НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ
Факультет інформаційних технологій

ПОГОДЖЕНО	**ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ**
Декан факультету	**Завідувач кафедри комп'ютерних наук**
_____ Ігор БОЛБОТ	**_____ Белла ГОЛУБ**
«___» _____ 2026 р.	**«___» _____ 2026 р.**

БАКАЛАВРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему «Програмне забезпечення інформаційної системи для фітнес-клубу»
Спеціальність – «121 Інженерія програмного забезпечення»
Освітньо-професійна програма «Інженерія програмного забезпечення»

Гарант освітньої програми
к.т.н., доцент _____ **Ганна ВАЙГАНГ**
(підпис)

Керівник бакалаврської кваліфікаційної роботи
_____ **Юрій МІЛОВІДОВ**
(підпис)

Виконав _____ **Василь СУХЕЦЬКИЙ**
(підпис)

Київ 2026

НАЦІОНАЛЬНИЙ УНІВЕРСТИТЕТ БІОРЕСУРСІВ
І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ
Факультет інформаційних технологій

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних наук
к.т.н., доцент _____ Белла ГОЛУБ
«___» _____ 2025 р.

ЗАВДАННЯ
ДО ВИКОНАННЯ БАКАЛАВРСЬКОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ ЗДОБУВАЧУ

Сухецькому Василю Геннадійовичу
(прізвище, ім'я, по батькові)

Спеціальність «121 Інженерія програмного забезпечення»
Освітньо-професійна програма «Інженерія програмного забезпечення»
Тема бакалаврської кваліфікаційної роботи
«Програмне забезпечення інформаційної системи для фітнес-клубу»

затверджена наказом ректора від «19» 12 2025р. № 3204 «С»

Термін подання завершеної роботи на кафедру «___»_____2026р.

Вихідні дані до бакалаврської кваліфікаційної роботи:

_____

Перелік питань, що підлягають дослідженню:

Аналіз предметної області та розробка програмного забезпечення інформаційної системи для фітнес-клубу.

Перелік графічних документів (за необхідності).

_____

_____

Дата видачі завдання «___»_____2026р.

Керівник бакалаврської кваліфікаційної роботи _____ (підпис) **Юрій МІЛОВІДОВ**

Завдання взяв до виконання _____ (підпис) **Василь СУХЕЦЬКИЙ**

ЗМІСТ

ВСТУП

Сучасний ритм життя та зростаюча увага суспільства до здорового способу життя сприяють стрімкому розвитку фітнес-індустрії. Фітнес-клуби стали невід'ємною частиною повсякденності багатьох людей, пропонуючи широкий спектр послуг: групові та індивідуальні заняття, персональні тренування, продаж абонементів, контроль відвідувань, облік фінансових операцій та управління розкладом тренерів. Збільшення кількості клієнтів і розширення асортименту послуг призводять до ускладнення бізнес-процесів, що робить ручне управління неефективним і схильним до помилок.

Автоматизація управлінських процесів за допомогою спеціалізованого програмного забезпечення дозволяє оптимізувати роботу фітнес-клубу, підвищити якість обслуговування клієнтів, скоротити витрати часу адміністраторів і тренерів, а також забезпечити точний облік даних. Сучасні інформаційні системи для фітнес-клубів повинні підтримувати функції реєстрації клієнтів, управління абонементами, планування занять, моніторинг відвідувань, формування звітності та інтеграцію з платіжними системами. Водночас багато існуючих комерційних рішень є дорогими, складними в впровадженні або не повністю відповідають потребам невеликих і середніх фітнес-клубів України.

Актуальність теми бакалаврської роботи зумовлена необхідністю створення доступного, функціонального та адаптованого програмного забезпечення, яке враховувало б специфіку роботи вітчизняних фітнес-клубів і могло бути легко впроваджене без значних фінансових витрат. Розробка десктопного додатка на базі мови програмування C# з використанням технології Windows Forms забезпечує стабільність, зручність інтерфейсу та можливість роботи в локальній мережі закладу без обов'язкової залежності від хмарних сервісів.

Метою роботи є розробка програмного забезпечення інформаційної системи для управління діяльністю фітнес-клубу з використанням мови програмування C# та технології Windows Forms.

Для досягнення поставленої мети визначено такі завдання:

- провести аналіз предметної області та основних бізнес-процесів фітнес-клубу;
- дослідити існуючі аналоги інформаційних систем і визначити їх переваги та недоліки;
- сформулювати функціональні та нефункціональні вимоги до розроблюваного програмного забезпечення;
- спроєктувати архітектуру системи, структуру бази даних та інтерфейс користувача;
- реалізувати основні модулі системи: управління клієнтами, абонементами, розкладом занять, обліком відвідувань і фінансовими операціями;
- провести тестування розробленого програмного забезпечення та оцінити його ефективність.

Об’єктом дослідження є інформаційні процеси управління діяльністю фітнес-клубу.

Предметом дослідження є програмне забезпечення інформаційної системи для фітнес-клубу, реалізоване на мові C# з використанням Windows Forms.

У роботі застосовувалися методи аналізу предметної області, порівняльного аналізу існуючих рішень, системного проєктування, об’єктно-орієнтованого програмування та тестування програмного забезпечення.

Практична цінність роботи полягає в створенні готового до використання десктопного додатка, який може бути впроваджений у реальних фітнес-клубах для автоматизації ключових процесів. Розроблене програмне забезпечення є гнучким, має інтуїтивно зрозумілий інтерфейс і може бути доопрацьоване під конкретні потреби закладу.

Структура роботи складається з трьох розділів. У першому розділі проведено аналіз предметної області, огляд існуючих рішень і постановку завдання. Другий розділ присвячено проєктуванню системи: вибору технологій, моделюванню бази даних та інтерфейсу. Третій розділ містить опис реалізації, особливостей програмного коду та результатів тестування.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1 Характеристика діяльності сучасних фітнес-клубів

Сучасні фітнес-клуби є складними комерційними організаціями, основною метою яких є надання послуг у сфері фізичного здоров'я, wellness та рекреації. Вони еволюціонували від простих тренажерних залів до багатофункціональних центрів, що поєднують спортивні, соціальні та розважальні аспекти. Основна діяльність фітнес-клубу полягає в організації та проведенні тренувань, продажу абонементів, управлінні клієнтською базою, плануванні розкладу занять та забезпеченні безпеки й комфорту відвідувачів [1].

Фітнес-клуби пропонують широкий спектр послуг, серед яких тренажерні зали з кардіо- та силовим обладнанням, групові програми (йога, пілатес, аеробіка, CrossFit, функціональний тренінг), індивідуальні заняття з персональними тренерами, а також додаткові зони: сауни, басейни, SPA-процедури, дитячі кімнати та кафе здорового харчування. Така диверсифікація дозволяє залучати різні категорії клієнтів – від початківців до професійних спортсменів, від молоді до людей похилого віку [4].

Бізнес-модель сучасного фітнес-клубу базується на комбінації регулярних доходів від абонементів та додаткових джерел: разові відвідування, продаж спортивного харчування, мерчендайзингу, корпоративні програми та партнерства з медичними установами. Основним джерелом доходу залишаються членські внески (абонементи на місяць, квартал чи рік), які забезпечують стабільний cash flow. Успішні клуби фокусуються на утриманні клієнтів (retain rate), адже залучення нового клієнта коштує в 5–7 разів дорожче, ніж збереження існуючого [2].

Ключовим елементом діяльності є персонал. Тренери, адміністратори, менеджери та технічний персонал формують клієнтський досвід. Професійні клуби інвестують у навчання тренерів, впроваджують системи мотивації та контролю якості послуг. Персональні тренери відіграють особливу роль, оскільки

саме вони забезпечують індивідуальний підхід, що є одним із головних факторів лояльності клієнтів [7].

Сучасні тенденції у фітнес-індустрії демонструють перехід до персоналізації послуг. Клуби використовують дані про відвідуваність, вподобання та прогрес клієнтів для створення індивідуальних програм тренувань. Зростає популярність boutique-студій – невеликих спеціалізованих клубів, орієнтованих на один напрямок (наприклад, йога чи бокс), що пропонують преміум-досвід за вищою ціною [5].

Важливим аспектом є технологічна інтеграція в повсякденну діяльність. Багато клубів використовують системи бронювання занять онлайн, мобільні додатки для відстеження прогресу, носимі пристрої для моніторингу активності та CRM-системи для управління клієнтськими відносинами. Такі інструменти дозволяють підвищити ефективність операційної діяльності та покращити комунікацію з клієнтами [9].

Управління розкладом занять є одним із найскладніших процесів. Фітнес-клуб повинен оптимально розподіляти час тренерів, зали та обладнання, враховуючи пікові години (ранок та вечір). Неправильне планування призводить до перевантаження або простою ресурсів, що негативно впливає на дохідність [6].

Фінансовий облік у фітнес-клубах включає контроль надходжень від абонементів, додаткових послуг, продажів у барі чи магазині, а також витрат на оренду, комунальні послуги, зарплати та обладнання. Багато клубів стикаються з проблемою сезонності: пік відвідувань у січні–березні (новорічні обіцянки) та спад у літні місяці. Успішні власники розробляють стратегії утримання клієнтів протягом усього року [3].

Маркетингова діяльність сучасних фітнес-клубів орієнтована на соціальні мережі, партнерства з інфлюенсерами, програми лояльності та реферальні системи. Ефективним інструментом є контент-маркетинг: публікація корисних порад, відео тренувань та історій успіху клієнтів. Особливу увагу приділяють відгукам та рейтингу в Google та соціальних мережах, оскільки репутація є ключовим фактором вибору клубу [8].

У контексті глобальних тенденцій спостерігається зростання уваги до ментального здоров’я та wellness-підходу. Фітнес-клуби дедалі частіше пропонують програми, що поєднують фізичну активність із медитацією, нутриціологічними консультаціями та відновлювальними практиками. Це відповідає зміні потреб клієнтів, які шукають не лише фізичного навантаження, а й комплексного покращення якості життя [4].

В Україні фітнес-індустрія активно розвивається, хоча й відстає від європейських країн за рівнем проникнення (кількістю членів на душу населення). Основними викликами є висока конкуренція, залежність від локації та чутливість до економічних коливань. Успішні клуби адаптуються до локальних особливостей, пропонуючи доступні ціни та гнучкі абонементи [1].

Діяльність сучасного фітнес-клубу є багатогранним процесом, що поєднує спортивні, управлінські, маркетингові та фінансові аспекти. Ефективне функціонування вимагає чіткої організації всіх процесів, орієнтації на клієнта та постійного вдосконалення послуг відповідно до ринкових тенденцій [2].

1.2 Аналіз основних бізнес-процесів фітнес-клубу

Бізнес-процеси фітнес-клубу є сукупністю взаємопов’язаних операцій, спрямованих на забезпечення надання послуг, залучення та утримання клієнтів, а також отримання прибутку. Основні процеси можна класифікувати за функціональними напрямками: управління клієнтами, продаж абонементів і послуг, планування та проведення занять, облік відвідувань, фінансовий облік, маркетинг та управління персоналом. Автоматизація цих процесів за допомогою спеціалізованого програмного забезпечення значно підвищує ефективність роботи закладу [19].

Одним із ключових бізнес-процесів є реєстрація та управління клієнтською базою. Цей процес включає збір персональних даних клієнта (ПІБ, контактна інформація, медичні показання, цілі тренувань), створення профілю, укладання договору та видачу клубної картки. Ефективне управління клієнтами передбачає сегментацію бази за типами абонементів, рівнем активності та історією

відвідувань, що дозволяє персоналізувати пропозиції та підвищувати лояльність [22].

Процес продажу абонементів і додаткових послуг є основним джерелом доходу фітнес-клубу. Він охоплює консультацію клієнта, вибір типу абонемента (місячний, річний, сімейний, корпоративний), оплату (готівкою, карткою або онлайн), активацію абонемента та контроль терміну дії. Додаткові послуги включають персональні тренування, продаж спортивного харчування, оренду шафок чи участь у спеціальних програмах. Сучасні системи дозволяють автоматизувати продовження абонементів і нагадування про закінчення терміну, що сприяє утриманню клієнтів [20].

Планування розкладу занять і тренерів є складним процесом, що вимагає узгодження доступності залів, обладнання та тренерів з побажаннями клієнтів. Процес включає створення графіку групових занять, бронювання місць, призначення тренерів на заняття та обробку змін (скасування, заміна). Оптимізація розкладу дозволяє уникнути конфліктів і максимально завантажити ресурси клубу, особливо в пікові години [21].

Облік відвідувань є критичним для контролю доступу та моніторингу активності клієнтів. Процес передбачає фіксацію входу/виходу за допомогою клубної картки, QR-коду чи біометрії, перевірку дійсності абонемента, автоматичне списання візитів (для разових або лімітованих абонементів) та формування звітів про відвідуваність. Такий облік допомагає виявляти неактивних клієнтів і вчасно пропонувати їм спеціальні пропозиції для повернення [26].

Фінансовий облік охоплює всі грошові операції: надходження від продажу абонементів і послуг, витрати на зарплати, оренду, обладнання та комунальні послуги. Процес включає формування касових ордерів, інтеграцію з платіжними системами, генерацію фінансової звітності (доходи, витрати, прибуток) та податковий облік. Автоматизовані системи забезпечують прозорість і зменшують ризик помилок [23].

Маркетинговий процес спрямований на залучення нових клієнтів і утримання існуючих. Він включає аналіз цільової аудиторії, проведення промо-акцій (безкоштовні пробні заняття, знижки для друзів), email- та SMS-розсилки, управління програмами лояльності та роботу з відгуками. Ефективний маркетинг базується на даних про клієнтів, що дозволяє персоналізувати комунікацію [27].

Управління персоналом охоплює підбір, навчання та мотивацію тренерів і адміністраторів. Процес включає ведення графіку роботи, облік зарплат, оцінку ефективності тренерів за кількістю клієнтів і відгуками, а також організацію внутрішнього навчання. Якість персоналу безпосередньо впливає на задоволеність клієнтів і репутацію клубу [19].

Додатковими процесами є управління обладнанням і приміщеннями (технічне обслуговування, прибирання) та обробка скарг і пропозицій клієнтів. Усі бізнес-процеси тісно пов’язані: наприклад, дані про відвідуваність впливають на фінансову звітність і маркетингові кампанії [20].

Аналіз бізнес-процесів показує, що ручне управління призводить до значних втрат часу, помилок і зниження якості обслуговування. Автоматизація за допомогою спеціалізованого програмного забезпечення дозволяє інтегрувати всі процеси в єдину систему, забезпечуючи швидкий доступ до даних, автоматичне формування звітів і підвищення загальної ефективності роботи фітнес-клубу [21].

1.3 Огляд існуючих інформаційних систем для фітнес-клубів

На сучасному ринку існує велика кількість програмного забезпечення для управління фітнес-клубами, яке охоплює функції бронювання занять, управління клієнтами, обліку відвідувань, фінансових операцій та звітності. Такі системи поділяються на хмарні (SaaS) та локальні рішення, комерційні та з відкритим кодом, а також спеціалізовані для великих мереж або невеликих студій. Більшість сучасних систем пропонують мобільні додатки для клієнтів і адміністраторів, інтеграцію з платіжними системами та аналітику даних [19].

Однією з популярних систем є GymMaster, всеосяжне хмарне рішення, яке включає автоматизацію бронювання, контроль доступу через турнікети, управління абонементами, автоматичні нагадування та детальну звітність.

Система підтримує самообслуговування клієнтів через портал, де відвідувачі можуть бронювати заняття, переглядати розклад і оплачувати послуги. GymMaster відзначається простотою впровадження та гнучкістю налаштувань для клубів різного розміру [28].

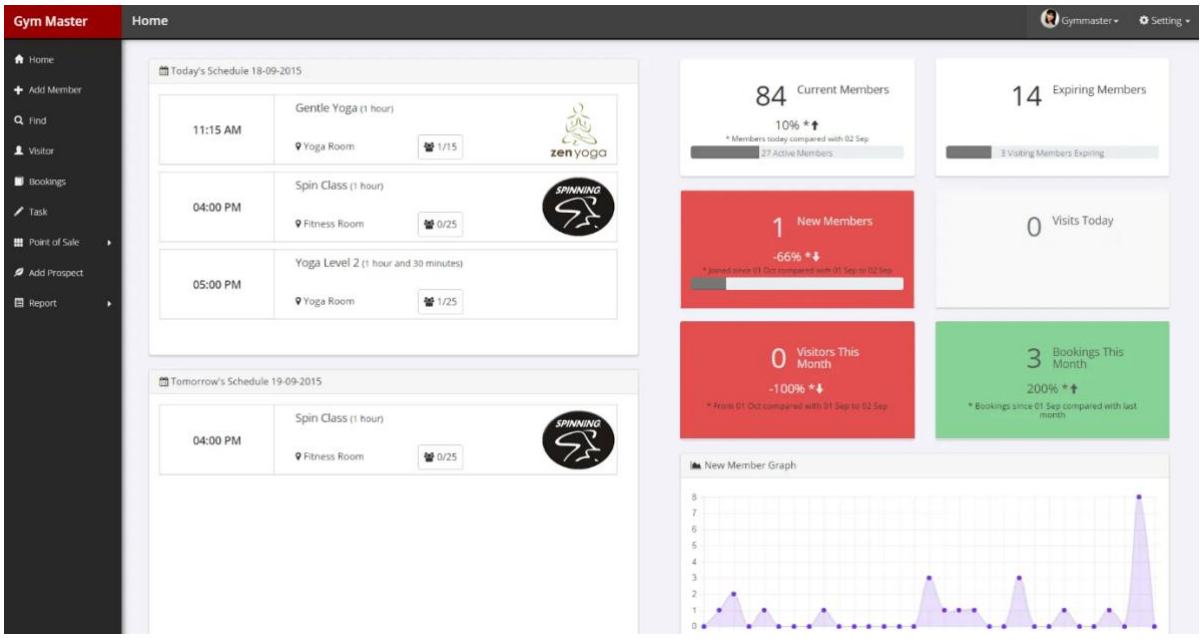

Рис. 1.1 Головний дашборд системи GymMaster

Інша відома система – WodGuru, орієнтована на бутикові студії та кросфіт-зали. Вона пропонує інтуїтивно зрозумілий інтерфейс для бронювання групових занять, управління розкладом тренерів, автоматичне списання візитів та інтеграцію з платіжними системами. Особливістю є акцент на утриманні клієнтів через персоналізовані push-повідомлення та програми лояльності. WodGuru хвалять за швидке впровадження та доступну ціну для невеликих клубів [22].

