

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАДИМА ГЕТЬМАНА
Навчально-науковий інститут
«Інститут інформаційних технологій в економіці»
Кафедра інформаційних систем в економіці**

ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА «КОМП'ЮТЕРНІ НАУКИ»
галузь знань 12 «Інформаційні технології»
спеціальність 122 «Комп'ютерні науки»

Форма навчання: денна

КВАЛІФІКАЦІЙНА БАКАЛАВРСЬКА РОБОТА

на тему: «Розроблення веб-застосунку для автоматизації взаємодії між
постачальниками послуг та клієнтами.»

здобувача Ляховича Ігоря Юрійовича _____

Науковий керівник :

к.т.н., доцент

_____Васильєва Л.В.

**Робота допущена до захисту перед
екзаменаційною комісією з атестації
здобувачів вищої освіти**

завідувач кафедри:

к.е.н., доцент

_____Тішков Б.О.

Київ 2025

Міністерство освіти і науки України
Київський національний економічний університет імені Вадима Гетьмана
Навчально-науковий інститут «Інститут інформаційних технологій в економіці»
Кафедра інформаційних систем в економіці

ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА «КОМП'ЮТЕРНІ НАУКИ»

галузь знань 12 «Інформаційні технології»

спеціальність 122 «Комп'ютерні науки»

ПОГОДЖЕНО:

Керівник проєктної групи(гарант)
освітньо-професійної програми

_____Помазун О.М.

“ _____ ” _____ 2025 р.

ЗАТВЕРДЖУЮ:

Завідувач кафедри

_____Тішков Б.О.

“ _____ ” _____ 2025 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

здобувача вищої освіти **Ляховича Ігоря Юрійовича**

очної (денної) форми навчання

на підготовку кваліфікаційної бакалаврської роботи

на тему: «Розроблення веб-застосунку для автоматизації взаємодії між постачальниками послуг та клієнтами»

Тему затверджено наказом ректора Університету від « 7 » березня 2025 р. № 466-ст.

Кваліфікаційна бакалаврська робота виконується на матеріалах
навчальних посібників та матеріалів з мережі Інтернет.

План кваліфікаційної бакалаврської роботи

Розділ I. Характеристика та аналіз предметної галузі

Розділ II. Розробка вимог і моделювання інформаційної системи

Розділ III. Проектування та реалізація компонентів системи

Об'єктом дослідження є процеси розробки веб-орієнтованих інформаційних систем, призначених для автоматизації взаємодії між постачальниками послуг та клієнтами.

Предметом дослідження є методи, інструменти та підходи до проектування і реалізації веб-додатків, які забезпечують двосторонню взаємодію між користувачами з різними ролями (клієнтами та постачальниками послуг)

Мета кваліфікаційної бакалаврської роботи розробка веб-застосунку, який автоматизує взаємодію між постачальниками послуг та клієнтами.

Конкретні завдання, які здобувач повинен виконати для досягнення поставленої мети:

У розділі I потрібно охарактеризувати і визначити об'єкт дослідження. Провести огляд літературних джерел і проаналізувати практичне застосування інформаційних систем у даній сфері.

У розділі II необхідно сформулювати вимоги до системи та її моделюванню. Деталізувати постановку задачі, описати вхідні й вихідні дані, навести алгоритм розв'язання. Побудувати моделі, що описують поведінку та структуру системи.

У розділі III потрібно здійснити проєктування та реалізацію основних компонентів. Описати інформаційне забезпечення, вибір СКБД, інфологічну та даталогічну моделі, а також програмні засоби, використані під час розробки.

**Завдання підготував
науковий керівник**

_____ **Васильєва Людмила Володимирівна**
«_10_» березня 2025 р.

**Завдання одержав
здобувач**

_____ **Ляхович Ігор Юрійович**
«_10_» березня 2025 р..

Відгук

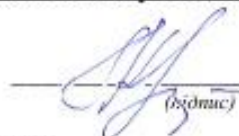
про кваліфікаційну бакалаврську роботу
здобувача навчально-наукового інституту
«Інститут інформаційних технологій в економіці»
освітньо-професійної програми
«Комп'ютерні науки»

Ляховича Ігоря Юрійовича

на тему: **«Розробка веб-додатку для автоматизації взаємодії між постачальниками послуг та клієнтами»**

1. Актуальність теми: Тема роботи спрямована на вирішення однієї з ключових задач сучасної сфери індивідуальних послуг – автоматизацію процесу онлайн-бронювання. У контексті зростаючого попиту на цифрові рішення, особливо серед малого та середнього бізнесу, створення єдиної платформи, є практично важливим і перспективним напрямом.
2. Позитивні риси кваліфікаційної бакалаврської роботи: Робота логічно побудована, містить аналіз предметної області та має орієнтацію на кінцевого користувача. Автор запропонував сучасну архітектуру системи, що дозволяє в реальному часі здійснювати пошук послуг, перегляд вільних слотів та бронювання у декількох постачальників одночасно. Особливо варто відзначити реалізацію роздільної логіки для клієнта і постачальника, що забезпечує зручність для обох сторін. Серед переваг створеної системи варто виділити інтуїтивно зрозумілий інтерфейс, високу продуктивність обробки запитів, а також адаптивність до потреб різних груп користувачів.
3. Наявність самостійних розробок автора: Здобувач самостійно реалізував ключові компоненти веб-додатку, що демонструють базову функціональність системи взаємодії між клієнтами та постачальниками послуг. Зокрема, на поточному етапі реалізовано механізм реєстрації та авторизації користувачів, вибору ролі, а також створення, перегляду й фільтрації послуг, які додаються постачальниками. Клієнти мають можливість переглядати доступні послуги від усіх постачальників і здійснювати запис.
4. Цінність теоретичних висновків та практичних рекомендацій: Робота має вагомий практичний значущість, оскільки результати можуть бути використані для створення реального веб-додатку, який сприятиме автоматизації взаємодії між клієнтами та постачальниками. Запропоноване рішення є універсальним і може адаптуватися до різних сфер індивідуальних послуг, забезпечуючи ефективну організацію онлайн-бронювання та керування послугами.
5. Наявність недоліків: реалізаційна частина потребує подальшого доопрацювання та завершення окремих функціональних модулів; не повністю реалізовані деякі нефункціональні вимоги, такі як підтримка багатомовності та забезпечення високого рівня безпеки даних.
6. Загальна оцінка кваліфікаційної бакалаврської роботи та її допущення до захисту перед ЕК: Представлена робота відповідає встановленим вимогам до кваліфікаційних бакалаврських робіт. Автор Ляхович І.Ю. продемонстрував здатність до самостійної розробки інформаційної системи, володіння навичками моделювання, аналізу та прототипування. Кваліфікаційна бакалаврська робота рекомендується до захисту в екзаменаційній комісії з позитивною оцінкою.

Науковий керівник



(підпис)

к.т.н., доцент Васильєва Л.В.

“10” ____ 06 ____ 2025 р.

Рецензія
на кваліфікаційну бакалаврську роботу
здобувача вищої освіти
Ляховича Ігоря Юрійовича
(прізвище, ім'я, по батькові)

Тема «Розробка веб-додатку для автоматизації взаємодії між постачальниками послуг та клієнтами»

Актуальність теми кваліфікаційної бакалаврської роботи і доцільність її розроблення: тема є актуальною, оскільки автоматизація взаємодії між клієнтом та постачальником послуг відповідає сучасним потребам цифрової трансформації та розвитку економіки. Доцільність її розроблення обґрунтована зростаючим попитом на онлайн-сервіси для запису, що забезпечують зручність, оперативність та ефективність взаємодії між учасниками процесу.

Якість проведеного дослідження: у роботі представлено чіткий аналіз предметної області, використано сучасні інструменти — Java, PostgreSQL, Bootstrap. Надано UML-діаграми, моделі бази даних, описано функціонал системи. Проект логічно структурований, технічно обґрунтований.

Позитивні риси кваліфікаційної бакалаврської роботи: проект демонструє високу реалістичність і практичність завдяки застосуванню сучасних технологій. Варто відзначити якісно підготовлену технічну документацію, що містить наочні діаграми та добре структуровану базу даних.

Зауваження: було б доцільно більше розповісти про тестування системи, також більш детально описати використані технології для розробки.

Практична значимість висновків і рекомендацій: розроблена система є фундаментом для майбутньої повноцінної системи, яку можна буде впровадити для всіх послуг малого та середнього бізнесу. Вона закладає основу для спрощення запису клієнтів, управління послугами та зберігання інформації, що в кінцевому підсумку позитивно вплине на ефективність роботи постачальників та швидкість запису клієнтів.

Зав. відділення «Програмування»
Відокремленого структурного підрозділу
«Фаховий коледж інформаційних
систем і технологій»

Світлана Миколаївна ЯНЧУК

“10” червня 2025 р.



