролювати доступ до кожного виду обладнання окремо.

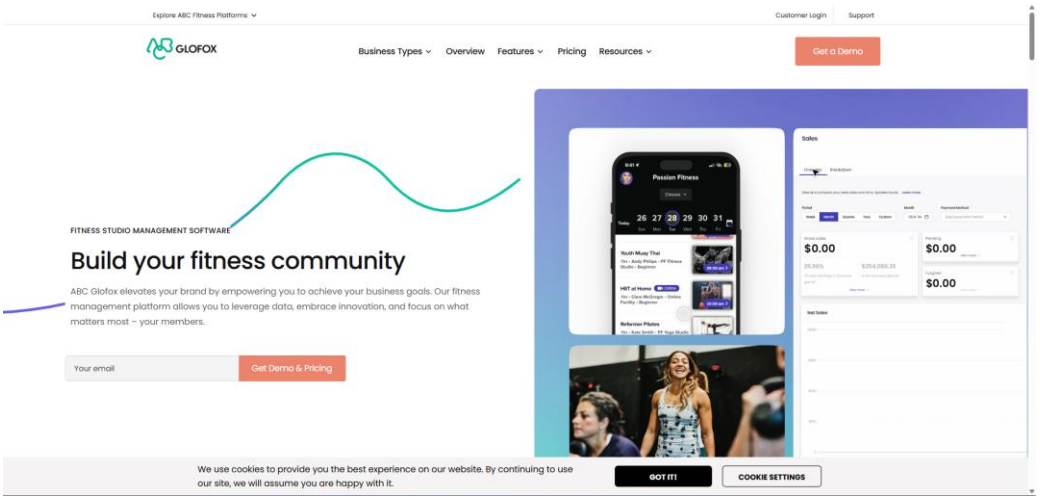

Рисунок 1.3 – Інтерфейс додатку «Glofox»

Підсумовуючи, можна зазначити, що всі розглянуті аналоги мають низку сильних сторін, таких як зручний інтерфейс, підтримка мобільних додатків, автоматичні сповіщення та аналітика. Водночас, жодна з систем не пропонує повноцінного рішення для детального бронювання саме спортивного інвентарю, що є актуальним для багатьох сучасних спортзалів. Це підкреслює доцільність розробки власної інформаційної системи, яка враховуватиме специфіку роботи з різними видами обладнання, забезпечить гнучкість налаштувань та простоту використання для всіх категорій користувачів.

1.3 Формування технічного завдання

Технічне завдання для інформаційної системи бронювання та використання спортивного інвентарю у спортзалах сформовано на основі реального функціоналу, реалізованого у дослідженні. Основна мета системи – забезпечити сучасний, зручний і безпечний веб-сервіс, який дозволяє користувачам переглядати доступний спортивний інвентар, бронювати його на визначений час, отримувати персоналізовані рекомендації та взаємодіяти з адміністрацією закладу.

Система побудована на сучасних веб-технологіях, що забезпечують швидкодію, адаптивність та масштабованість. Інтерфейс розроблений таким чином, щоб користувач міг легко зареєструватися, увійти до системи, переглянути каталог обладнання, скористатися фільтрами для пошуку потрібного інвентарю, переглянути детальну інформацію про кожну одиницю та здійснити бронювання. Для зручності реалізовано історію бронювань, можливість скасування або зміни замовлення, а також отримання сповіщень про статус бронювання.

Адміністрація спортзалу повинна мати розширені можливості для керування каталогом обладнання, додавання та редагування інформації про інвентар, контролю технічного стану, а також перегляду статистики використання. Система має надавати аналітичні дані щодо завантаженості обладнання, популярності різних категорій інвентарю, що дозволить приймати обґрунтовані рішення щодо оновлення чи закупівлі нового обладнання.

Особливістю системи є інтеграція модуля штучного інтелекту, який аналізує вподобання користувача, історію його тренувань та бронювань і надає персоналізовані рекомендації щодо вибору обладнання чи вправ. Також реалізовано багатомовний інтерфейс, що дозволяє перемикати мову системи для зручності різних категорій користувачів.

Таким чином, технічне завдання охоплює створення багатофункціональної, інтуїтивно зрозумілої, безпечної та гнучкої у налаштуванні системи, яка автоматизує всі основні процеси бронювання та використання спортивного інвентарю, враховує сучасні вимоги до дизайну, персоналізації, багатомовності та забезпечує простоту подальшого розширення функціоналу.

1.4 Висновки

У цьому розділі було проведено аналіз предметної області, розглянуто сучасний стан організації процесу бронювання спортивного інвентарю.

Порівняння існуючих рішень виявило їх обмеженість у керуванні окремими одиницями інвентарю та відсутність персоналізації. Запропонована система, на відміну від аналогів, забезпечує комплексний підхід, поєднуючи функції бронювання з AI-рекомендаціями, що підвищує ефективність використання ресурсів спортзалу та якість обслуговування клієнтів.

На основі отриманих результатів сформульовано технічне завдання, яке враховує актуальні потреби користувачів і адміністрації спортзалів, а також сучасні тенденції розвитку інформаційних технологій.

2 ВИБІР ОПТИМАЛЬНИХ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

2.1 Аналіз можливостей мови програмування TypeScript

TypeScript – це сучасна мова програмування, яка поєднує у собі переваги JavaScript [10] із потужністю статичної типізації. Однією з найважливіших характеристик TypeScript [11] є можливість визначати типи даних ще на етапі написання коду. Це дозволяє розробнику одразу бачити потенційні помилки, уникати неочікуваних ситуацій під час виконання програми та підвищувати загальну надійність програмного продукту.

Важливою особливістю TypeScript є підтримка об’єктно-орієнтованого підходу. Мова дозволяє використовувати класи, інтерфейси, наслідування, абстракції, що значно спрощує структурування коду, робить його більш зрозумілим і легким для підтримки. Завдяки цьому розробники можуть створювати масштабовані архітектури, які легко розширювати та модифікувати у майбутньому.

Ще однією перевагою TypeScript є його сумісність із JavaScript. Це означає, що будь-який існуючий код на JavaScript може бути використаний у проєкті на TypeScript без необхідності повної переписування. Така гнучкість дозволяє поступово впроваджувати нові підходи у вже існуючі системи, не порушуючи їхню працездатність.

TypeScript має розвинену систему універсальних типів, що дає змогу створювати функції та класи, які працюють із різними типами даних. Це підвищує гнучкість коду, дозволяє уникати дублювання та робить програму більш адаптивною до змін [12].

Варто відзначити, що TypeScript активно підтримується спільнотою розробників і має широку екосистему. Для цієї мови існує безліч бібліотек, інструментів для тестування, автоматизації, аналізу коду, а також типових визначень для популярних фреймворків. Завдяки цьому розробники отримують

доступ до сучасних інструментів, які значно спрощують процес створення, тестування та супроводу програмного забезпечення.

TypeScript також сприяє підвищенню продуктивності команди. Завдяки автодоповненню, підсвічуванню синтаксису, швидкому переходу до визначень та рефакторингу, розробники можуть ефективніше працювати з великими кодовими базами, швидше знаходити та виправляти помилки.

Окремо слід згадати про безпеку. Завдяки суворій типізації та можливості визначати власні типи, TypeScript дозволяє контролювати, які саме дані потрапляють у систему, і запобігати багатьом поширеним вразливостям, пов’язаним із неправильним використанням типів або некоректною обробкою даних.

Ключові особливості TypeScript:

- статична типізація, що дозволяє виявляти помилки ще на етапі розробки;
- підтримка об’єктно-орієнтованого програмування для побудови масштабованих архітектур;
- повна сумісність із JavaScript та можливість поступової міграції існуючих проєктів;
- використання універсальних типів для створення функцій і класів;
- розвинена екосистема, велика кількість типових визначень для бібліотек і фреймворків;
- підтримка сучасних стандартів ECMAScript та трансляція у сумісний код;
- зручність командної роботи, автодоповнення, підсвічування синтаксису та інші інструменти для підвищення продуктивності;
- підвищення безпеки та надійності програмного забезпечення завдяки суворому контролю типів.

Отже, TypeScript – це мова, яка поєднує сучасні підходи до розробки, забезпечує високу якість, безпеку та масштабованість програмних рішень. Її

використання дозволяє створювати надійні, гнучкі та зручні у підтримці інформаційні системи, що відповідають актуальним вимогам індустрії.

2.2 Аналіз середовища розробки Visual Studio Code та Android Studio

Visual Studio Code [13] — це сучасне, кросплатформне середовище розробки, яке поєднує у собі зручність текстового редактора з потужністю повноцінної IDE. Однією з ключових переваг Visual Studio Code є його легкість, швидкість запуску та гнучкість налаштувань. Завдяки розширенням розробник може адаптувати середовище під власні потреби: додавати підтримку мов програмування, інтегрувати інструменти для тестування, роботи з системами контролю версій та хмарними сервісами.

Важливою особливістю Visual Studio Code є інтелектуальні підказки (IntelliSense), які допомагають уникати синтаксичних і логічних помилок ще на етапі написання коду. Інтегрований термінал дозволяє виконувати команди безпосередньо у вікні редактора, а підтримка Git забезпечує зручний контроль версій та командну роботу над проєктом. Завдяки цьому розробник отримує єдине середовище для написання, тестування, відлагодження та розгортання програмного продукту.

Capacitor [14] – це сучасна бібліотека у Visual Studio Code, яка інтегрується у веб-проєкт і дозволяє запускати його як нативний мобільний застосунок для Android [15] та iOS [16]. Завдяки Capacitor розробник може використовувати такі веб-технології як: JavaScript/TypeScript та отримувати доступ до нативних можливостей пристрою, швидко переносити веб-додаток на мобільні платформи без втрати функціональності.

Важливою перевагою Capacitor є простота інтеграції з такими середовищами, як Android Studio. Після генерації мобільного проєкту розробник може відкривати його у Android Studio для подальшого налаштування, тестування та публікації.

Android Studio [17] – це офіційне середовище розробки для Android, яке забезпечує повний цикл створення мобільних застосунків: від написання коду до збірки, тестування та розгортання на пристроях. Завдяки вбудованому емулятору, візуальному редактору інтерфейсів та потужним інструментам для профілювання продуктивності, Android Studio дозволяє створювати якісні, оптимізовані та сучасні мобільні застосунки.

2.3 Аналіз фреймворку React та системи збірки Vite

React [18] – це одна з найпопулярніших бібліотек для розробки інтерфейсів користувача, яка вже багато років залишається лідером у сфері фронтенд-розробки. Її головна ідея полягає у створенні додатків на основі незалежних компонентів, кожен з яких відповідає за власну частину інтерфейсу та логіки. Такий підхід дозволяє легко масштабувати проєкт, повторно використовувати код, а також розділяти роботу між різними членами команди. Компонентна архітектура сприяє підвищенню читабельності та підтримки коду, що особливо важливо для великих і динамічних систем.

Однією з ключових технологічних переваг React є використання віртуального DOM. Замість того, щоб безпосередньо змінювати структуру сторінки у браузері, React створює віртуальну копію DOM, аналізує зміни у стані компонентів і оновлює лише ті елементи, які дійсно змінилися. Це дозволяє значно підвищити продуктивність додатків, зменшити навантаження на браузер і забезпечити плавну роботу навіть при великій кількості інтерактивних елементів.

React підтримує концепцію одностороннього потоку даних, що робить логіку додатку більш передбачуваною та спрощує відстеження змін. Завдяки цьому розробники можуть легко контролювати, як дані передаються між компонентами, і уникати неочікуваних побічних ефектів. Для управління станом додатку існує велика кількість додаткових бібліотек, таких як Redux, MobX,

Zustand, які дозволяють організувати складні сценарії взаємодії між різними частинами системи.

Ще однією важливою перевагою React є його універсальність. Бібліотека може використовуватися не лише для створення класичних веб-додатків, а й для розробки мобільних застосунків, десктопних програм, а також для серверного рендерингу. Це відкриває широкі можливості для створення багатоплатформених рішень на основі єдиної технології.

Для ефективної роботи з React важливо використовувати сучасні інструменти збірки, які забезпечують швидке оновлення коду, оптимізацію ресурсів та зручність розробки. Однією з найінноваційніших систем збірки є Vite. На відміну від традиційних інструментів для збирання коду, Vite використовує нативні можливості браузера для роботи з ES-модулями, що дозволяє майже миттєво запускати проєкт і отримувати оновлення під час розробки без затримок. Це особливо цінно для великих проєктів, де час компіляції може суттєво впливати на продуктивність команди.

Ключові особливості React:

- компонентна архітектура, що дозволяє розбивати інтерфейс на незалежні, повторно використовувані частини;
- використання віртуального DOM для підвищення продуктивності та ефективного оновлення сторінки;
- односторонній потік даних, який забезпечує передбачуваність логіки та спрощує відстеження змін у стані додатку;
- можливість створення динамічних, інтерактивних інтерфейсів із мінімальними витратами ресурсів;
- велика екосистема додаткових бібліотек для управління станом, маршрутизації, тестування та інтеграції з іншими технологіями;
- підтримка серверного рендерингу, мобільної та десктопної розробки на основі єдиної технології;
- активна спільнота та постійний розвиток, що гарантує актуальність та підтримку фреймворку.

Vite [19] – це сучасний інструмент для збірки та розробки веб-додатків, який за короткий час здобув популярність серед розробників завдяки своїй швидкодії, простоті налаштування та підтримці новітніх стандартів JavaScript. Однією з головних переваг Vite є його принцип роботи: під час розробки він використовує нативні ES-модулі браузера, що дозволяє уникнути тривалого процесу попередньої збірки та забезпечує майже миттєве оновлення сторінки при зміні коду. Це значно підвищує продуктивність розробника, оскільки всі зміни відображаються у браузері практично без затримок.

Vite має інтуїтивно зрозумілу структуру конфігурації, що дозволяє легко підключати додаткові плагіни, налаштовувати підтримку різних препроцесорів стилів, інтегрувати TypeScript, TailwindCSS та інші сучасні технології. Завдяки цьому розробники можуть швидко адаптувати інструмент під потреби конкретного проєкту, не витрачаючи час на складні налаштування.

Ще однією важливою особливістю Vite є його оптимізація для продакшн-збірки. На етапі фінального збирання Vite використовує потужний інструмент Rollup, який забезпечує мінімізацію розміру файлів, розділення коду на частини (code splitting), оптимізацію завантаження ресурсів та підтримку сучасних стандартів JavaScript. Це дозволяє створювати швидкі, легкі та ефективні веб-додатки, які швидко завантажуються навіть при повільному інтернет-з’єднанні.

Vite також підтримує гаряче оновлення модулів, що дає змогу розробникам бачити зміни у додатку в реальному часі без повного перезавантаження сторінки. Це особливо корисно при роботі з великими проектами, де час компіляції може суттєво впливати на швидкість розробки.

Завдяки відкритій архітектурі та активній спільноті, Vite постійно розвивається, отримує нові можливості та інтеграції з популярними фреймворками. Його використання у зв’язці з React дозволяє створювати сучасні, продуктивні та масштабовані веб-додатки, які відповідають найвищим стандартам якості у сфері розробки.

Ключові особливості Vite:

- надзвичайно швидкий старт проєкту та миттєве оновлення сторінки під час розробки завдяки використанню нативних ES-модулів;
- гаряче оновлення модулів (hot module replacement), що дозволяє бачити зміни у додатку без повного перезавантаження сторінки;
- простота налаштування та розширення за допомогою плагінів і зрозумілої конфігурації;
- оптимізована продакшн-збірка з мінімізацією розміру файлів, розділенням коду та швидким завантаженням додатку;
- підтримка сучасних технологій, таких як TypeScript, TailwindCSS, а також інтеграція з популярними фреймворками;
- відкрита архітектура та активна спільнота, що забезпечує постійний розвиток і впровадження нових можливостей;
- зручність роботи з великими проектами завдяки високій продуктивності та гнучкості інструменту.

2.4 Аналіз бібліотеки стилізації TailwindCSS

TailwindCSS [20] – це сучасна утилітарна CSS-бібліотека, яка суттєво змінила підхід до стилізації веб-додатків. На відміну від класичних CSS-фреймворків, де використовуються готові компоненти та шаблони, TailwindCSS надає розробнику набір низькорівневих утилітарних класів, які можна комбінувати для створення унікального дизайну без написання власних CSS-правил.

Однією з ключових особливостей TailwindCSS є принцип "utility-first". Це означає, що для кожної властивості стилю (колір, відступ, розмір шрифту, тінь, рамка тощо) існує окремий клас, який можна застосовувати безпосередньо у розмітці. Такий підхід дозволяє швидко створювати адаптивні, сучасні інтерфейси, не перемикаючись між HTML і CSS-файлами. В результаті розробник отримує повний контроль над виглядом елементів і може легко змінювати дизайн без ризику виникнення конфліктів у стилях.

Ще однією важливою перевагою TailwindCSS є його гнучкість та масштабованість. Бібліотека дозволяє легко налаштовувати тему проєкту, змінювати кольорову палітру, шрифти, розміри, а також додавати власні утилітарні класи. Завдяки цьому можна створювати унікальні інтерфейси, які відповідають вимогам бренду чи специфіці конкретного проєкту. TailwindCSS також підтримує темну тему, що є актуальним трендом у сучасному веб-дизайні.

TailwindCSS ідеально підходить для розробки адаптивних інтерфейсів. Бібліотека містить вбудовану систему брейкпоінтів, що дозволяє легко створювати мобільні, планшетні та десктопні версії сторінок. Для кожного розміру екрана можна застосовувати окремі стилі, що забезпечує коректне відображення додатку на будь-якому пристрої.

Важливою перевагою є й оптимізація продуктивності. TailwindCSS використовує механізм "purge", який автоматично видаляє невикористані стилі під час фінальної збірки проєкту. Завдяки цьому розмір CSS-файлів залишається мінімальним, що позитивно впливає на швидкість завантаження сторінок і загальну продуктивність додатку.