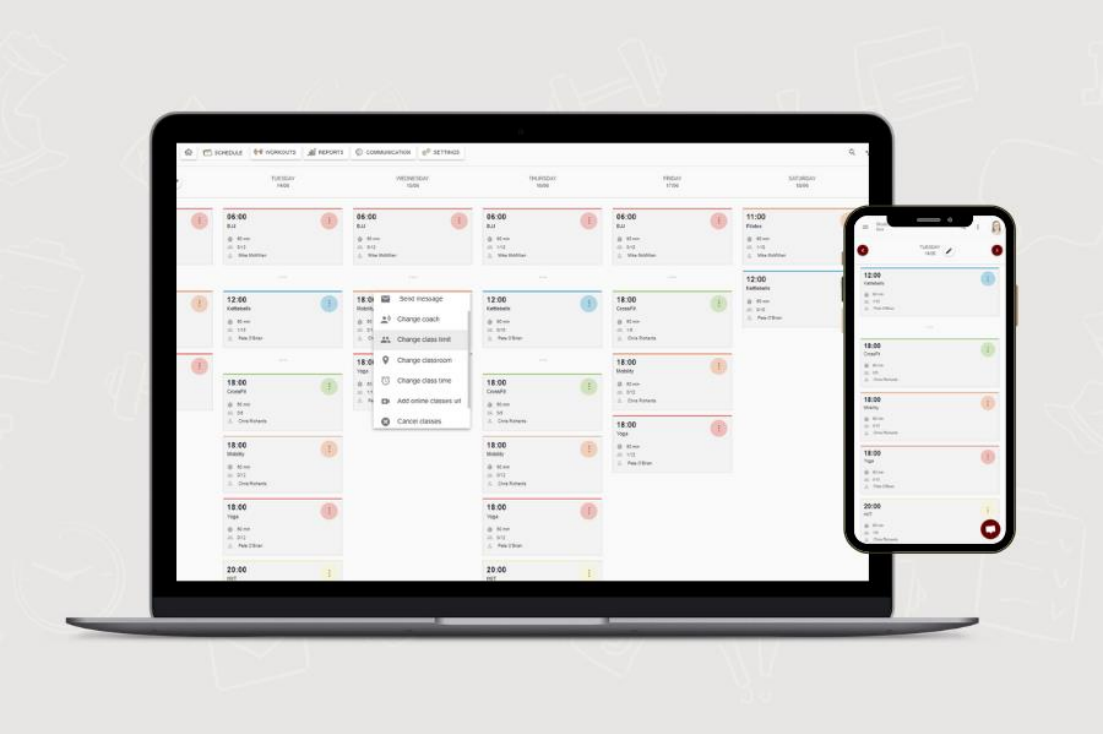

Рис. 1.2 Модуль бронювання занять у системі WodGuru

Club Automation є потужним рішенням для великих фітнес-мереж і enterprise-операторів. Система включає повний цикл управління: від CRM та білінгу до маркетингових інструментів і аналітики. Вона підтримує інтеграцію з біометричним доступом, автоматизовані кампанії email-розсилок та детальні фінансові звіти. Club Automation відзначається високим рівнем безпеки даних та можливістю масштабування для мереж з кількома філіями [24].

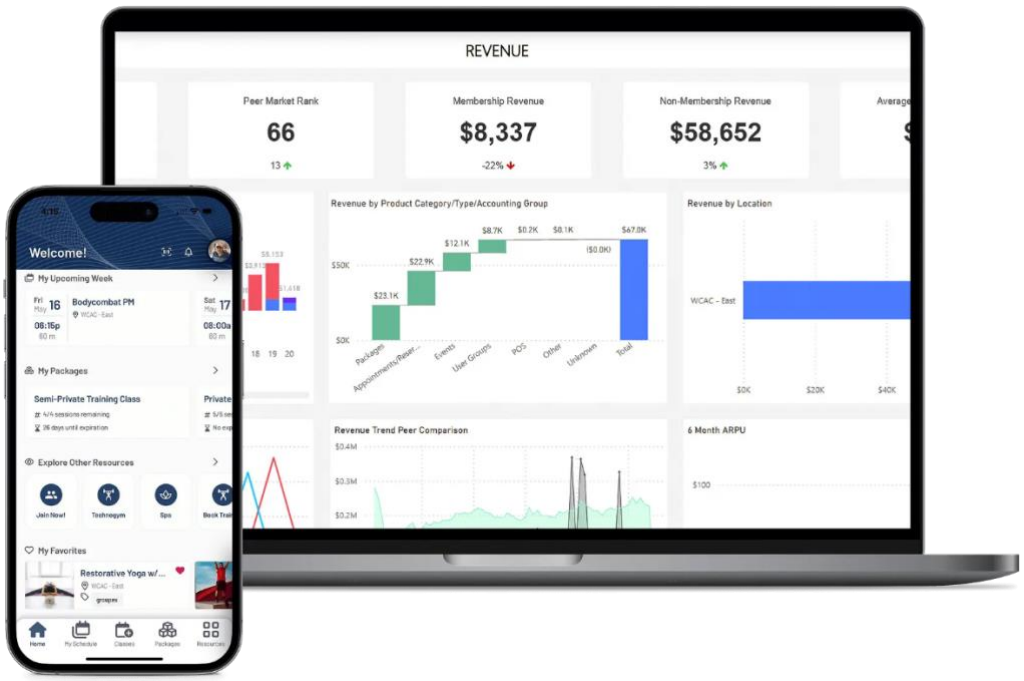

Рис. 1.3 Фінансова звітність у Club Automation

Gymdesk пропонує сучасний підхід до управління, з акцентом на утримання членів клубу. Система автоматизує процеси продовження абонементів, відстежує неактивних клієнтів і пропонує інструменти для реферальних програм. Серед функцій – онлайн-реєстрація, мобільний додаток для клієнтів, управління розкладом та інтеграція з популярними платіжними шлюзами. Gymdesk особливо ефективна для покращення показників retention rate [26].

Xplor Gym (раніше відомий як PerfectGym в деяких регіонах) є комплексною платформою, яка охоплює всі аспекти роботи фітнес-клубу: від продажу абонементів до управління персоналом. Система пропонує модулі для автоматизованого маркетингу, онлайн-бронювання, контролю доступу та генерації звітів. Вона підтримує наявність багатьох філій та інтеграцію з зовнішніми сервісами, що робить її придатною для середніх і великих клубів [27].

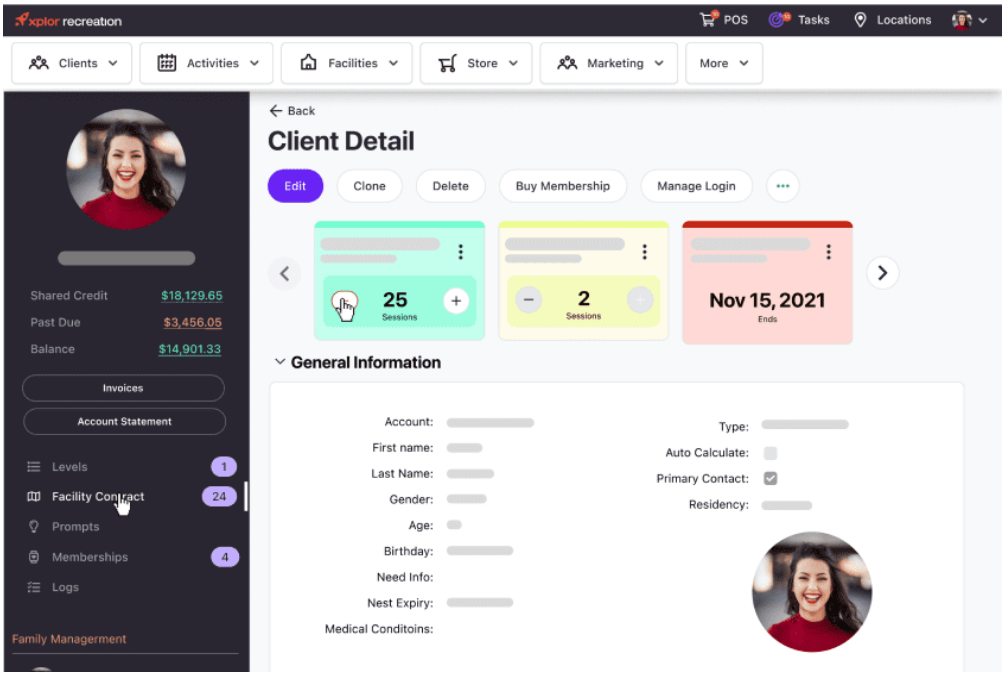

Рис. 1.4 Інтерфейс управління клієнтами в Xplor Gym

Серед безкоштовних або бюджетних рішень варто відзначити відкриті альтернативи та freemium-моделі, такі як ті, що описані в оглядах 2025–2026 років. Деякі системи пропонують базовий функціонал безкоштовно з обмеженнями за кількістю клієнтів, що підходить для стартапів або невеликих студій. Однак безкоштовні версії часто обмежені в аналітиці та інтеграціях [25].

Багато систем, таких як Gymflow чи Booking Ninjas, акцентують на впливі програмного забезпечення на зростання бізнесу: автоматизація зменшує адміністративне навантаження, підвищує задоволеність клієнтів і сприяє збільшенню доходів за рахунок кращого утримання та upsell додаткових послуг [20][23].

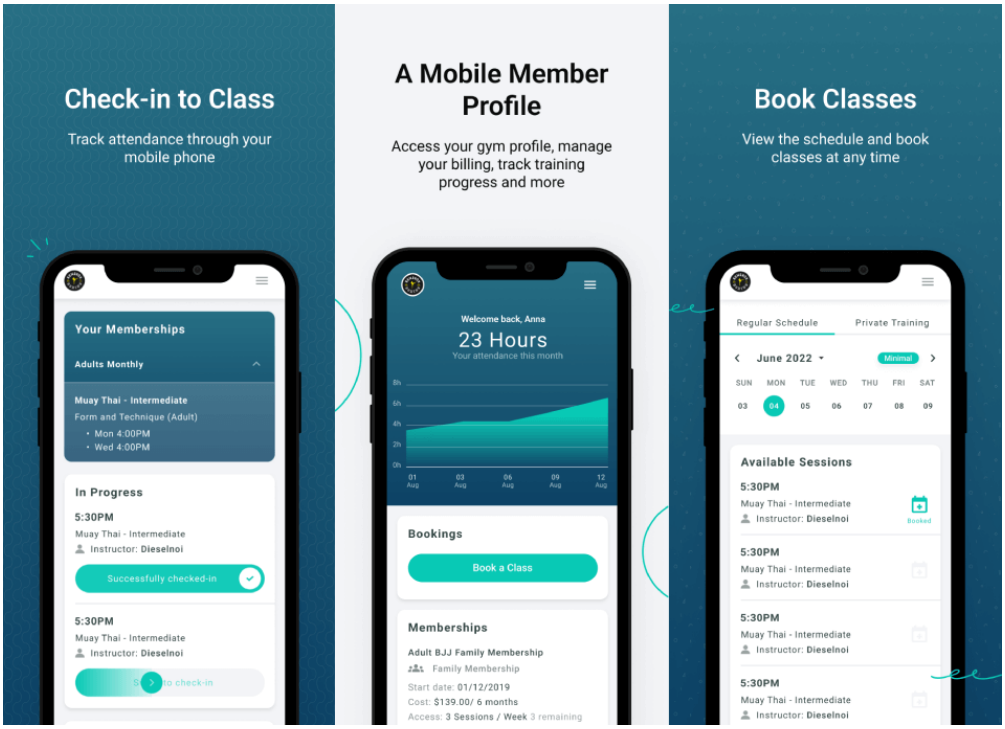

Рис. 1.5 Мобільний додаток клієнта в системі Gymdesk

Порівнюючи системи, можна виділити спільні тенденції: перехід до хмарних технологій, мобільна доступність, автоматизація комунікацій та акцент на даних для маркетингу. Водночас комерційні рішення часто є дорогими в обслуговуванні, вимагають щомісячної абонплати та можуть бути надмірно складними для невеликих українських клубів. Локальні десктопні системи, на відміну від хмарних, забезпечують незалежність від інтернету та нижчі витрати в довгостроковій перспективі, але поступаються в мобільності [19][21].

Існуючі інформаційні системи пропонують широкий спектр функцій, але часто не враховують специфіку локального ринку, мають високу вартість або залежність від постійного інтернет-з’єднання. Це обґрунтовує необхідність розробки доступного десктопного рішення на базі C# та Windows Forms, адаптованого до потреб середніх і невеликих фітнес-клубів України [28].

1.4 Порівняльний аналіз функціональних можливостей аналогічних систем

Порівняльний аналіз функціональних можливостей існуючих інформаційних систем для фітнес-клубів дозволяє виявити їх сильні та слабкі сторони, а також визначити напрямки для розробки нової системи. Розглянуто п'ять популярних рішень: GymMaster, WodGuru, Club Automation, Gymdesk та Xplor Gym. Усі вони є хмарними платформами, що забезпечують широкий спектр функцій, але відрізняються за глибиною реалізації, орієнтацією на певні типи клубів та вартістю [19][20].

GymMaster пропонує комплексний набір інструментів для середніх і великих клубів, включаючи глибоку базу членів, автоматизацію платежів, інтеграцію з системами контролю доступу та детальну звітність. Система добре підходить для 24/7 клубів завдяки власному обладнанню для доступу [28].

WodGuru орієнтований на бутикові студії та кросфіт-зали, акцентуючи на бронюванні занять, брендованому мобільному додатку, автоматизованому білінгу та маркетингових інструментах. Він відзначається простотою інтерфейсу та інструментами для утримання клієнтів [22][21].

Club Automation є enterprise-рішенням для великих мереж, з потужними CRM, маркетинговими кампаніями, аналітикою та інтеграціями. Система забезпечує високий рівень масштабованості та безпеки даних [24].

Gymdesk виділяється інтуїтивним інтерфейсом, автоматизацією лідогенерації та управлінням членами, особливо для martial arts та невеликих бутиків. Вона ефективна для утримання клієнтів завдяки інструментам реферальних програм [20][26].

Xplor Gym (раніше PerfectGym у деяких регіонах) фокусується на преміум-досвіді з красивим мобільним UI, відстеженням шляху клієнта та автоматизованим маркетингом. Підходить для люксових студій, але має обмежену кастомізацію [27].

Таблиця 1.1 – Порівняльна характеристика функціональних можливостей інформаційних систем для фітнес-клубів

Функціональна можливість	GymMaster	WodGuru	Club Automation	Gymdesk	Xplor Gym	Розроблена система
Управління клієнтами (CRM, профілі)	Так	Так	Так	Так	Так	Так
Управління абонементами та білінгом	Так	Так	Так	Так	Так	Так
Планування та онлайн-бронювання занять	Так	Так	Так	Так	Так	Ні
Облік відвідувань та check-in	Так	Так	Так	Так	Так	Так
Фінансова звітність та аналітика	Так	Так	Так	Так	Так	Так
Маркетингові інструменти (email/SMS)	Так	Так	Так	Так	Так	Частково
Брендований мобільний додаток	Так	Так	Так	Так	Так	Ні

для клієнтів						
Контроль доступу (door access integration)	Так	Так	Так	Частково	Частково	Частково (план)
Автоматизація процесів (нагадування, тощо)	Так	Так	Так	Так	Так	Так
Тип розгортання	Хмарне	Хмарне	Хмарне	Хмарне	Хмарне	Локальне (десктоп)
Самообслуговування клієнтів (портал)	Так	Так	Так	Так	Так	Ні

Аналіз таблиці показує, що всі розглянуті системи мають повний набір базових функцій для управління клієнтами, абонементами, розкладом та фінансами. Вони переважно хмарні, що забезпечує мобільний доступ, онлайн-бронювання та інтеграції, але вимагає постійного інтернет-з’єднання та щомісячної абонплати [19][25].

Перевагою комерційних рішень є розвинені маркетингові інструменти та мобільні додатки, які сприяють утриманню клієнтів та самообслуговуванню. Однак для невеликих і середніх фітнес-клубів України такі системи можуть бути надмірно дорогими, складними у впровадженні та залежними від іноземних серверів [23][27].

Розроблена система на базі C# з Windows Forms є локальним десктопним додатком, що не вимагає інтернету для базових операцій, має нижчу вартість

впровадження та забезпечує повну незалежність даних. Вона реалізує ключові функції (управління клієнтами, абонементами, розкладом, відвідуваннями та фінансами), але без хмарних елементів, таких як мобільний додаток чи онлайн-портал. Це робить її оптимальною для локального ринку, де стабільність та простота важливіші за розширену мобільність [21][28].

Таким чином, існуючі системи пропонують розвинений функціонал, орієнтований на хмарні технології, тоді як розроблена система заповнює нішу доступних локальних рішень, адаптованих до потреб невеликих фітнес-клубів без значних експлуатаційних витрат [20][24].

1.5 Визначення вимог до програмного забезпечення інформаційної системи

На основі проведеного аналізу предметної області, бізнес-процесів фітнес-клубу та огляду існуючих рішень сформульовано вимоги до розроблюваного програмного забезпечення. Вимоги поділяються на функціональні (що саме система повинна виконувати) та нефункціональні (якісні характеристики системи). Функціональні вимоги визначають ключові модулі та операції, необхідні для автоматизації основних процесів фітнес-клубу, такі як управління клієнтами, абонементами, розкладом, відвідуваннями та фінансами. Вони враховують типові потреби невеликих і середніх клубів України, де потрібна проста, надійна та локальна система без залежності від хмарних сервісів [19][20].

Функціональні вимоги охоплюють підтримку повного циклу роботи з клієнтами: від реєстрації до обліку активності, автоматизацію продажу та контролю абонемів, планування занять, фіксацію відвідувань та формування базової звітності. Система повинна забезпечувати зручний інтерфейс для адміністраторів і тренерів, а також можливість розмежування доступу за ролями (адміністратор, тренер) [21][22].

Таблиця 1.2 – Функціональні вимоги до інформаційної системи

№	Код вимоги	Опис вимоги	Пріоритет
1	FR-01	Реєстрація та управління	Високий

		профілями клієнтів (ПІБ, контакти, фото, медичні дані)	
2	FR-02	Управління типами абонементів (створення, редагування, контроль терміну дії)	Високий
3	FR-03	Продаж абонементів та додаткових послуг з фіксацією платежів	Високий
4	FR-04	Планування розкладу групових та індивідуальних занять	Високий
5	FR-05	Бронювання місць на заняття та призначення тренерів	Середній
6	FR-06	Облік відвідувань (реєстрація входу/виходу, списання візитів)	Високий
7	FR-07	Формування звітів (відвідуваність, фінансові показники, активність клієнтів)	Середній
8	FR-08	Розмежування прав доступу (адміністратор, тренер)	Високий
9	FR-09	Пошук та	Середній

		фільтрація даних по клієнтам, абонементам, заняттям	

Нефункціональні вимоги визначають якісні характеристики системи, такі як зручність використання, продуктивність, надійність та безпека. Оскільки система реалізується як десктопний додаток на базі Windows Forms, особлива увага приділяється інтуїтивно зрозумілому графічному інтерфейсу, швидкій роботі з локальною базою даних та підтримкою офлайн-режиму. Технологія Windows Forms забезпечує стабільність, простоту розробки та сумісність з операційними системами Windows, що є поширеними в українських фітнес-клубах [29][30].