АНОТАЦІЯ

кваліфікаційної бакалаврської роботи
здобувача першого (бакалаврського) рівня вищої освіти, 4 курсу, Ляховича
Ігоря Юрійовича

виконаної на тему: «Розроблення веб-застосунку для автоматизації
взаємодії між постачальниками послуг та клієнтами»

Київ: кафедра інформаційних систем в економіці, 2025 р.

Бакалаврська робота присвячена актуальній проблемі автоматизації взаємодії між клієнтами та постачальниками індивідуальних послуг шляхом створення сучасного веб-додатку. Розроблена система сприяє підвищенню ефективності обслуговування, скороченню часу на організаційні процеси, забезпеченню прозорості комунікації та цифровій трансформації сфери послуг.

Робота складається з трьох логічно пов'язаних розділів.

У першому розділі охарактеризовано і визначено об'єкт дослідження. Проведено огляд літературних джерел і проаналізовано практичне застосування інформаційних систем у даній сфері.

Другий розділ присвячено формуванню вимог до системи та її моделюванню. Деталізовано постановку задачі, описано вхідні й вихідні дані, наведено алгоритм розв'язання. Побудовано моделі, що описують поведінку та структуру системи.

У третьому розділі здійснено проєктування та реалізацію основних компонентів. Описано інформаційне забезпечення, вибір СКБД, інфологічну та даталогічну моделі, а також програмні засоби, використані під час розробки.

РЕФЕРАТ

Кваліфікаційна бакалаврська робота містить 117 сторінок, 11 таблиць, 41 рисунок, список використаних джерел з 25 найменувань, додатків.

«Розроблення веб-застосунку для автоматизації взаємодії між постачальниками послуг та клієнтами»

Перелік ключових слів: інформаційна система, веб-додаток, клієнт, постачальник, розробка, вимоги, діаграми, проектування.

Об'єктом дослідження є процеси розробки веб-орієнтованих інформаційних систем, призначених для автоматизації взаємодії між постачальниками послуг та клієнтами. Це включає бронювання, управління послугами, записами та організацію робочих процесів у цифровому середовищі, з урахуванням потреб малого і середнього бізнесу в Україні.

Предметом дослідження є методи, інструменти та підходи до проектування і реалізації веб-додатків, які забезпечують двосторонню взаємодію між користувачами з різними ролями (клієнтами та постачальниками послуг), зокрема функціональність реєстрації, створення та перегляду замовлень, управління профілями, фільтрація послуг, інтеграція з календарем тощо.

Метою дослідження є розробка веб-застосунку, який автоматизує взаємодію між постачальниками послуг та клієнтами. Застосунок має забезпечити ефективне управління послугами, зручне бронювання, перегляд і редагування записів, а також оптимізувати процеси комунікації та обслуговування. Рішення орієнтоване на потреби малого та середнього бізнесу в Україні.

Завданнями кваліфікаційного бакалаврської роботи є:

- Проаналізувати потреби клієнтів і постачальників послуг, а також сучасний стан ринку індивідуальних послуг з метою визначення ключових функцій веб-застосунку
- Дослідити існуючі аналогічні системи, призначені для автоматизації послуг.

- Сформувати вимоги до інформаційної системи та визначити основний функціонал веб-застосунку.
- Спроекувати зручний та інтуїтивно зрозумілий інтерфейс веб-додатку, орієнтований на максимальне задоволення потреб користувачів.
- Обґрунтувати вибір методологій для проєктування веб-застосунку.
- Визначити вхідні та вихідні дані, необхідні для функціонування системи.
- Сформувати структуру інформаційного забезпечення та здійснити його опис.
- Реалізувати основні компоненти веб-застосунку на програмному рівні.

Апаратні засоби, що використовувались при проєктуванні: Acer Nitro 5

Використані методи дослідження включали аналіз існуючих інформаційних систем, формування функціональних вимог, побудову моделей поведінки та структури системи, а також застосування сучасних програмних середовищ для реалізації ключового функціоналу.

Новизна полягає у проєктуванні веб-застосунку, який автоматизує взаємодію між клієнтами та постачальниками індивідуальних послуг, забезпечує зручний пошук, бронювання послуг і управління розкладом. Рішення спрямоване на підвищення ефективності обслуговування, оптимізацію часу та покращення досвіду користувача шляхом використання сучасних технологій і методів моделювання.

На даний момент система перебуває на етапі активної розробки, а остаточний рівень впровадження буде визначено після завершення всіх етапів проєкту.

Одержані результати можуть бути використані в рамках глобальної цифрової системи, що підтримує малий та середній бізнес в Україні. Розроблений веб-застосунок сприяє цифровій трансформації сфери послуг, підвищує ефективність взаємодії між клієнтами та постачальниками, автоматизує процеси бронювання й управління розкладом, що є особливо актуальним для локального сервісного бізнесу.

Рік виконання кваліфікаційної бакалаврської роботи – 2025.

Рік захисту роботи – 202

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ.....	4
ВСТУП.....	5
РОЗДІЛ 1 ХАРАКТЕРИСТИКА ТА АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ	9
1.1 Характеристика предметної галузі та об'єкта дослідження	9
1.2 Аналіз літературних джерел та практичного досвіду використання ІС і технологій в предметній галузі.....	13
РОЗДІЛ 2	17
РОЗРОБКА ВИМОГ І МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ.....	17
2.1 Аналіз і специфікація вимог до інформаційної системи	17
2.1.1 Бізнес-вимоги до системи	17
2.1.2 Функціональні вимоги до системи.....	18
2.1.3 Нефункціональні вимоги до системи.....	24
2.2 Постановка та алгоритм розв'язання задачі.....	25
2.2.1 Постановка задачі	25
2.2.2 Вихідні дані	27
2.2.3 Вхідні дані.....	31
2.2.4 Алгоритм розв'язання задачі	34
2.3 Моделювання інформаційної системи.....	38
2.3.2 Моделювання структури системи	40
РОЗДІЛ 3 ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ ...	42
3.1 Інформаційне забезпечення	42
3.1.1 Загальна характеристика інформаційного забезпечення.....	42
3.1.2 Організація збору і передання первинної інформації	46
3.1.3 Побудова системи класифікації та кодування	47
3.1.4 Структура інформаційних масивів.....	48
3.1.5 Вибір СКБД	55
3.1.6 Інфологічна модель бази (сховища) даних	56

3.1.7 Даталогічна модель бази (сховища)даних.....	57
3.2 Технічне забезпечення	59
3.2.1 Загальні положення та схема автоматизації.....	59
3.2.2 Структура комплексу технічних засобів	60
3.2.3 Опис автоматизованого робочого місця.....	63
3.3 Програмне забезпечення	64
3.3.1 Структура програмного забезпечення	64
3.3.2 Системне програмне забезпечення	66
3.3.3 Прикладне програмне забезпечення	68
3.3.4 Програмна документація.....	69
3.3 Результати реалізації інформаційної системи	70
ВИСНОВКИ.....	81
ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	83
ДОДАТКИ.....	86

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

IS – інформаційна система

СКБД – система керування базами даних

БД – на основі Даніеля

HTML – мова гіпертекстової розмітки

CSS – каскадні таблиці стилів

JS

UI – інтерфейс користувача

API – інтерфейс прикладного програмування

CRUD – Створення, читання, оновлення, видалення

ORM – об’єктно-реляційне відображення

REST – передача репрезентативного стану

ВСТУП

У сучасних умовах цифровізації важливим завданням є автоматизація взаємодії між постачальниками послуг та клієнтами. Існуючі рішення здебільшого орієнтовані на великі компанії або вузькі сфери, тоді як малий та середній бізнес, особливо в Україні, потребує доступних і гнучких платформ для організації онлайн-запису, управління послугами та комунікації з клієнтами.

У зв'язку з цим розробка веб-додатку, який дозволяє ефективно координувати попит і пропозицію в сервісній сфері, є актуальною. Враховуючи специфіку українського ринку, така система сприятиме підвищенню ефективності бізнесу та зручності для користувачів. Висока вартість існуючих міжнародних сервісів, відсутність їх локалізації під українські реалії та складність використання є суттєвими перепонами для малого та середнього бізнесу при впровадженні онлайн-бронювання.

Отже, існує потреба у створенні доступного, простого та локалізованого веб-додатку, який суттєво спростить комунікацію між постачальниками послуг і клієнтами, дозволить заощадити час та підвищить ефективність роботи малого й середнього бізнесу в Україні.

Сучасні дослідження у сфері інформаційних систем зосереджуються на принципах гнучкості, масштабованості та орієнтації на кінцевого користувача. Зокрема, Рой Філдінг у своїй докторській дисертації заклав теоретичні основи REST-архітектури, яка стала фундаментом для розробки веб-додатків [1]. Мартін Фаулер у праці *Patterns of Enterprise Application Architecture* описав архітектурні патерни, що забезпечують надійність та масштабованість прикладного програмного забезпечення [2, с. 285]. У контексті українських реалій актуальною є праця Соловйової Н. В. та Руденка А. О., присвячена цифровій трансформації малого бізнесу в Україні, де підкреслюється необхідність локалізованих цифрових рішень [3].