Ще одним плюсом є активна спільнота та велика кількість додаткових плагінів, які розширюють можливості TailwindCSS. Серед них — готові компоненти, анімації, інтеграція з формами, таблицями, графікою тощо. Це дозволяє швидко впроваджувати нові функції та підтримувати сучасний вигляд інтерфейсу без зайвих зусиль.

TailwindCSS також добре інтегрується з сучасними фреймворками та інструментами розробки, такими як React, Vite, тощо. Це забезпечує зручність використання бібліотеки у будь-якому сучасному стеку технологій.

Ключові особливості TailwindCSS:

- утилітарний підхід до стилізації, що дозволяє створювати унікальні інтерфейси без написання власних CSS-файлів;
- гнучкість налаштувань та можливість кастомізації теми проєкту;
- вбудована підтримка адаптивного дизайну та брейкпоінтів;

- оптимізація розміру фінального CSS-файлу завдяки автоматичному видаленню невикористаних класів;
- активна спільнота, велика кількість плагінів і готових рішень;
- легка інтеграція з популярними фреймворками та інструментами розробки.

TailwindCSS є потужним інструментом для створення сучасних, адаптивних і продуктивних веб-інтерфейсів, який дозволяє розробникам швидко впроваджувати нові ідеї, підтримувати чистоту коду та забезпечувати високу якість кінцевого продукту.

2.5 Технології інтеграції AI та геолокаційних сервісів

Інтеграція штучного інтелекту (AI) у сучасні інформаційні системи є одним із найактуальніших напрямів розвитку цифрових сервісів. AI-технології дозволяють автоматизувати аналіз великих обсягів даних, забезпечувати персоналізований підхід до користувача, підвищувати ефективність бізнес-процесів і створювати інноваційні функції, які раніше були недоступними.

Сучасна інтеграція AI у веб- та мобільні додатки зазвичай здійснюється через використання зовнішніх хмарних сервісів або API, які надають доступ до потужних моделей машинного навчання. Найпоширенішими є сервіси Google Gemini [21], OpenAI, Microsoft Azure AI, Hugging Face та інші. Взаємодія з такими сервісами відбувається через стандартні протоколи REST API, що дозволяє легко підключати AI-функціонал до будь-якої сучасної системи незалежно від її архітектури чи мови програмування.

Технічно процес інтеграції AI виглядає так: клієнтська або серверна частина додатку формує запит із необхідними даними (наприклад, текст, зображення, історія дій користувача) і надсилає його до AI-сервісу. На стороні хмари відбувається обробка даних за допомогою складних моделей машинного навчання, після чого у відповідь повертається результат – це можуть бути рекомендації, аналітичні висновки, класифікація, прогнозування чи навіть

згенерований текст або зображення. Такий підхід дозволяє не перевантажувати локальні ресурси пристрою користувача, а використовувати потужності дата-центрів для виконання складних обчислень.

Важливою перевагою сучасних AI-сервісів є їхня масштабованість, гнучкість і можливість швидкого оновлення моделей без необхідності змін у основному коді додатку. Це дає змогу поступово розширювати функціонал, впроваджувати нові сценарії використання AI, адаптувати систему до змін у поведінці користувачів чи ринку.

AI-технології можуть застосовуватися для різних завдань: персоналізовані рекомендації, аналіз поведінки користувачів, прогнозування попиту, автоматизація спілкування через чат-боти, розпізнавання зображень, голосу, тексту, а також для аналітики великих даних. Усе це сприяє підвищенню якості сервісу, залученості користувачів і конкурентоспроможності цифрових продуктів.

Варто також враховувати питання безпеки та етики при впровадженні штучного інтелекту. Необхідно забезпечити захист персональних даних, прозорість алгоритмів і можливість контролю з боку користувача. Дотримання сучасних стандартів конфіденційності та етичного використання AI є обов’язковою умовою для впровадження таких технологій у будь-якій сфері.

Ключові особливості інтеграції AI:

- можливість персоналізованих рекомендацій для користувачів на основі аналізу їхніх дій, вподобань та історії взаємодії з системою;
- використання зовнішніх AI-сервісів через REST API, що дозволяє виконувати складні обчислення на стороні хмарних платформ і не навантажувати клієнтський додаток;
- гнучкість у виборі та зміні AI-моделей без необхідності суттєвих змін у архітектурі основної системи;
- масштабованість рішень: можливість обробки великої кількості запитів та адаптації під зростаючі потреби користувачів;

- висока швидкість отримання результатів завдяки оптимізованим алгоритмам та потужностям хмарних сервісів;
- безпека та конфіденційність: передача лише необхідних даних для обробки, дотримання сучасних стандартів захисту інформації;
- можливість поступового розширення функціоналу AI – від простих підказок до складних аналітичних і прогностичних модулів;
- підвищення якості сервісу та залученості користувачів завдяки інтелектуальній взаємодії та адаптації системи до індивідуальних потреб.

Штучний інтелект в даному дослідженні реалізовано через окремий модуль, який надає користувачам персоналізовані рекомендації щодо вибору спортивного інвентарю та вправ. Система надсилає дані вибраного спортивного інвентарю до зовнішнього AI-сервісу через API, а у відповідь отримує індивідуальні поради, які відображаються у зручному вигляді. Такий підхід дозволяє зробити сервіс більш адаптивним і сучасним.

Далі розглянемо технології інтеграції геолокаційних сервісів. Інтеграція геолокаційних сервісів у сучасні інформаційні системи стала важливим інструментом для підвищення зручності, функціональності та персоналізації користувацького досвіду. Геолокація дозволяє визначати місцезнаходження користувача або об’єктів у просторі, що особливо актуально для систем, пов’язаних із фізичними локаціями, такими як спортзали, фітнес-центри чи спортивне обладнання.

Однією з основних переваг використання геолокаційних сервісів є можливість візуалізації розташування об’єктів на інтерактивній карті. Це значно спрощує навігацію для користувачів, дозволяє швидко знаходити потрібне обладнання, орієнтуватися у великому приміщенні чи навіть планувати маршрут до найближчого вільного тренажера. Для нових користувачів або гостей закладу така функція є особливо корисною, оскільки зменшує час на пошук і підвищує загальний рівень задоволеності сервісом.

Технічно інтеграція геолокаційних сервісів у веб-додатки зазвичай реалізується за допомогою спеціалізованих API, таких як OpenStreetMap [22] та

Leaflet [23]. Ці сервіси надають інструменти для відображення інтерактивних карт, додавання маркерів, зон, підказок, а також для роботи з координатами у реальному часі. Веб-додаток може отримувати координати користувача через стандартний інтерфейс браузера, а потім відображати його місцезнаходження на карті або пропонувати найближчі об’єкти.

Важливою особливістю сучасних геолокаційних сервісів є можливість кастомізації вигляду карти, додавання власних шарів, інтеграція з іншими даними (наприклад, станом обладнання, розкладом занять, бронюваннями). Це дозволяє створювати унікальні рішення, які повністю відповідають потребам конкретного закладу чи користувача.

Геолокаційні сервіси також можуть використовуватися для аналітики – наприклад, для відстеження популярності певних зон у спортзалі, аналізу маршрутів переміщення відвідувачів, оптимізації розміщення обладнання. Зібрані дані допомагають адміністрації приймати обґрунтовані рішення щодо організації простору, планування закупівель чи проведення ремонтів.

Ще одним важливим аспектом є безпека та конфіденційність. При впровадженні геолокаційних функцій необхідно дотримуватися сучасних стандартів захисту персональних даних, інформувати користувачів про збір та використання їхньої локації, а також надавати можливість відмовитися від цієї функції.

Ключові особливості інтеграції геолокаційних сервісів:

- можливість визначення та відображення місцезнаходження користувача або об’єктів у реальному часі;
- візуалізація розташування обладнання, зон чи інших важливих об’єктів на інтерактивній карті;
- підтримка інтеграції з популярними картографічними сервісами (OpenStreetMap, Leaflet);
- гнучкість налаштування вигляду карти, додавання власних маркерів, шарів та інформаційних підказок;

– використання стандартних API браузера для отримання координат користувача;

– можливість аналітики переміщень, популярності зон, оптимізації розміщення обладнання;

– підвищення зручності навігації для користувачів, особливо у великих або незнайомих просторах;

– дотримання стандартів безпеки та конфіденційності при роботі з персональними даними про локацію;

– легка інтеграція з іншими модулями інформаційної системи для розширення функціоналу.

Інтеграція геолокаційних сервісів у інформаційні системи дозволяє значно підвищити зручність користування, зробити сервіс більш інтуїтивним і персоналізованим, а також надати адміністрації нові інструменти для аналітики та оптимізації роботи закладу.

2.6 Вибір оптимальної IT-інфраструктури

Архітектура інформаційної системи побудована за багаторівневою моделлю, що забезпечує розділення відповідальностей і спрощує підтримку та розвиток додатку (рис. 2.1).

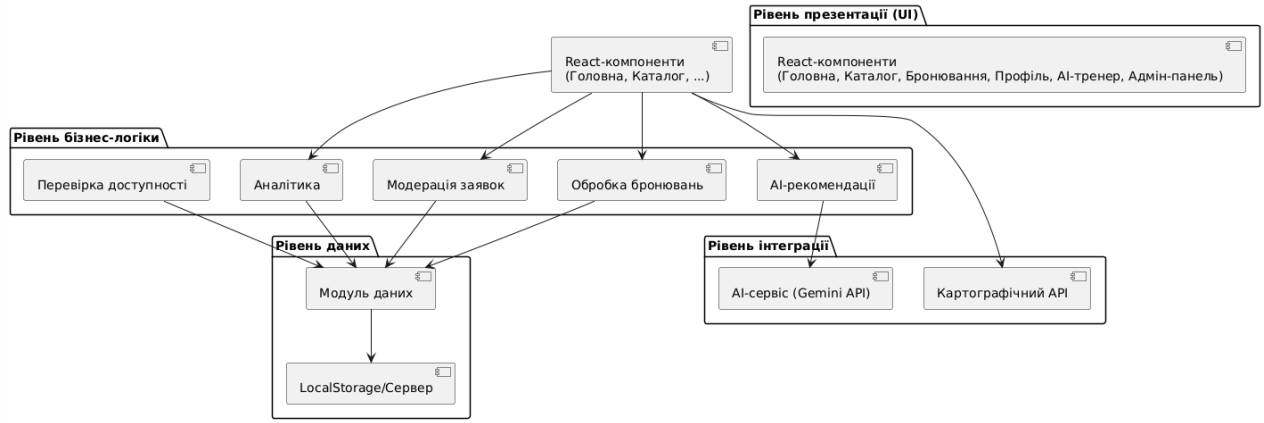

Рисунок 2.1 – Архітектура інформаційної системи

Основні рівні архітектури:

- рівень презентації (Presentation Layer): Відповідає за відображення інтерфейсу користувача, обробку взаємодії та передачу запитів до бізнес-логіки. Реалізований за допомогою React-компонентів, що забезпечують адаптивність, багатомовність і сучасний дизайн;
- рівень бізнес-логіки (Business Logic Layer): Містить основні алгоритми роботи додатку: обробку бронювань, перевірку доступності інвентарю, управління профілями, генерацію AI-рекомендацій, аналітику та модерацію заявок. Всі дії проходять через цей рівень, що гарантує цілісність і безпеку даних;
- рівень даних (Data Layer): Відповідає за зберігання та обробку інформації про користувачів, інвентар, бронювання, історію дій. Дані зберігаються у локальному сховищі (localStorage) або можуть бути інтегровані з віддаленим сервером для розширення можливостей;
- рівень інтеграції (Integration Layer): Забезпечує взаємодію з зовнішніми сервісами AI-модулем для генерації рекомендацій, картографічними API для відображення локацій, а також іншими сторонніми сервісами для розширення функціоналу.

На рисунку 2.2 представлено блок-схему зв’язків програмного забезпечення інформаційної системи для бронювання спортивного інвентарю (рис. 2.2).

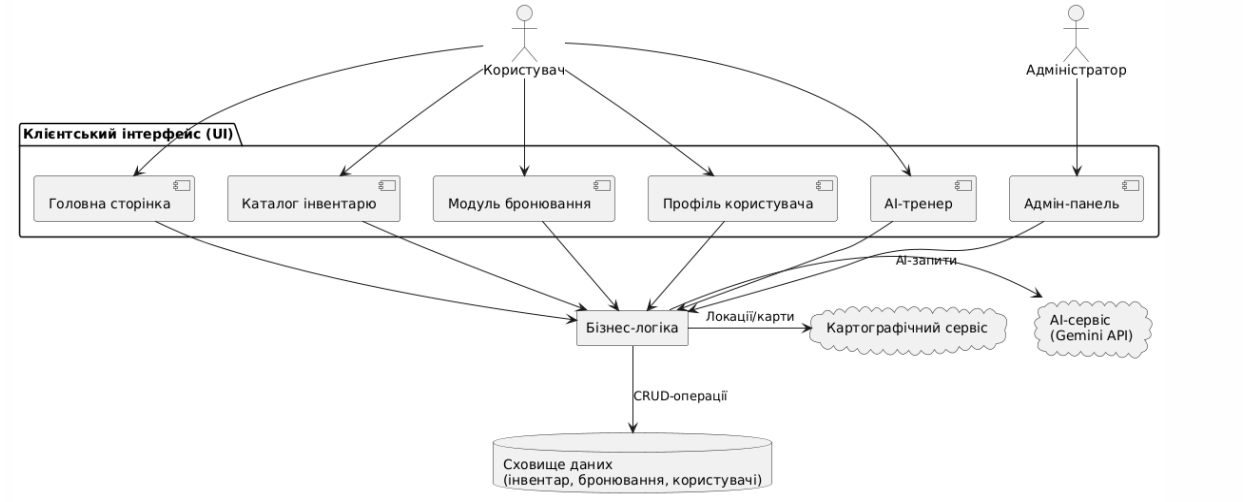

Рисунок 2.2 – Блок-схема зв’язків програмного забезпечення

Система складається з декількох основних модулів, які взаємодіють між собою для забезпечення повного циклу роботи додатку, зокрема:

- користувач взаємодіє з інтерфейсом додатку через мобільний або веб-клієнт;
- головна сторінка надає доступ до каталогу інвентарю, пошуку, фільтрації та переходу до інших розділів;
- каталог інвентарю дозволяє переглядати, фільтрувати та обирати спортивне обладнання;
- модуль бронювання відповідає за створення, перевірку та збереження бронювань, а також за відображення історії бронювань користувача;
- AI-тренер інтегрується із зовнішнім AI-сервісом (наприклад, Google Gemini API) для генерації персоналізованих рекомендацій щодо тренувань;
- профіль користувача містить особисті дані, налаштування і статистику;
- адміністративна панель надає адміністратору можливість керувати інвентарем, модерацією заявок, переглядом аналітики та зміною розкладу доступності;
- модуль даних забезпечує зберігання інформації про користувачів, інвентар, бронювання та історію дій у локальному сховищі або на сервері;

– зовнішні сервіси (AI, карти) використовуються для розширення функціоналу системи.

Архітектура інформаційної системи для бронювання та використання спортивного інвентарю побудована на сучасних принципах модульності, масштабованості та гнучкості. Мобільний додаток реалізовано з урахуванням різних ролей користувачів, що дозволяє ефективно розподіляти функціонал між клієнтами та адміністраторами спортзалу. Ключові компоненти та функції системи:

1. реєстрація та авторизація користувачів: система дозволяє новим користувачам створювати обліковий запис, а зареєстрованим входити до особистого кабінету. Реалізовано перевірку введених даних. Всі дії користувачів захищені механізмами авторизації, що обмежують доступ до приватних сторінок і функцій;

2. каталог спортивного інвентарю: мобільний додаток містить структурований каталог обладнання з детальними описами, фотографіями, категоріями та статусами доступності. Користувачі можуть фільтрувати інвентар за категоріями, станом, популярністю, а також шукати потрібне обладнання за ключовими словами. Для кожної одиниці інвентарю доступна сторінка з детальною інформацією та розкладом зайнятості;

3. система бронювання: користувачі можуть бронювати обладнання на визначений час, обираючи зручний часовий слот. Система автоматично перевіряє доступність інвентарю, запобігає конфліктам у розкладі. Реалізовано історію бронювань, можливість скасування, а також отримання сповіщень про статус бронювання;

4. особистий кабінет користувача: у профілі користувача відображається історія бронювань, налаштування профілю, можливість переглянути особисті дані, змінити мову інтерфейсу;

5. адміністрування: адміністратор має розширені права: може додавати, редагувати, видаляти обладнання, підтверджувати або відхиляти заявки на бронювання, переглядати статистику використання інвентарю, керувати

розкладом доступності, а також отримувати аналітичні звіти щодо завантаженості обладнання;

6. інтеграція з AI: у системі реалізовано модуль штучного інтелекту, який аналізує вибраний спортивний інвентар і надає персоналізовані рекомендації щодо вибору обладнання чи вправ. Взаємодія з AI здійснюється через зовнішній сервіс за допомогою REST API, що дозволяє масштабувати функціонал без навантаження на клієнтську частину;

7. інтерактивна карта: мобільний додаток містить інтерактивну карту, яка відображає розташування спортивних закладів. Це спрощує навігацію для користувачів, дозволяє швидко знаходити і планувати маршрут у приміщенні;

8. багатомовний інтерфейс: система підтримує кілька мов, що дозволяє користувачам перемикати мову інтерфейсу для зручності та доступності.