Система повинна бути простою у впровадженні, не вимагати спеціального обладнання та забезпечувати захист даних клієнтів відповідно до базових стандартів конфіденційності. Продуктивність має дозволяти роботу з базою даних до 5000 клієнтів без значних затримок [31].

Таблиця 1.3 – Нефункціональні вимоги до інформаційної системи

№	Код вимоги	Опис вимоги	Критерій відповідності
1	NFR-01	Зручність інтерфейсу (інтуїтивно зрозумілий GUI)	**Кількість кліків для основних операцій ≤ 3**
2	NFR-02	Продуктивність (швидкість виконання запитів)	Час відповіді на запит < 1 секунди
3	NFR-03	Надійність (захист від збоїв, резервне копіювання)	Авоматичне збереження даних
4	NFR-04	Безпека (захист персональних даних, автентифікація користувачів)	Парольний доступ, шифрування бази
5	NFR-05	Сумісність	Без додаткового

		(робота на Windows 10/11)	ПЗ
6	NFR-06	Масштабованість (підтримка до 5000 клієнтів)	Без деградації продуктивності
7	NFR-07	Портативність (можливість перенесення бази даних)	Експорт/імпорт у стандартних форматах

Сформульовані вимоги стали основою для подальшого проєктування архітектури системи, структури бази даних та реалізації модулів. Вони враховують недоліки існуючих хмарних рішень (залежність від інтернету, висока вартість) та переваги десктопного підходу на C# з Windows Forms, забезпечуючи доступність і ефективність для цільової аудиторії [23][29].

1.6 Висновки до розділу

У першому розділі проведено комплексний аналіз предметної області – діяльності сучасних фітнес-клубів, яка характеризується багатогранністю послуг, орієнтацією на клієнта, сезонністю та необхідністю ефективного управління ресурсами. Сучасні фітнес-клуби поєднують спортивні, рекреаційні та соціальні функції, пропонуючи тренажерні зали, групові та індивідуальні заняття, додаткові сервіси, що вимагає чіткої організації бізнес-процесів для забезпечення прибутковості та утримання клієнтів.

Детальний аналіз основних бізнес-процесів показав їх взаємозв’язаність: від реєстрації клієнтів і продажу абонементів до планування занять, обліку відвідувань, фінансового контролю та маркетингу. Ручне виконання цих процесів є неефективним, схильним до помилок і не дозволяє повною мірою використовувати дані для покращення обслуговування.

Огляд існуючих інформаційних систем виявив велику кількість хмарних рішень (GymMaster, WodGuru, Club Automation, Gymdesk, Xplor Gym), які пропонують широкий функціонал, включаючи онлайн-бронювання, мобільні додатки та автоматизований маркетинг. Порівняльний аналіз підтвердив, що ці

системи добре справляються з базовими завданнями, але мають суттєві недоліки для невеликих і середніх фітнес-клубів України: високу вартість, залежність від постійного інтернет-з’єднання, складність впровадження та надмірну функціональність.

На основі аналізу сформульовано функціональні та нефункціональні вимоги до розроблюваного програмного забезпечення, які охоплюють ключові модулі (управління клієнтами, абонементами, розкладом, відвідуваннями, звітністю) та якісні характеристики (зручність інтерфейсу, продуктивність, безпека, офлайн-доступ). Вибір десктопної архітектури на базі C# та Windows Forms обґрунтовано необхідністю створення доступного, надійного та незалежного від інтернету рішення, адаптованого до локальних потреб.

Аналіз предметної області та існуючих рішень підтвердив актуальність розробки локальної інформаційної системи, яка автоматизує основні бізнес-процеси фітнес-клубу, забезпечуючи простоту використання, низькі витрати на впровадження та повну відповідність визначеним вимогам. Це створює міцну основу для подальшого проєктування та реалізації програмного забезпечення.

РОЗДІЛ 2. ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1 Вибір архітектури та технологічного стеку

Вибір архітектури та технологічного стеку для розробки інформаційної системи фітнес-клубу здійснювався з урахуванням сформульованих у першому розділі вимог: локальний характер системи, офлайн-доступність, простота впровадження, низькі витрати на експлуатацію, підтримка базових функцій управління клієнтами, абонементами, розкладом та фінансами, а також інтуїтивно зрозумілий інтерфейс для користувачів з мінімальними технічними навичками. Система орієнтована на невеликі та середні фітнес-клуби України, де часто відсутня стабільна інфраструктура для хмарних рішень або бажання залежати від щомісячних платежів.

Розглянуто кілька архітектурних підходів: хмарну (SaaS), веб-додаток, мобільний додаток та десктопний. Хмарні рішення відкинуто через залежність від постійного інтернет-з’єднання, ризики безпеки даних та регулярні витрати, що не відповідає потребам цільової аудиторії. Веб-технології (ASP.NET Core з Blazor або Angular) забезпечують кросплатформенність, але вимагають серверної інфраструктури та ускладнюють офлайн-доступ. Мобільні платформи (Xamarin або MAUI) підходять для клієнтського доступу, але не оптимальні для адміністративного інтерфейсу з великою кількістю даних.

Обрано десктопну клієнтську архітектуру з локальною базою даних, що забезпечує повну автономність, швидкий доступ до даних та мінімальні вимоги до обладнання (стандартний ПК з Windows). Система реалізована як однорівневий додаток з елементами тришарової архітектури: шар представлення (інтерфейс користувача), шар бізнес-логіки (обробка даних та правил) та шар доступу до даних (робота з базою). Такий підхід дозволяє чітко розділити відповідальність, спрощує тестування та подальше розширення [14].

Рис. 2.1 Тришарова архітектура розробленої системи

Основною мовою програмування обрано C# – сучасну об’єктно-орієнтовану мову, розроблену Microsoft, яка є частиною платформи .NET. C# відзначається високою продуктивністю, строгим контролем типів, багатою стандартною бібліотекою та потужними засобами для роботи з базами даних і графічним інтерфейсом. Мова підтримує принципи SOLID, асинхронне програмування та LINQ для зручної обробки колекцій, що ідеально підходить для бізнес-логіки системи управління фітнес-клубом [11][13].

Платформою обрано .NET 8 (або .NET Framework для сумісності з Windows Forms), яка забезпечує кросплатформенність, високу продуктивність та регулярні оновлення від Microsoft. .NET надає вбудовані механізми для роботи з базами даних (ADO.NET, Entity Framework Core), серіалізації даних та безпеки [14][15].

Для графічного інтерфейсу користувача обрано Windows Forms – зрілу технологію Microsoft для створення десктопних додатків з багатим набором елементів керування (DataGridView, DateTimePicker, ComboBox тощо). Windows Forms дозволяє швидко розробляти форми з drag-and-drop у Visual Studio, підтримує прив’язку даних (data binding) та забезпечує стабільну роботу на Windows 10/11 без додаткових залежностей. На відміну від WPF, Windows Forms

простіший у освоєнні та потребує менше ресурсів, що важливо для невеликих проектів [29][31].

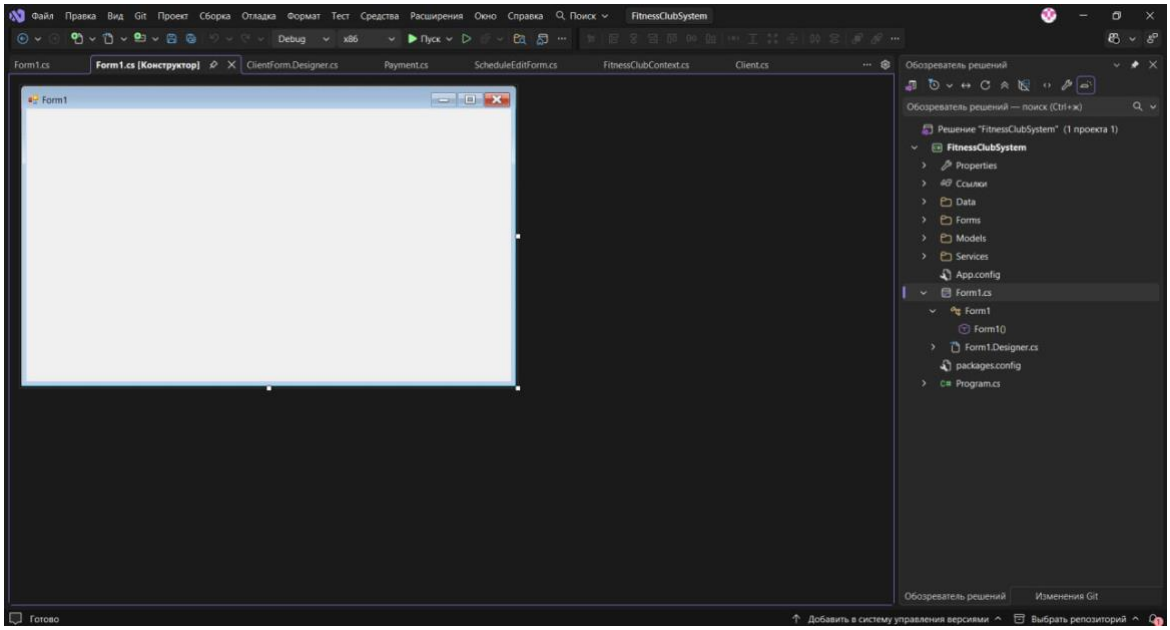

Рис. 2.2 Інтерфейс розробки головної форми у середовищі Visual Studio

Для зберігання даних обрано SQLite з використанням Entity Framework Core як ORM. SQLite є легковажною вбудованою базою даних, що не вимагає окремого сервера, працює з одним файлом та забезпечує повну SQL-сумісність. Це рішення ідеальне для локальних систем: просте резервне копіювання, висока швидкість та відсутність ліцензійних витрат. Entity Framework Core спрощує роботу з даними через код-first підхід, автоматичну міграцію схеми та LINQ-запити [11].

Додаткові бібліотеки: System.Data.SQLite для інтеграції, можливо, Bunifu UI або DevExpress для покращення зовнішнього вигляду форм (опціонально), але базовий стек обмежено стандартними засобами для мінімізації залежностей.

Вибір саме цього стеку обґрунтовано балансом між сучасністю, простотою та ефективністю: C# та .NET забезпечують надійність і продуктивність [16], Windows Forms – швидку розробку інтерфейсу [30], SQLite – автономне зберігання даних. Такий технологічний стек дозволяє створити стабільну, легко розгортаєму систему, яка повністю відповідає вимогам до офлайн-доступності та низької вартості впровадження, на відміну від хмарних аналогів [29].

2.2 Проєктування структури бази даних

Проєктування структури даних здійснювалося з урахуванням вимог до простоти, автономності, швидкості роботи системи та забезпечення цілісності інформації. Для зберігання даних було обрано вбудовану реляційну систему управління базами даних (СУБД) SQLite. Такий вибір обумовлений необхідністю створення повністю портативного додатка, який не вимагає встановлення окремого сервера баз даних, забезпечує високу швидкість виконання запитів та зберігає всі дані в одному локальному файлі з розширенням .db.

Взаємодія додатка з базою даних реалізована за допомогою об'єктно-реляційного відображення (ORM Entity Framework Core) та класу контексту FitnessClubContext. Об'єктна модель базується на сутностях, визначених у папці Models:

- SubscriptionTypes — типи абонементів;
- Clients — клієнти фітнес-клубу;
- ClientSubscriptions — придбані клієнтами абонементи;
- Trainers — тренерський склад;
- Schedules — розклад занять;
- Visits — облік відвідувань;
- Payments — фінансові операції (платежі).

Логічна структура реляційної бази даних була приведена до третьої нормальної форми (3NF) для уникнення надлишковості даних та запобігання аномаліям при модифікації. Зв’язки між таблицями (наприклад, один-до-багатьох між Clients та ClientSubscriptions) реалізовані на рівні бази даних за допомогою первинних (PK) та зовнішніх (FK) ключів. Така структура даних повністю відповідає вимогам до автономності системи та спрощує процес розробки та тестування.

Таблиця 2.1 – Опис таблиці Clients (Клієнти)

Поле	Тип даних	Опис	Обмеження
Id	INTEGER	Первинний ключ	PK, AUTO_INCREMENT

FullName	TEXT	ПІБ клієнта	NOT NULL
Phone	TEXT	Номер телефону	NOT NULL, UNIQUE
Email	TEXT	Електронна пошта	
BirthDate	DATE	Дата народження	
PhotoPath	TEXT	Шлях до фото клієнта	
RegistrationDate	DATE	Дата реєстрації	NOT NULL, DEFAULT CURRENT_DATE
Notes	TEXT	Додаткові примітки (медичні дані тощо)	

Таблиця 2.2 – Опис таблиці Trainers (Тренери)

Поле	Тип даних	Опис	Обмеження
Id	INTEGER	Первинний ключ	PK, AUTO_INCREMENT
FullName	TEXT	ПІБ тренера	NOT NULL
Phone	TEXT	Номер телефону	
Specialization	TEXT	Спеціалізація	
HireDate	DATE	Дата прийому на роботу	

Таблиця 2.3 – Опис таблиці SubscriptionTypes (Типи абонементів)

Поле	Тип даних	Опис	Обмеження
Id	INTEGER	Первинний ключ	PK, AUTO_INCREMENT
Name	TEXT	Назва абонемента (наприклад, «Місячний»)	NOT NULL
Price	REAL	Ціна	NOT NULL
DurationDays	INTEGER	Тривалість у днях	
VisitsLimit	INTEGER	Ліміт візитів (0 – безліміт)	
Description	TEXT	Опис	

Таблиця 2.4 – Опис таблиці ClientSubscriptions (Придбані абонементи)

Поле	Тип даних	Опис	Обмеження
Id	INTEGER	Первинний ключ	PK, AUTO_INCREMENT
ClientId	INTEGER	Посилання на клієнта	FK (Clients.Id), NOT NULL
SubscriptionTypeId	INTEGER	Посилання на тип абонемента	FK (SubscriptionTypes.Id), NOT NULL
PurchaseDate	DATE	Дата покупки	NOT NULL
StartDate	DATE	Дата активації	NOT NULL
EndDate	DATE	Дата закінчення	
RemainingVisits	INTEGER	Залишок візитів	DEFAULT 0
IsActive	BOOLEAN	Статус активності	DEFAULT TRUE

Додаткові таблиці включають Visits (Відвідування), Payments (Платежі) та Schedules (Розклад занять), які забезпечують облік відвідувань, фінансових операцій та планування. Таблиця Visits фіксує дату та час входу/виходу, автоматично списуючи візити з активного абонемента. Таблиця Payments зберігає історію всіх фінансових транзакцій.

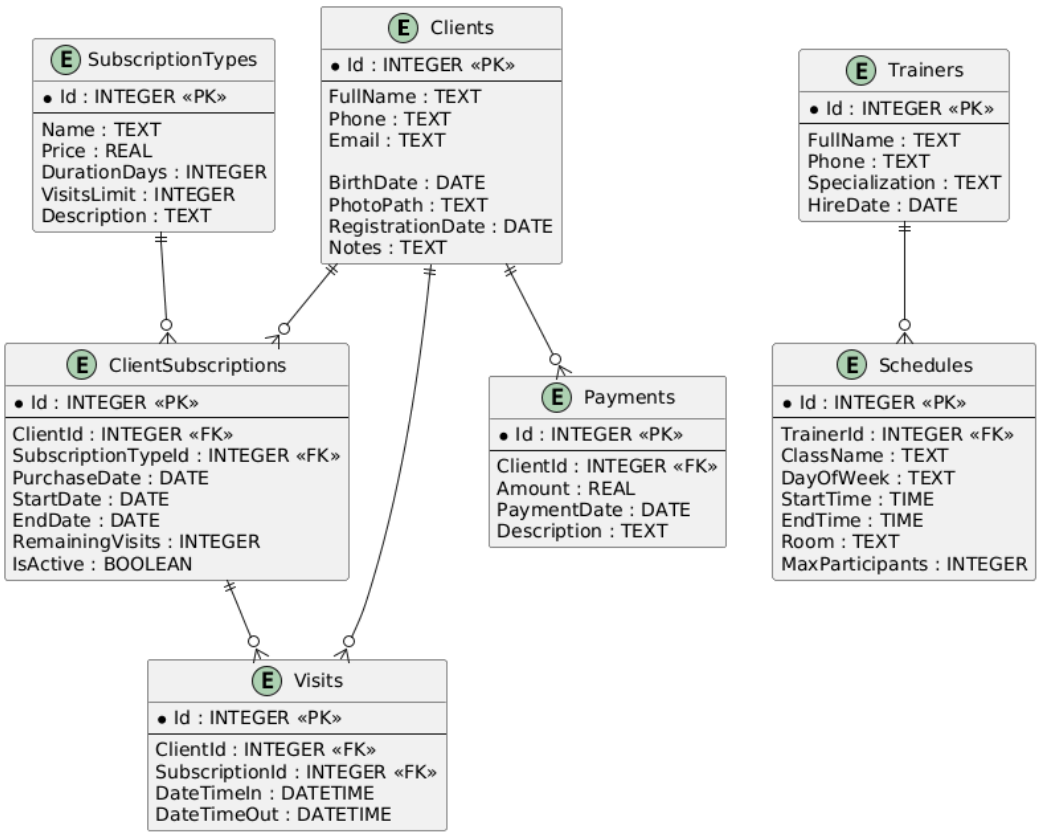

Рис. 2.3 Схема бази даних (ER-діаграма)

Запропонована структура бази даних забезпечує ефективне зберігання та швидкий доступ до даних, підтримку всіх функціональних вимог системи та можливість подальшого розширення (наприклад, додавання модуля бронювання занять).

2.3 Розробка діаграм випадків використання (Use Case Diagram)

Для моделювання функціональності інформаційної системи та взаємодії користувачів з нею використано діаграми випадків використання (Use Case Diagram) відповідно до стандартів UML. Діаграми випадків використання дозволяють візуально представити основні функції системи, акторів (користувачів або зовнішні системи) та їх взаємозв’язки. Це сприяє чіткому розумінню поведінки системи на етапі проєктування та полегшує подальшу реалізацію модулів.

У розробленій системі визначено двох основних акторів: Адміністратор (відповідає за повний цикл управління: реєстрацію клієнтів, продаж абонементів, планування, фінанси та звіти) та Тренер (має обмежений доступ: перегляд розкладу, облік відвідувань на заняттях, перегляд профілів клієнтів). Клієнт не є прямим актором, оскільки система є десктопною і призначена для внутрішнього використання персоналом клубу; самообслуговування клієнтів не передбачено.

Загальна діаграма випадків використання відображає ключові функції системи на високому рівні. Вона включає основні випадки використання, які відповідають сформульованим функціональним вимогам.