Попри наявність міжнародних комерційних сервісів, таких як Fresha[5] чи Booksy [4], залишається не вирішеною проблема створення безкоштовного українського веб-додатку, адаптованого до потреб вітчизняного малого й середнього бізнесу. Ця кваліфікаційна робота спрямована на вирішення саме цієї проблеми.

Метою цього дослідження є розробка веб-застосунку, який автоматизує взаємодію між постачальниками послуг та клієнтами. Застосунок має забезпечити ефективне управління послугами, зручне бронювання, перегляд і редагування записів, а також оптимізувати процеси комунікації та обслуговування. Рішення орієнтоване на потреби малого та середнього бізнесу в Україні, який потребує доступних і гнучких цифрових інструментів.

Основними завданнями, необхідними для досягнення мети, є:

1. Провести аналіз предметної області, визначити актуальні проблеми та особливості організації сервісної взаємодії між клієнтами і постачальниками послуг. Ознайомитися з існуючими інформаційними системами в цій сфері.
2. Сформулювати функціональні та нефункціональні вимоги до системи, описати алгоритми її роботи, а також побудувати відповідні діаграми (використання, класів, активності тощо) для моделювання системи.
3. Спроекувати архітектуру веб-застосунку та реалізувати його основні компоненти з використанням сучасних технологій (Spring Boot, PostgreSQL, Thymeleaf, HTML/CSS/JS). Забезпечити інтеграцію з календарем, фільтрацію послуг та розмежування прав доступу для клієнтів і постачальників.

Розроблений веб-застосунок має практичну цінність, оскільки сприяє цифровій трансформації сервісної сфери, підвищує ефективність роботи підприємств і покращує досвід користувачів. Його впровадження дозволить скоротити час взаємодії, мінімізувати помилки в бронюванні, підвищити якість обслуговування та сприяє підвищенню конкурентоспроможності бізнесу. Об'єктом дослідження є процеси розробки веб-орієнтованих інформаційних систем, призначених для автоматизації взаємодії між постачальниками послуг та

клієнтами. Така взаємодія охоплює бронювання, управління послугами, записами та організацію робочих процесів у цифровому середовищі. Особливу увагу приділено створенню рішень, адаптованих до потреб малого і середнього бізнесу в Україні, з урахуванням обмежених ресурсів і високих вимог до зручності користування.

Предметом дослідження є методи, інструменти та підходи до проєктування і реалізації веб-додатків, які забезпечують двосторонню взаємодію між користувачами з різними ролями (клієнтами та постачальниками послуг), включаючи функціональність реєстрації, створення та перегляду замовлень, управління профілями, фільтрації послуг, інтеграції з календарем тощо.

У ході дослідження застосовано комплекс методів, що забезпечують всебічний аналіз та реалізацію поставлених завдань. Теоретичний аналіз предметної області проведено за допомогою систематизації та узагальнення наукових джерел, вітчизняних і зарубіжних публікацій, а також огляду існуючих інформаційних систем. Для моделювання системи використано методи об'єктно-орієнтованого проєктування з побудовою UML-діаграм (використання, класів, активності). Практична частина базується на застосуванні сучасних технологій розробки веб-застосунків (Spring Boot, PostgreSQL, Thymeleaf, HTML/CSS/JS), що забезпечують реалізацію функціоналу автоматизації взаємодії між клієнтами та постачальниками послуг.

Теоретична значущість полягає у систематизації сучасних підходів до розробки веб-орієнтованих інформаційних систем для автоматизації взаємодії в сервісній сфері, з урахуванням особливостей малого і середнього бізнесу в Україні. У роботі розглянуто основи REST-архітектури та патернів корпоративних додатків, адаптованих під локальні умови.

Методична значущість полягає у розробці підходу до моделювання та створення веб-застосунку з урахуванням потреб малого і середнього бізнесу, що може бути використаний як приклад для подальших розробок у цій області.

Практична значущість полягає у створенні функціонального веб-застосунку, який дозволяє автоматизувати процеси бронювання, управління послугами та комунікації між клієнтами і постачальниками, що може сприяти підвищенню ефективності бізнесу та покращенню обслуговування на українському ринку.

Інформаційна база дослідження охоплює наукові праці вітчизняних та зарубіжних фахівців (зокрема Р. Філдінга, М. Фаулера), навчальні посібники з проектування інформаційних систем і баз даних, публікації у фахових виданнях щодо цифровізації економіки, а також сучасні інтернет-ресурси. Використано офіційні сайти провайдерів послуг (Booksy, Fresha, BeautyPro), приклади системного проектування (UML-діаграми, архітектурні стилі, тришарова архітектура), інструменти проектування (Enterprise Architect), документацію з використання PostgreSQL, JPA, Spring Boot та MVC-підходу, а також онлайн-матеріали з функціональних і нефункціональних вимог, хмарних технологій і веброзробки.

Робота складається зі вступу, трьох розділів, висновків, списку використаних джерел та додатків.

РОЗДІЛ 1 ХАРАКТЕРИСТИКА ТА АНАЛІЗ СФЕРИ ІНДИВІДУАЛЬНИХ ПОСЛУГ

1.1 Характеристика предметної галузі та об'єкта дослідження

Предметна галузь дослідження охоплює сферу індивідуальних послуг в Україні, до якої належать такі види діяльності, як перукарські та косметичні послуги (манікюр, педикюр), клінінгові послуги, ремонтні та побутові роботи, а також інші послуги, що надаються безпосередньо клієнтам. Основними типами підприємств та організацій, що працюють у цій сфері, є салони краси та перукарні, клінінгові компанії, майстерні з ремонту побутової техніки та електроніки, сервісні служби з ремонту меблів та сантехніки, агентства домашнього обслуговування, фітнес-центри та тренажерні зали, навчальні студії, служби доставки, технічні майстерні та інші підприємства, що надають послуги клієнтам. Ця галузь є ключовою складовою малого та середнього бізнесу, вона формує значну частку економіки країни, створює робочі місця та забезпечує попит на різноманітні сервіси для населення.

Економічна сутність задачі, яку було розглядано у рамках цього дослідження, полягає у підвищенні ефективності бізнес-процесів постачальників індивідуальних послуг за рахунок впровадження сучасних цифрових технологій. Зокрема, автоматизація процесів онлайн-запису клієнтів, управління графіками надання послуг, а також налагодження комунікації між клієнтами та постачальниками, що суттєво спрощує організаційні питання та дозволяє оптимізувати використання ресурсів.

У сфері індивідуальних послуг приймаються різні рішення, які спрямовані на оптимізацію обслуговування клієнтів, ефективне планування робочого часу, управління навантаженням персоналу та підтримку високого рівня сервісу. Інколи можуть виникати проблемні ситуації, унаслідок нерівномірного розподілу записів, скасування послуг в останній момент, неузгодженості графіків або браку комунікації між клієнтами й постачальниками.

У цій кваліфікаційній бакалаврській роботі здійснено розробку веб-застосунку — централізованої інформаційної системи для автоматизації взаємодії між постачальниками послуг і клієнтами. Система включатиме такі модулі:

- реєстрація користувачів та постачальників послуг;
- створення послуг;
- перегляд послуг;
- управління профілем;
- управління графіками;
- онлайн-запис;
- сповіщення про зміни;
- ведення клієнтської бази та статистики.

Об'єктом дослідження є інформаційна система для управління індивідуальними послугами, а предметом — технології, що забезпечують ефективну взаємодію між постачальниками послуг і клієнтами через автоматизацію. Робота передбачає розробку прототипу, який може бути використаний як державна система для підтримки малого бізнесу в Україні, сприяючи автоматизації та покращенню взаємодії між постачальниками послуг і клієнтами.

Інформаційна система для управління індивідуальними послугами базуватиметься на кількох ключових ролях:

- адміністратор;
- постачальник послуг;
- клієнт;

Кожна з визначених ролей у системі має власний набір функціональних можливостей, що відповідають їхнім завданням та зонам відповідальності (табл. 1.1.). Це дозволить ефективно розподіляти права доступу, забезпечувати зручність користування та підтримувати високу якість обслуговування для всіх учасників процесу.

Таблиця 1.1 – Функціональна структура

Робоче місце	Функції
Адміністратор системи	- Моніторинг роботи системи та баз даних.
	- Забезпечення стабільної роботи платформи.
	- Оновлення програмного забезпечення та резервне копіювання даних.
Постачальник послуг	- Управління графіками надання послуг.
	- Оновлення доступних послуг та їх опису.
	- Перегляд записів клієнтів та коригування розкладу.
Клієнт	- Реєстрація та управління профілем користувача.
	- Запис на послугу через онлайн-форму.
	- Отримання сповіщень про зміни в графіку або статусі замовлення.
Спеціаліст техпідтримки	- Надання консультацій користувачам через чат, електронну пошту або телефон.
	- Вирішення технічних проблем, що виникають у процесі запису або використання системи.
	- Ведення бази знань для вирішення поширених запитів та проблем.