2.7 Висновки

У другому розділі було проведено аналіз інформаційних технологій, які є найбільш доцільними для створення ефективної, масштабованої та зручної у використанні інформаційної системи. Розглянуто можливості мови програмування TypeScript, яка забезпечує високу надійність, безпеку та структурованість коду завдяки статичній типізації та підтримці об’єктно-орієнтованого підходу. Проаналізовано переваги фреймворку React, що дозволяє будувати компонентні, динамічні та легко підтримувані інтерфейси, а також системи збірки Vite, яка значно прискорює розробку та оптимізує продуктивність додатку. Обґрунтовано вибір середовища розробки Visual Studio Code, яке завдяки своїй легкості, швидкості запуску та гнучкості налаштувань підвищило ефективність розробки. Важливою складовою технологічного стеку стала бібліотека Capacitor, що дозволила трансформувати веб-додаток у повноцінний мобільний застосунок для платформ Android та iOS без переписування коду. Для подальшого налаштування, тестування та публікації мобільного додатку використовувалося середовище Android Studio, яке надає вбудований емулятор,

візуальний редактор інтерфейсів та інструменти для профілювання продуктивності. Здійснено аналіз бібліотеки стилізації TailwindCSS, яка дає змогу швидко створювати сучасні, адаптивні та унікальні інтерфейси без надлишкового коду. Розглянуто технології інтеграції штучного інтелекту, що відкривають нові можливості для персоналізації сервісу, а також геолокаційних сервісів, які підвищують зручність навігації та аналітики у цифрових продуктах. Здійснено вибір IT-інфраструктури. Також розроблено архітектуру інформаційної системи, яка побудована за принципами модульності та розділення обов'язків. Кожен модуль від аутентифікації до AI-тренера має чітко визначену роль і взаємодіє з іншими компонентами через стандартизовані інтерфейси.

3 РОЗРОБЛЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ ДЛЯ БРОНЮВАННЯ ТА ВИКОРИСТАННЯ СПОРТИВНОГО ІНВЕНТАРЮ

3.1 Розроблення структури інформаційної системи

Для документування та аналізу структури системи використовуються UML-діаграми, зокрема діаграма випадків використання (Use Case Diagram) (рис. 3.1), яка ілюструє основні сценарії взаємодії користувачів із системою, розмежування функціоналу між клієнтами та адміністраторами, а також ключові бізнес-процеси.

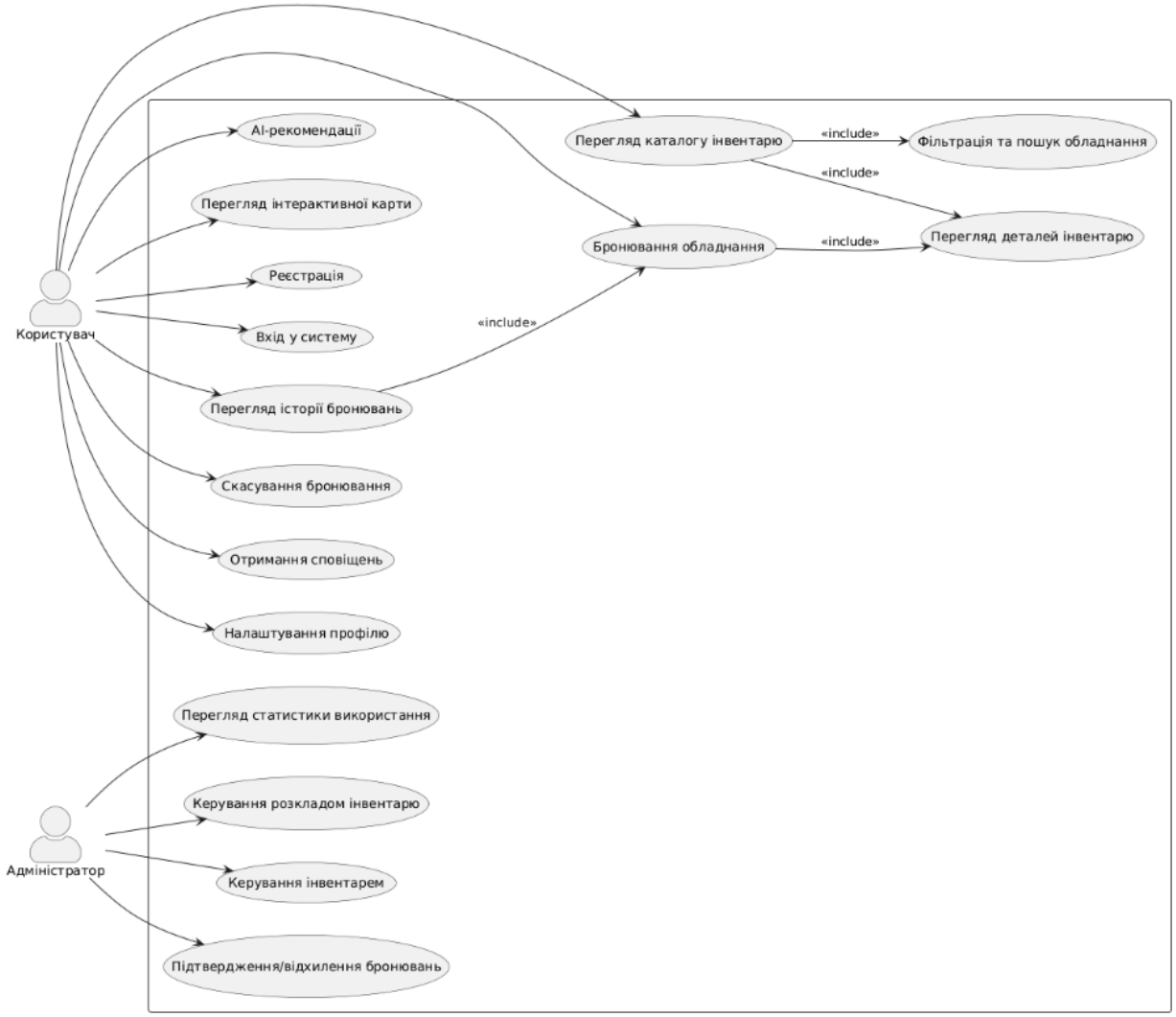

Рисунок 3.1 – Діаграма випадків використання (Use Case Diagram)

На діаграмі послідовності (рис. 3.2) відображено порядок взаємодії між користувачем і системою під час процесу бронювання спортивного інвентарю.

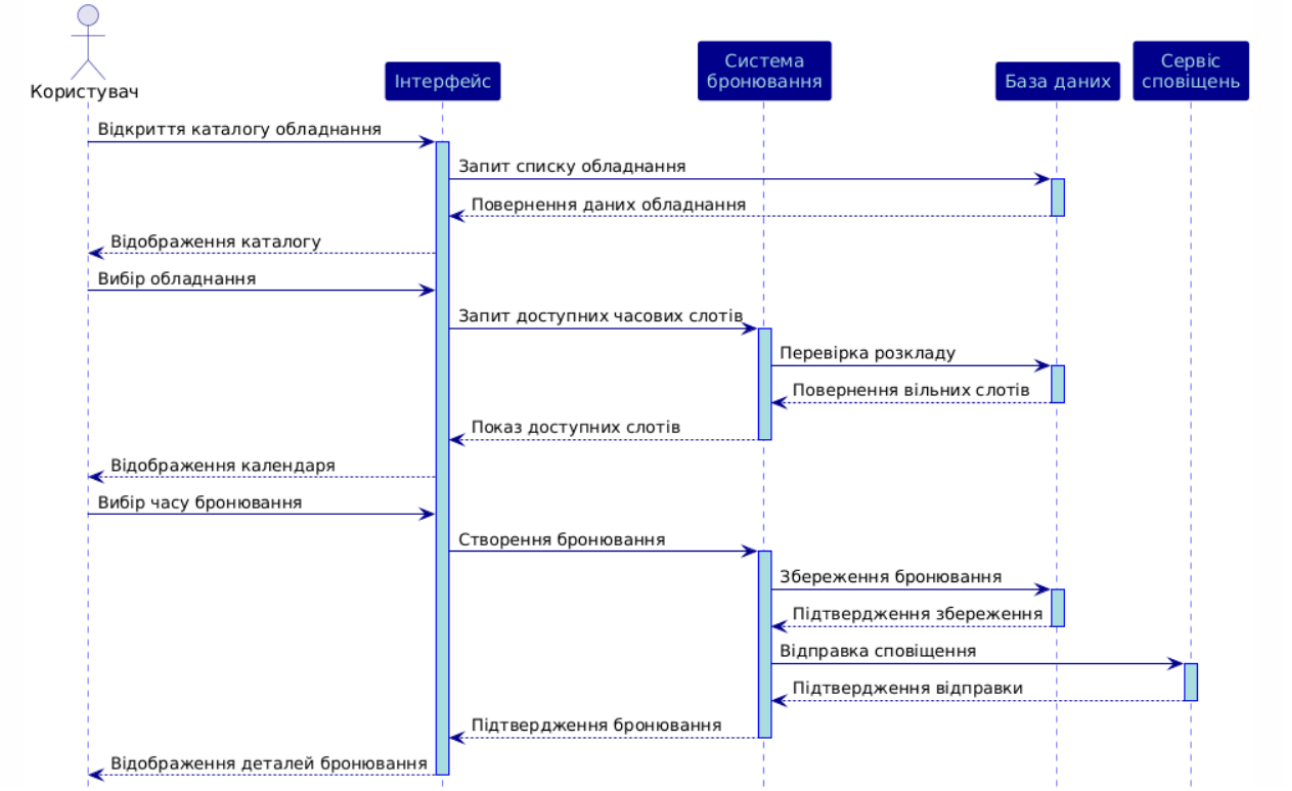

Рисунок 3.2 – Діаграма послідовності бронювання спортивного інвентарю

Порядок взаємодії між користувачем і системою під час процесу бронювання спортивного інвентарю наступний:

1. запит користувача: користувач відкриває каталог обладнання через інтерфейс додатку;
2. відображення каталогу: система показує список доступного спортивного інвентарю з описом та статусом;
3. вибір обладнання: користувач обирає потрібний тренажер чи інвентар для бронювання;
4. запит часових слотів: система відображає календар з доступними часовими слотами для обраного обладнання;
5. вибір часу: користувач обирає бажаний час для бронювання інвентарю;

6. перевірка доступності: система перевіряє актуальність обраного часового слоту;

7. створення бронювання: система формує запис про нове бронювання в базі даних;

8. оновлення статусу: база даних оновлює статус доступності обладнання на обраний період.

9. підтвердження бронювання: система генерує підтвердження бронювання.

10. відправка сповіщення: система надсилає push-повідомлення про успішне бронювання.

11. збереження в історії: система зберігає інформацію про бронювання в історії користувача.

12. відображення деталей: користувач отримує повну інформацію про своє бронювання.

13. відображення статусу: інтерфейс відображає статус "Заброньовано" для обраного обладнання.

Діаграма послідовності демонструє логічний процес бронювання спортивного інвентарю, забезпечуючи надійну взаємодію між користувачем та системою, а також гарантуючи коректне оновлення статусів обладнання.

На діаграмі класів (рис. 3.3) відображено основні компоненти системи – користувачі (User), адміністратори (Admin), спортивний інвентар (Equipment), бронювання (Booking) та сповіщення (Notification), а також сервіси AITrainer (персоналізовані рекомендації та тренування на основі штучного інтелекту) і MapService (інтеграція з картою для пошуку та навігації до спортзалів).

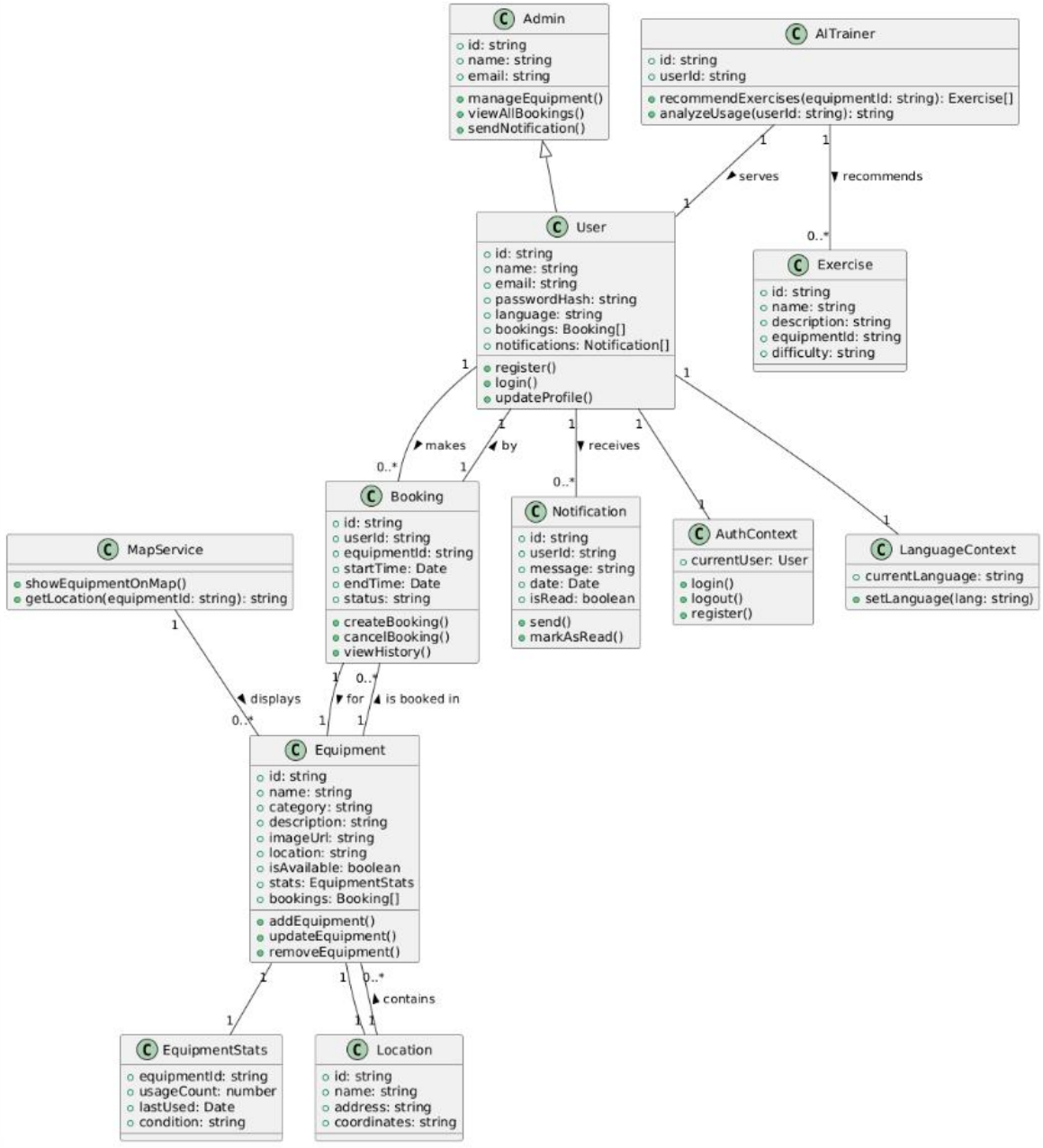

Рисунок 3.3 – Діаграма класів

Ключові компоненти системи:

– User – представляє звичайного користувача, містить дані профілю, історію бронювань, може здійснювати бронювання, переглядати інвентар, отримувати сповіщення;

– Admin – розширює User, має додаткові права: керування інвентарем, підтвердження/скасування бронювань, перегляд аналітики;

– Equipment – описує одиницю спортивного інвентарю: назва, категорія, стан, доступність, розташування;

– Booking – містить інформацію про бронювання: користувач, інвентар, час початку/завершення, статус;

– Notification – відповідає за сповіщення користувачів про події (бронювання, нагадування, зміни статусу);

– AITrainer – сервіс, що надає рекомендації користувачам на основі їхніх цілей та вибору;

– MapService – сервіс для відображення розташування спортзалів на карті.

Зв'язки між компонентами системи формують структуровану архітектуру даних. User має множинні зв'язки з Booking та Notification, забезпечуючи персоналізацію сервісу. Admin функціонує як розширена версія User з додатковими привілеями. Equipment може асоціюватися з багатьма об'єктами Booking, причому сама сутність Booking виступає зв'язуючою ланкою між User та Equipment, документуючи деталі бронювання. Notification завжди належить конкретному User. AITrainer та MapService взаємодіють з користувачами через API, надаючи рекомендації та геодані, але працюють без прямого збереження в базі даних системи.

Окремі результати розробки та підходи до побудови системи було представлено у вигляді тез: «Інформаційна система для бронювання та використання спортивного інвентарю в спортзалах» на Міжнародній науково-практичній інтернет-конференції «Молодь в науці: дослідження, проблеми, перспективи» (Вінниця, 2024-2025 рр.) [3].

3.2 Розроблення загального алгоритму роботи додатку та структурної схеми функціонування інформаційної системи

На рисунку 3.4 представлено блок-схему алгоритму роботи мобільного додатку (рис. 3.4).