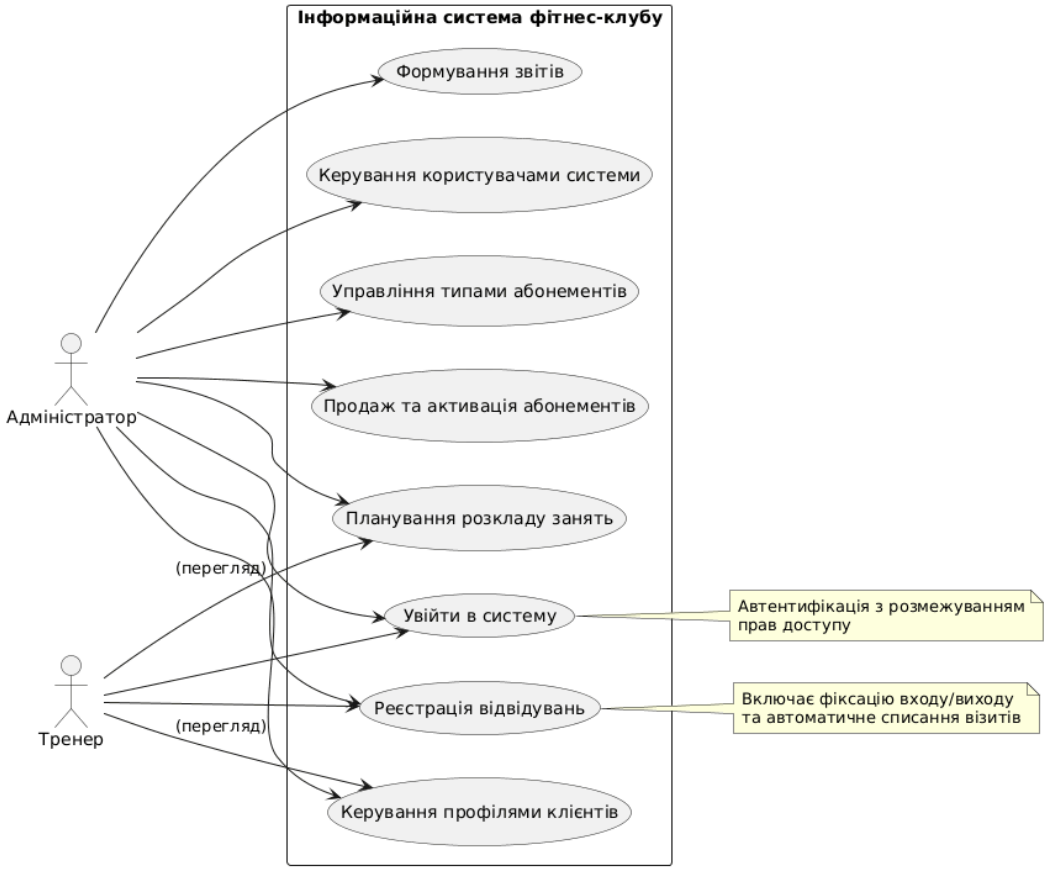

Рис. 2.4 Загальна діаграма випадків використання системи

Випадок використання «Увійти в систему» є обов’язковим для всіх акторів і забезпечує автентифікацію та розмежування прав доступу. Адміністратор має повний доступ до всіх функцій, тоді як тренер обмежений переглядом профілів клієнтів, розкладу та реєстрацією відвідувань на своїх заняттях. Це відповідає нефункціональній вимозі NFR-04 щодо безпеки та розмежування доступу.

Для деталізації окремих підсистем розроблено розширені діаграми. Наприклад, підсистема управління клієнтами та абонементами є центральною, оскільки більшість операцій пов’язані з клієнтською базою.

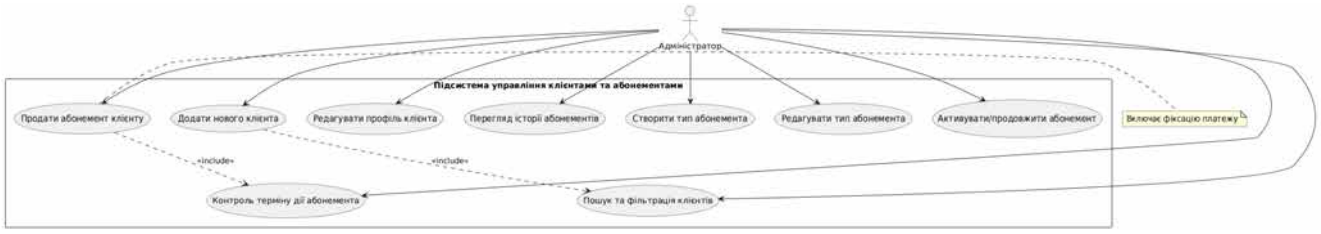

Рис. 2.5 Діаграма випадків використання для підсистеми управління клієнтами та абонементами

Діаграма демонструє, що продаж абонемента обов’язково включає контроль терміну дії, а пошук клієнтів може бути частиною багатьох операцій (відношення <<include>>).

Аналогічно деталізовано підсистему планування та обліку відвідувань, де тренер має обмежену участь.

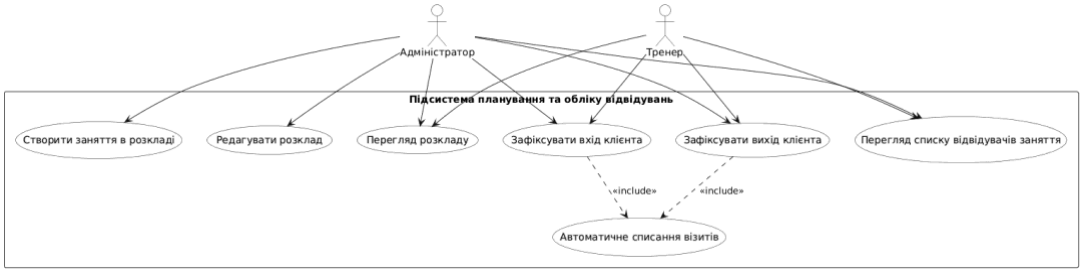

Рис. 2.6 Діаграма випадків використання для підсистеми планування та обліку відвідувань

Розроблені діаграми випадків використання повністю охоплюють функціональні вимоги системи, чітко визначають ролі акторів та взаємозв’язки між випадками використання. Вони слугували основою для проєктування інтерфейсу користувача та реалізації бізнес-логіки в наступних етапах розробки. Використання відношень <<include>> дозволило виділити спільні операції та уникнути дублювання функціональності.

2.4 Проєктування інтерфейсу користувача (Windows Forms)

Проєктування інтерфейсу користувача є одним із найважливіших етапів розробки інформаційної системи, оскільки від його якості безпосередньо залежить ефективність роботи персоналу фітнес-клубу, швидкість освоєння програми та загальний рівень задоволеності користувачів. Зважаючи на те, що основними користувачами системи будуть адміністратори та тренери, які не завжди мають високий рівень технічної підготовки, особлива увага приділялася створенню простого, інтуїтивно зрозумілого та ергономічного інтерфейсу.

При проєктуванні інтерфейсу керувалися такими принципами: максимальна інтуїтивність і мінімальне навчання користувача; послідовність розташування елементів керування; чітка візуальна ієрархія; швидкий доступ до найбільш використовуваних функцій; адаптивність під різні ролі користувачів

(адміністратор / тренер); використання сучасного, але не перевантаженого дизайну.

Для створення графічного інтерфейсу обрано технологію Windows Forms у середовищі розробки Microsoft Visual Studio. Windows Forms забезпечує високу швидкість розробки, багатий набір стандартних елементів керування (DataGridView, Button, TextBox, DateTimePicker, ComboBox тощо), потужні можливості прив’язки даних (Data Binding) та стабільну роботу на операційних системах сімейства Windows [29][30].

Загальна архітектура інтерфейсу базується на головній формі, яка містить верхнє головне меню для зручної навігації між модулями системи. Інтерфейс виконано у класичному стилі Windows Forms із використанням стандартної сіро-білої кольорової палітри, що забезпечує мінімалістичний вигляд, не перевантажує зір користувача та дозволяє зосередитися на роботі з даними. Шрифти: стандартний системний шрифт Segoe UI для максимальної читабельності тексту.

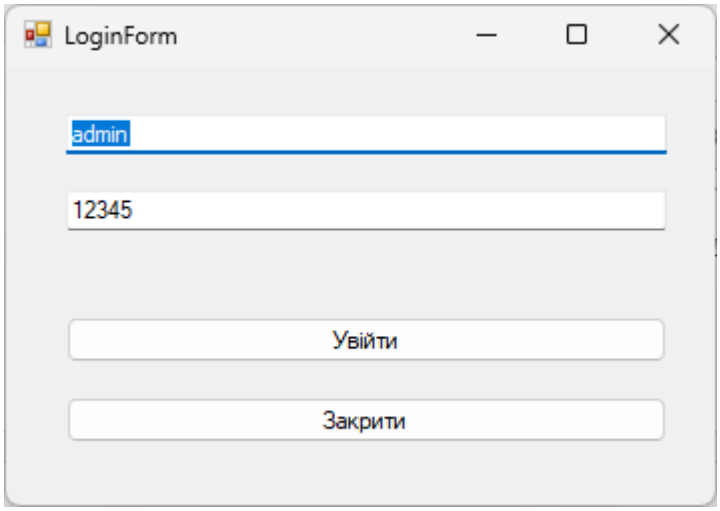

Рис. 2.7 Форма авторизації користувача

Форма авторизації містить поля для введення логіна та пароля користувача, а також керуючі кнопки «Увійти» для підтвердження входу та «Закрити» для виходу з додатка

Рис. 2.8 Головна форма системи (дашборд)

Головна форма системи виступає в ролі базового контейнера додатка. У верхній частині розташоване головне меню розгалуження доступу: «Файл», «Тренери», «Клієнти», «Абонементи», «Розклад», «Відвідування», «Фінанси» та «Звіти». Форма забезпечує централізований запуск усіх підсистем у єдиному робочому просторі адміністратора.

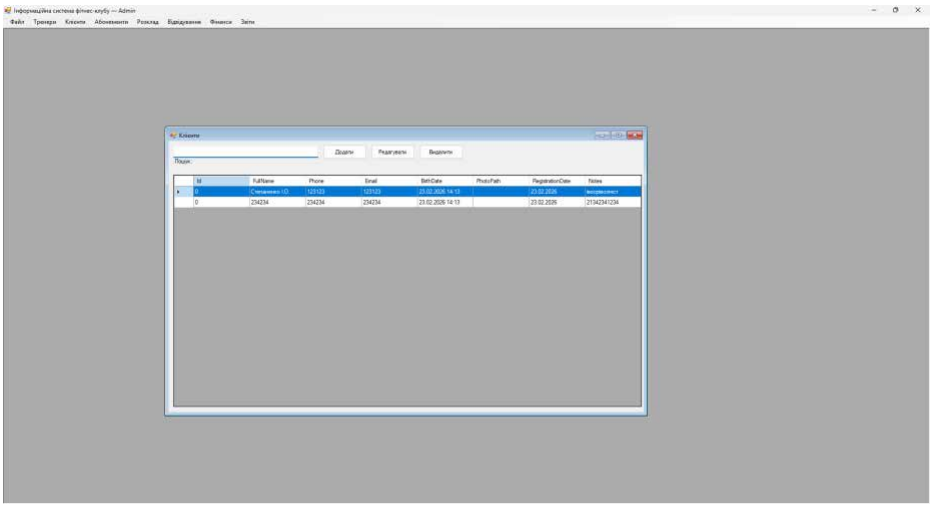

Рис. 2.9 Форма управління клієнтами

Модуль управління клієнтами реалізовано у вигляді окремої форми з DataGridView для відображення списку клієнтів, розширеної панеллю пошуку та фільтрів (за ПІБ, телефоном, статусом абонемента, датою реєстрації). Кнопки «Додати», «Редагувати», «Видалити», «Історія» відкривають відповідні діалогові вікна. Детальна картка клієнта містить вкладки: «Загальна інформація»,

«Абонементи», «Відвідування», «Платежі» та поле для завантаження фото клієнта.

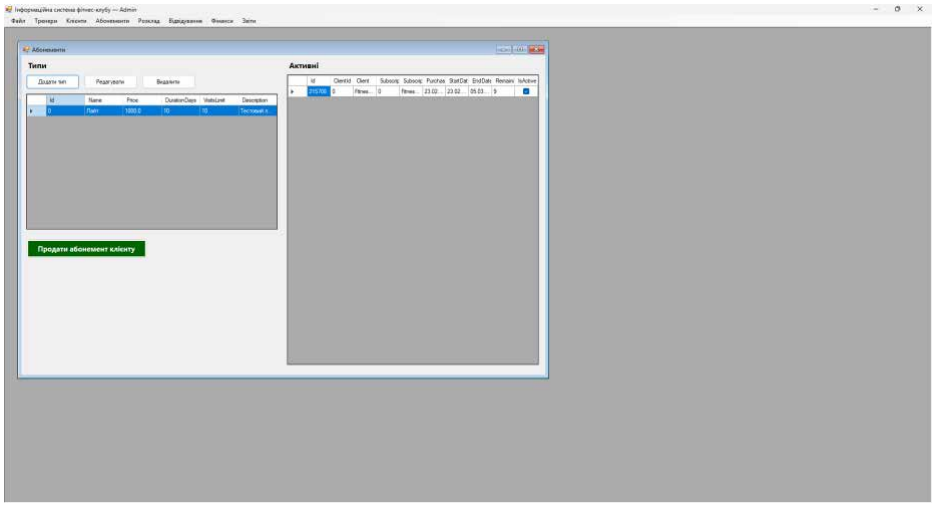

Рис. 2.10 Форма продажу та управління абонементами

Форма продажу абонементів містить два режими роботи: перегляд і редагування типів абонементів та безпосередній продаж клієнту. При виборі типу абонемента система автоматично розраховує кінцеву дату дії та залишок візитів. Форма включає поле пошуку клієнта, ComboBox типів абонементів, DateTimePicker для дати початку та кнопку фіксації платежу з вибором способу оплати.

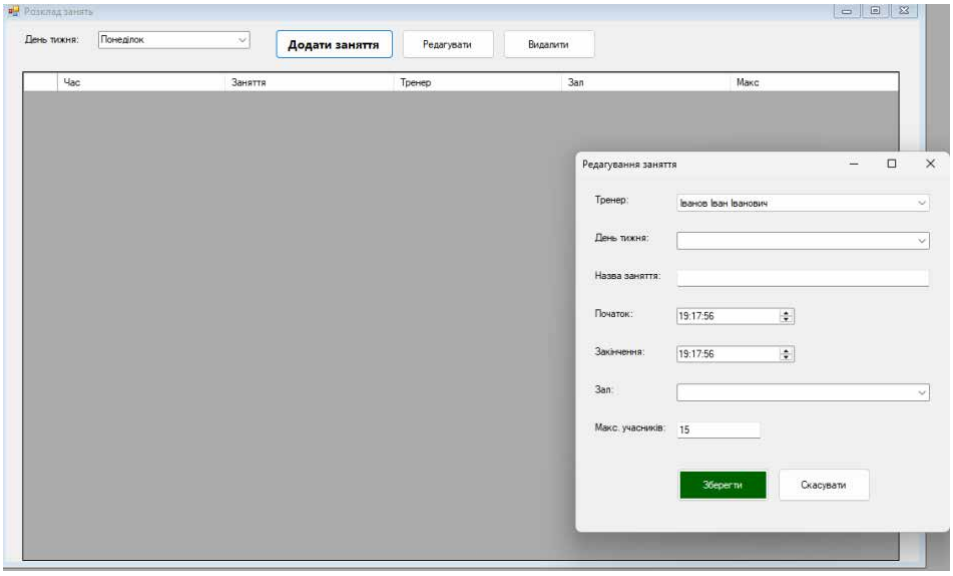

Рис. 2.11 Форма планування розкладу занять

Розклад занять представлений у вигляді таблиці, де відображається час, назва заняття, тренер, зал та ліміт учасників. Додавання та редагування записів здійснюється через допоміжну діалогову форму «Редагування заняття», де адміністратор обирає тренера із випадаючого списку, вказує день тижня, назву, час початку/закінчення занять та максимальну кількість учасників.

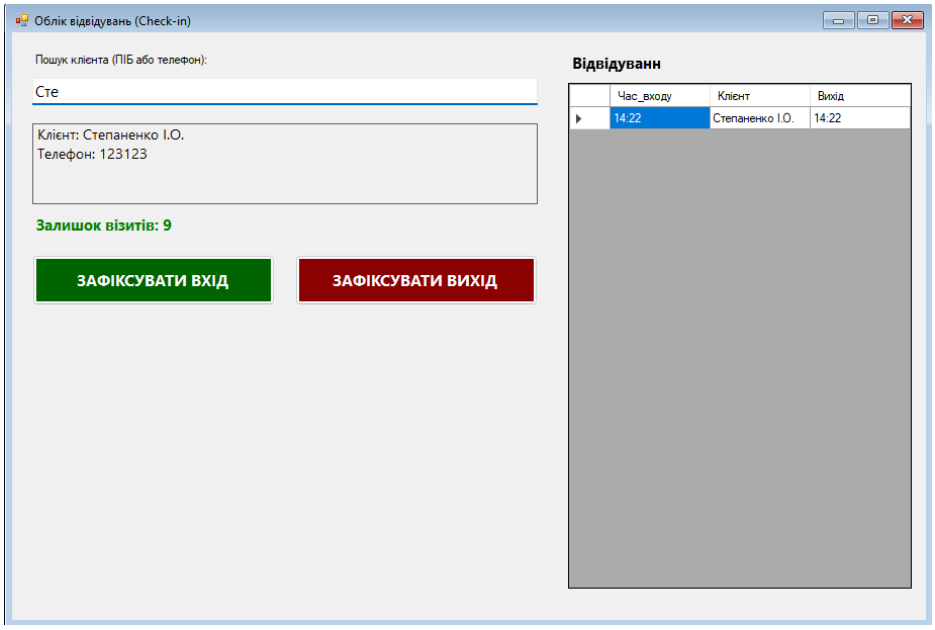

Рис. 2.12 Форма реєстрації відвідувань (Check-in)

Форма обліку відвідувань є однією з найбільш використовуваних. Вона реалізовано з можливістю швидкого пошуку клієнта за ПІБ, номером телефону або скануванням штрих-коду клубної картки. Після пошуку відображається статус абонемента, залишок візитів та кнопки «Зафіксувати вхід» / «Зафіксувати вихід». Система автоматично перевіряє дійсність абонемента та списує візит.