Джерело: сформоване автором на основі виконаних досліджень

У розробленій системі множина інформації включає такі сутності, як користувач(клієнт, постачальник), послуга, бронювання клієнта, розклад постачальника, слоти для розкладу. Всі ці сутності взаємопов'язані через систему баз даних, що забезпечує збереження та обробку даних про кожну з них. Взаємодія між різними ролями системи отримується зарахунок обміну інформацією про клієнтів, послуги, постачальників, що гарантує безперебійність та точність роботи платформи.

Проаналізовано існуючі інформаційні системи для управління послугами і виявлено їхні недоліки, такі як недостатня популярність, повільна обробка запитів та недостатня підтримка клієнтів. Це може призводити до затримок у виконанні записів і відсутності необхідних послуг, що негативно впливає на якість обслуговування. Тому розробляється система з високою швидкістю обробки запитів, покращеною підтримкою користувачів і широкою популярністю серед клієнтів яка буде отримана зарахунок орієнтованості на український ринок

Запроваджено рішення, що допомагають підвищити якість сервісу, збільшити задоволеність клієнтів і сприяти цифровізації економіки, підтримуючи малий бізнес в Україні та створюючи зручні умови для користувачів.

У системі яка автоматизує взаємодію між клієнтами та постачальниками реалізовано переважно рішення закритого типу з детермінованою логікою для автоматизації запису та управління графіками. Водночас враховано відкриті ситуації, пов'язані з ризиками скасування та непередбачуваною поведінкою користувачів, що потребує адаптивності. Передбачено оперативне втручання у кризових ситуаціях, таких як технічні збої чи конфлікти в розкладах. Таким чином, система поєднує детерміновані рішення з прийняттям рішень за умов ризику та нечітких цілей, забезпечуючи надійну та гнучку роботу в динамічному середовищі індивідуальних послуг.

1.2 Аналіз літературних джерел та практичного досвіду використання ІС і технологій в предметній галузі

У сучасних умовах цифровізації сфери послуг більшої актуальності набуває проблема ефективної автоматизації процесів взаємодії між клієнтами та постачальниками індивідуальних послуг. Аналіз літературних джерел та практики впровадження інформаційних систем у цій галузі свідчить про існування різноманітних підходів до побудови сервісів онлайн-запису, управління графіками, клієнтською базою та внутрішньою логістикою обслуговування.

Серед найпоширеніших цифрових рішень у сфері надання індивідуальних послуг можна виокремити такі системи, як Booksy [4], Fresha [5], SimplyBook.me, Treatwell, а також локальні українські сервіси на кшталт Beauty Pro [24], Zapis тощо. У сучасних зарубіжних дослідженнях розглядаються практики застосування хмарних платформ [6], що надають постачальникам сервісів повний цикл управління — від планування розкладів до обробки платежів та збору зворотного зв'язку. У цих роботах акцент зроблено на ефективність алгоритмів планування, автоматизацію обробки запитів та забезпечення якості обслуговування.

- обмежена локалізація (відсутність україномовного інтерфейсу);
- залежність від іноземної технічної підтримки;
- недостатня адаптація до українських умов ведення малого бізнесу;
- низька гнучкість налаштувань для мікропідприємств.

Вітчизняні сервіси, хоч і враховують специфіку місцевого ринку, часто не витримують навантаження або мають обмежений функціонал. Проблеми, що фіксуються в оглядах користувачів та тематичних статтях, включають: повільну обробку запитів, відсутність інтеграції з популярними платіжними системами та мізерну технічну підтримку.

Таблиця 1.2 – Схожі системи

Критерій	Booksy	Fresha	SimplyBook.me
Платформа	Веб, iOS, Android	Веб, iOS, Android	Веб, iOS, Android
Цільова аудиторія	Салони краси, барбершопи, майстри	Салони, масажисти, незалежні фахівці	Універсальна: будь-які послуги за записом
Локалізація (українська мова)	Немає	Немає	Часткова підтримка
Інтеграція з платіжними системами	Так (Stripe, Square)	Так (Stripe, інші)	Так (Stripe, PayPal, інші)
Технічна підтримка	24/7(англійською мовою)	24/7(англійською мовою)	Онлайн-підтримка
Гнучкість налаштувань	Обмежена	Обмежена	Висока: плагіни, адаптація інтерфейсу
Хмарне зберігання даних	Так	Так	Так
Ціноутворення	Платна модель, пробний період	Безкоштовна модель + комісія з платежів	Безкоштовна + платні плагіни
Аналітика та звітність	Є	Є	Є
Мобільний додаток для клієнтів	Є	Є	Є
Надійність при навантаженні	Висока	Висока	Може бути обмежена в безкоштовній версії
Підтримка локального бізнесу (UA)	Не враховано специфіку українського ринку	Не враховано специфіку українського ринку	Частково враховано

Джерело: сформоване автором на основі виконаних досліджень

Аналіз структури інформаційних систем у зазначеній галузі вказує, що повноцінне вирішення задач планування, запису, комунікації та підтримки потребує не лише створення базового вебінтерфейсу, але й продуманої

багаторольової архітектури з чітким розподілом функціональних обов'язків між користувачами. Типова структура успішних систем передбачає такі компоненти:

- модуль реєстрації та автентифікації з багаторівневим доступом;
- інтерфейс управління графіками з адаптивним календарем;
- система сповіщень, інтегрована з e-mail, SMS або месенджерами;
- аналітичний модуль, що формує звіти для постачальника;
- чат або інтеграція з підтримкою, яка дозволяє швидко реагувати на проблеми.

проблеми.

Методи, які застосовуються для побудови таких систем, часто включають використання RESTful API для клієнт-серверної взаємодії, баз даних типу PostgreSQL або MongoDB, а також фреймворків (Spring Boot, React, Angular тощо), які забезпечують швидкість розробки та масштабованість.

З точки зору інформаційних потоків, системи повинні забезпечувати обробку вхідних даних (запитів на запис, змін профілю, скасування) та вихідних (сповіщень, звітів, оновлень графіків). Структура цих даних має бути стандартизована, щоб мінімізувати помилки під час обробки.

Ключовими користувачами такої системи є:

- постачальники послуг, які приймають рішення про розклад, типи послуг, ціни;
- клієнти, які ініціюють запити та оцінюють якість обслуговування;
- адміністратори, які відповідають за коректність роботи системи;
- розробники та технічна підтримка, що вирішують критичні ситуації та впроваджують оновлення.

На основі аналізу можна дійти висновку, що незважаючи на наявність певних рішень на ринку, потреба в адаптованих, локалізованих і технологічно гнучких системах залишається актуальною. Саме цю нішу покликаний заповнити запропонований у кваліфікаційній роботі веб-застосунок.

Таким чином, сформульовані такі проблемні питання:

Як забезпечити стабільну роботу системи при зростанні навантаження?

Яким чином реалізувати інтерфейс, зручний як для клієнтів, так і для постачальників?

Вирішення цих питань обумовлює цілі та задачі дослідження, які полягають у розробці адаптивної інформаційної системи управління індивідуальними послугами з урахуванням сучасних вимог, українського контексту та технологічного розвитку.

РОЗДІЛ 2

РОЗРОБКА ВИМОГ І МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1 Аналіз і специфікація вимог до інформаційної системи

2.1.1 Бізнес-вимоги до системи

Усі вимоги до застосунку поділено на 3 групи: бізнес-вимоги, функціональні вимоги та нефункціональні вимоги.

Бізнес-вимоги визначають основну функціональність та вимоги, що стосуються діяльності додатку з автоматизації. Для цього була розроблена діаграма бізнес-вимог (рисунок 2.1), яка відображає основні аспекти функціонування та вимоги до системи з точки зору бізнесу.

У бізнес-вимогах було відокремлено чотири основні вимоги.

1. Має бути підвищена зручність замовлення послуг.
2. Має бути реалізовано спрощене управління розкладом.
3. Має зменшитись кількість ручних дій.
4. Має реалізована бути комунікація між сторонами автоматична.

Для уточнення вимог основні вимоги були розбиті на підвимоги.

Підвищена зручність замовлення послуг:

- Має бути пошук та запис на послуги.
- Має бути календар доступності.

Спрощене управління розкладом:

- Має бути функції для створення та редагування послуг.
- Має бути можливість управління вільним часом.

Зменшення ручних дій:

- Має бути автоматичне збереження.

Комунікації між сторонами:

- Має бути повідомлення про запис.
- Має бути повідомлення про відмову.