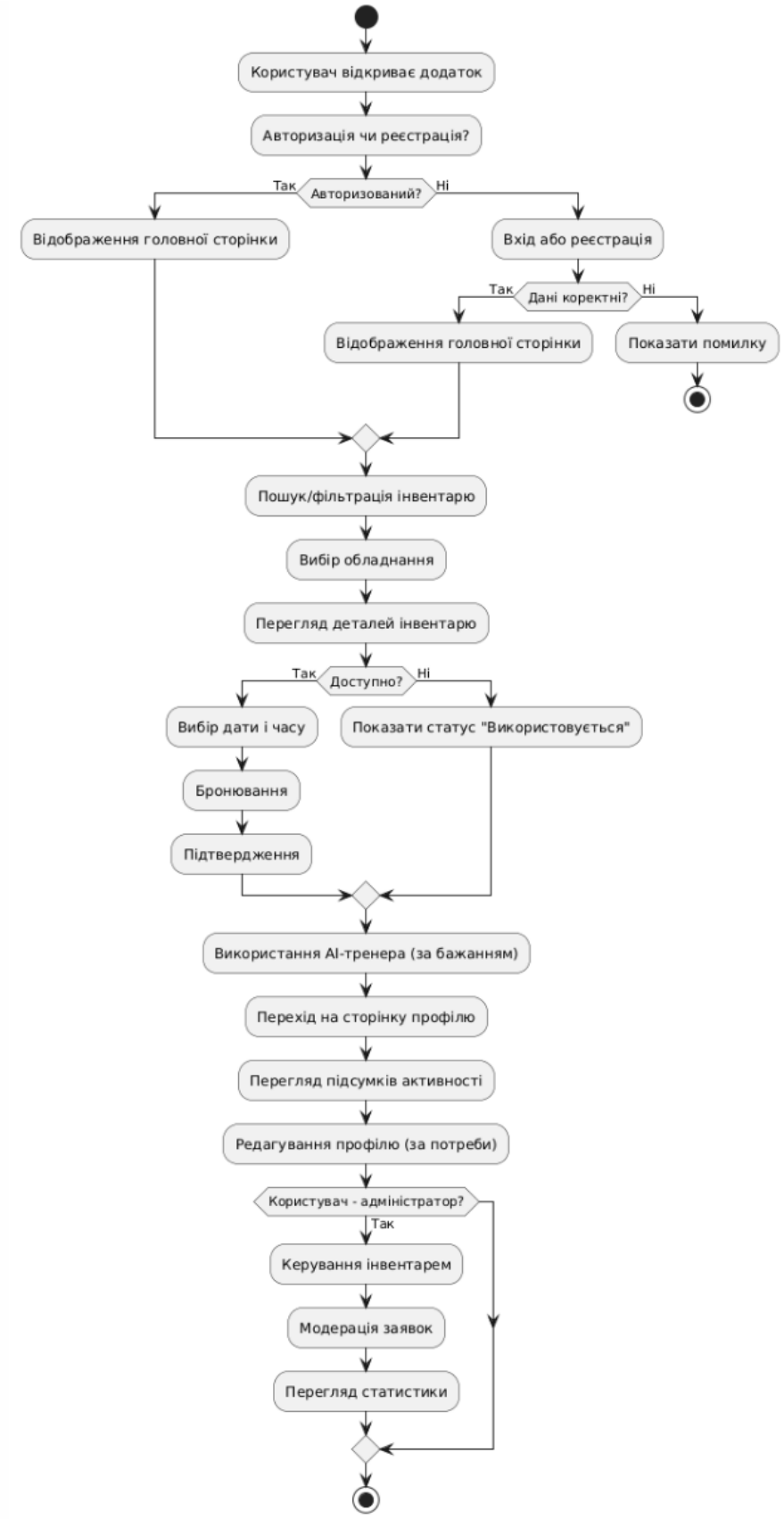

Рисунок 3.4 – Блок-схема алгоритму роботи додатку

Опис алгоритму роботи мобільного додатку наступний:

- користувач проходить авторизацію або реєстрацію для отримання доступу до системи;
- на головній сторінці здійснюється пошук і фільтрація спортивного інвентарю;
- після вибору обладнання користувач переглядає його деталі та, за наявності, оформлює бронювання із зазначенням дати й часу;
- система перевіряє доступність інвентарю та підтверджує бронювання або повідомляє про неможливість операції;
- за бажанням користувач може скористатися AI-тренером для отримання персоналізованих рекомендацій;
- у профілі відображається історія бронювань, статистика та налаштування;
- для адміністратора доступні функції керування інвентарем, модерації заявок і перегляду аналітики.

На рисунку 3. 5 представлено структурну схему функціонування додатку, яка ілюструє основні складові мобільного додатку та їх взаємозв’язки. Користувацький інтерфейс включає головну сторінку, модуль бронювання, AI-тренер, профіль користувача та адміністративну панель. Бізнес-логіка відповідає за авторизацію, управління бронюваннями, генерацію AI-рекомендацій і локалізацію інтерфейсу. Дані про інвентар та бронювання зберігаються у відповідних модулях даних. Додаток взаємодіє із зовнішніми сервісами, такими як Google Gemini API для отримання рекомендацій та картографічними сервісами для відображення локацій.

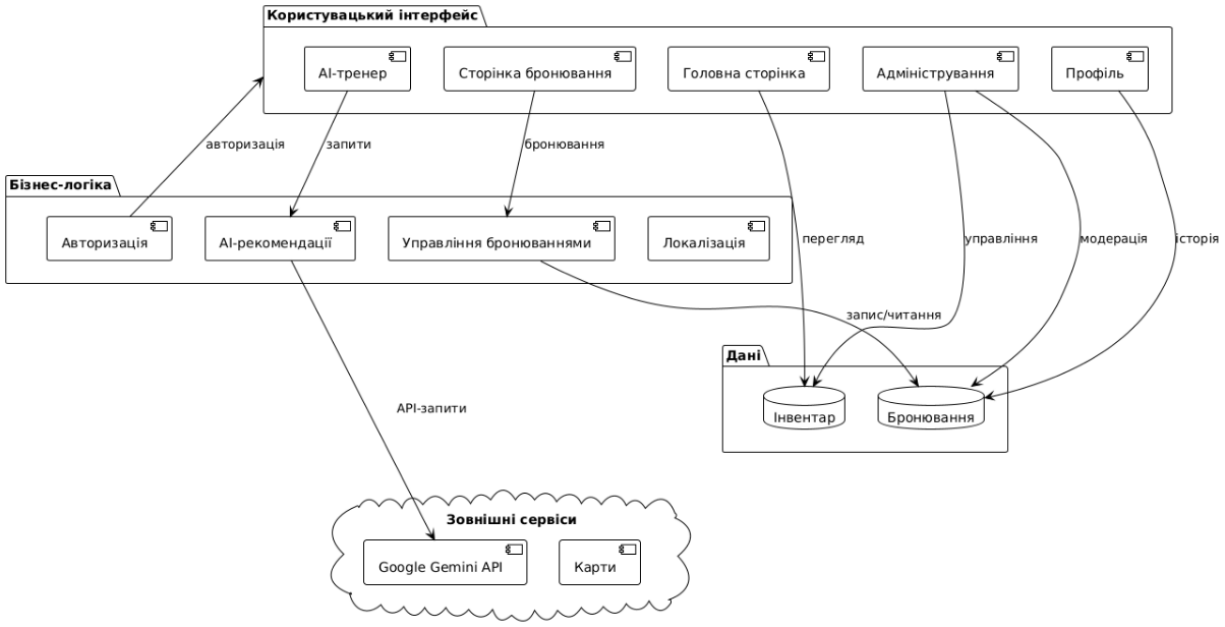

Рисунок 3.5 – Структурна схема функціонування додатку

3.3 Програмна реалізація мобільного додатку

Для розробки інформаційної системи бронювання та використання спортивного інвентарю було використано сучасне середовище розробки Visual Studio Code. Основний стек технологій включає TypeScript, React, Vite для швидкої збірки, TailwindCSS для стилізації інтерфейсу, а також набір сучасних бібліотек для UI-компонентів та інтеграції з AI-сервісами.

Сторінка реєстрації RegisterPage.tsx (рис. 3.6) представляє код сторінки реєстрації, яка дозволяє новим користувачам створити обліковий запис, ввівши основні дані.

```tsx
import { useNavigate } from "react-router-dom";
import { useForm } from "react-hook-form";
import { zodResolver } from "@hookform/resolvers/zod";
import * as z from "zod";
import { useAuth } from "@/context/AuthContext";
import { Button } from "@/components/ui/button";
import { useLanguage } from "@/context/LanguageContext";
import {
  Form,
  FormControl,
  FormField,
  FormItem,
  FormLabel,
  FormMessage,
} from "@/components/ui/form";
import { Input } from "@/components/ui/input";
import { Card, CardHeader, CardContent, CardFooter } from "@/components/ui/card";
import { toast } from "sonner";

const formSchema = z
  .object({
    name: z.string().min(2, "Name must be at least 2 characters"),
    email: z.string().email("Invalid email address"),
    password: z.string().min(6, "Password must be at least 6 characters"),
    confirmPassword: z.string(),
  })
  .refine((data) => data.password === data.confirmPassword, {
    message: "Passwords don't match",
    path: ["confirmPassword"],
  });

export default function RegisterPage() {
  const { register } = useAuth();
  const navigate = useNavigate();
  const { t } = useLanguage();
```

Рисунок 3.6 – Код сторінки RegisterPage.tsx

Основні функції та компоненти:

– форма реєстрації з полями для введення персональних даних;

– перевірка коректності введених даних;

– відображення повідомлень про помилки та успішну реєстрацію;

– кнопка для переходу на сторінку входу, якщо обліковий запис вже існує;

– інтеграція з контекстом автентифікації для створення нового користувача.

Сторінка ProfilePage.tsx (рис. 3.7) відображає персональні дані користувача, історію бронювань та налаштування профілю. Вона дозволяє переглядати особисту інформацію, в тому числі минулі та активні бронювання, а також змінювати налаштування облікового запису.

```tsx
import React, { useState, useEffect } from 'react';
import { useNavigate } from 'react-router-dom';
import { User, Settings, LogOut, Clock, Calendar, Timer, Globe } from 'lucide-react';
import Navbar from '@/components/Navbar';
import { userBookings, saveBookings } from '@/data/equipmentData';
import { useToast } from '@/hooks/use-toast';
import { useLanguage } from '@/context/LanguageContext';
import { useAuth } from '@/context/AuthContext';

const STATS_STORAGE_KEY = 'training_stats';

interface TrainingStats {
  completedCount: number;
  totalTrainingHours: number;
}

const getStoredStats = (): TrainingStats => {
  const stored = localStorage.getItem(STATS_STORAGE_KEY);
  return stored ? JSON.parse(stored) : { completedCount: 0, totalTrainingHours: 0 };
};

const saveStats = (stats: TrainingStats) => {
  localStorage.setItem(STATS_STORAGE_KEY, JSON.stringify(stats));
};

const ProfilePage = () => {
  const { toast } = useToast();
  const { t, locale, changeLocale } = useLanguage();
  const navigate = useNavigate();
  const { user, logout } = useAuth();
  const [upcomingCount, setUpcomingCount] = useState(0);
  const [stats, setStats] = useState<TrainingStats>(getStoredStats);

  const toggleLanguage = () => {
    changeLocale(locale === 'en' ? 'uk' : 'en');
    toast({
      title: t('languageChanged'),
      description: locale === 'en' ? 'Мову змінено на українську' : 'Language changed to English',
```

Рисунок 3.7 – Код сторінки ProfilePage.tsx

Основні функції та компоненти:

– відображає ім’я, email користувача, статистику тренувань (кількість завершених бронювань, загальний час тренувань);

– дозволяє змінити мову (UA/EN) та вийти з акаунта;

– статистика зберігається у localStorage;

– є секція налаштувань та кнопка виходу;

– використовує контексти автентифікації та мови, кастомний toast;

– оформлення – сучасний дизайн із TailwindCSS;

– внизу – навігаційна панель.

На рисунку 3.8 зображено код головної сторінки Index.tsx, яка містить назву додатку, пошук, фільтри, сортування, каталог інвентарю та інтеграцію з картою. Основна логіка сторінки полягає у фільтрації, пошуку та відображенні доступного спортивного обладнання.

```tsx
import React, { useState, useEffect } from 'react';
import { Search, SlidersHorizontal, Check, X, ArrowDownUp } from 'lucide-react';
import CategoryFilter from '@/components/CategoryFilter';
import EquipmentCard from '@/components/EquipmentCard';
import Navbar from '@/components/Navbar';
import MapSection from '@/components/MapSection';
import AdvancedFilters, { FilterOptions } from '@/components/AdvancedFilters';
import { equipmentData, EquipmentCategory, Equipment } from '@/data/equipmentData';
import { useLanguage } from '@/context/LanguageContext';
import { Button } from '@/components/ui/button';
import { Tabs, TabsContent, TabsList, TabsTrigger } from "@/components/ui/tabs";

const Index = () => {
  const [selectedCategory, setSelectedCategory] = useState<EquipmentCategory | null>(null);
  const [searchQuery, setSearchQuery] = useState('');
  const [filteredEquipment, setFilteredEquipment] = useState(equipmentData);
  const [showFilters, setShowFilters] = useState(false);
  const [activeFilters, setActiveFilters] = useState<FilterOptions>({
    location: null,
    minRating: null,
    availableOnly: false,
    category: null,
  });
  const [sortBy, setSortBy] = useState<'name' | 'rating' | 'popularity'>('name');
  const [sortOrder, setSortOrder] = useState<'asc' | 'desc'>('asc');
  const { t, locale } = useLanguage();

  useEffect(() => {
    let result = equipmentData;


    if (selectedCategory) {
      result = result.filter(item => item.category === selectedCategory);
    }

    if (searchQuery) {
      const query = searchQuery.toLowerCase();
      result = result.filter(item =>
```

Рисунок 3.8 – Код сторінки Index.tsx

Сторінка BookingPage.tsx (рис. 3.9) надає користувачам можливість обирати спортивний інвентар, встановлювати дату та час, а також підтверджувати бронювання.

```tsx
import React, { useState, useEffect } from 'react';
import { useNavigate } from 'react-router-dom';
import { Calendar, Check, Clock, X } from 'lucide-react';
import Navbar from '@/components/Navbar';
import { userBookings, Booking, saveBookings } from '@/data/equipmentData';
import { useToast } from '@/hooks/use-toast';
import { useLanguage } from '@/context/LanguageContext';

const BookingPage = () => {
  const navigate = useNavigate();
  const { toast } = useToast();
  const [bookings, setBookings] = useState<Booking[]>([]);
  const { t } = useLanguage();

  useEffect(() => {
    setBookings(userBookings);
  }, []);

  const updateBookings = (updatedBookings: Booking[]) => {
    setBookings(updatedBookings);
    saveBookings(updatedBookings);
    userBookings.length = 0;
    userBookings.push(...updatedBookings);
  };

  const cancelBooking = (bookingId: string) => {
    const updatedBookings = bookings.map(booking =>
      booking.id === bookingId
        ? { ...booking, status: 'cancelled' as const }
        : booking
    );

    updateBookings(updatedBookings);

    const bookingToCancel = bookings.find(booking => booking.id === bookingId);
    toast({
      title: t('cancelledStatus'),
      description: t('youHaveBooked')
```

Рисунок 3.9 – Код сторінки BookingPage.tsx

Функціонал:

– відображення списку доступного інвентарю з фільтрами (категорія, доступність, локація);

– вибір конкретного обладнання для бронювання;

– форма для вибору дати, часу початку та тривалості бронювання (інтеграція з календарем);

– перевірка доступності обраного інвентарю на вибраний час;

– підтвердження бронювання та відображення повідомлення про успіх/помилку;

– відображення активних та минулих бронювань користувача (історія бронювань).

Сторінка AITrainerPage.tsx (рис. 3.10) надає користувачам можливість отримати персоналізовані рекомендації щодо тренувань на основі введеного спортивного інвентарю за допомогою інтегрованого AI-тренера.

```tsx
import React, { useState } from 'react';
import { Sparkles, ArrowLeft, Dumbbell } from 'lucide-react';
import { Button } from '@/components/ui/button';
import Navbar from '@/components/Navbar';
import AITrainer from '@/components/AITrainer';
import { useLanguage } from '@/context/LanguageContext';
import { equipmentData } from '@/data/equipmentData';

const AITrainerPage: React.FC = () => {
  const { t } = useLanguage();
  const [equipmentName, setEquipmentName] = useState<string>('');
  const [currentEquipment, setCurrentEquipment] = useState<string | null>(null);
  const [hasSearched, setHasSearched] = useState<boolean>(false);

  const categorizedEquipment = equipmentData.reduce((acc, equipment) => {
    if (!acc[equipment.category]) {
      acc[equipment.category] = [];
    }
    acc[equipment.category].push(equipment);
    return acc;
  }, {} as Record<string, typeof equipmentData>);

  const handleEquipmentSelect = (name: string) => {
    setEquipmentName(name);
    setCurrentEquipment(name);
    setHasSearched(true);
  };
  const handleReset = () => {
    setEquipmentName('');
    setCurrentEquipment(null);
    setHasSearched(false);
  };

  return (
    <div className="min-h-screen bg-gray-50 pb-20">
      <div className="px-4 py-6">
        <div className="flex items-center mb-6">
          <Sparkles className="w-6 h-6 text-primary mr-2" />
```

Рисунок 3.10 – Код сторінки AITrainerPage.tsx

Функціональні можливості та технічна реалізація AI-тренера:

– основний функціонал: використання AI-тренера для отримання персоналізованих порад щодо тренувань, підбору вправ, рекомендацій з використання обладнання та формування індивідуальних програм;

– технології: інтеграція з Google Gemini API, використання React Hook для керування станом чату, контексту автентифікації для персоналізації відповідей;

– особливості реалізації: підтримка багатомовності, toast-повідомлення для інформування про статус запитів, адаптивний дизайн, збереження історії діалогу у localStorage для зручності користувача;

– додатково: обробка помилок API, індикатор завантаження під час очікування відповіді, сучасний UI з використанням Tailwind CSS.