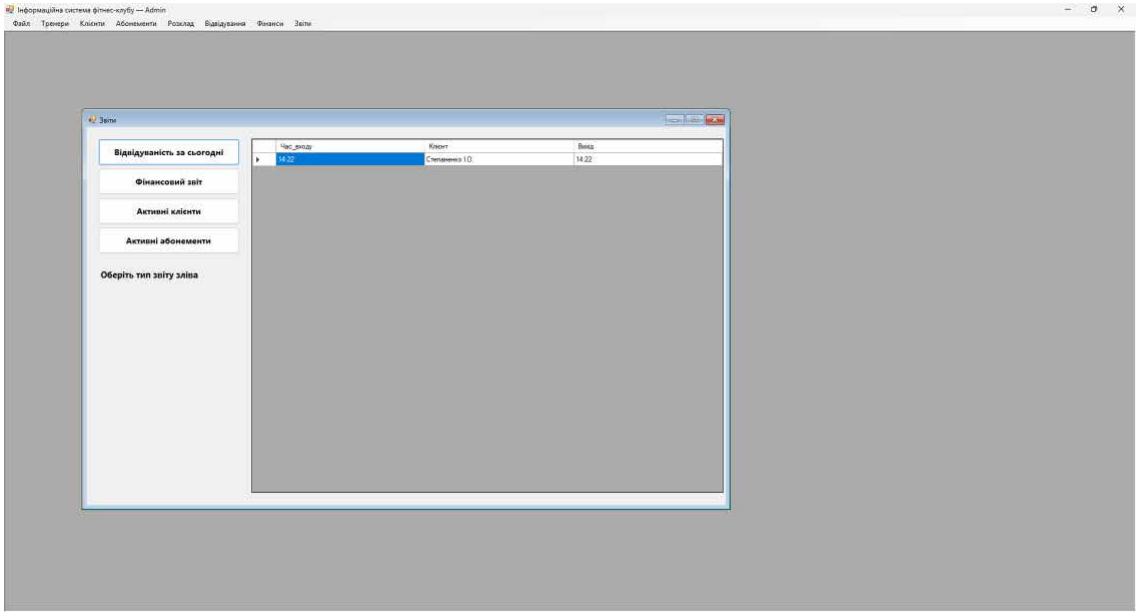

Рис. 2.13 Форма формування звітів

Модуль звітів містить навігаційну панель із кнопками вибору готових шаблонів: «Відвідуваність за сьогодні», «Фінансовий звіт», «Активні клієнти» та «Активні абонементи». При виборі відповідного типу зліва, система автоматично робить вибірку з бази даних та виводить оперативну інформацію в табличному вигляді на екран.

Усі форми мають єдиний стиль оформлення, динамічне розмежування доступу (тренер не бачить фінансові модулі та кнопки видалення), валідацію полів у реальному часі та повідомлення про успішне виконання операцій. Інтерфейс адаптовано під роздільну здатність від 1366×768 пікселів і вище.

Проєктований інтерфейс користувача повністю відповідає сучасним вимогам до ергономіки десктопних додатків, забезпечує високу швидкість виконання операцій, мінімізує ймовірність помилок та відповідає всім сформульованим функціональним і нефункціональним вимогам, зокрема NFR-01 щодо зручності інтерфейсу. Використання технології Windows Forms дозволило створити стабільний, швидкий і візуально приємний інтерфейс, повністю адаптований до специфіки роботи невеликих і середніх фітнес-клубів.

2.5 Опис основних модулів системи

Інформаційна система для фітнес-клубу побудована за модульним принципом. Це забезпечує чітке розділення функціональності, зручність

підтримки та можливість подальшого розширення. Кожен модуль реалізовано як окрему форму, що інтегрована в загальну ієрархію додатка. Взаємодія між модулями та централізований доступ до бази даних SQLite здійснюється через об'єктний контекст FitnessClubContext з використанням технології Entity Framework Core. Нижче наведено детальний опис усіх основних модулів системи.

Модуль автентифікації та управління доступом є точкою входу в систему. Він забезпечує безпечну авторизацію користувачів і розмежування прав доступу залежно від ролі (адміністратор або тренер). Після успішного входу система автоматично визначає права користувача і приховує недоступні елементи меню. Модуль також підтримує створення облікового запису адміністратора за замовчуванням при першому запуску програми.

Основні можливості модуля:

- Авторизація користувачів за логіном і паролем;
- Автоматичне визначення ролі користувача при вході;
- Обмеження доступу до адміністративних функцій;
- Розмежування прав доступу для персоналу клубу.

Модуль управління клієнтами є одним з центральних елементів системи. Він призначений для ведення повної клієнтської бази, включаючи реєстрацію нових клієнтів, редагування їх даних, пошук та перегляд історії. Форма містить зручну таблицю, поле швидкого пошуку та інструменти управління записами. Редагування профілю клієнта відбувається у окремому діалоговому вікні з можливістю завантаження фотографії.

Основні можливості модуля:

- Додавання, редагування та видалення клієнтів;
- Швидкий пошук за ПІБ або номером телефону;
- Зберігання додаткової інформації (дата народження, медичні примітки, фото);
- Перегляд повної історії абонементів і відвідувань клієнта;
- Автоматичне збереження змін у локальній базі даних за допомогою Entity Framework Core.

Модуль управління абонементами відповідає за створення різних типів абонементів та їх продаж клієнтам. Модуль розділений на два блоки: управління типами абонементів та перегляд активних абонементів. При продажу система автоматично розраховує дату закінчення дії та залишок візитів.

Основні можливості модуля:

- Створення та редагування типів абонементів (ціна, термін, ліміт візитів);
- Оформлення та продаж абонемента клієнту з вибором типу;
- Автоматичний розрахунок кінцевої дати дії абонемента;
- Перегляд списку активних абонементів та контролю їхнього статусу.

Модуль планування розкладу занять і тренерів забезпечує зручне формування графіка роботи клубу. Модуль включає два взаємопов’язаних компоненти: управління списком тренерів та безпосереднє планування занять. Форма розкладу дозволяє фільтрувати заняття за днями тижня та виконувати всі необхідні операції.

Основні можливості модуля:

- Ведення списку тренерів клубу та їхніх спеціалізацій;
- Формування та редагування розкладу занять через діалогові форми;
- Відображення дня тижня, часу, залу та ліміту учасників для кожного заняття.

Модуль обліку відвідувань (Check-in) оптимізований для швидкої роботи адміністратора на ресепшні. Форма дозволяє швидко знайти клієнта, перевірити статус його абонемента та зафіксувати вхід або вихід. При фіксації входу відбувається автоматичне списання візиту.

Основні можливості модуля:

- Швидкий пошук клієнта за ПІБ або телефоном;
- Миттєва перевірка дійсності абонемента;
- Фіксація часу входу та виходу;
- Автоматичне списання візитів з активного абонемента;
- Відображення поточної кількості відвідувачів у залі.

Фінансовий модуль забезпечує повний облік грошових операцій клубу. Він відображає ключові фінансові показники в реальному часі та зберігає історію всіх платежів. Додавання нового платежу відбувається через окрему форму.

Основні можливості модуля:

- Фіксація платежів за абонементи та додаткові послуги;
- Перегляд повної історії платежів;
- Автоматичний розрахунок доходу за день, місяць і загальний період;
- Формування простих фінансових звітів;

Модуль звітності та аналітики призначений для формування управлінських звітів. Користувач може швидко отримати інформацію про відвідуваність, активних клієнтів, доходи та активні абонементи. Результати відображаються у зручній таблиці.

Основні можливості модуля:

- Формування звіту про відвідуваність за поточний день;

Фінансовий звіт з групуванням за датами;

- Звіт про активних клієнтів та абонементи;
- Швидке перемикання між типами звітів;
- Можливість копіювання даних у буфер обміну.

Усі модулі системи тісно інтегровані між собою, що забезпечує єдину базу даних і автоматичне оновлення інформації в реальному часі. Модульна архітектура дозволяє легко додавати нові функції без зміни існуючого коду. Завдяки використанню технології Windows Forms і програмному створенню інтерфейсу вдалося досягти високої швидкості роботи, зручності для кінцевого користувача та повної автономності програми.

Описана модульна структура повністю реалізує всі поставлені завдання, забезпечує надійну роботу системи та відповідає вимогам до сучасного програмного забезпечення для фітнес-клубів.

2.6 Діаграма класів системи

Діаграма класів є одним із найважливіших артефактів об’єктно-орієнтованого проєктування. Вона надає повне уявлення про статичну структуру інформаційної системи, демонструє основні класи, їхні атрибути, методи, а також усі типи взаємозв’язків між ними (асоціація, агрегація, композиція та залежність). Діаграма класів дозволяє чітко визначити доменну модель системи, розподілити відповідальність між компонентами та забезпечити правильну логічну організацію коду.

На розробленій діаграмі чітко виділено три логічні пакети:

- Models — містить доменні сутності (основні бізнес-об’єкти);
- Services — сервісний шар, який абстрагує бізнес-логіку та взаємодію з контекстом бази даних;
- Forms — презентаційний шар (інтерфейс користувача Windows Forms).

Центральним елементом архітектури є клас DataManager. Він виступає в ролі єдиної точки доступу (патерн Фасад / Репозиторій), через яку всі графічні форми звертаються до даних. Клас DataManager інкапсулює в собі роботу з об'єктним контекстом FitnessClubContext, здійснюючи завантаження (LoadData<T>) та збереження (SaveData<T>) сутностей у локальну базу даних SQLite за допомогою інструментів Entity Framework Core.

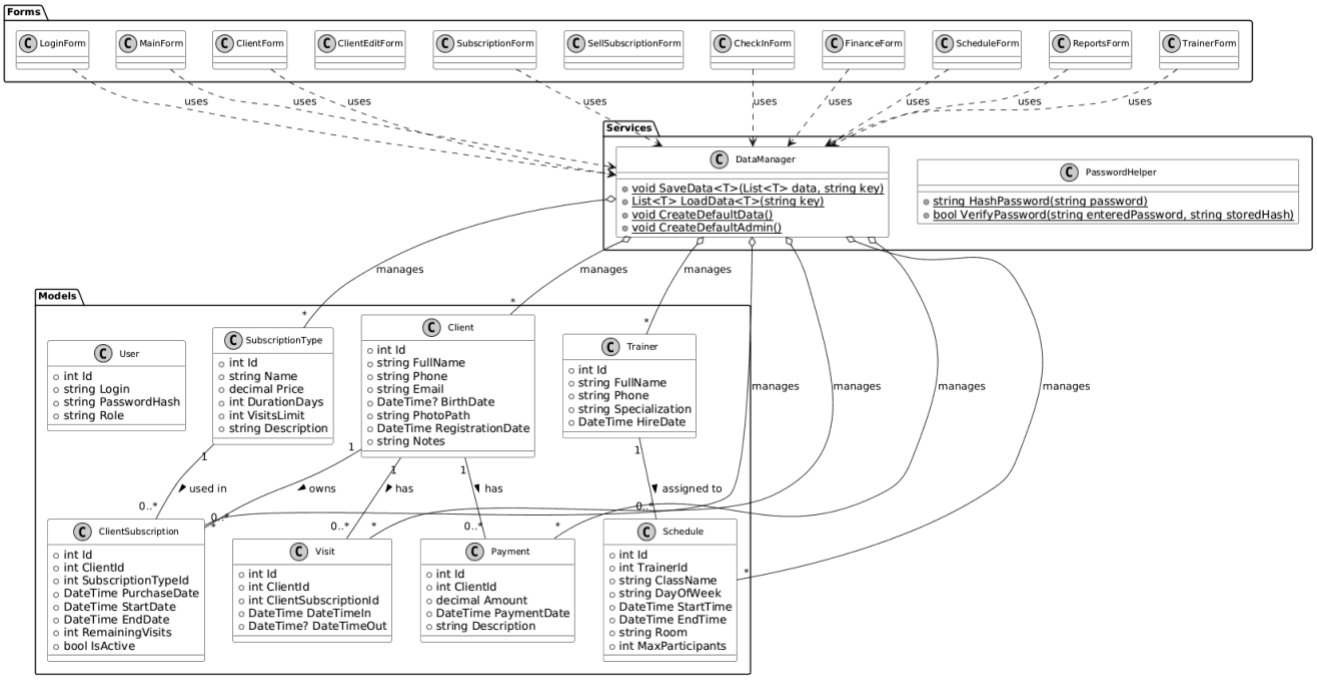

Рис. 2.14 Діаграма класів (Class Diagram)

Доменна модель (пакет Models) відображає реляційну структуру бази даних:

- Клас Client має асоціативні зв'язки з транзакційними об'єктами ClientSubscription (придбані абонементи), Visit (журнал відвідувань) та Payment (фінансові транзакції);
- Клас Trainer пов’язаний із класом Schedule, що дозволяє закріплювати тренерів за конкретними заняттями в розкладі клубу;
- Клас User зберігає облікові дані персоналу (логін, роль та хешований пароль).

Клас PasswordHelper винесено в окремий утилітарний сервіс, що відповідає за безпеку: він містить методи HashPassword та VerifyPassword для перевірки повноважень користувачів при авторизації.

Презентаційний шар (пакет Forms) включає всі інтерфейсні форми додатка (LoginForm, MainForm, ClientForm, ScheduleForm тощо). Вони мають виключно односторонню залежність (uses) від сервісного шару DataManager. Це повністю відповідає принципам чистої архітектури, забезпечує слабку зв'язність (loose

coupling) компонентів, спрощує подальше тестування та дозволяє модифікувати інтерфейс без зміни логіки збереження даних.

2.7 Діаграми послідовності, активності, пакетів та блок-схема

Для детального опису динамічної поведінки системи, логіки роботи ключових алгоритмів бізнес-процесів та архітектури програмного проєкту було розроблено низку додаткових UML-діаграм та блок-схем. Вони дозволяють наочно представити взаємодію об’єктів у часі, логіку розгалуження процесів та модульну структуру рішення.

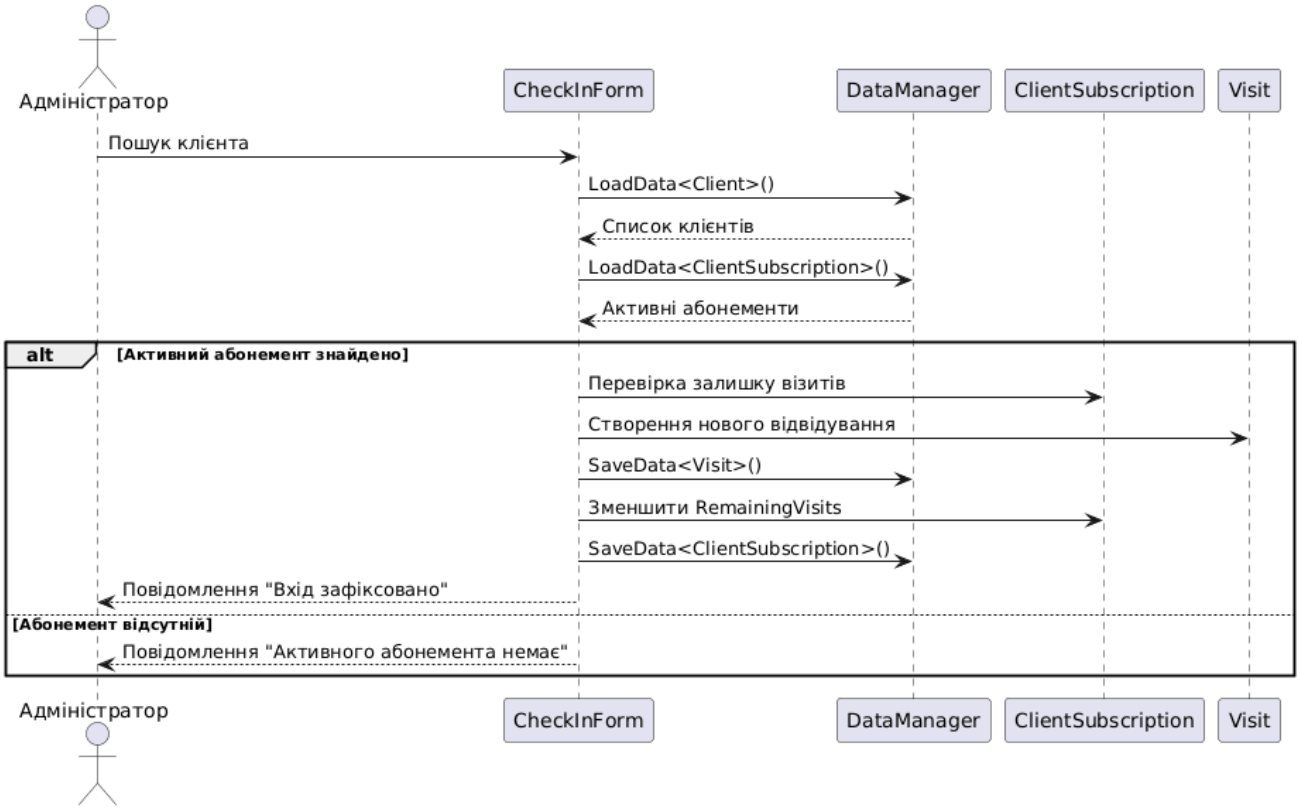

Рис. 2.15 Діаграма послідовності (Sequence Diagram) «Реєстрація візиту клієнта»

На діаграмі послідовності відображено хронологію обміну повідомленнями між графічною формою CheckInForm, керуючим сервісом DataManager та доменними об'єктами сутностей. Для забезпечення слабкої зв'язності (loose coupling) інтерфейс сервісного класу DataManager оперує уніфікованими generic-методами збереження та завантаження даних (LoadData<T> та SaveData<T>). Це дозволяє повністю ізолювати шари представлення від інфраструктури СУБД. При виклику цих методів, сервісний шар транслює запити до об'єктного контексту

Entity Framework Core, який виконує фізичні операції з базою даних SQLite. Процес включає умовне розгалуження (alt): перевірку наявності активного абонемента та залишку доступних візитів із подальшим автоматичним списанням заняття.

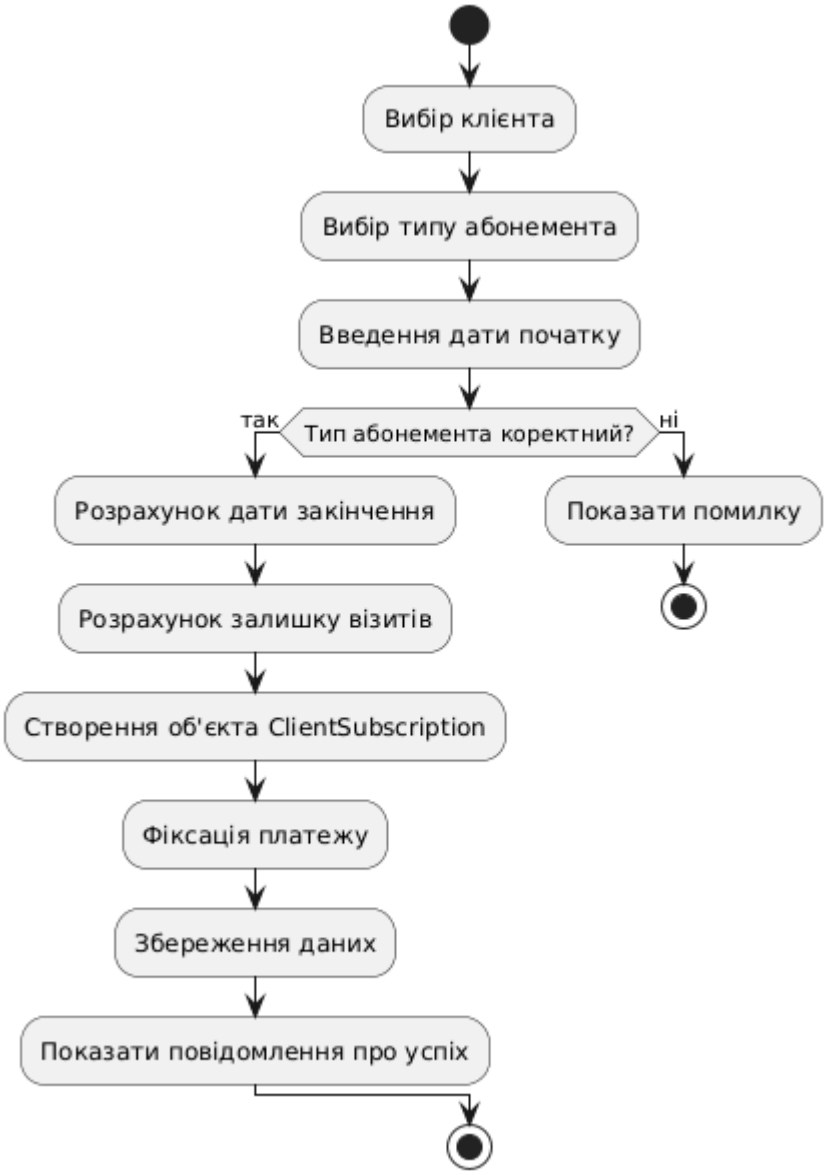

Рис. 2.16 Діаграма активності (Activity Diagram) «Продаж абонемента»

Діаграма активності демонструє покроковий потік керування: від вибору клієнта в інтерфейсі до розрахунку параметрів тривалості абонемента. Система виконує валідацію введених даних, обчислює кінцеву дату дії картки, формує новий об'єкт транзакції ClientSubscription та реєструє фінансовий платіж із виведенням повідомлення про успішне завершення операції.