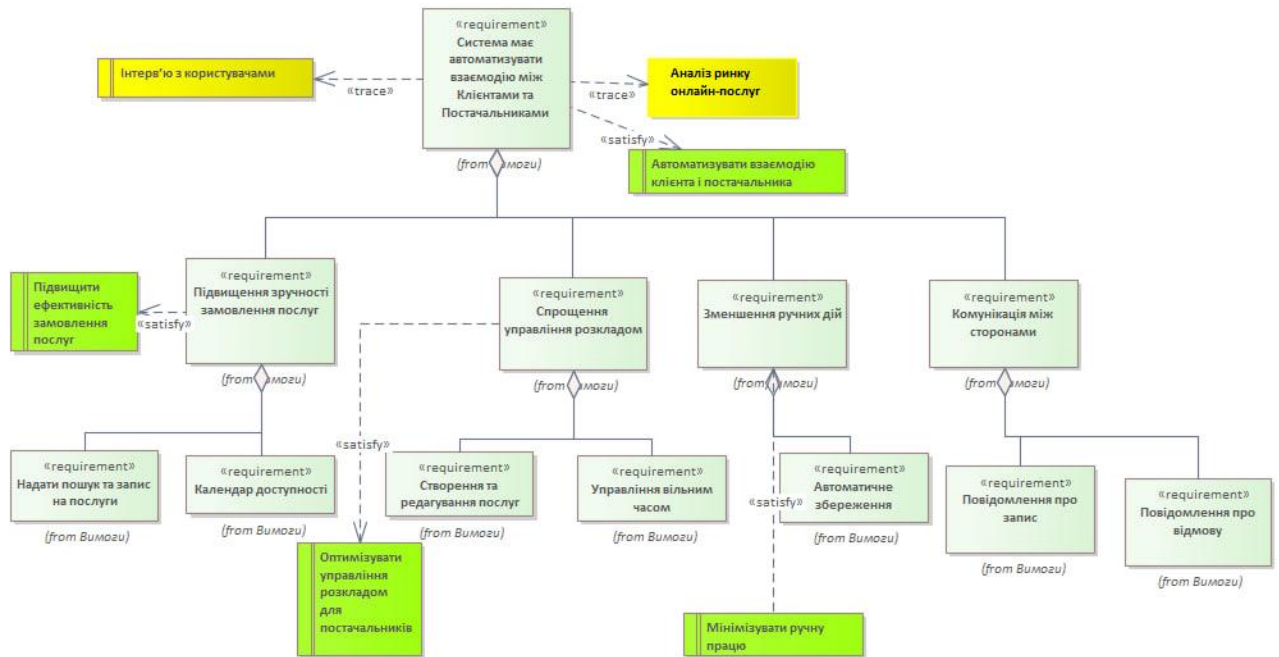


Рисунок 2.1 - Бізнес-вимоги

Джерело: розроблено автором на основі виконаних досліджень

Жовтим кольором позначені джерела.

Зеленим кольором позначені бізнес-цілі ,які виконуються .

2.1.2 Функціональні вимоги до системи

Функціональна вимога - це формулювання того, як система повинна поводитися. Він визначає, що повинна робити система, щоб задовольнити потреби або очікування користувача. Функціональні вимоги можна розглядати як функції, які виявляє користувач. [7] Детальний виклад цих вимог міститься у Специфікації

вимог до програмного забезпечення (Software Requirements Specification), де вони документуються з необхідним ступенем повноти. Цей документ (SRS) є основою і використовується на різних етапах життєвого циклу проєкту, включаючи процеси розробки, тестування, забезпечення якості продукту, управління проєктом та інші супутні функції. У даній моделі представлено 25 функціональні вимоги, які розподілені між вісьмома логічними групами (підпакетами).

Нижче наведені основні функціональні вимоги до системи.

1. SPA-навігація: Сайт має SPA-навігацію між сторінками: Головна, Профіль, Послуги, Мої записи.
2. Автоматичний вхід: Успішна реєстрація автоматично входить користувача в систему та перенаправляє в кабінет.
3. Бронювання послуги користувачем: Користувач бронює послуги.
4. Вибір ролі користувача: Користувач має обрати роль: Клієнт або Постачальник послуг при відкритті сайту.
5. Видалення бронювань: Постачальник і клієнт може видалити бронювання.
6. Видалення послуг: Можливість видалення записаних послуг зі списку.
7. Додавання послуг: Клієнт може додавати послуги до записів.
8. Календар: Головна сторінка містить календар поточного місяця.
9. Календар постачальника: Постачальник працює з календарем: задає слоти, редагує чи видаляє їх.
10. Нагадування про оновлення: Блок "Сьогоднішні записи" показує записи на поточну дату або повідомляє про їх відсутність.
11. Перегляд замовлень: Користувач може переглядати/фільтрувати замовлення за датами.
12. Перегляд записів постачальника: Постачальник бачить записи в хронологічному порядку з можливістю фільтрації за діапазоном дат.
13. Пошуковий рядок: Пошуковий рядок на всіх сторінках — пошук за ключовими словами серед послуг.

14. Редагування даних компанії: Постачальник може редагувати дані компанії.
15. Редагування особистих даних: Користувач може редагувати особисті дані: ім'я, прізвище, вік, контакти.
16. Редагування послуг: Можливість редагування кожної послуги.
17. Реєстрація: Форма для реєстрації з відповідними полями.
18. Рекомендовані послуги: У профілі клієнта — блок з рекомендованими послугами на основі історії.
19. Список послуг: Відображення списку доступних послуг із повною інформацією.
20. Список послуг постачальника: Постачальник бачить список власних послуг.
21. Статистика постачальника: Відображення статистики постачальника за останній період у вигляді ключових показників.
22. Створення нової послуги: Створення нової послуги через форму, з миттєвим додаванням до списку.
23. Сьогоднішні записи: Постачальнику система нагадує про потребу оновити розклад, якщо є прогалини (Ймовірно, це стосується Календаря постачальника або Нагадування про оновлення).
24. Форма входу: Після вибору ролі відкривається форма входу з кнопками "Увійти" та "New account".
25. Швидкість пошуку: Результати пошуку повинні з'являтися до 500 мс, при відсутності збігів — повідомлення.

Для візуалізації було вирішено реалізувати таку діаграму функціональних вимог у вигляді 8 пакетів які мають в собі всі функціональні вимоги на мові SysML в середовищі Enterprise Architect.