Сторінка EquipmentDetailPage.tsx (рис. 3.11) відображає детальну інформацію про обраний спортивний інвентар: фото, опис, характеристики, рейтинг, доступність, а також кнопку для бронювання. Вона інтегрується з даними та дозволяє користувачу швидко перейти до бронювання.

```tsx
import React, { useState } from 'react';
import { useParams, useNavigate } from 'react-router-dom';
import { ArrowLeft, Calendar, Clock, MapPin, Star } from 'lucide-react';
import Navbar from '@/components/Navbar';
import EquipmentStats from '@/components/EquipmentStats';
import RecommendedExercises from '@/components/RecommendedExercises';
import AITrainer from '@/components/AITrainer';
import { equipmentData, userBookings, Booking, saveBookings } from '@/data/equipmentData';
import { useToast } from '@/hooks/use-toast';
import { useLanguage } from '@/context/LanguageContext';
import { Badge } from '@/components/ui/badge';
import { toast } from 'sonner';

const EquipmentDetailPage = () => {
  const { id } = useParams<{ id: string }>();
  const navigate = useNavigate();
  const { toast: uiToast } = useToast();
  const { t } = useLanguage();
  const [selectedDate, setSelectedDate] = useState<string>('');
  const [selectedTime, setSelectedTime] = useState<string>('');

  const equipment = equipmentData.find(item => item.id === id);

  if (!equipment) {
    return (
      <div className="flex flex-col items-center justify-center h-screen bg-gray-50">
        <p className="text-lg text-gray-600 mb-4">{t('equipmentNotFound')}</p>
        <button
          onClick={() => navigate('/')}
          className="px-4 py-2 bg-primary text-white rounded-lg"
        >
          {t('goBackToHome')}
        </button>
      </div>
    );
  }

  const handleBooking = () => {
```

Рисунок 3.11 – Код сторінки EquipmentDetailPage.tsx

3.4 Тестування роботи інформаційної системи

Після встановлення додатку на головному екрані з’явиться ярлик (рис. 3.12).

Рисунок 3.12 – Ярлик додатку

Далі новий користувач здійснює реєстрацію (рис. 3.13), заповнює форму з особистими даними та створює обліковий запис. Після успішної реєстрації система автоматично входить у профіль або пропонує увійти.

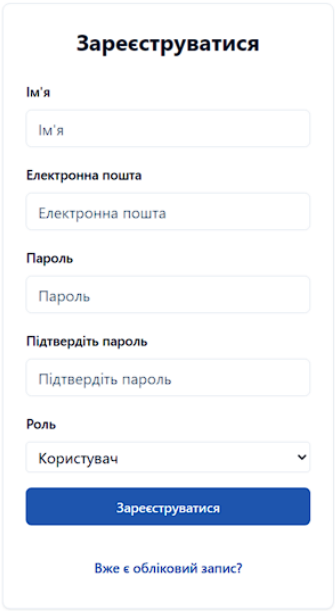

Рисунок 3.13 – Сторінка реєстрації

Процес реєстрації нового користувача та валідація даних:

1. Відображення форми реєстрації:
   – користувач бачить поля для введення імені, електронної пошти, пароля, підтвердження пароля.
2. Валідація введених даних:

– при спробі відправити форму перевіряється коректність email, довжина пароля, співпадіння паролів;

– разі помилки під полями з’являються підказки, а поле підсвічується червоним.

3. Відправка даних на сервер:

– після успішної валідації дані надсилаються на сервер або зберігаються у локальному сховищі.

4. Реакція системи:

– у разі успіху з’являється toast-повідомлення про успішну реєстрацію;

– користувач перенаправляється на сторінку входу або головну;

– у разі помилки (email вже зайнятий, проблеми з мережею) – toast з відповідним повідомленням.

Після успішної реєстрації користувач переходить на сторінку входу (рис. 3.14), вводить свої облікові дані та натискає кнопку для авторизації. У разі успіху відбувається перенаправлення на головну або профіль, у разі помилки – відображається відповідне повідомлення.

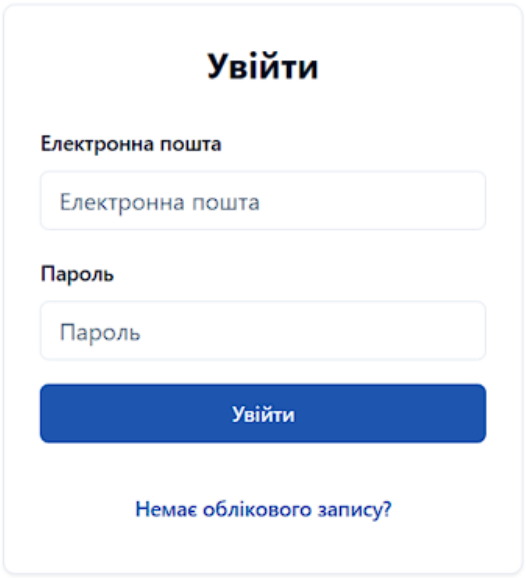

Рисунок 3.14 – Сторінка входу

Процес авторизації користувача та основні етапи автентифікації:

1. Відображення форми входу:
   – поля для email та пароля, кнопка "Увійти";
   – посилання на сторінку реєстрації.
2. Валідація введених даних:
   – перевірка формату email, наявності пароля;
   – у разі помилки – підказки під полями.
3. Аутентифікація:
   – після натискання "Увійти" дані перевіряються на сервері або у локальному сховищі.
4. Реакція системи:
   – успішний вхід – toast-повідомлення, перенаправлення на головну;
   – помилка (невірний пароль, неіснуючий email) – toast з описом.

Наступним кроком користувач бачить навігаційну панель (рис. 3.15) у нижній частині кожної сторінки додатку. Вона містить логотип, основні розділи (Головна, Інвентар, AI-тренер, Профіль). При натисканні на відповідний пункт меню відбувається перехід на обрану сторінку без перезавантаження додатку.

Рисунок 3.15 – Навігаційна панель

Після реєстрації користувач відкриває додаток і потрапляє на головну сторінку (рис 3.16 – 3.17), де бачить логотип, фільтра, обладнання і карту з місцезнаходженням. Звідси можна перейти до перегляду списку інвентарю або скористатися пошуком для швидкого знаходження потрібного обладнання.

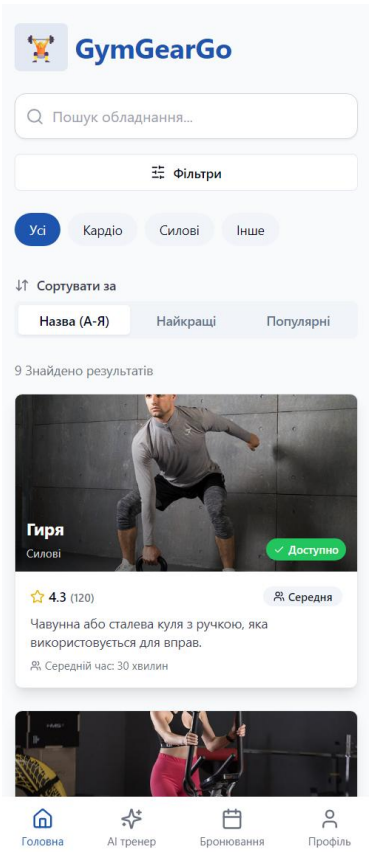

Рисунок 3.16 – Головна сторінка з відображенням обладнанням

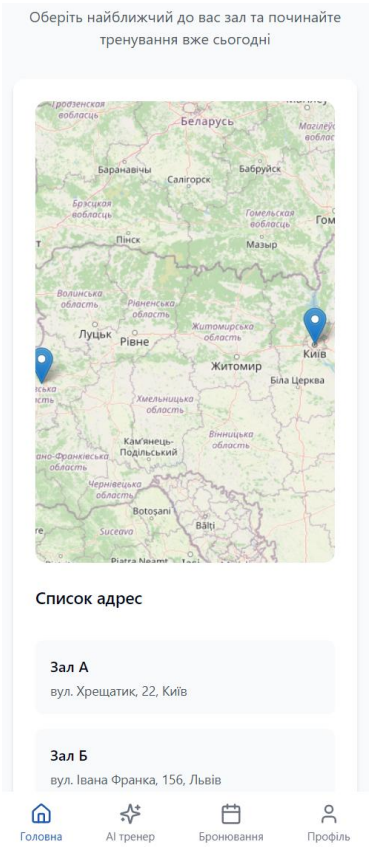

Рисунок 3.17 – Головна сторінка з відображенням карти з місцерозташуванням спортивних залів

Процес відображення каталогу інвентарю та інтерактивної карти:

1. Завантаження списку інвентарю:
   - після входу користувача видно список доступного спортивного обладнання.
2. Відображення основного інтерфейсу:
   - користувач бачить сітку карток інвентарю з фото, назвою, категорією, статусом доступності;
   - у верхній частині фільтри за категоріями, пошук, за сортуванням;
   - у нижній частині під списком обладнання, відображається карта з місцерозташуванням спортивних залів.
3. Фільтрація та пошук:
   - при виборі категорії або введенні запиту у пошук список карток оновлюється в реальному часі;
   - якщо результатів немає – відображається повідомлення "Інвентар не знайдено".
4. Взаємодія з карткою інвентарю:
   - клік по картці відкриває сторінку деталей інвентарю;
   - якщо інвентар недоступний – кнопка "Забронювати" неактивна.
5. Інтеграції:
   - підтримка багатомовності;
   - toast-повідомлення при помилках завантаження або відсутності даних.

Далі користувач переходить у розділ профілю (рис. 3.18) через меню навігації. На сторінці відображаються його основні дані (ім’я, email, аватар), а також список усіх бронювань. За потреби користувач натискає кнопку "Редагувати профіль", змінює свої дані та зберігає зміни. Для виходу з акаунту натискає кнопку "Вийти" і повертається на сторінку входу.

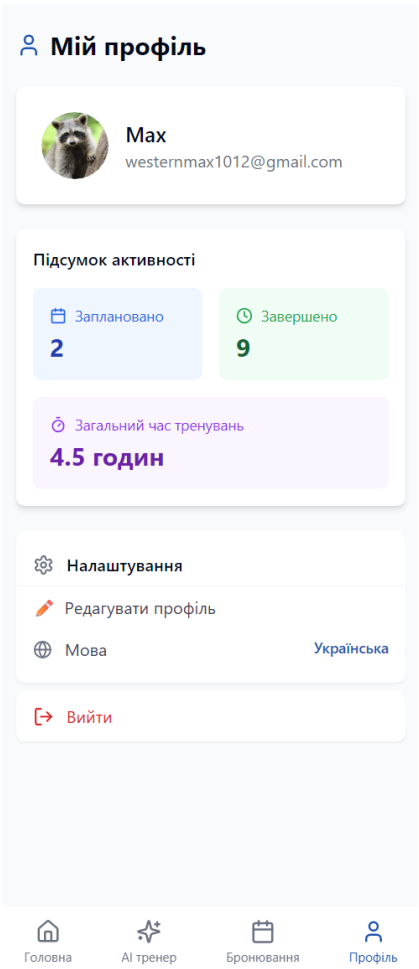

Рисунок 3.18 – Сторінка профілю користувача

На сторінці профілю користувач може виконувати такі дії:

1. Завантаження профілю:
   - після переходу на сторінку система отримує дані користувача з контексту або локального сховища;
   - відображається аватар, ім’я, email, налаштування та підсумок активності.
2. Редагування профілю:
   - користувач може натиснути "Редагувати профіль", змінити ім’я, email, фото, також змінити мову інтерфейсу;
   - після збереження – toast-повідомлення про успіх або помилку.
3. Перегляд історії бронювань:

– відображається кількість запланованих, завершених та загальний час тренувань.

4. Вихід з акаунту:

– кнопка "Вийти" очищає дані користувача, перенаправляє на сторінку входу.

На наступному етапі користувач потрапляє на сторінку «Мої бронювання» (рис. 3.19). На сторінці «Мої бронювання» користувач бачить список усіх своїх бронювань із деталями, статусом та можливістю керування кожним бронюванням.

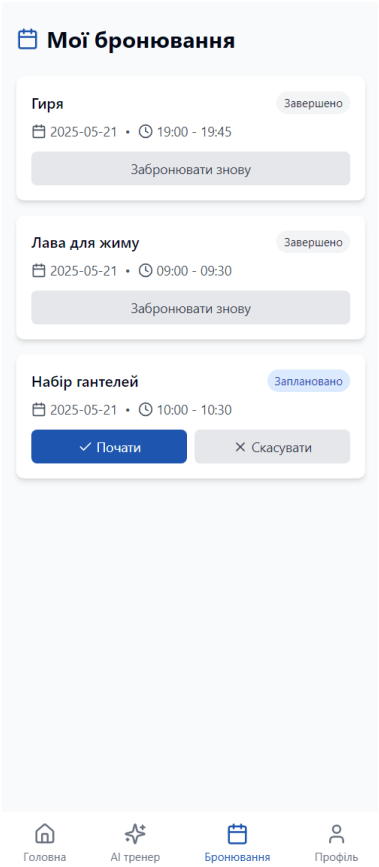

Рисунок 3.19 – Сторінка бронювання

Перегляд та керування статусами замовлень:

1. Відображення бронювань:

– всі бронювання користувача відображаються у вигляді списку.

2. Для кожного бронювання показано:

– назву інвентарю;

– дату та час (початок – кінець);

– статус (заплановано, активно, завершено, скасовано) з кольоровим бейджем;

– якщо бронювань немає – з’являється повідомлення та кнопка "Знайти обладнання" для переходу на головну сторінку.

3. Дії з бронюванням:

– кнопка "Почати" – змінює статус на "Активно";

– кнопка "Скасувати" – змінює статус на "Скасовано";

– кнопка "Завершити" – змінює статус на "Завершено";

– кнопка "Забронювати знову" – переходить на сторінку інвентарю для повторного бронювання.

Далі користувач відкриває сторінку AI-тренера (рис. 3.21), вибирає доступне спорядження.

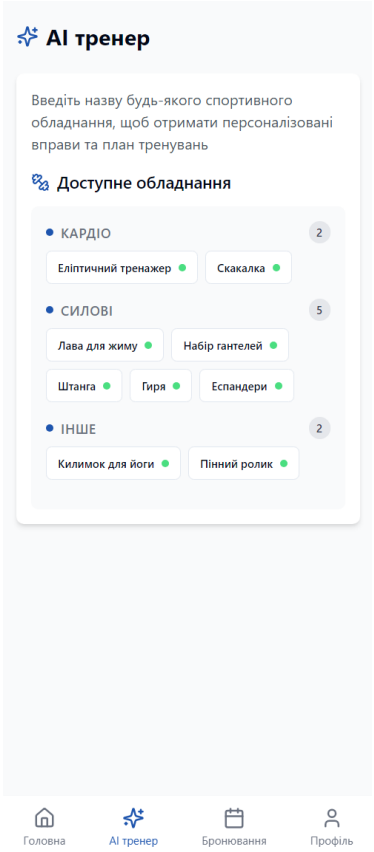

Рисунок 3.20 – Сторінка AI-тренера

Як видно на рисунку 3.21, система генерує персоналізовані рекомендації та вправи, які можна додати до свого плану.

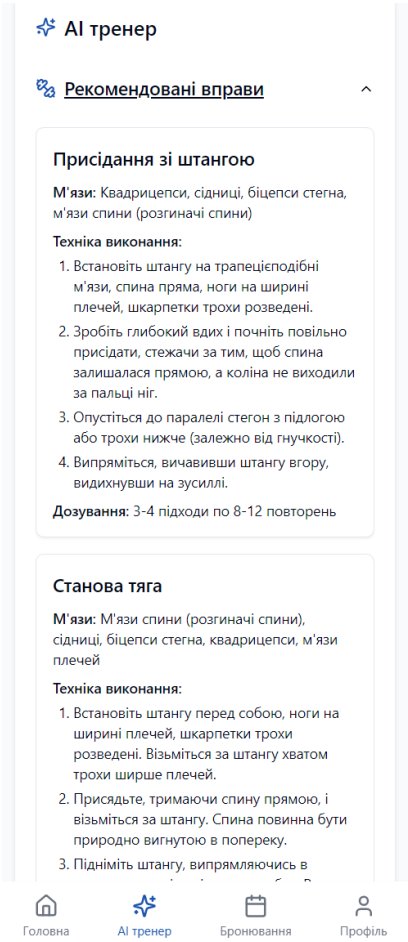

Рисунок 3.21 – Вигляд згенерованих вправ за допомогою AI-тренера

Генерація персоналізованих тренувальних рекомендацій виглядає так:

1. Введення запиту:
   – користувач бачить поле, де може вибирати доступне обладнання.
2. Генерація рекомендацій:
   – після натискання "Згенерувати" запит надсилається до AI (Gemini API);
   – відображається індикатор завантаження.

3. Відображення результату:
   – на сторінці з’являється список рекомендованих вправ, опис, поради;
   – кожну вправу можна згенерувати безліч разів.
4. Повідомлення та інтеграції:
   – Toast-повідомлення про успіх/помилку генерації;
   – підтримка багатомовності.

Наступним етапом є сторінка деталей інвентарю. На головній сторінці або у списку інвентарю користувач натискає на картку певного обладнання (наприклад, "Штанга") (рис. 3.22). Далі відбувається перехід на сторінку з детальною інформацією про цей інвентар: опис, характеристики, доступність, фото та кнопка для бронювання.