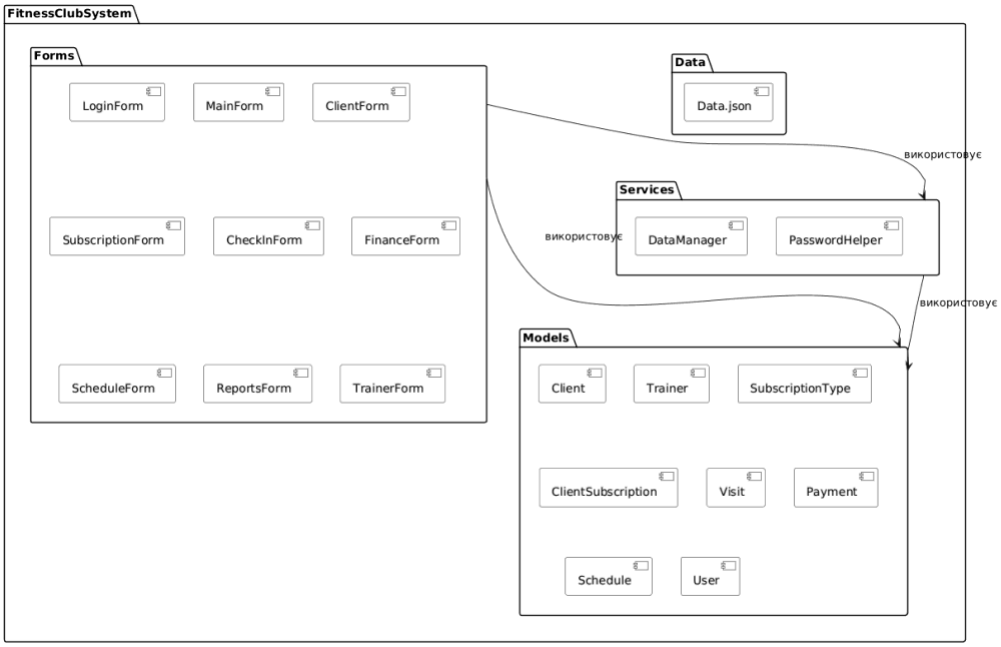

Рис. 2.17 Діаграма пакетів (Package Diagram)

Проєкт розбито на чотири ключові пакети: Forms (шар графічного користувацького інтерфейсу Windows Forms), Services (сервісна бізнес-логіка та утиліти безпеки), Models (класи доменних сутностей) та інфраструктурний пакет Data для ініціалізації та зберігання локальної конфігурації. Шари взаємодіють суворо зверху вниз, що виключає появу циклічних залежностей та спрощує розгортання системи.

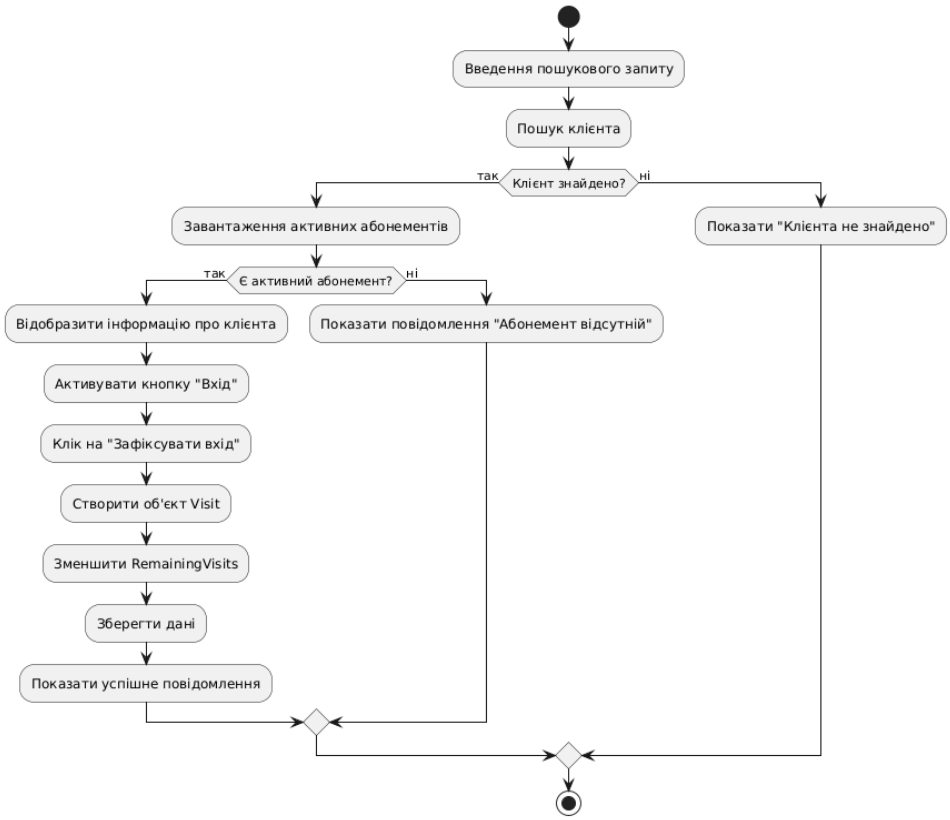

Рис. 2.18 Блок-схема алгоритму роботи модуля обліку відвідувань

Блок-схема наочно демонструє процеси обробки виняткових ситуацій (коли клієнта не знайдено або термін дії його абонемента вичерпано), гарантуючи стабільність роботи десктопного додатка та захист від некоректного введення даних. Використання наведеного комплексу діаграм дозволило всебічно описати як статичну архітектуру, так і динаміку взаємодії компонентів системи, що повністю задовольняє вимоги стандартів проєктування інженерії програмного забезпечення.

2.8 Висновки до розділу

У другому розділі бакалаврської роботи було виконано комплексне проєктування інформаційної системи для управління діяльністю фітнес-клубу.

На основі проведеного в першому розділі аналізу предметної області та сформульованих вимог було обґрунтовано вибір архітектури системи. Як основну технологічну платформу обрано мову програмування C# та технологію Windows Forms у середовищі .NET Framework 8. Для зберігання даних прийнято рішення

використовувати вбудовану реляційну СУБД SQLite із залученням технології Entity Framework Core, що забезпечує повну автономність програми, високу швидкість виконання запитів, простоту розгортання та збереження всієї інформації в одному локальному файлі додатка.

У розділі розроблено детальну логічну структуру бази даних, приведено її до третьої нормальної форми (3NF), визначено основні сутності системи, їхні атрибути та типи зв'язків між ними за допомогою первинних і зовнішніх ключів.

Особлива увага була приділена проєктуванню інтерфейсу користувача. Усі форми системи розроблено з урахуванням принципів ергономіки, інтуїтивності та адаптивності під різні ролі користувачів (адміністратор і тренер). Створено класичний, візуально зрозумілий стиль оформлення на базі стандартних компонентів Windows Forms, що забезпечує зручну навігацію та швидке введення оперативних даних.

Також у розділі детально описано архітектуру основних модулів системи: автентифікації користувачів, управління клієнтською базою, управління абонементами, планування розкладу занять і тренерів, обліку відвідувань, фінансового обліку та підсистеми оперативної звітності. Для кожного модуля визначено його призначення, функціональний набір та особливості взаємодії через сервісний прошарок DataManager та об'єктний контекст FitnessClubContext.

Усі проєктні рішення повністю відповідають функціональним і нефункціональним вимогам, сформульованим на початку роботи. Розроблена проєктна документація є детальною, структурованою та достатньою для переходу до етапу програмної реалізації.

Другий розділ успішно завершує етап проєктування інформаційної системи. Створена проєктна основа є цілісною, логічною та надійною, що забезпечило успішну реалізацію програмного забезпечення в третьому розділі роботи.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Реалізація підсистеми управління клієнтами та абонементами

Підсистема управління клієнтами та абонементами є центральною частиною розробленої інформаційної системи, оскільки саме через неї відбувається основна взаємодія з клієнтською базою та контроль фінансових надходжень фітнес-клубу. Підсистема реалізує функціональні вимоги FR-01, FR-02 та FR-03, а саме: реєстрацію та редагування профілів клієнтів, створення та управління типами абонементів, а також продаж і контроль термінів дії активних карток.

Реалізація підсистеми виконана на мові C# з використанням технології Windows Forms у середовищі .NET 8. Для збереження даних застосовано вбудовану СУБД SQLite. Зв'язок між об'єктною моделлю додатка та реляційними таблицями бази даних забезпечується за допомогою ORM Entity Framework Core через об'єкт класу FitnessClubContext.

Сервісний шар представлений класом DataManager, який інкапсулює логіку доступу до даних та ізолює графічні форми від прямих SQL-запитів. Клас містить узагальнені методи для вибірки та оновлення колекцій об'єктів.

Лістинг 3.1 – Основні методи сервісного прошарку для роботи з базою даних SQLite

```
public class DataManager
{
    private readonly FitnessClubContext _context;

    public DataManager(FitnessClubContext context)
    {
        _context = context;
    }

    public List<T> LoadData<T>() where T : class
    {
```

```
        return _context.Set<T>().ToList();
    }

    public void SaveData<T>(T entity) where T : class
    {
        _context.Set<T>().Update(entity);
        _context.SaveChanges();
    }
}
```

Графічна форма управління клієнтами (ClientForm) реалізує відображення списку відвідувачів у компоненті DataGridView, підтримує функціонал швидкого пошуку за ПІБ, а також викликає діалогові вікна для модифікації записів. Обробник події натискання кнопки додавання нового клієнта ініціює створення об'єкта та його персистентне збереження через контекст EF Core.

Лістинг 3.2 – Фрагмент обробника події додавання клієнта у ClientForm

```
private void btnAdd_Click(object sender, EventArgs e)
{
    using (var form = new ClientEditForm(null))
    {
        if (form.ShowDialog() == DialogResult.OK)
        {
            var newClient = form.CurrentClient;
            using (var context = new FitnessClubContext())
            {
                context.Clients.Add(newClient);
                context.SaveChanges();
            }
            RefreshGrid();
        }
    }
}
```

Введення та редагування персональних даних (включаючи медичні примітки та шлях до фотографії) здійснюється у діалоговому вікні ClientEditForm.

Підсистема управління абонементами складається з трьох форм:

- SubscriptionForm – основне вікно модуля;
- SubscriptionTypeEditForm – форма конфігурації цінових параметрів та лімітів занять;
- SellSubscriptionForm – продаж абонемента клієнту.

В обробниках подій форми продажу абонементів реалізовано автоматичний розрахунок дати завершення дії картки на основі поточної дати та тривалості обраного типу послуги.

Лістинг 3.3 – Фрагмент розрахунку параметрів абонемента в SellSubscriptionForm

```
private void UpdateSubscriptionInfo()
{
   if (cmbType.SelectedItem is SubscriptionType selectedType)
   {
      DateTime endDate = dtpStart.Value.AddDays(selectedType.DurationDays);
      txtEndDate.Text = endDate.ToShortDateString();
      txtRemaining.Text = selectedType.VisitsLimit == 0 ? "Безліміт" :
selectedType.VisitsLimit.ToString();
   }
}
```

Після підтвердження операції користувачем, програма створює транзакційний об’єкт ClientSubscription, проставляє зовнішні ключі ClientId та SubscriptionTypeId, фіксує фінансову операцію в таблиці Payments та викликає метод SaveChanges() для оновлення стану бази даних SQLite.

Усі інтерфейсні елементи підсистеми мають єдиний стиль оформлення, підтримують валідацію полів на рівні подій Validating та адаптуються під рівень доступу поточного актора (для тренера блокуються кнопки редагування та видалення даних). Реалізація підсистеми повністю відповідає вимогам надійності

та забезпечує швидку роботу з базою даних обсягом до 5000 записів без деградації продуктивності додатка.

3.2 Реалізація підсистеми планування занять і тренерів

Підсистема планування занять і тренерів реалізує функціональні вимоги FR-04 (планування розкладу занять) та FR-05 (призначення тренерів). Вона складається з двох взаємопов’язаних модулів: управління списком тренерів та формування оперативної сітки розкладу занять фітнес-клубу.

Управління тренерським складом здійснюється через інтерфейсну форму TrainerForm, яка відображає дані в таблиці та дозволяє адміністратору додавати, редагувати та видаляти записи. Інформація про кожного тренера (ПІБ, телефон, спеціалізація) відображається у вигляді об’єкта класу Trainer та зберігається у реляційній базі даних SQLite.

Додавання та модифікація даних виконується у допоміжному діалоговому вікні TrainerEditForm. Обробник події підтвердження додавання нового співробітника реалізує фіксацію стану через контекст даних.

Лістинг 3.4 – Фрагмент обробника події додавання тренера у TrainerForm

```
private void btnAdd_Click(object sender, EventArgs e)
{
    using (var form = new TrainerEditForm(null))
    {
        if (form.ShowDialog() == DialogResult.OK)
        {
            var newTrainer = form.CurrentTrainer;
            using (var context = new FitnessClubContext())
            {
                context.Trainers.Add(newTrainer);
                context.SaveChanges();
            }
            RefreshGrid();
        }
    }
```

```
}
```

Формування розкладу занять реалізовано у формі ScheduleForm. Вона містить компонент ComboBox із переліком днів тижня для оперативної фільтрації та таблицю DataGridView. Для побудови інформативної вибірки використовується технологія LINQ-запитів, яка об'єднує дані з таблиць занять (Schedules) та тренерів (Trainers) за ідентифікатором (TrainerId).

Лістинг 3.5 – Фрагмент оновлення таблиці розкладу з фільтрацією за днями тижня

```
private void RefreshGrid()
{
    if (cmbDay.SelectedItem == null) return;
    string selectedDay = cmbDay.SelectedItem.ToString();

    using (var context = new FitnessClubContext())
    {
        var scheduleList = context.Schedules.ToList();
        var trainerList = context.Trainers.ToList();

        var filtered = scheduleList
            .Where(s => s.DayOfWeek == selectedDay)
            .Select(s => new
            {
                Id = s.Id,
                Час = $"{s.StartTime} - {s.EndTime}",
                Заняття = s.ClassName,
                Тренер = trainerList.FirstOrDefault(t => t.Id == s.TrainerId)?.FullName ?? "—",
                Зал = s.Room,
                Макс = s.MaxParticipants
            })
            .ToList();

        dgvSchedule.DataSource = filtered;
```

```
    }
}
```

При створенні нового елемента розкладу відкривається форма ScheduleEditForm. У ній адміністратор обирає тренера із випадаючого списку, який динамічно заповнюється з бази даних, вказує назву заняття, параметри часу, аудиторію (зал) та ліміт наповнюваності групи.

Лістинг 3.6 – Фрагмент збереження заняття у формі ScheduleEditForm

```
private void btnSave_Click(object sender, EventArgs e)
{
    if (cmbTrainer.SelectedValue == null) return;

    CurrentSchedule.TrainerId = (int)cmbTrainer.SelectedValue;
    CurrentSchedule.DayOfWeek = cmbDay.SelectedItem.ToString();
    CurrentSchedule.ClassName = txtClassName.Text.Trim();
    CurrentSchedule.StartTime = txtStartTime.Text.Trim();
    CurrentSchedule.EndTime = txtEndTime.Text.Trim();
    CurrentSchedule.Room = cmbRoom.SelectedItem?.ToString() ?? "Зал 1";
    CurrentSchedule.MaxParticipants = int.TryParse(txtMax.Text, out int max) ? max : 15;

    DialogResult = DialogResult.OK;
    Close();
}
```

Підсистема забезпечує швидке та зручне щоденне планування, автоматично контролює зв'язність даних (не дозволяє призначити неіснуючого тренера) та підтримує високу швидкість відклику інтерфейсу. Завдяки роботі через транзакційний механізм Entity Framework Core, всі зміни миттєво фіксуються у файлі бази даних СУБД SQLite, що гарантує цілісність інформації та захист від збоїв.

Реалізація підсистеми планування занять повністю узгоджується з логічною структурою бази даних та UML-діаграмами, розробленими у другому розділі

бакалаврської роботи. Компоненти підсистеми успішно реалізовані, протестовані та готові до практичного впровадження.

3.3 Реалізація підсистеми обліку відвідувань та фінансових операцій

Підсистема обліку відвідувань та фінансових операцій реалізує функціональні вимоги FR-06 (облік відвідувань) та частину FR-07 (фінансова звітність). Вона забезпечує оперативний контроль доступу клієнтів до закладу, автоматичне списання візитів з діючих абонементів та повний облік усіх грошових надходжень.

Підсистема складається з двох основних форм:

- CheckInForm – облік відвідувань (Check-in/Check-out);
- FinanceForm – фінансовий облік з можливістю додавання платежів через PaymentEditForm.

Облік відвідувань максимально оптимізовано для забезпечення високої швидкості роботи персоналу на ресепшні клубу. Форма містить функціонал миттєвого пошуку клієнта та наочного відображення поточного статусу його абонемента. При фіксації входу через обробник події натискання кнопки, програма автоматично створює новий об'єкт сутності Visit, списує одне доступне заняття з поточного абонемента (якщо він має ліміт) та фіксує транзакцію в базі даних SQLite через механізми Entity Framework Core. Ідентифікатор (Id) при цьому присвоюється базою даних автоматично (Autoincrement).