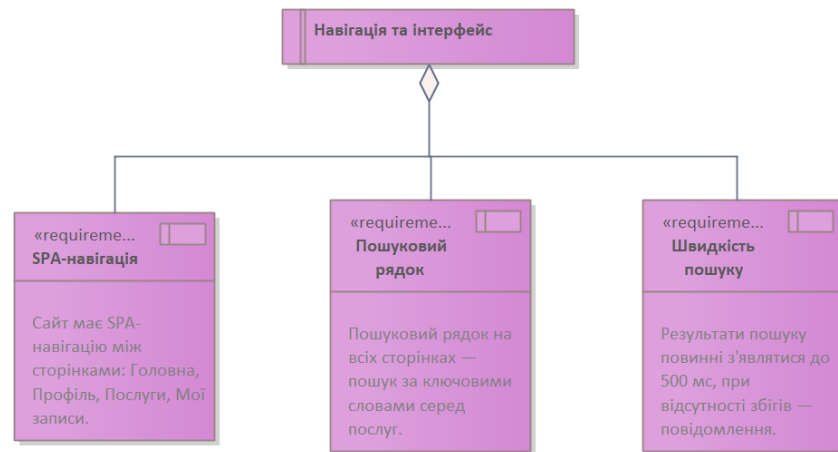


Рисунок 2.2 – Пакет функціональних вимог до навігації та інтерфейсу

Джерело: розроблено автором на основі виконаних досліджень

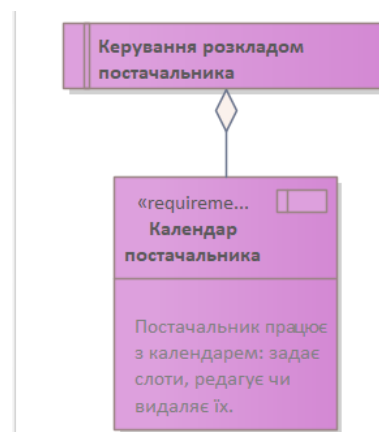


Рисунок 2.3 – Пакет функціональних вимог керування розкладом постачальника

Джерело: розроблено автором на основі виконаних досліджень

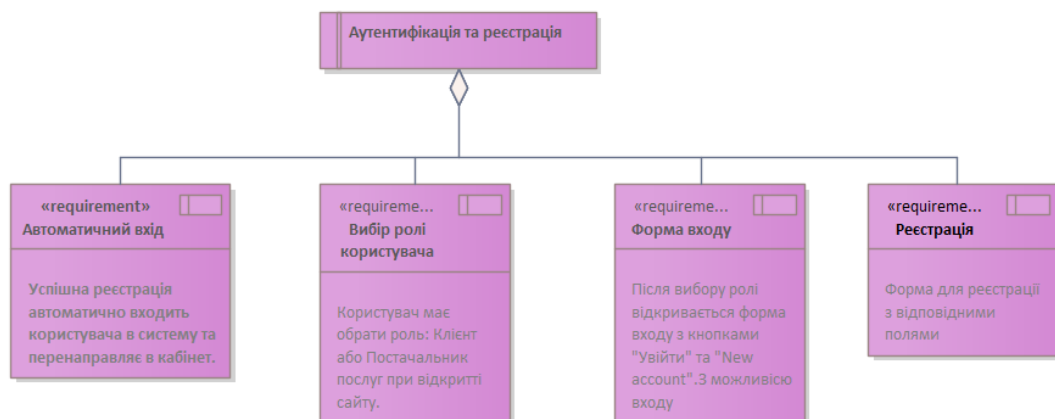


Рисунок 2.4 – Пакет функціональних вимог до аутентифікації та реєстрації

Джерело: розроблено автором на основі виконаних досліджень

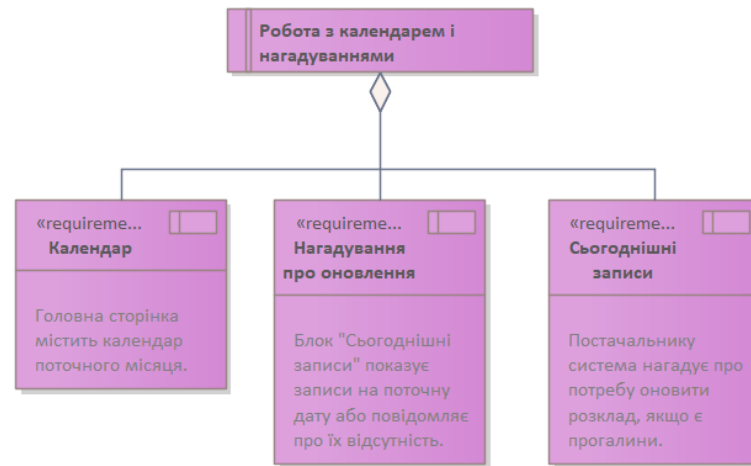


Рисунок 2.5 – Пакет функціональних вимог до роботи з календарем та нагадуванням
Джерело: розроблено автором на основі виконаних досліджень

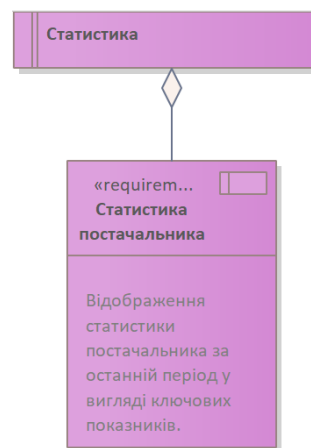


Рисунок 2.6 – Пакет функціональних вимог статистика
Джерело: розроблено автором на основі виконаних досліджень



Рисунок 2.7 – Пакет функціональних вимог управління записами
Джерело: розроблено автором на основі виконаних досліджень

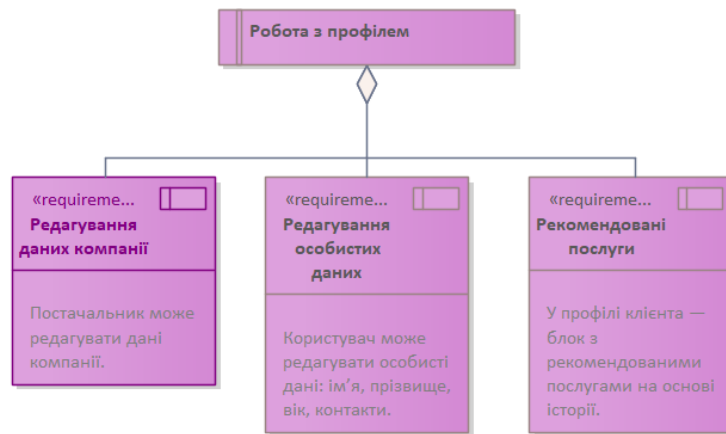


Рисунок 2.8 – Пакет функціональних вимог робота з профілем

Джерело: розроблено автором на основі виконаних досліджень

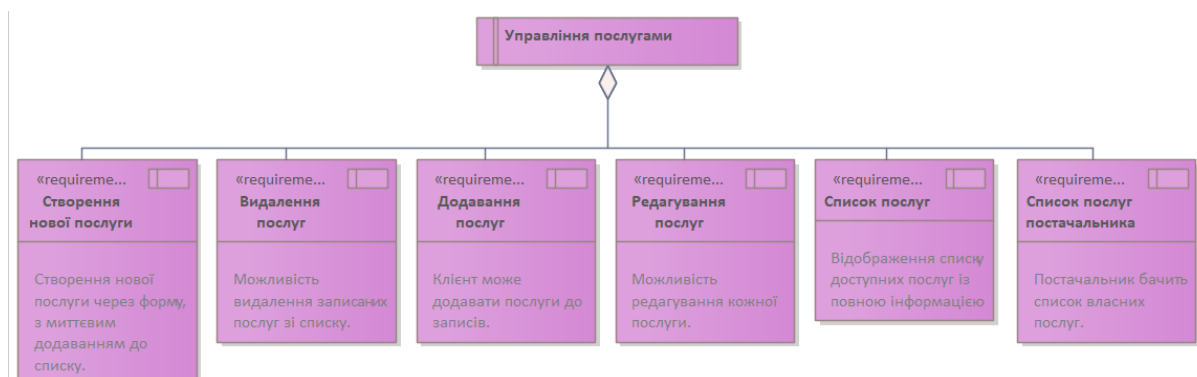


Рисунок 2.9 – Пакет функціональних вимог для управління послугами

Джерело: розроблено автором на основі виконаних досліджень

У данній роботі головна діаграма функціональних вимог не представлена. Проте наведені вісім функціональних підсистем які є складовими частинами загальної цілі функціональних вимог інформаційної системи.

Ці підсистеми відображають ключові напрями роботи застосунку, які спрямовані на досягнення основної мети — автоматизації взаємодії між клієнтом та постачальником послуг.

Отже, кожна підсистема — аутентифікація та реєстрація, керування профілем користувача та компанії, керування послугами, бронювання та робота із записами, календар і розклад, пошук і фільтрація, статистика постачальника, SPA-навігація та інтерфейс — є невід’ємною складовою загальної функціональної структури системи.

Разом вони формують цілісну багаторівневу систему функціональних вимог, що забезпечує реалізацію бізнес-логіки і відповідає потребам користувачів у зручному та ефективному автоматизованому сервісі.

2.1.3 Нефункціональні вимоги до системи

Нефункціональні вимоги (Non-Functional Requirements, NFR) визначають якісні характеристики системи, що впливають на її продуктивність, безпеку, зручність використання, масштабованість та відповідність нормативним актам. Відповідно до стандарту ISO/IEC 25000, такі вимоги відносяться до категорії вимог якості системи та програмного забезпечення [8].

Основні нефункціональні вимоги, що покладені в основу розробки системи автоматизації взаємодії між постачальниками послуг та клієнтами, наведені нижче:

1. Продуктивність

Час відгуку ключових операцій не повинен перевищувати 2 секунди у 95% випадків, що забезпечує комфортну роботу користувача.

Система має підтримувати не менше 100 одночасних активних сесій без зниження продуктивності та виникнення збоїв.

2. Безпека

Доступ до системи здійснюється лише після успішної автентифікації користувача (логін/пароль або альтернативний метод).

Запроваджено контроль доступу до даних і функцій на основі ролей користувачів (адміністратор, клієнт, постачальник).

3. Зручність використання

Інтерфейс користувача повинен бути інтуїтивно зрозумілим, що дозволяє мінімізувати час навчання.

Забезпечується повна адаптація інтерфейсу до мобільних пристроїв (смартфони, планшети).

4. Масштабованість

Архітектура системи передбачає можливість додавання нових функцій і збільшення обсягів даних без необхідності повного перепроєктування.

Мобільна адаптивність сприяє розширенню кола користувачів за рахунок підтримки різних платформ.

5. Відповідність законодавству

Система реалізована з урахуванням вимог Закону України «Про захист персональних даних» та інших нормативних актів у сфері інформаційної безпеки.

Обробка та зберігання даних здійснюються відповідно до чинного українського законодавства.

Для наочності нефункціональних вимог розроблено діаграму на мові SysML у середовищі Enterprise Architect (рис. А.1, Додаток А). Центральним елементом діаграми є компонент «Вимоги до якості», що агрегує ключові аспекти: продуктивність, безпеку, масштабованість, зручність та відповідність законодавству. Така візуалізація дозволяє комплексно оцінити якісні параметри системи, необхідні для забезпечення її стабільної та безпечної роботи, а також задоволення потреб кінцевих користувачів.