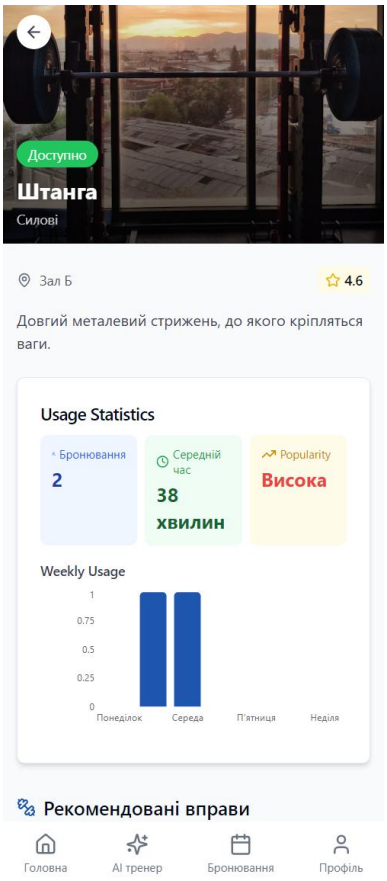

Рисунок 3.22 – Сторінка деталей обладнання

Сторінка з детальною інформацією про вибраний інвентар містить наступну інформацію:

1. Завантаження даних:
   - після переходу на сторінку система отримує детальну інформацію про обраний інвентар (назва, фото, опис, характеристики, статус).
2. Основний інтерфейс:
   - картка з фото, описом, технічними характеристиками, статистикою використання;
   - кнопка "Забронювати" (активна лише для доступного інвентарю).
3. Додаткові функції:
   - перегляд рекомендованих вправ для цього інвентарю.
4. Повідомлення та реакція:
   - toast-повідомлення про успішне бронювання, помилки;
   - у разі помилки завантаження – з’являється повідомлення з пропозицією повернутися на головну.

Сторінка адміністратора (рис. 3.23) призначена для повного керування спортивним інвентарем, заявками на бронювання, перегляду статистики та управління розкладом доступності обладнання.

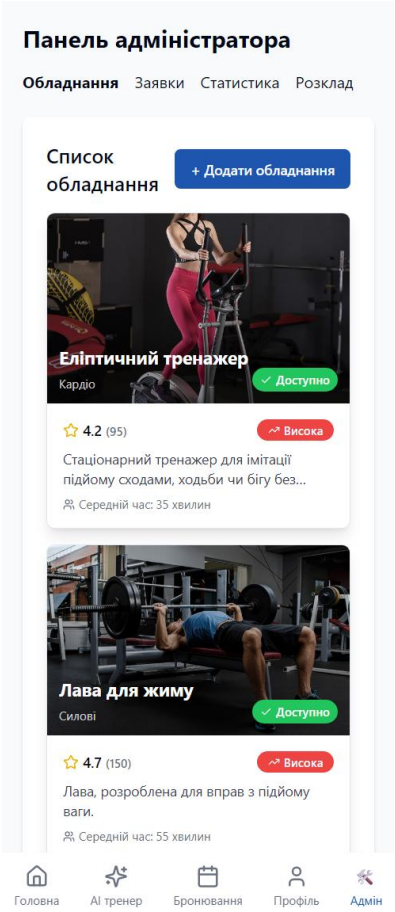

Рисунок 3.23 – Сторінка адміністратора

Функціональні можливості адміністративної панелі та управління ресурсами наступні:

1. Навігація між вкладками:
   – адміністратор бачить вкладки: "Обладнання", "Заявки", "Статистика", "Розклад";
   – може швидко перемикатися між розділами для виконання різних адміністративних дій.
2. Керування обладнанням:
   – відображається список усього обладнання з детальною інформацією;
   – є кнопка "+ Додати обладнання" — відкриває вікно для створення нового інвентарю (назва, категорія, опис, зображення, локація, доступність);

– для кожного елементу доступні кнопки "Редагувати" та "Видалити";

– редагування відкриває модальне вікно з формою, видалення одразу прибирає інвентар зі списку;

– зміни зберігаються у локальному сховищі та синхронізуються з іншими сторінками.

3. Модерація заявок на бронювання:

– вкладка "Заявки" містить список усіх бронювань;

– для заявок зі статусом "Очікує підтвердження" доступні кнопки "Підтвердити" та "Відхилити";

– після дії статус заявки змінюється, відображається toast-повідомлення про результат.

5. Статистика та аналітика:

– вкладка "Статистика" показує детальну статистику по кожному обладнанню: кількість бронювань, середня тривалість, популярність, рейтинг;

– додатково формується аналітичний звіт: загальна кількість обладнання, кількість бронювань, найпопулярніше та найменш популярне обладнання.

6. Керування розкладом доступності:

– вкладка "Розклад" дозволяє обрати обладнання зі списку та змінити його доступність для бронювання (перемикач "Доступне/Недоступне").

3.5 Висновки

Розроблена інформаційна система успішно вирішує задачу автоматизації процесів бронювання спортивного інвентарю, демонструючи високу ефективність та надійність. Модульна архітектура системи забезпечує

оптимальну взаємодію компонентів, а інтеграція з AI-сервісами значно підвищує якість персоналізованих рекомендацій для користувачів. Реалізований механізм синхронізації з localStorage гарантує стабільну роботу та безпеку даних, хоча і створює певні обмеження для масштабування системи. Тестування підтвердило інтуїтивність інтерфейсу та коректність обробки даних, що особливо важливо для кінцевих користувачів. Система демонструє високу продуктивність у обробці запитів та генерації аналітичних даних для адміністраторів, що підтверджує ефективність обраних технологічних рішень.

ВИСНОВКИ

У ході виконання бакалаврської кваліфікаційної роботи проведено аналіз предметної області, розглянуто сучасний стан організації процесу бронювання спортивного інвентарю та доведена актуальність питання автоматизації бронювання та використання спортивного інвентарю в спортзалах.

Розглянуто основні можливості, переваги і недоліки існуючих інформаційних систем для бронювання та використання спортивного інвентарю в спортзалах. Порівняльний аналіз існуючих рішень показав наявність базового функціоналу для управління спортивними закладами, проте виявив відсутність спеціалізованих систем для детального бронювання спортивного інвентарю.

Здійснено опис та обґрунтування оптимальних інформаційних технології, зокрема використання TypeScript, React, Vite, TailwindCSS, а також інтеграція AI та геолокаційних сервісів, забезпечило створення гнучкої, масштабованої та зручної у використанні системи. Особливу увагу приділено багатомовності, адаптивності інтерфейсу, швидкодії та безпеці даних користувачів.

IT-інфраструктура розробленої системи базується на сучасних веб-технологіях і багаторівневій архітектурі, що забезпечує чітке розділення відповідальностей між інтерфейсом, бізнес-логікою, даними та інтеграцією із зовнішніми сервісами. Використання TypeScript, React, Vite та TailwindCSS дозволило створити швидкий, адаптивний і багатомовний додаток, який легко підтримувати та розширювати. Інтеграція з AI та картографічними сервісами підвищила персоналізацію і зручність для користувачів.

Розроблена інформаційна система значно підвищує ефективність управління спортивним інвентарем, забезпечуючи автоматизацію процесів бронювання та спрощення взаємодії між користувачами та адміністрацією спортзалів. Модульна архітектура системи, що включає взаємопов'язані компоненти від аутентифікації до AI-тренера, демонструє високу надійність та гнучкість у масштабуванні. Порівняно з існуючими рішеннями, система відрізняється глибшою інтеграцією компонентів через стандартизовані

інтерфейси, що забезпечує кращу продуктивність та зручність використання. Застосування сучасних технологій та AI-рішень дозволило досягти високого рівня автоматизації та персоналізації сервісу, що підтверджується позитивними результатами тестування у реальних умовах спортивних закладів.

Структура додатку побудована на сучасних принципах модульності, що дозволяє легко розширювати функціонал і підтримувати систему у майбутньому.

Реалізовано повний цикл користувацьких сценаріїв: від реєстрації та авторизації до бронювання, отримання персоналізованих рекомендацій, перегляду статистики та адміністрування інвентарю. Інтеграція AI-модуля надала можливість формувати індивідуальні тренувальні плани, а використання інтерактивної карти підвищило зручність навігації для користувачів.

Проведено тестування додатку, яке підтвердило відповідність системи поставленим вимогам щодо функціональності, надійності та зручності використання. Всі основні задачі, визначені на початку роботи, виконано у повному обсязі.

За результатами виконання роботи опубліковано тези доповідей на тему: «Інформаційна система для бронювання та використання спортивного інвентарю в спортзалах» на Міжнародній науково-практичній інтернет-конференції «Молодь в науці: дослідження, проблеми, перспективи (Вінниця, 2024-2025 рр.).

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Барановський М. Д., Войцеховська О. О. Інформаційна система для бронювання та використання спортивного інвентарю в спортзалах. *Міжнародна науково-практична інтернет-конференція студентів, аспірантів та молодих науковців «Молодь в науці: дослідження, проблеми, перспективи»:* збірник матеріалів конференції. Вінниця, 2025. URL: https://conferences.vntu.edu.ua/index.php/mn/mn2025/paper/view/25057/20715 (дата звернення: 12.05.2025).

2. Shan, G. B. Sport Equipment Evaluation and Optimization – A Review of the Relationship between Sport Science Research and Engineering. *The Open Sports Sciences Journal*. 2008. Vol.1, p. 5-11. DOI: http://dx.doi.org/10.2174/1875399X00801010005 (дата звернення: 11.05.2025).

3. Anis, M., AlTaher, G., Sarhan, W., Elsemary, M. Sports Equipment. In: *Nanovate*. Cham: Springer, 2017. p. 183–192. DOI: https://doi.org/10.1007/978-3-319-44863-3_10 (дата звернення: 11.05.2025).

4. 2025 Global Fitness Trends: Strategies to Thrive in a Changing Industry. URL: https://corehandf.com/2025-global-fitness-trends-strategies-to-thrive-in-a-changing-industry/ (дата звернення: 12.05.2025).

5. Gym Usage Monitoring Systems. URL: https://gymequipmentpro.com/gym-usage-monitoring-systems/ (дата звернення: 12.05.2025).

6. Fitness Studio Software and Gym Booking Software: A Comprehensive Guide. URL: https://blog.zamstudios.com/fitness-studio-software-and-gym-booking-software-a-comprehensive-guide/ (дата звернення: 13.05.2025).

7. Pitchbooking. URL: https://pitchbooking.com/ (дата звернення: 13.05.2025).

8. Wodify. URL: https://www.wodify.com/ (дата звернення: 13.05.2025).

9. Glofox. URL: https://www.glofox.com/ (дата звернення: 13.05.2025).

10. Osmani A. Learning JavaScript Design Patterns: A JavaScript and React Developer's Guide. Sebastopol: O'Reilly Media, 2023. – 298 p. URL: https://dokumen.pub/learning-javascript-design-patterns-a-javascript-and-react-developers-guide-2nbsped-9781098139872.html (дата звернення: 14.05.2025).

11. Cherny B. Programming TypeScript: Making Your JavaScript Applications Scale. Sebastopol: O'Reilly Media, 2019. – 324 p. URL: https://github.com/kpyszkowski/kpyszkowski/blob/main/books/Boris%20Cherny%20-%20Programming%20TypeScript%3A%20Making%20Your%20JavaScript%20Applications%20Scale%20-%20O%E2%80%B2Reilly%20(2019).pdf (дата звернення: 14.05.2025).

12. TypeScript. URL: https://www.typescriptlang.org/ (дата звернення: 15.05.2025).

13. Visual Studio Code. URL: https://code.visualstudio.com/ (дата звернення: 15.05.2025).

14. Capacitor. URL: https://capacitorjs.com/ (дата звернення: 15.05.2025).

15. Android Developers. URL: https://developer.android.com/develop (дата звернення: 16.05.2025).

16. Apple Developer Documentation. URL: https://developer.apple.com/documentation/ (дата звернення: 16.05.2025).

17. Android Studio. URL: https://developer.android.com/studio (дата звернення: 16.05.2025).

18. Banks A., Porcello E. Learning React: Modern Patterns for Developing React Apps. Sebastopol: O'Reilly Media, 2020. – 310 p. URL: https://github.com/kpyszkowski/kpyszkowski/blob/main/books/Alex%20Banks%20%26%20Eve%20Porcello%20-%20Learning%20React%3A%20Learning%20React%20Modern%20Patterns%20for%20Developing%20React%20Apps%20-%20O%E2%80%B2Reilly%20(2020).pdf (дата звернення: 17.05.2025).

19. Vite. URL: https://vite.dev/ (дата звернення: 17.05.2025).

20. Bhat K.E. Ultimate Tailwind CSS Handbook: Build Sleek and Modern Websites with Immersive UIs Using Tailwind CSS. Delhi: Orange Education Pvt Limited, 2023. – 294 p. URL: https://dokumen.pub/ultimate-tailwind-css-handbook-build-sleek-and-modern-websites-with-immersive-uis-using-tailwind-css-team-ira-9388590767-9789388590761.html (дата звернення: 17.05.2025).

21. Gemini Google. URL: https://gemini.google.com/app (дата звернення: 17.05.2025).

22. OpenStreetMap. URL: https://www.openstreetmap.org/#map=6/48.54/31.17 (дата звернення: 18.05.2025).

23. Leaflet. URL: https://leafletjs.com/ (дата звернення: 18.05.2025).

Додаток А

(обов’язковий)

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації

ЗАТВЕРДЖУЮ
Завідувач кафедри САІТ
_____ д.т.н., проф. Віталій МОКІН
“____”_____ 2025 р.

ТЕХНІЧНЕ ЗАВДАННЯ

на бакалаврську кваліфікаційну роботу

ІНФОРМАЦІЙНА СИСТЕМА ДЛЯ БРОНЮВАННЯ ТА ВИКОРИСТАННЯ СПОРТИВНОГО ІНВЕНТАРЮ В СПОРТЗАЛАХ

08-34.БКР.001.00.000 ТЗ

Керівник: Ph.D., ст. викл. каф. САІТ
_____ Ольга ВОЙЦЕХОВСЬКА
«__» _____ 2025 р.
Розробив студент гр. 2ІСТ-21б
_____ Максим БАРАНОВСЬКИЙ
«__» _____ 2025 р.

Вінниця 2025

1. Підстава для проведення робіт.

Підставою для виконання роботи є наказ №__по ВНТУ від «_» _____2025р., та індивідуальне завдання на БКР, затверджене протоколом №_ засідання кафедри САІТ від «_» _____2025р.

2. Джерела розробки:

1) Shan, G. B. Sport Equipment Evaluation and Optimization – A Review of the Relationship between Sport Science Research and Engineering. *The Open Sports Sciences Journal*. 2008. Vol.1, p. 5-11. DOI: http://dx.doi.org/10.2174/1875399X00801010005;

2) Anis, M., AlTaher, G., Sarhan, W., Elsemary, M. Sports Equipment. In: *Nanovate*. Cham: Springer, 2017. p. 183–192. DOI: https://doi.org/10.1007/978-3-319-44863-3_10.

3. Мета і призначення роботи.

Метою дослідження є розроблення інформаційної системи для бронювання та використання спортивного інвентарю в спортзалах.

4. Вихідні дані для проведення робіт: дані про спортивний інвентарь, сценарії використання спортивного інвентарю, стандарти спортивного інвентарю.

Методи дослідження:

1) Аналіз вимог, аналіз потреб користувачів, порівняльний аналіз існуючих рішень та технологій, аналіз технічних можливостей інтеграції AI-сервісів, аналіз статистики завантаженості інвентарю, структурний аналіз архітектури інформаційних систем, розроблення UML діаграм, тестування.

5. Етапи роботи і терміни їх виконання:

a) Аналіз предметної області _____–_____
b) Огляд аналогів _____–_____
c) Вибір оптимальних інформаційних технологій _____–_____
d) Розроблення інформаційної системи для бронювання та використання спортивного інвентарю в спортзалах _____–_____
e) Оформлення матеріалів до захисту БКР _____–_____

6. Очікувані результати та порядок реалізації

Отримання інформаційної системи для бронювання та використання спортивного інвентарю в спортзалах.

7. Вимоги до розробленої документації

Текстова та ілюстративна частини роботи оформлені у відповідності до вимог «Методичних вказівок до виконання бакалаврських кваліфікаційних робіт для студентів спеціальностей: 124 «Системний аналіз», 126 «Інформаційні системи та технології» (освітня програма «Прикладні інформаційні технології»)».

8. Порядок приймання роботи

Публічний захист «___» _____ 2025 р.
Початок роботи «___» _____ 2025 р.
Граничні терміни виконання БКР «___» _____ 2025 р.

Розробив студент групи 2ІСТ-21б _____ Максим БАРАНОВСЬКИЙ

Додаток Б
(обов'язковий)

ПРОТОКОЛ ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Назва роботи: «Інформаційна система для бронювання та використання спортивного інвентарю в спортзалах»

Тип роботи: бакалаврська кваліфікаційна робота

Підрозділ: кафедра САІТ, ФІІТА, гр. 2ІСТ-21б

Коефіцієнт подібності текстових запозичень, виявлених у роботі системою StrikePlagiarism 2,04 %

Висновок щодо перевірки кваліфікаційної роботи (відмітити потрібне):

- ■ Запозичення, виявлені у роботі, є законними і не містять ознак плагіату, фабрикації, фальсифікації. Роботу прийняти до захисту
- □ У роботі не виявлено ознак плагіату, фабрикації, фальсифікації, але надмірна кількість текстових запозичень та/або наявність типових розрахунків не дозволяють прийняти рішення про оригінальність та самостійність її виконання. Роботу направити на доопрацювання.
- □ У роботі виявлено ознаки плагіату та/або текстових маніпуляцій як спроб укриття плагіату, фабрикації, фальсифікації, що суперечить вимогам законодавства та нормам академічної доброчесності. Робота до захисту не приймається.