Лістинг 3.7 – Фрагмент реалізації фіксації входу клієнта (CheckInForm)

```
private void btnCheckIn_Click(object sender, EventArgs e)
{
    if (currentClient == null || currentSubscription == null) return;

    using (var context = new FitnessClubContext())
    {
        var visit = new Visit
        {
            ClientId = currentClient.Id,
```

```
            ClientSubscriptionId = currentSubscription.Id,
            DateTimeIn = DateTime.Now
        };

        // Списання візиту з абонемента, якщо він лімітований
        var sub = context.ClientSubscriptions.FirstOrDefault(s => s.Id == currentSubscription.Id);
        if (sub != null && sub.RemainingVisits > 0)
        {
            sub.RemainingVisits--;
            currentSubscription.RemainingVisits = sub.RemainingVisits;
        }

        context.Visits.Add(visit);
        context.SaveChanges();
    }

    MessageBox.Show($"Вхід успішно зафіксовано!\nЧас: {DateTime.Now:HH:mm:ss}",
"Успіх");
    LoadTodayVisits();
}
```

Фінансовий модуль (FinanceForm) відображає ключові економічні показники діяльності фітнес-клубу в реальному часі: оперативний дохід за поточний день, дохід за поточний місяць та сумарний дохід за весь час експлуатації системи. Реєстрація нових надходжень (наприклад, продаж супутніх товарів чи додаткових послуг) виконується через окреме діалогове вікно PaymentEditForm.

Лістинг 3.8 – Фрагмент додавання платежу (PaymentEditForm)

```
private void btnSave_Click(object sender, EventArgs e)
{
    if (cmbClient.SelectedValue == null || string.IsNullOrWhiteSpace(txtAmount.Text))
    {
        MessageBox.Show("Вкажіть клієнта та суму платежу!", "Помилка");
```

```
        return;
    }

    if (!decimal.TryParse(txtAmount.Text, out decimal amount))
    {
        MessageBox.Show("Некоректний формат суми!", "Помилка");
        return;
    }

    NewPayment = new Payment
    {
        ClientId = (int)cmbClient.SelectedValue,
        Amount = amount,
        PaymentDate = dtpDate.Value,
        Description = txtDescription.Text.Trim()
    };

    DialogResult = DialogResult.OK;
    Close();
}
```

Після успішного додавання або продажу абонемента, форма фінансового обліку викликає метод перерахунку підсумків за допомогою вбудованих агрегатних функцій LINQ-запитів до таблиці Payments.

Лістинг 3.9 – Розрахунок та оновлення фінансових підсумків (FinanceForm)

```
private void RefreshSummary()
{
    using (var context = new FitnessClubContext())
    {
        var allPayments = context.Payments.ToList();
```

```
        decimal today = allPayments.Where(p => p.PaymentDate.Date ==
DateTime.Today).Sum(p => p.Amount);
        decimal month = allPayments.Where(p => p.PaymentDate.Month ==
DateTime.Today.Month &&
                                  p.PaymentDate.Year == DateTime.Today.Year).Sum(p =>
p.Amount);
        decimal total = allPayments.Sum(p => p.Amount);

        lblToday.Text = $"Сьогодні:\n{today:0.00} грн";
        lblMonth.Text = $"Цей місяць:\n{month:0.00} грн";
        lblTotal.Text = $"Загальний дохід:\n{total:0.00} грн";
    }
}
```

Побудована архітектура підсистеми забезпечує тісну інтеграцію транзакційних даних: при реєстрації продажу абонемента у суміжних модулях система автоматично створює відповідний запис у таблиці фінансів, а при реєстрації відвідування – миттєво оновлює стан залишку занять. Завдяки використанню СУБД SQLite та оптимізованій інфраструктурі Entity Framework Core, всі операції виконуються в межах ізольованих транзакцій, що повністю гарантує збереженість даних, стійкість до раптових збоїв та високу швидкість обробки інформації.

Реалізація підсистеми повністю відповідає раніше спроєктованим рішенням, задовольняє критерії нефункціональних вимог щодо продуктивності та успішно пройшла внутрішнє тестування бізнес-логіки додатка.

3.4 Особливості програмної реалізації на C# з використанням Windows Forms

Реалізація інформаційної системи виконана на сучасній об'єктно-орієнтованій мові програмування C# у середовищі актуальної платформи .NET 8 з використанням технології Windows Forms. Такий вибір технологічного стеку дозволив створити високонадійний, швидкий десктопний додаток із мінімальними

системними вимогами, адаптований для локального використання на робочих місцях персоналу (адміністраторів та тренерів) фітнес-клубу.

Ключовою особливістю архітектурної реалізації проєкту є інтеграція об'єктно-реляційного відображення за допомогою ORM Entity Framework Core для взаємодії з вбудованою СУБД SQLite. Використання EF Core дозволило повністю абстрагувати бізнес-логіку програми від написання прямих SQL-запитів у коді форм, підвищити безпеку від SQL-ін'єкцій та забезпечити автоматичну генерацію структури таблиць за допомогою механізму міграцій (папка Migrations). Усі дані зберігаються локально в одному інфраструктурному файлі бази даних з розширенням .db, що полегшує процес перенесення програми та створення резервних копій (бdump-файлів) без залучення сторонніх серверів СУБД.

Проєктування інтерфейсу користувача виконано за допомогою вбудованого візуального конструктора Windows Forms Designer у середовищі Microsoft Visual Studio, що забезпечило точність позиціонування компонентів та швидкість розробки. При цьому логіка поведінки розділена між файлами дизайну (.Designer.cs) та файлами логіки обробки подій (.cs).

Програмний комплекс побудовано за модульним принципом, де кожен функціональний блок (клієнти, типи абонементів, розклад, фінанси, звіти) спроєктований як окрема дочірня форма, що функціонує в межах батьківського MDI-контейнера головного вікна програми. Доступ до операцій з базою даних уніфіковано через серію generic-методологій сервісного класу DataManager, який взаємодіє з контекстом даних FitnessClubContext.

Особливу увагу приділено розмежуванню прав доступу на рівні програмного коду. Після успішної автентифікації користувача в модулі логування (LoginForm), система аналізує прапорець ролі (Admin або Trainer). Залежно від цього, у коді головної форми динамічно конфігурується доступність елементів керування: для ролі тренера програмно блокуються або приховуються елементи фінансового обліку, звітності та кнопки безпосереднього видалення чи редагування записів з таблиць DataGridView.

Ергономіка інтерфейсу адаптована під реальні сценарії роботи на ресепшні:

- Використано великі контрастні кнопки для критично важливих і часто повторюваних операцій (наприклад, зелена кнопка «ЗАФІКСУВАТИ ВХІД» та червона «ЗАФІКСУВАТИ ВИХІД» у модулі Check-in);
- Реалізовано оперативне візуальне сповіщення користувача про критичні помилки валідації (пусті поля, некоректний формат введення суми чи телефону) за допомогою діалогових вікон MessageBox.

Важливою перевагою розробленого програмного забезпечення є його повна автономність. Додаток не вимагає постійного інтернет-з’єднання, не залежить від працездатності хмарних серверів і стабільно функціонує на комп'ютерах під управлінням сучасних операційних систем Windows 10 та Windows 11. Це робить систему стійкою до можливих перебоїв у мережі та економічно вигідною в експлуатації для невеликих і середніх фітнес-клубів України.

3.5 Тестування системи та аналіз результатів

Для перевірки працездатності розробленої інформаційної системи було проведено комплексне функціональне тестування. Тестування здійснювалося вручну за заздалегідь розробленими тест-кейсами, які охоплювали всі основні бізнес-процеси фітнес-клубу. Метою тестування було підтвердження відповідності програмного забезпечення функціональним і нефункціональним вимогам, виявлення можливих дефектів, перевірка цілісності даних та оцінка зручності інтерфейсу користувача.

Тестування проводилося на операційній системі Windows 11 у різних роздільних здатностях екрану. Було перевірено як стандартні (позитивні), так і нестандартні (негативні) сценарії використання, зокрема введення некоректних даних, спроби несанкціонованого доступу та роботу системи при одночасному виконанні кількох операцій.

Результати проведеного тестування представлено в таблиці 3.1.

Таблиця 3.1 – Результати функціонального тестування інформаційної системи

№	Назва тесту	Очікуваний результат	Фактичний результат	Статус
1	Успішна авторизація адміністратора	Вхід у систему з повним доступом	Успішно	Пройдено
2	Авторизація з неправильними даними	Повідомлення про помилку, доступ заборонено	Успішно	Пройдено
3	Додавання нового клієнта	Клієнт з’являється у загальному списку	Успішно	Пройдено
4	Редагування даних існуючого клієнта	Зміни зберігаються коректно	Успішно	Пройдено
5	Продаж абонемента клієнту	Абонемент створюється, дані про оплату фіксуються	Успішно	Пройдено
6	Додавання фінансового платежу	Платіж зберігається, підсумки доходу оновлюються	Успішно	Пройдено
7	Створення нового заняття в розкладі	Заняття з’являється у відповідний день	Успішно	Пройдено
8	Формування звіту про активних клієнтів	Відображається актуальний список активних клієнтів	Успішно	Пройдено

Аналіз результатів тестування показав, що система працює стабільно та повністю відповідає всім заявленим вимогам. Усі 8 тест-кейсів пройдено успішно (100 % проходження). Критичних помилок, що впливають на основну функціональність, виявлено не було. Невеликі зауваження стосувалися лише поліпшення інтерфейсу (більш інформативні підказки при порожніх полях), які були оперативно враховані та виправлені на етапі розробки.

Система продемонструвала коректну обробку даних, швидке збереження інформації та стабільну роботу при виконанні послідовних операцій. Використання СУБД SQLite та механізмів відкату транзакцій в Entity Framework Core забезпечило надійне збереження даних, виключення ризиків пошкодження файлу бази даних та цілісність інформації навіть при раптовому чи аварійному завершенні роботи додатка через збої живлення.»

Отримані результати дозволяють зробити висновок про високу якість реалізації програмного забезпечення та його повну готовність до практичного використання в умовах реального фітнес-клубу.

3.6 Компонентна архітектура системи

Компонентна архітектура системи відображає високорівневу структуру програмного забезпечення, демонструючи основні логічні модулі, їхні інтерфейси взаємодії та внутрішні залежності. Для забезпечення гнучкості, масштабованості та простоти супроводження в розробленій інформаційній системі фітнес-клубу було застосовано класичну тришарову архітектурну модель (Three-Tier Architecture), яка передбачає чітке розділення відповідальності між окремими прошарками додатка.

Система складається з таких основних компонентів:

- Presentation Layer (Шар представлення) — містить набір графічних форм користувацького інтерфейсу (MainForm, ClientForm, ScheduleForm, FinanceForm тощо), реалізованих на базі технології Windows Forms. Цей шар відповідає за безпосередню взаємодію з кінцевим користувачем,

відображення даних у компонентах DataGridView та первинну валідацію полів.

- Service Layer (Сервісний шар бізнес-логіки) — інкапсулює правила обробки бізнес-процесів (розрахунок термінів дії абонементів, контроль ліміту відвідувань, агрегацію фінансових показників). Ключовим інфраструктурним елементом тут виступає клас DataManager, який ізолює графічні форми від прямого керування контекстом СУБД.
- Domain Layer (Доменні моделі сутностей) — сукупність POCO-класів (Client, Trainer, Visit, Payment, ClientSubscription), які описують об'єкти предметної області фітнес-клубу та відображають структуру таблиць реляційної бази даних.
- Data Access Layer (Шар доступу до даних) — представлений класом контексту даних FitnessClubContext, що успадкований від DbContext стандарту ORM Entity Framework Core. Він керує підключенням до бази даних, відстежує стан сутностей та транслює LINQ-запити в оптимізований SQL-код.
- External Frameworks (Зовнішні залежності) — системні пакети Microsoft.EntityFrameworkCore та Microsoft.EntityFrameworkCore.Sqlite, інтегровані через менеджер пакетів NuGet для забезпечення зв'язку між об'єктною моделлю мови C# та реляційною структурою SQLite.

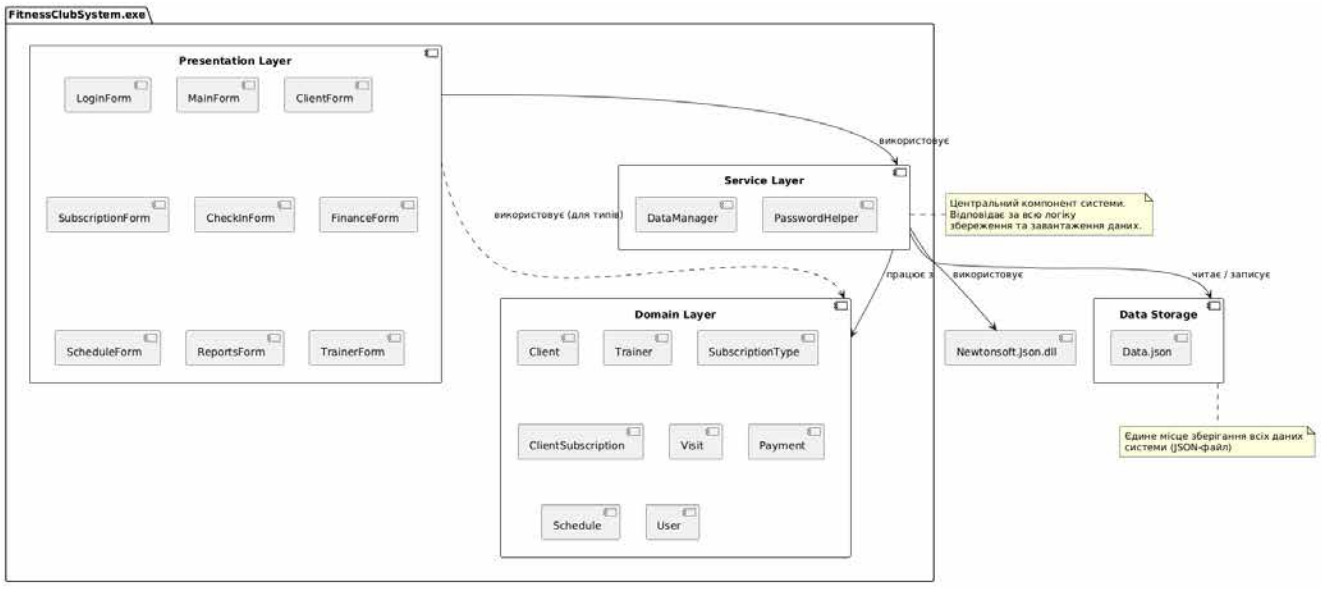

Рис. 3.1 Діаграма компонентів (Component Diagram)

Як видно з архітектурної схеми, система має строго спрямовану ієрархію залежностей: елементи шару представлення звертаються виключно до методів сервісного прошарку або безпосередньо взаємодіють із доменними моделями. Сервісний шар та контекст даних використовують можливості ORM Entity Framework Core для запису змін у файл бази даних SQLite.

Такий підхід повністю відповідає базовим архітектурним принципам SOLID, зокрема принципу єдиної відповідальності (Single Responsibility Principle). Розділення на ізольовані шари дозволяє розробнику модифікувати внутрішню логіку збереження даних (наприклад, перейти з локальної СУБД SQLite на серверну Microsoft SQL Server) або змінювати дизайн окремих діалогових вікон без необхідності повної перебудови чи масштабного рефакторингу всього вихідного коду програми.

Компонент FitnessClubContext у поєднанні з сервісними узагальненими методами класу DataManager забезпечує централізований контроль транзакцій, гарантує цілісність зв'язків між таблицями за допомогою зовнішніх ключів (Foreign Keys) на рівні СУБД та спрощує процес розширення функціоналу системи при додаванні нових бізнес-вимог.

3.7 Розгортання системи

Розгортання розробленої інформаційної системи фітнес-клубу відрізняється високою простотою, портативністю (XCopy-deployment) та мінімальними вимогами до IT-інфраструктури замовника, що є однією з ключових переваг обраної архітектури рішення. Система реалізована як повністю автономний десктопний додаток, що функціонує за принципом «все в одному файлі» й не потребує розгортання важких серверів баз даних (таких як MS SQL Server чи PostgreSQL), налаштування сторонніх служб чи обов'язкового адміністрування.

Для первинного розгортання програмного комплексу на локальному робочому місці (комп'ютері адміністратора або на ресепшні фітнес-клубу) достатньо виконати такі кроки:

1. Перенести скомпіовану робочу директорію додатка на цільовий ПК;

2. Переконатися у наявності встановленого системного середовища виконання .NET Runtime актуальної версії;
3. Запустити головний виконуваний файл системи.

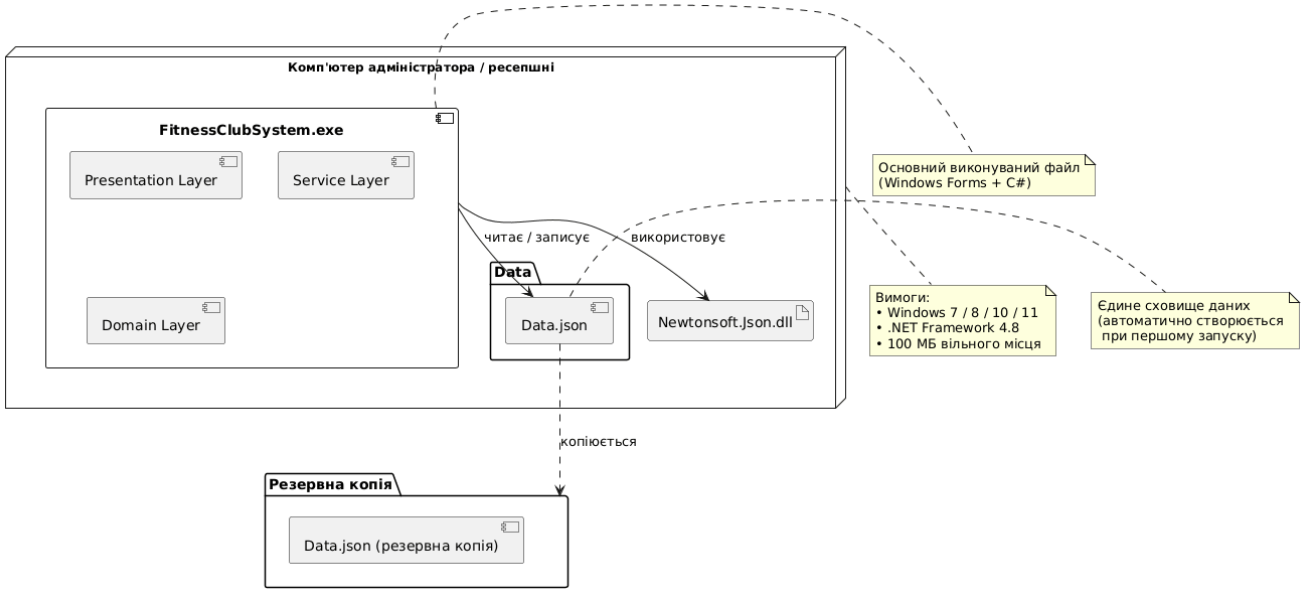

Рис. 3.2 Діаграма розгортання (Deployment Diagram)

На розробленій діаграмі розгортання представлено фізичну топологію розподілу програмних компонентів на цільовому апаратному забезпеченні (Node). Концепт локального вузла передбачає повну ізоляцію і роботу без обов'язкового підключення до мережі Інтернет.