2.2 Постановка та алгоритм розв'язання задачі

2.2.1 Постановка задачі

Актуальність розробки даної системи зумовлена необхідністю автоматизації процесів запису клієнтів на індивідуальні послуги в малих та середніх

підприємствах. Впровадження електронно-обчислювальних засобів дозволяє значно підвищити ефективність управління бронюваннями, зменшити витрати часу та ресурсів, а також покращити якість взаємодії між клієнтами і постачальниками послуг. Об'єктами управління є клієнти, постачальники, послуги, бронювання та профілі користувачів. Вихідна інформація системи використовується для відображення персональних даних, підтвердження дій користувачів, виведення списків послуг, результатів пошуку та статистичних даних. Обробка інформації відбувається в режимі реального часу за запитом користувача з миттєвою видачею результатів. Автоматизація процесів припиняється при втраті зв'язку з сервером, завершенні сесії або виникненні технічних неполадок. У майбутньому система повинна бути інтегрована з іншими інформаційними ресурсами, зокрема платіжними та CRM-системами, що дозволить розширити функціональність та підвищити якість обслуговування користувачів. Розподіл функцій між користувачами, адміністраторами та технічними засобами передбачається з метою забезпечення стабільності та безперервності роботи системи. В інформаційній моделі задачі на рис. 2.10. відображені вихідні та вхідні повідомлення, а також процеси, що перетворюють вхідні повідомлення на вихідні, та зовнішні сутності, що взаємодіють із системою.

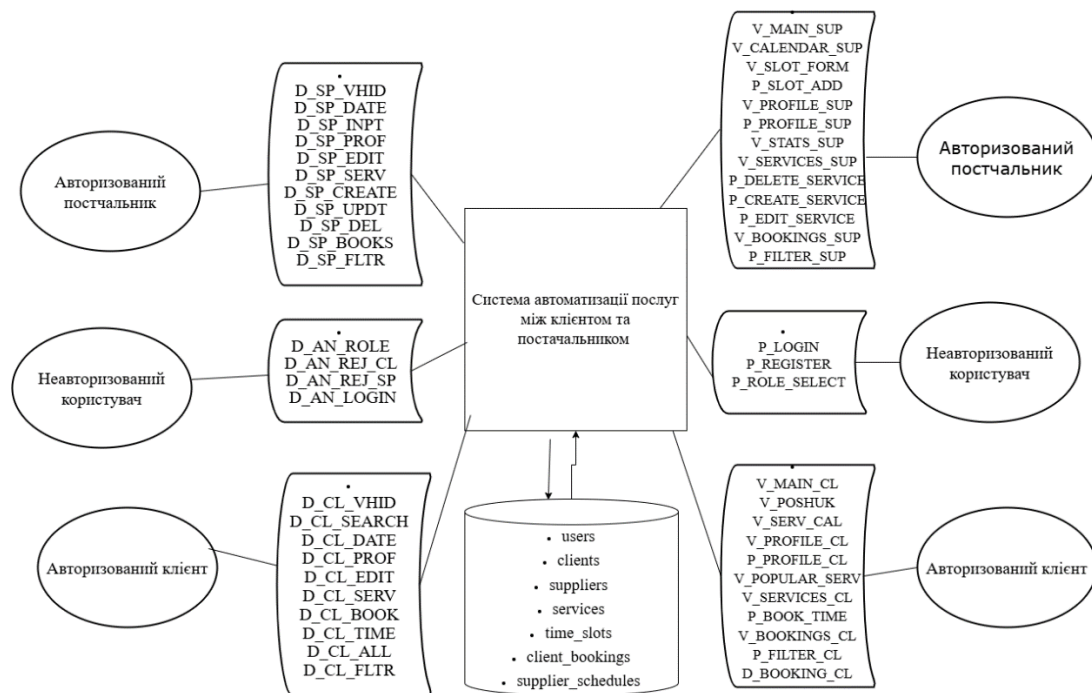


Рисунок 2.10 – Інформаційна модель задачі

Джерело: розроблено автором на основі виконаних досліджень

2.2.2 Вихідні дані

Вихідна інформація системи складається з результатів взаємодії клієнтів та постачальників з функціональними можливостями платформи. Вона включає підтвердження операцій, відомості про стан облікового запису, результати пошукових запитів, деталізовані відомості про об'єкти пошуку, а також повідомлення про зміни та оновлення, що відбуваються в системі.

Зокрема, до вихідної інформації належать такі типи даних:

Підтвердження успішного входу в систему, яке отримує користувач після введення коректних облікових даних. Цей процес забезпечує контроль доступу та безпеку інформації.

Відомості про профіль користувача, які система формує та відображає відповідно до дій користувача. Ці дані доступні як для самого користувача, так і для адміністративних модулів системи.

Результати пошукових операцій, які формуються на основі введених критеріїв та відображають релевантні записи. Ця інформація призначена для кінцевого користувача та інших зацікавлених сторін.

Детальна інформація про окремі послуги, що з'являється у відповідь на запити користувачів і дозволяє отримати повний опис, характеристики та додаткові відомості.

Підтвердження виконаних операцій, таких як додавання нового елемента в список, зміна налаштувань, створення або редагування записів, що фіксуються для користувача і системи.

Відображення оновлень інтерфейсу та функціональних модулів, що розширюють можливості користувача та підвищують зручність використання платформи.

Дані зворотного зв'язку, отримані від користувачів через відповідні форми, що допомагають вдосконалювати роботу системи та підвищувати якість сервісу.

Таким чином, вихідна інформація не лише забезпечує функціональність системи, але й є основою для аналізу ефективності, контролю якості, а також подальшого розвитку та оптимізації продукту.

Повний перелік та опис основних вихідних повідомлень наведено у таблиці 2.1.

Таблиця 2.1 – Вихідні дані

Назва вихідного повідомлення	Ідентифікатор	Форма подання	Термін і частота надання	Призначення
Підтвердження входу в систему	P_LOGIN	Повідомлення	Миттєво після дії	Неавторизований користувач
Підтвердження реєстрації	P_REGISTER	Повідомлення	Миттєво після дії	Неавторизований користувач
Підтвердження вибору ролі	P_ROLE_SELECT	Повідомлення	Миттєво після дії	Неавторизований користувач

Продовження таблиці 2.1

Відображення головної сторінки	V_MAIN_CL	Екранна форма	За потребою	Авторизований клієнт
Результати пошуку	V_SEARCH	Екранна форма	За потребою	Авторизований клієнт
Вивід всіх послуг в конкретній даті календаря	V_SERV_CAL	Екранна форма	За потребою	Авторизований клієнт
Вивід профілю	V_PROFILE_CL	Екранна форма	За потребою	Авторизований клієнт
Підтвердження зміни профілю	P_PROFILE_CL	Повідомлення	Миттєво після дії	Авторизований клієнт
Вивід популярних послуг	V_POPULAR_SERVICES	Екранна форма	За потребою	Авторизований клієнт
Вивід сторінки послуг	V_SERVICES_CL	Екранна форма	За потребою	Авторизований клієнт
Підтвердження запиту на бронювання на конкретний час	P_BOOK_CONFIRM	Повідомлення	Миттєво після дії	Авторизований клієнт
Вивід сторінки записи	V_BOOKINGS_CL	Екранна форма	За потребою	Авторизований клієнт
Підтвердження фільтрування записів	D_BOOKINGS_CL	Повідомлення	Миттєво після дії	Авторизований клієнт
Видалення записів	V_DELETE_BOOKING	Повідомлення	Миттєво після дії	Авторизований клієнт
Вивід головної сторінки	V_MAIN_SUP	Екранна форма	За потребою	Авторизований постачальник
Вивід нагадувань про календар	V_REMINDERS	Екранна форма	Щодня	Авторизований постачальник

Продовження таблиці 2.1

Вивід форми для додавання слотів	V_SLOT_FORM	Екранна форма	За потребою	Авторизований постачальник
Підтвердження додавання слотів для графіку	P_SLOT_ADD	Повідомлення	Миттєво після дії	Авторизований постачальник
Вивід профілю	V_PROFILE_SUP	Екранна форма	За потребою	Авторизований постачальник
Підтвердження зміни профілю	P_UPDATE_PROFILE_SUP	Повідомлення	Миттєво після дії	Авторизований постачальник
Вивід статистики постачальника	V_STATISTICS	Екранна форма	За потребою	Авторизований постачальник
Вивід сторінки послуг постачальника	V_SUP_SERVICES	Екранна форма	За потребою	Авторизований постачальник
Підтвердження видалення послуги	P_DELETE_SERVICE	Повідомлення	Миттєво після дії	Авторизований постачальник
Підтвердження створення послуги	P_CREATE_SERVICE	Повідомлення	Миттєво після дії	Авторизований постачальник
Підтвердження редагування послуги	P_EDIT_SERVICE	Повідомлення	Миттєво після дії	Авторизований постачальник
Вивід сторінки записів	V_SUP_BOOKINGS	Екранна форма	За потребою	Авторизований постачальник
Підтвердження фільтрування записів	P_FILTER_SUP_BOOKINGS	Повідомлення	Миттєво після дії	Авторизований постачальник

Джерело: розроблено автором на основі виконаних досліджень

2.2.3 Вхідні дані

Вхідні дані представляють собою інформацію, яку користувач або система надсилає для певних дій чи отримання необхідної інформації від платформи. Вони подаються через екранні форми та обробляються миттєво за потребою користувача чи системи.