Експертна комісія:

Віталій МОКІН, зав. каф. САІТ _____ (підпис)

Євгеній КРИЖАНОВСЬКИЙ, доц. каф. САІТ _____ (підпис)

Особа, відповідальна за перевірку _____ (підпис) Сергій ЖУКОВ

З висновком експертної комісії ознайомлений(-на)

Керівник _____ (підпис) Ольга ВОЙЦЕХОВСЬКА, PhD., ст. викл. каф. САІТ

Здобувач _____ (підпис) Максим БАРАНОВСЬКИЙ

Додаток В
(довідниковий)
Фрагмент лістингу програми

Код файлу додатку Index.tsx

```
import React, { useState, useEffect } from 'react';
import { Search, SlidersHorizontal, Check, X, ArrowDownUp } from 'lucide-react';
import CategoryFilter from '@/components/CategoryFilter';
import EquipmentCard from '@/components/EquipmentCard';
import Navbar from '@/components/Navbar';
import MapSection from '@/components/MapSection';
import AdvancedFilters, { FilterOptions } from '@/components/AdvancedFilters';
import { equipmentData as initialEquipmentData, Equipment, categories, EquipmentCategory } from '@/data/equipmentData';
import { useLanguage } from '@/context/LanguageContext';
import { Button } from '@/components/ui/button';
import { Tabs, TabsContent, TabsList, TabsTrigger } from "@/components/ui/tabs";

const Index = () => {
  const [selectedCategory, setSelectedCategory] = useState<EquipmentCategory | null>(null);
  const [searchQuery, setSearchQuery] = useState('');
  const [filteredEquipment, setFilteredEquipment] = useState<Equipment[]>([]);
  const [showFilters, setShowFilters] = useState(false);
  const [activeFilters, setActiveFilters] = useState<FilterOptions>({
    location: null,
    minRating: null,
    availableOnly: false,
    category: null,
  });
  const [sortBy, setSortBy] = useState<'name' | 'rating' | 'popularity'>('name');
  const [sortOrder, setSortOrder] = useState<'asc' | 'desc'>('asc');
  const { t, locale } = useLanguage();
  const [equipmentList, setEquipmentList] = useState<Equipment[]>(() => {
    const stored = localStorage.getItem('equipmentData');
    if (stored) {
      try {
        return normalizeEquipmentList(JSON.parse(stored));
      } catch {
        return initialEquipmentData;
      }
    }
```

```
  return initialEquipmentData;
});

const normalizeEquipmentList = (list: any[]): Equipment[] =>
list.map(eq => ({
  ...eq,
  category: eq.category as EquipmentCategory
}));

useEffect(() => {
  setFilteredEquipment(
    selectedCategory
      ? equipmentList.filter(eq => eq.category === selectedCategory)
      : equipmentList
  );
}, [selectedCategory, equipmentList]);

useEffect(() => {
  let result = equipmentList;


  if (selectedCategory) {
    result = result.filter(item => item.category === selectedCategory);
  }

  if (searchQuery) {
    const query = searchQuery.toLowerCase();
    result = result.filter(item =>
      item.name.toLowerCase().includes(query) ||
      item.description.toLowerCase().includes(query)
    );
  }

  if (activeFilters.location) {
    result = result.filter(item => item.location === activeFilters.location);
  }

  if (activeFilters.minRating) {
    result = result.filter(item => (item.rating || 0) >= (activeFilters.minRating || 0));
  }

  if (activeFilters.availableOnly) {
```

```
    result = result.filter(item => item.available);
  }

  result = sortEquipment(result, sortBy, sortOrder);

  setFilteredEquipment(result);
}, [selectedCategory, searchQuery, activeFilters, sortBy, sortOrder, equipmentList]);

useEffect(() => {
  const handleLocationSelect = (event: CustomEvent<{locationName: string}>) => {
    setActiveFilters(prev => ({
      ...prev,
      location: event.detail.locationName
    }));
  };

  window.addEventListener('locationSelect', handleLocationSelect as EventListener);

  return () => {
    window.removeEventListener('locationSelect', handleLocationSelect as EventListener);
  };
}, []);

const handleFilterApply = (filters: FilterOptions) => {
  setActiveFilters(filters);

  if (filters.category !== selectedCategory) {
    setSelectedCategory(filters.category);
  }
};

const toggleFilters = () => {
  setShowFilters(!showFilters);
};

const handleSortChange = (value: string) => {
  const [sort, order] = value.split('-') as [('name' | 'rating' | 'popularity'), ('asc' | 'desc')];
  setSortBy(sort);
  setSortOrder(order);
};
```

Код файлу додатку AITrainer.tsx

```
import React, { useState } from 'react';
import { Sparkles, Loader2, Dumbbell, BookOpen, Lightbulb } from 'lucide-react';
import { Button } from '@/components/ui/button';
import { Accordion, AccordionContent, AccordionItem, AccordionTrigger } from '@/components/ui/accordion';
import { Card, CardContent } from '@/components/ui/card';
import { generateExercises } from '@/lib/gemini';
import { useLanguage } from '@/context/LanguageContext';
import { toast } from 'sonner';

interface AITrainerProps {
  equipmentName: string;
}

interface ExerciseData {
  name?: string;
  muscles?: string;
  technique?: string[];
  repsSets?: string;
  [key: string]: any;
}

type Exercise = string | ExerciseData;

const AITrainer: React.FC<AITrainerProps> = ({ equipmentName }) => {
  const { locale, t } = useLanguage();  const [loading, setLoading] = useState(true);
  const [exercises, setExercises] = useState<Exercise[]>([]);
  const [tips, setTips] = useState<string[]>([]);
  const [workoutPlan, setWorkoutPlan] = useState<string>('');
  const [isGenerated, setIsGenerated] = useState(false);

  React.useEffect(() => {
    handleGenerateExercises();
  }, [equipmentName]);

  const handleGenerateExercises = async () => {
    setLoading(true);
    try {
      const loadingDelay = new Promise(resolve => setTimeout(resolve, 1500));

      const resultPromise = generateExercises(equipmentName, locale as 'en' | 'uk');
```

```
const [result] = await Promise.all([resultPromise, loadingDelay]);
if (result && result.exercises && result.exercises.length > 0) {

 const firstExercise = result.exercises[0];
 if (
  typeof firstExercise === 'string' &&
  (firstExercise.includes('ліміту запитів') ||
  firstExercise.includes('Невалідний ключ API') ||
  firstExercise.includes('не вдалось згенерувати'))
 ) {

  toast.error(firstExercise);
  setIsGenerated(false);
 } else {

  const processedExercises = result.exercises.map((exercise: any) => {

   if (typeof exercise === 'object' && exercise !== null) {
    return exercise;
   }

   if (typeof exercise === 'string') {
    try {

     if (exercise.includes('{') && exercise.includes('}')) {
      const jsonMatch = exercise.match(/\{[\s\S]*\}/);
      if (jsonMatch) {
       try {
        return JSON.parse(jsonMatch[0]);
       } catch (e) {
        return exercise;
       }
      }
     }
     return exercise;
    } catch (e) {
     return exercise;
    }
   }

   return exercise;
  });
```

```
        setExercises(processedExercises);
        setTips(result.tips || []);
        setWorkoutPlan(result.workoutPlan || '');          setIsGenerated(true);
        toast.success(t('aiTrainerSuccess'), {
          description: `${equipmentName} - ${processedExercises.length} ${t('recommendedExercises').toLowerCase()}`
        });
      }
    } else {

      console.error('Отримана порожня або некоректна відповідь від API');
      toast.error(t('aiTrainerError'));
    }
  } catch (error) {
    console.error('Помилка в AI тренері:', error);
    toast.error(t('aiTrainerError'));


    if (process.env.NODE_ENV === 'development') {
      console.error('Деталі помилки:', error);
    }
  } finally {
    setLoading(false);
  }
};

return (
  <div className="mt-8 bg-white rounded-lg shadow-md p-5">
    <div className="flex items-center justify-between mb-4">
      <h2 className="text-xl font-semibold flex items-center">
        <Sparkles className="w-5 h-5 mr-2 text-primary" />
        {t('aiTrainer')}
      </h2>
```

Додаток Г
(обов’язковий)

ІЛЮСТРАТИВНА ЧАСТИНА

ІНФОРМАЦІЙНА СИСТЕМА ДЛЯ БРОНЮВАННЯ ТА ВИКОРИСТАННЯ СПОРТИВНОГО ІНВЕНТАРЮ В СПОРТЗАЛАХ

Нормоконтроль: к.т.н., доцент
_____Сергій ЖУКОВ
«___» _____2025 р.

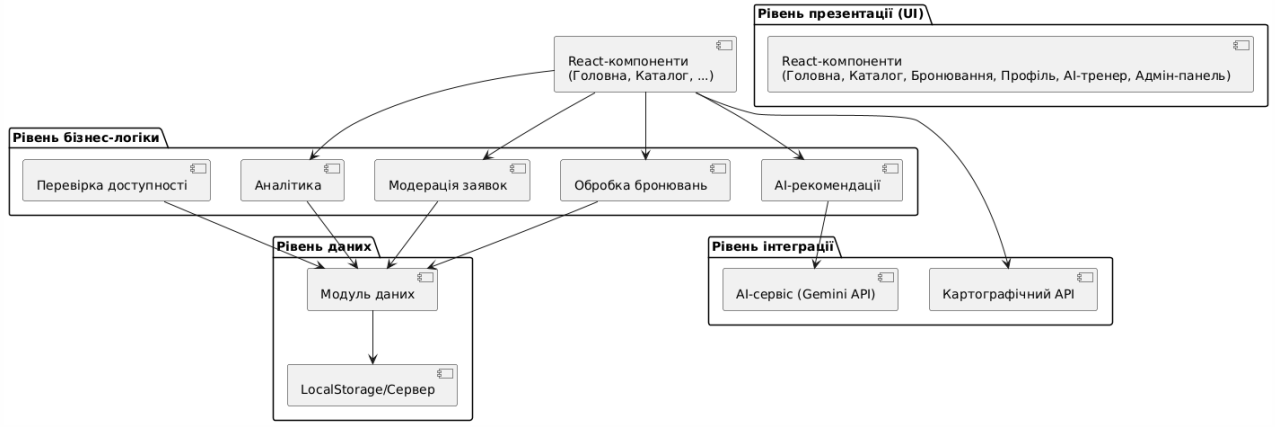

Рисунок Г.1 – Архітектура інформаційної системи

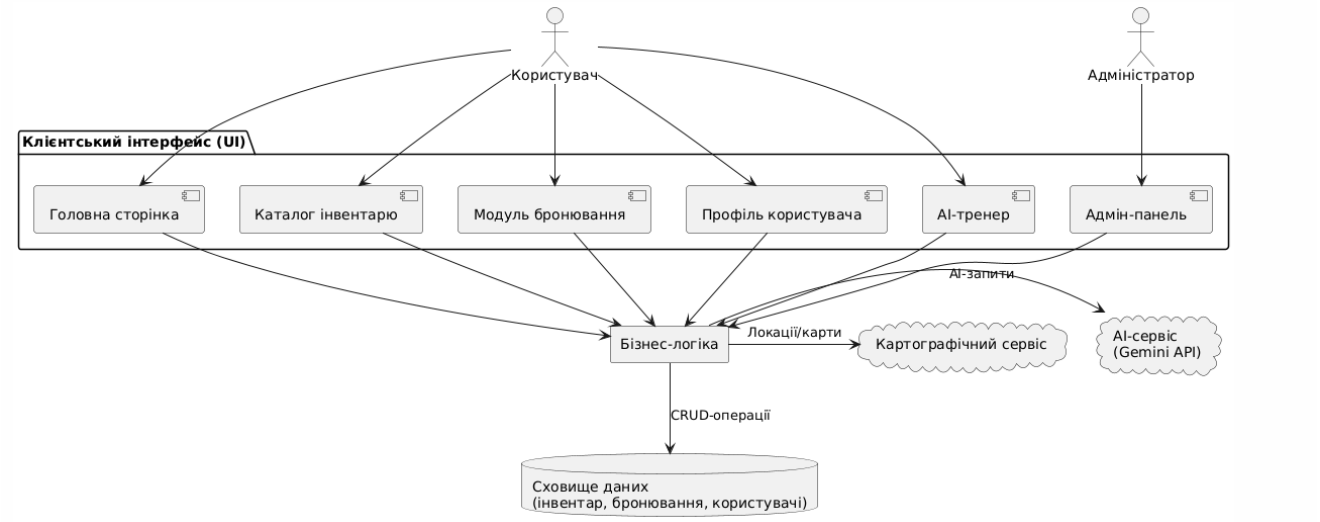

Рисунок Г.2 – Блок-схема зв’язків програмного забезпечення

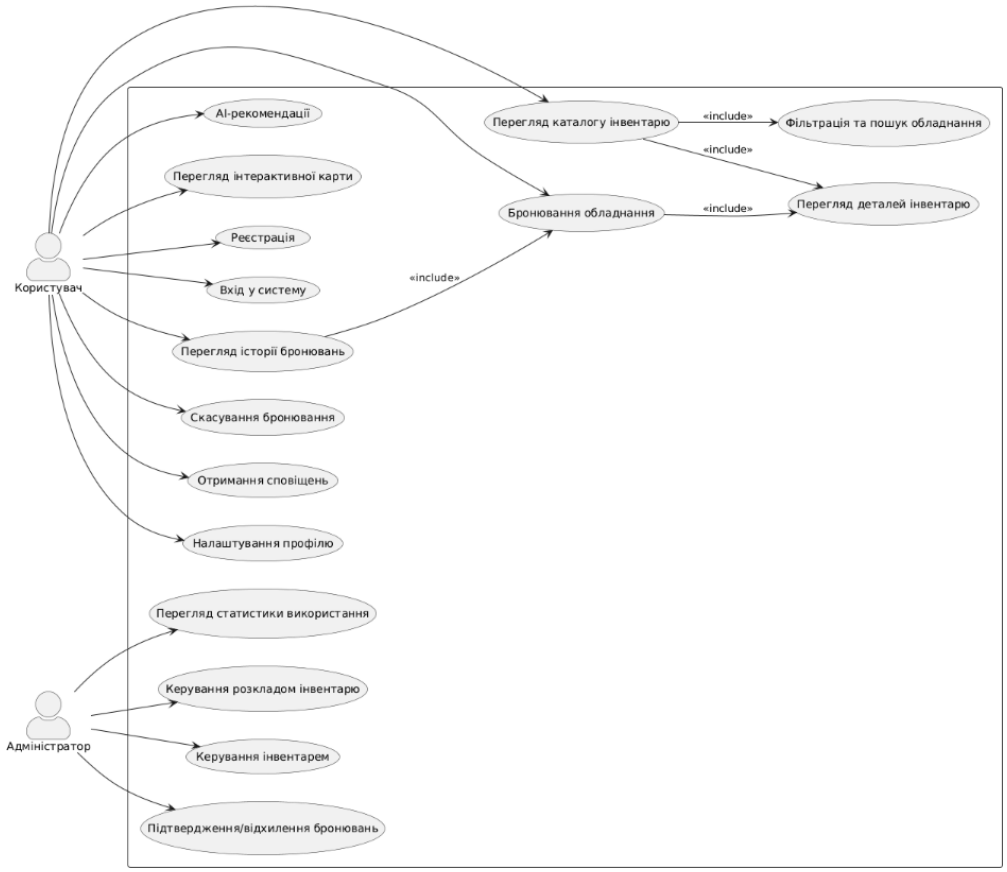

Рисунок Г.3 – Діаграма випадків використання (Use Case Diagram)