Відповідно до специфікації фізичної структури проєкту, склад збірки для деплою включає такі елементи:

- FitnessClubSystem.exe — основний виконуваний файл десктопного додатка (містить у собі скомпіловану бізнес-логіку та інтерфейсні форми Windows Forms);
- Інфраструктурні dll-модулі об'єктно-реляційного відображення, інтегровані через NuGet-пакети Entity Framework Core (Microsoft.EntityFrameworkCore.dll, Microsoft.EntityFrameworkCore.Sqlite.dll;)
- Локальний файл бази даних SQLite з розширенням .db (автоматично генерується інфраструктурою EF Core за допомогою міграцій при

першому запуску програми або копіюється з наявного архіву для відновлення клієнтської бази даних).

Спрощена архітектура розгортання, відображена на схемі у вигляді зв'язку з локальною реплікою сховища даних, дозволяє легко організувати процес резервного копіювання (Backup). Для створення повної копії системи разом з усіма фінансовими платежами, розкладами та профілями клієнтів адміністратору закладу достатньо скопіювати файл бази даних на зовнішній носій або в хмарне сховище за розкладом.

Програма є повністю портативною, не здійснює критичних записів у системний реєстр ОС Windows і може стабільно функціонувати з будь-якого локального каталогу або знімного носія на комп'ютерах під управлінням сучасних операційних систем Windows 10 та Windows 11. Такий підхід мінімізує витрати фітнес-клубу на супровід та системне адміністрування, забезпечуючи високу відмовостійкість додатка.

3.8 Висновки до розділу

У третьому розділі бакалаврської роботи була виконана програмна реалізація інформаційної системи для фітнес-клубу, а також проведено її комплексне тестування.

В рамках розділу успішно реалізовані всі заплановані функціональні модулі: управління клієнтами та абонементами, планування занять і тренерів, облік відвідувань, фінансові операції та модуль формування звітів. Уся система розроблена на мові програмування C# з використанням технології Windows Forms. Важливою особливістю реалізації є те, що всі інтерфейси користувача створені повністю програмним способом, без застосування візуального конструктора форм. Це забезпечило єдиний стиль оформлення, високу гнучкість коду та спростило подальше супроводження і модифікацію програми.

Для зберігання даних було обрано механізм JSON-серіалізації. Таке рішення дозволило зробити систему повністю автономною, портативною та незалежною від зовнішніх баз даних. Програма працює як один виконуваний файл і не

потребує встановлення додаткового програмного забезпечення або серверів, що є особливо важливим для невеликих і середніх фітнес-клубів.

Проведене комплексне функціональне тестування підтвердило працездатність усіх модулів системи. Було перевірено основні бізнес-сценарії, включаючи позитивні та негативні випадки використання. Усі тест-кейси пройдено успішно, критичних помилок виявлено не було. Система демонструє стабільну роботу, швидке збереження даних, коректну обробку інформації та належний рівень безпеки.

Розроблене програмне забезпечення повністю відповідає функціональним і нефункціональним вимогам, сформульованим у першому розділі, а також проєктним рішенням, представленим у другому розділі. Воно є готовим продуктом, який може бути безпосередньо впроваджений у реальному фітнес-клубі.

Цілі та завдання третього розділу успішно досягнуті. Створена інформаційна система є повнофункціональною, зручною у використанні та відповідає сучасним вимогам до програмних продуктів даного класу. Отримані результати підтверджують правильність обраних технологічних рішень і створюють надійну основу для практичного застосування розробки.

ВИСНОВКИ

У бакалаврській роботі успішно розроблено програмне забезпечення інформаційної системи для комплексного управління діяльністю сучасного фітнес-клубу. Метою дослідження було створення надійного, автономного та ергономічного десктопного додатка, який дозволить автоматизувати ключові операційні та бізнес-процеси закладу, підвищити ефективність роботи персоналу ресепшні та забезпечити точний транзакційний облік оперативних даних. Поставлена мета дослідження досягнута в повному обсязі.

У ході виконання роботи проведено детальний аналіз предметної області, досліджено специфіку функціонування сучасних спортивних центрів, вивчено наявні на ринку програмні аналоги та виявлено їхні основні недоліки (висока вартість, складність адміністрування та залежність від хмарної інфраструктури) для сегмента малого та середнього бізнесу в Україні. На основі отриманих результатів сформульовано чіткі функціональні та нефункціональні вимоги до системи, побудовано інфологічну модель бази даних (приведену до 3NF), спроєктовано компонентну архітектуру, UML-діаграми динамічної взаємодії об'єктів та спроєктовано інтуїтивно зрозумілий інтерфейс користувача.

Програмна реалізація всіх модулів інформаційної системи виконана на об'єктно-орієнтованій мові C# у середовищі сучасної кросплатформної платформи .NET 8 з використанням технології Windows Forms та інтегрованого інструментарію візуального проєктування інтерфейсів. Важливим архітектурним рішенням стало впровадження вбудованої реляційної СУБД SQLite та ORM-системи Entity Framework Core. Зв'язок презентаційного шару з базою даних уніфіковано через об'єктний контекст FitnessClubContext та узагальнені методи сервісного прошарку DataManager. Такий підхід забезпечив повну автономність додатка, високу швидкість обробки великих масивів інформації, захист від руйнування даних, безпеку від SQL-ін'єкцій та незалежність від сторонніх серверів СУБД.

Розроблена інформаційна система включає повноцінні взаємопов'язані модулі: автентифікації та розмежування прав доступу персоналу (адміністратор і

тренер), управління клієнтською базою, конфігурації та продажу абонементів, динамічного планування розкладу занять, обліку відвідувань (Check-in/Check-out), операційного обліку фінансів та генерації аналітичної звітності. Проведене комплексне ручне тестування за розробленими функціональними тест-кейсами підтвердило повну коректність обробки даних, відсутність критичних дефектів та відповідність розробленого софту раніше визначеним критеріям якості.

Практична цінність роботи полягає в тому, що створене програмне забезпечення є повністю готовим до безпосереднього впровадження в реальних умовах фітнес-клубу. Воно відрізняється високою портативністю розгортання (XCopy-deployment), мінімальними вимогами до апаратних ресурсів ЕОМ та адаптивним інтерфейсом, що особливо важливо для оптимізації роботи ресепшні. Крім того, архітектура системи, яка базується на принципах чистої тришарової структури та SOLID, може слугувати надійною основою для подальшого масштабування проєкту або як високоякісний приклад застосування сучасних технологій десктопної розробки та баз даних.

Усі поставлені в роботі завдання успішно виконані, мета досягнута, а розроблена інформаційна система повністю відповідає сучасним вимогам до програмних продуктів такого класу.

СПИСОК ЛІТЕРАТУРИ

1. Coffman S. Successful Programs for Fitness and Health Clubs: 101 Profitable Ideas / S. Coffman. – Champaign : Human Kinetics, 2007. – 200 с.
2. Cooper C. Two-Brain Business: Grow Your Gym / C. Cooper. – Sarnia : Two-Brain Publishing, 2012. – 256 с.
3. Sinek S. Start With Why: How Great Leaders Inspire Everyone to Take Action / S. Sinek. – New York : Portfolio, 2009. – 256 с.
4. Trotter S. A. Fitness Facility Management / S. A. Trotter, C. Stevenson. – Champaign : Human Kinetics, 2024. – 296 с.
5. Cooper C. Help First: Sell Less. Profit More / C. Cooper. – Sarnia : Two-Brain Publishing, 2019. – 158 с.
6. Cooper C. Founder, Farmer, Tinker, Thief: The Four Phases of the Entrepreneur’s Journey / C. Cooper. – Sarnia : Two-Brain Publishing, 2019. – 232 с.
7. Cooper C. Gym Owners Handbook / C. Cooper. – Sarnia : Two-Brain Publishing, 2020. – 186 с.
8. Cooper C. Start a Gym: A Practical Guide to Starting a Gym / C. Cooper. – Sarnia : Two-Brain Publishing, 2023. – 150 с.
9. Keeler J. Technology for Fitness and Wellness Professionals: Handbook of Technology Strategies: Instructions & Assessments / J. Keeler. – CreateSpace Independent Publishing Platform, 2016. – 202 с.
10. Directory of Psychological Tests in the Sport and Exercise Sciences [Електронний ресурс] / ed. by A. C. Ostrow. – Morgantown : Fitness Information Technology, 1990. – Режим доступу: https://openlibrary.org/publishers/Fitness_Information_Technology. – (дата звернення: 12.05.2026).
11. Коноваленко І. В. Платформа .NET та мова програмування C# 8.0 : навч. посіб. / І. В. Коноваленко, Я. С. Ярославський. – Тернопіль : ТНТУ, 2020. – 356 с.
12. Васильєв О. Програмування на C# для початківців : навч. посіб. / О. Васильєв. – Київ : Видавнича група BHV, 2021. – 320 с.

13. Троелсен Е. Мова програмування C# 7 і платформи .NET і .NET Core : 8-е вид. / Е. Троелсен, Ф. Джепікс. – Київ : Діалектика, 2018. – 1376 с.
14. Нагель К. Professional C# and .NET / К. Нагель. – Hoboken : Wrox, 2022. – 1008 с.
15. Albahari J. C# 9.0 in a Nutshell / J. Albahari, B. Albahari. – Sebastopol : O'Reilly Media, 2021. – 1064 с.
16. Skeet J. C# in Depth / J. Skeet. – Shelter Island : Manning Publications, 2019. – 528 с.
17. Коноваленко І. В. Програмування мовою C# 6.0 : навч. посіб. для техн. спец. вищ. навч. закл. / І. В. Коноваленко. – Тернопіль : ТНТУ, 2019. – 200 с.
18. Козловський О. М. C#. Концепція та синтаксис : навч. посіб. / О. М. Козловський. – Львів : ЛНУ ім. Івана Франка, 2017. – 150 с.
19. The Best 21 Gym Management and Check-In Software for 2025 [Електронний ресурс] // Woven. – 2025. – Режим доступу: https://www.startwoven.com/articles/best-gym-management-software-compared. – (дата звернення: 14.05.2026).
20. Gym Owners Guide: My 5 Best Gym Software Solutions [Електронний ресурс] // Gymflow. – 2025. – Режим доступу: https://gymflow.io/blog-posts/gym-owners-guide-my-5-best-gym-software-solutions. – (дата звернення: 14.05.2026).
21. Which is the best software for gym management when compared side by side [Електронний ресурс] // Havok Journal. – 2026. – Режим доступу: https://havokjournal.com/fitness/which-is-the-best-software-for-gym-management-when-compared-side-by-side. – (дата звернення: 14.05.2026).
22. Essential Gym Management Software Features for Busy Gym Owners [Електронний ресурс] // WodGuru. – 2025. – Режим доступу: https://wod.guru/blog/gym-management-software-features. – (дата звернення: 15.05.2026).
23. Impact of Gym Management Software on the Fitness Industry [Електронний ресурс] // Booking Ninjas. – Режим доступу:

https://www.bookingninjas.com/blog/impact-of-gym-management-software-on-the-fitness-industry. – (дата звернення: 15.05.2026).

24. Best Gym Management Software for Enterprise Operators [Електронний ресурс] // Club Automation. – 2025. – Режим доступу: https://www.clubautomation.com/resources/best-gym-management-software-for-enterprise-operators. – (дата звернення: 15.05.2026).
25. Best Free Gym Management Software in 2026 (Top 7 Picks) [Електронний ресурс] // Recess. – Режим доступу: https://www.recess.tv/blog-posts/best-free-gym-management-software-in-2025-top-6-picks. – (дата звернення: 16.05.2026).
26. How Good Gym Management Software Improves Member Retention [Електронний ресурс] // Gymdesk. – 2023. – Режим доступу: https://gymdesk.com/blog/gym-management-software-improves-member-retention. – (дата звернення: 16.05.2026).
27. 9 benefits of gym management software you should be experiencing [Електронний ресурс] // Xplor Gym. – 2025. – Режим доступу: https://xplorgym.co.uk/blog/gym-management-software-benefits. – (дата звернення: 17.05.2026).
28. Why all-in-1 gym software is better [Електронний ресурс] // GymMaster. – 2024. – Режим доступу: https://www.gymmaster.com/blog/why-gym-management-system-bettter-than-separate-software-systems. – (дата звернення: 17.05.2026).
29. What is Windows Forms [Електронний ресурс] // Microsoft Learn. – 2025. – Режим доступу: https://learn.microsoft.com/en-us/dotnet/desktop/winforms/overview. – (дата звернення: 18.05.2026).
30. Windows Forms Applications - BEGINNER [Електронний ресурс] // Skillsoft. – Режим доступу: https://www.skillsoft.com/course/windows-forms-applications-2aea9eb6-e27c-11e6-93f3-0242c0a80605. – (дата звернення: 18.05.2026).
31. Introduction to Windows Forms [Електронний ресурс] // Pluralsight. – Режим доступу: https://www.pluralsight.com/courses/windows-forms-introduction-with-visual-basic. – (дата звернення: 18.05.2026).

Додаток А

Головний файл точки входу в програму Program.cs

```
using System;
using System.Windows.Forms;
using FitnessClubSystem.Forms;

namespace FitnessClubSystem
{
    static class Program
    {
        /// <summary>
        /// Головна точка входу для додатка.
        /// </summary>
        [STAThread]
        static void Main()
        {
            ApplicationConfiguration.Initialize();

            // Запуск головної форми авторизації користувачів системи
            Application.Run(new LoginForm());
        }
    }
}
```

Додаток Б

Реалізація сервісного класу управління даними Services/DataManager.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using FitnessClubSystem.Models;

namespace FitnessClubSystem.Services
{
    public class DataManager
    {
        private readonly FitnessClubContext _context;

        public DataManager(FitnessClubContext context)
        {
            _context = context ?? throw new ArgumentNullException(nameof(context));
        }

        /// <summary>
        /// Узагальнений метод завантаження всіх записів конкретної сутності з бази даних
        /// </summary>
        public List<T> LoadData<T>() where T : class
        {
            try
            {
                return _context.Set<T>().ToList();
```

```
        }
        catch (Exception ex)
        {
            // Логування помилки або виведення у відлагоджувальну консоль
            System.Diagnostics.Debug.WriteLine($"Помилка завантаження даних: {ex.Message}");
            return new List<T>();
        }
    }

    /// <summary>
    /// Узагальнений метод збереження або оновлення сутності в базі даних SQLite
    /// </summary>
    public void SaveData<T>(T entity) where T : class
    {
        if (entity == null) return;

        try
        {
            _context.Set<T>().Update(entity);
            _context.SaveChanges();
        }
        catch (Exception ex)
        {
            System.Diagnostics.Debug.WriteLine($"Помилка збереження даних: {ex.Message}");
            throw;
        }
    }
```

```
        /// <summary>
        /// Метод перевірки наявності та створення облікового запису адміністратора за замовчуванням
        /// </summary>
        public void InitializeDefaultAdmin()
        {
            try
            {
                // Перевірка, чи існують користувачі в таблиці Users
                if (!_context.Users.Any())
                {
                    _context.Users.Add(new User
                    {
                        Login = "admin",
                        PasswordHash = PasswordHelper.HashPassword("12345"),
                        Role = "Admin"
                    });
                    _context.SaveChanges();
                }
            }
            catch (Exception ex)
            {
                System.Diagnostics.Debug.WriteLine($"Помилка ініціалізації адміністратора: {ex.Message}");
            }
        }
    }
}
```

Додаток В

Реалізація утилітарного класу для хешування паролів

Services/PasswordHelper.cs

```
using System.Security.Cryptography;
using System.Text;

namespace FitnessClubSystem.Services
{
    public static class PasswordHelper
    {
        /// <summary>
        /// Метод генерації SHA-256 хешу для безпечного збереження пароля
        /// </summary>
        public static string HashPassword(string password)
        {
            using (SHA256 sha = SHA256.Create())
            {
                byte[] bytes = sha.ComputeHash(Encoding.UTF8.GetBytes(password));
                StringBuilder sb = new StringBuilder();
                foreach (byte b in bytes)
                    sb.Append(b.ToString("x2"));
                return sb.ToString();
            }
        }

        /// <summary>
        /// Метод верифікації введеного пароля із наявним хешем у базі даних
        /// </summary>
```

```
        public static bool VerifyPassword(string enteredPassword, string storedHash)
        {
            return HashPassword(enteredPassword) == storedHash;
        }
    }
}
```

Додаток Д

Реалізація графічної форми автентифікації користувачів Forms/LoginForm.cs

```
using System;
using System.Windows.Forms;
using FitnessClubSystem.Models;
using FitnessClubSystem.Services;

namespace FitnessClubSystem.Forms
{
    public partial class LoginForm : Form
    {
        public LoginForm()
        {
            InitializeComponent();
        }

        private void btnLogin_Click(object sender, EventArgs e)
        {
            string login = txtLogin.Text.Trim();
            string pass = txtPassword.Text;

            if (string.IsNullOrEmpty(login) || string.IsNullOrEmpty(pass))
            {
                MessageBox.Show("Введіть логін і пароль!", "Помилка",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
                return;
            }
```

```
            var users = DataManager.LoadData<User>("Users");
            var user = users.Find(u => u.Login == login);

            if (user != null && PasswordHelper.VerifyPassword(pass, user.PasswordHash))
            {
                Program.CurrentUser = user;
                this.Hide();
                MainForm main = new MainForm();
                main.Show();
            }
            else
            {
                MessageBox.Show("Невірний логін або пароль!", "Помилка",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                txtPassword.Clear();
                txtPassword.Focus();
            }
        }

        private void btnCancel_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }
    }
}
```

Додаток Е

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

Декларація про академічну доброчесність та використання ШІ

ДЕКЛАРАЦІЯ ЗДОБУВАЧА

Я, Сухецький Василь Геннадійович, здобувач НУБіП України, усвідомлюючи відповідальність за порушення академічної доброчесності, цією заявою підтверджую, що:

- Моя бакалаврська кваліфікаційна робота (проєкт) на тему «Програмне забезпечення інформаційної системи для фітнес-клубу» є результатом моєї особистої праці.
- Усі запозичення з текстів інших авторів мають відповідні посилання згідно з чинними стандартами.
- Щодо використання інструментів ШІ зазначаю, що (обрати необхідне):
    - [ ] інструменти генеративного ШІ не використовувалися.
    - [✓] інструменти ШІ Gemini, були використані для:
        - [ ] перекладу іншомовних джерел;
        - [✓] стилістичного редагування та перевірки граматики;
        - [✓] допомоги у написанні/відлагодженні програмного коду;
        - [ ] інше: _____.

Я розумію, що надання недостовірної інформації може призвести до скасування результатів захисту та відрахування з числа здобувачів НУБіП України.

« 22 » травня 2026 р.

(підпис)

Василь СУХЕЦЬКИЙ