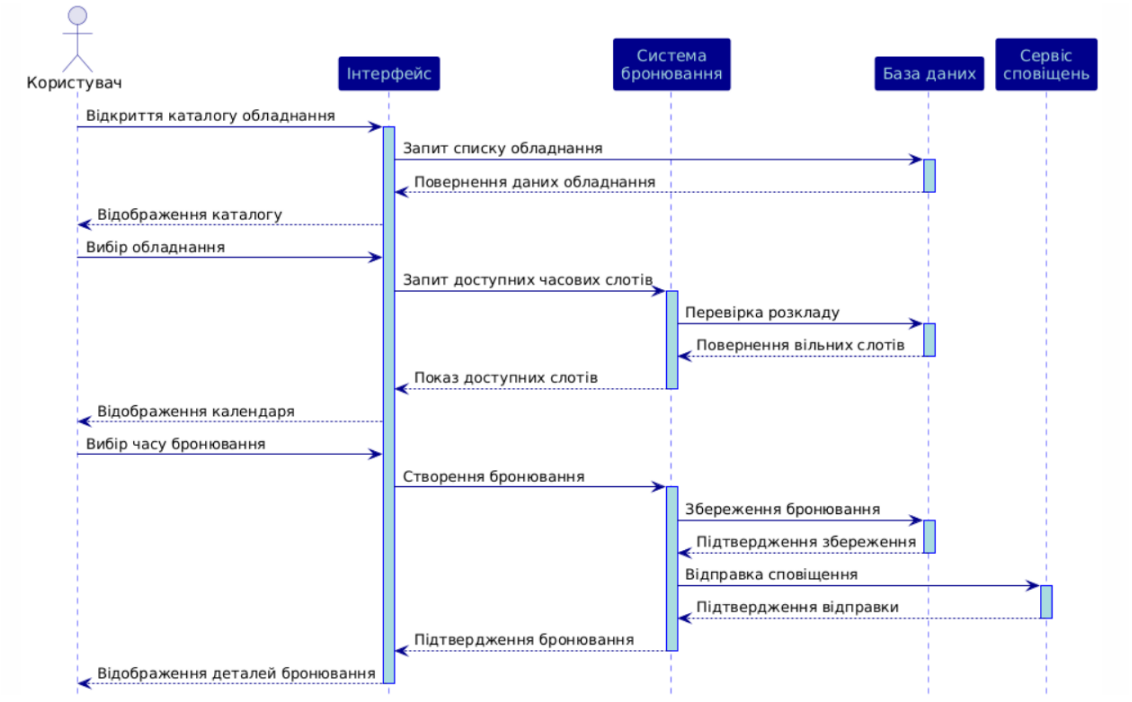

Рисунок Г.4 – Діаграма послідовності бронювання спортивного інвентарю

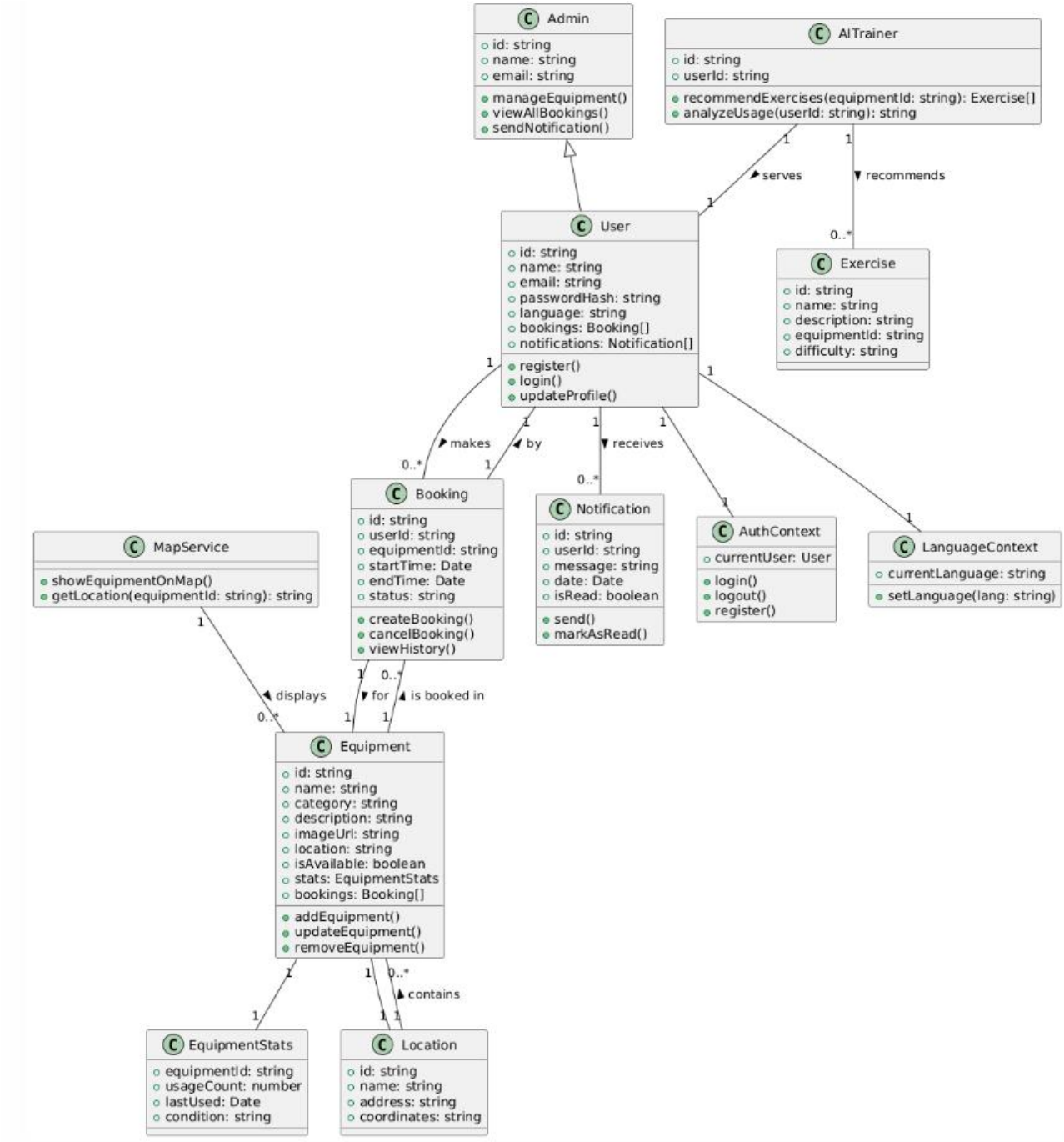

Рисунок Г.5 – Діаграма класів

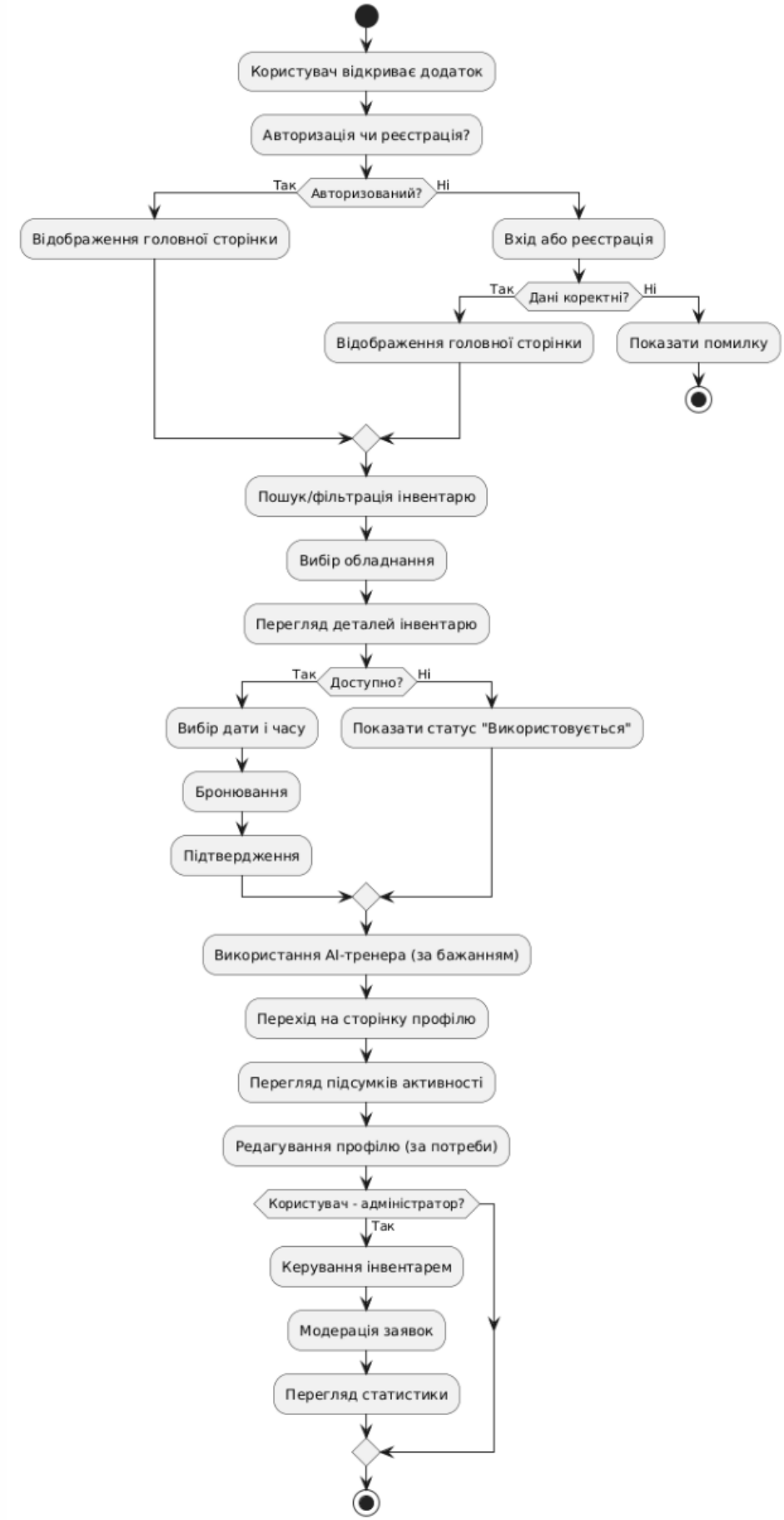

Рисунок Г.6 – Блок-схема алгоритму роботи додатку

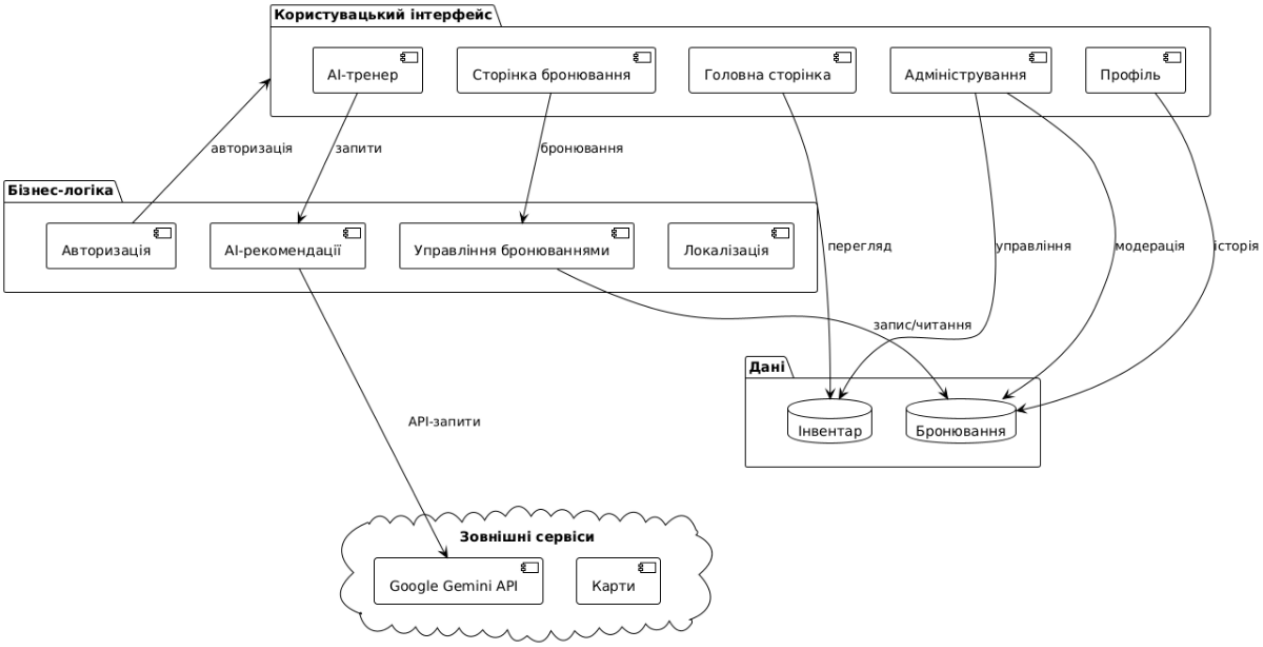

Рисунок Г.7 – Структурна схема функціонування додатку

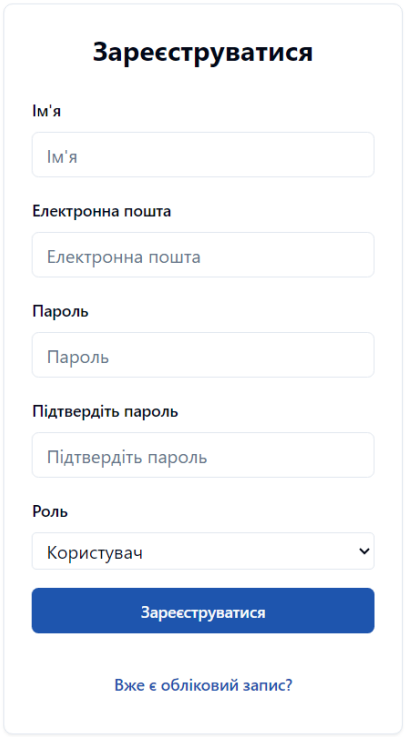

Рисунок Г8 – Вікно реєстрації

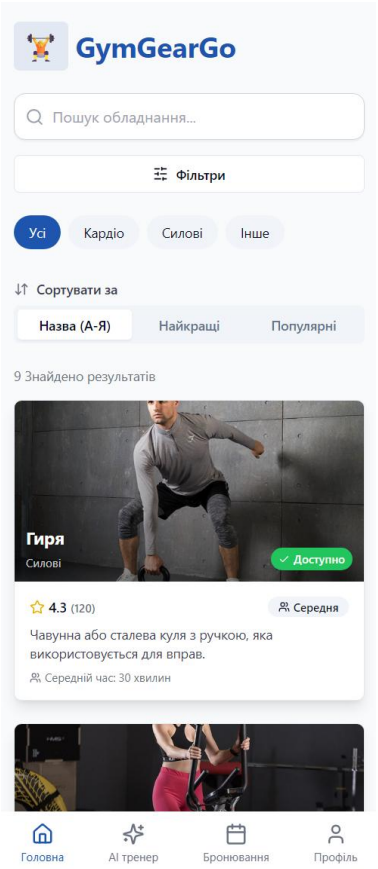

Рисунок Г9 – Головна сторінка з відображенням обладнанням

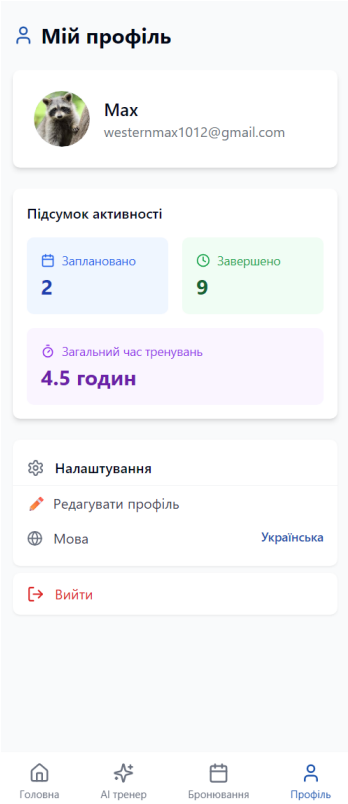

Рисунок Г10 – Вікно профілю користувача

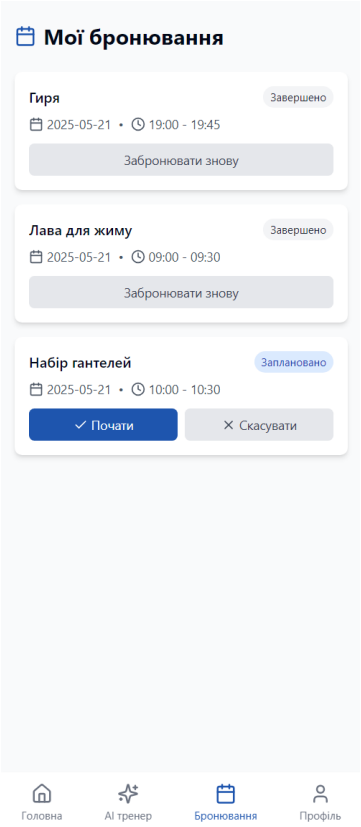

Рисунок Г11 – Вікно бронювання

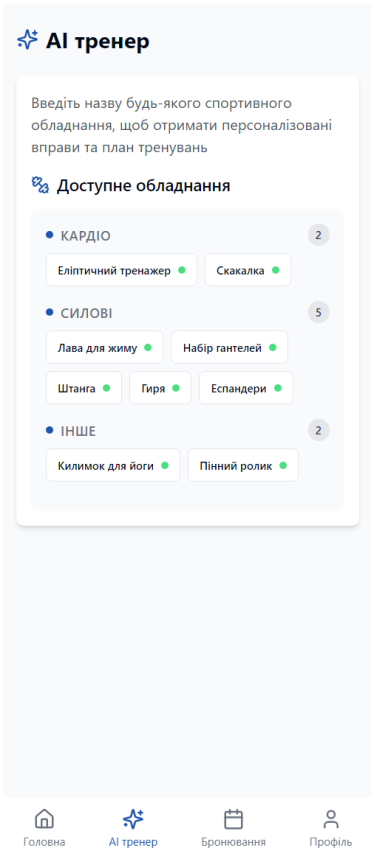

Рисунок Г12 – Вікно AI-тренера

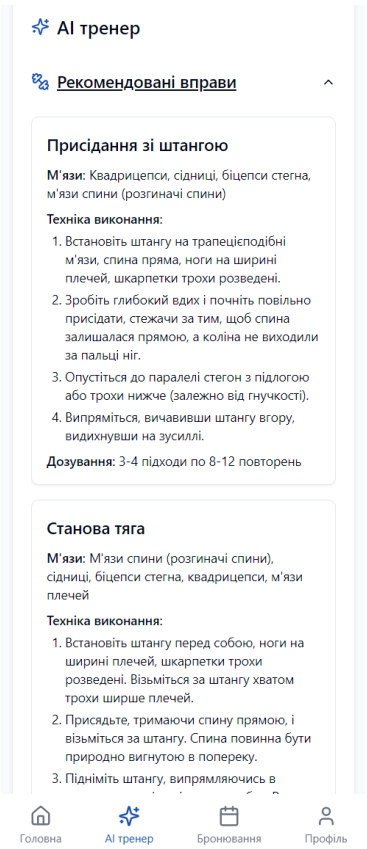

Рисунок Г13 – Вигляд згенерованих вправ за допомогою Al-тренера

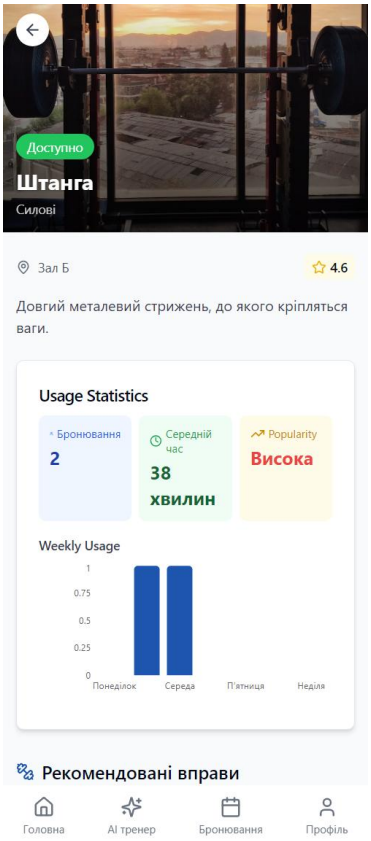

Рисунок Г14 – Вікно деталей обладнання