Основні типи вхідних даних включають:

- Дані для входу в систему: логін та пароль, які вводить неавторизований користувач для доступу до облікового запису.
- Дані для реєстрації клієнта та постачальника : інформація, що вводиться новим користувачем для створення облікового запису відповідно до ролі.
- Запит на вибір ролі : вибір ролі користувача під час реєстрації або першого входу.
- Запити на вивід сторінок : ініціюють відображення головних сторінок клієнта або постачальника після успішної авторизації.
- Пошукові запити : містять параметри пошуку, введені авторизованим користувачем.
- Запити на редакцію даних: містять інформацію для оновлення профілю користувача.
- Запити на керування послугами: включають дані для створення, оновлення або видалення послуг постачальника.
- Запити на бронювання та їх фільтрацію : дозволяють клієнтам та постачальникам працювати із записами бронювань.

Вхідні дані надходять від користувачів, які можуть бути як неавторизованими (під час входу або реєстрації), так і авторизованими (під час роботи з основним функціоналом). Вони є основою для формування відповідних запитів до системи і забезпечують взаємодію користувача з платформою.

Повний перелік та опис основних вихідних повідомлень наведено у таблиці 2.2

Таблиця 2.2 – Вхідні дані

Назва вхідного повідомлення	Ідентифікатор	Форма подання	Термін і частота надходження	Джерело
Дані для входу в систему	D_VHID	Екранна форма	За потребою, миттєва обробка	Неавторизований користувач
Дані для реєстрації клієнта	D_REJ_CL	Екранна форма	За потребою, миттєва обробка	Неавторизований користувач
Дані для реєстрації постачальника	D_REJ_SP	Екранна форма	За потребою, миттєва обробка	Неавторизований користувач
Запит на вибір ролі	D_AN_ROLE	Екранна форма	За потребою, миттєва обробка	Неавторизований користувач
Запит на вивід головної сторінки клієнта	D_CL_VHID	Екранна форма	За потребою, миттєва обробка	Авторизований клієнт
Пошуковий запит	D_CL_SEARCH	Екранна форма	За потребою, миттєва обробка	Авторизований клієнт
Запит на отримання записів на конкретну дату	D_CL_DATE	Екранна форма	За потребою, миттєва обробка	Авторизований клієнт
Запит на вивід профілю клієнта	D_CL_PROF	Екранна форма	За потребою, миттєва обробка	Авторизований клієнт

Продовження таблиці 2.2

Запит на редакцію профілю клієнта	D_CL_EDIT	Екранна форма	За потребою, миттєва обробка	Авторизований клієнт
Запит на вивід списку послуг	D_CL_SERV	Екранна форма	За потребою, миттєва обробка	Авторизований клієнт
Запит на бронювання послуги	D_CL_BOOK	Екранна форма	За потребою, миттєва обробка	Авторизований клієнт
Запит на конкретний час бронювання	D_CL_TIME	Екранна форма	За потребою, миттєва обробка	Авторизований клієнт
Запит на вивід всіх бронювань	D_CL_ALL	Екранна форма	За потребою, миттєва обробка	Авторизований клієнт
Запит на фільтрацію бронювань	D_CL_FLTR	Екранна форма	За потребою, миттєва обробка	Авторизований клієнт
Запит на вивід головної сторінки постачальника	D_SP_VHID	Екранна форма	За потребою, миттєва обробка	Авторизований постачальник
Запит на виведення графіків на окрему дату	D_SP_DATE	Екранна форма	За потребою, миттєва обробка	Авторизований постачальник
Введення даних для графіків	D_SP_INPT	Екранна форма	За потребою, миттєва обробка	Авторизований постачальник
Запит на вивід профілю постачальника	D_SP_PROF	Екранна форма	За потребою, миттєва обробка	Авторизований постачальник

Продовження таблиці 2.2

Запит на редакцію профілю постачальника	D_SP_EDIT	Екранна форма	За потребою, миттєва обробка	Авторизований постачальник
Запит на виведення сторінки послуг постачальника	D_SP_SERV	Екранна форма	За потребою, миттєва обробка	Авторизований постачальник
Запит на створення послуги	D_SP_CREATE	Екранна форма	За потребою, миттєва обробка	Авторизований постачальник
Запит на редакцію послуги	D_SP_UPDT	Екранна форма	За потребою, миттєва обробка	Авторизований постачальник
Запит на видалення послуги	D_SP_DEL	Екранна форма	За потребою, миттєва обробка	Авторизований постачальник
Запит на виведення сторінки записів	D_SP_BOOKS	Екранна форма	За потребою, миттєва обробка	Авторизований постачальник
Запит до фільтрації записів	D_SP_FLTR	Екранна форма	За потребою, миттєва обробка	Авторизований постачальник

Джерело: розроблено автором на основі виконаних досліджень

2.2.4 Алгоритм розв'язання задачі

Предметна область, що охоплює взаємодію між клієнтами та постачальниками послуг, є багатогранною та динамічною, оскільки включає різні сценарії обміну даними, рольові залежності й комплексні послідовності дій. Для

розробки вебзастосунку, здатного автоматизувати такі процеси в умовах українського ринку, було застосовано системний підхід та використано уніфіковані засоби опису логіки роботи системи.

Одним із основних методів моделювання бізнес-процесів у данній роботі стала уніфікована мова моделювання (UML — Unified Modeling Language), яка забезпечує єдиний стандарт для візуалізації структури та поведінки систем. Зокрема, для опису алгоритму розв’язання задачі було використано діаграми діяльності [9], що дають можливість детально зобразити послідовність дій, варіанти розвитку подій, точки прийняття рішень, а також паралельні або повторювані процеси.

Алгоритм розв’язання задачі можна розбити на три основні логічні етапи, кожен із яких моделюється окремою діаграмою діяльності:

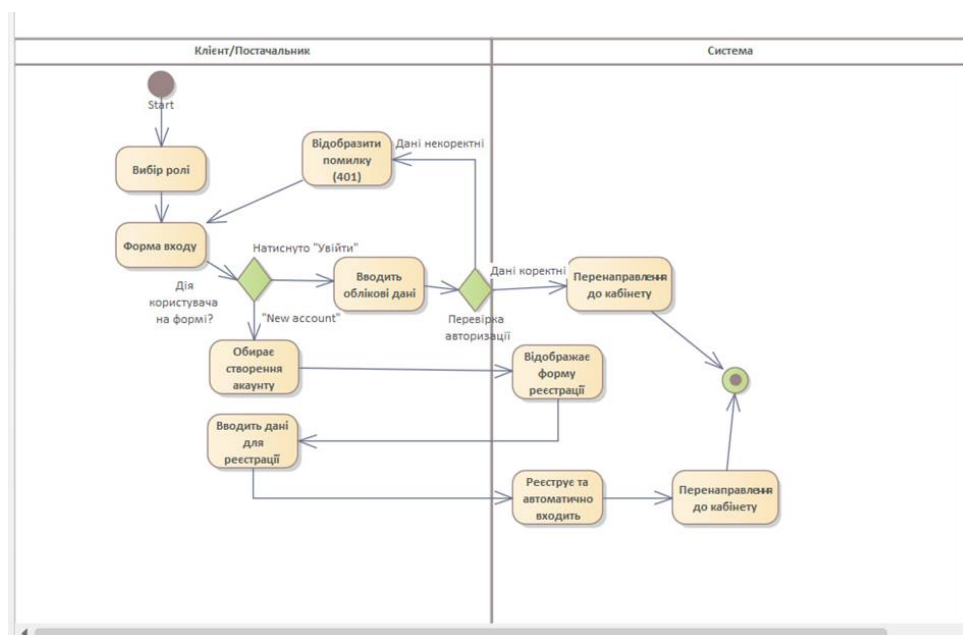


Рисунок 2.11 – Діаграма діяльності «Реєстрація та авторизація користувача»

Джерело: розроблено автором на основі виконаних досліджень

Процес починається на сторінці автентифікації, де користувач може увійти за наявними обліковими даними або зареєструвати новий акаунт.

При виборі входу користувач вводить логін і пароль. Система перевіряє їх і при коректних даних перенаправляє до особистого кабінету, інакше повідомляє про помилку.

При виборі реєстрації користувач заповнює форму з персональними даними та роллю (клієнт або постачальник). Після успішної реєстрації система автоматично авторизує користувача і відкриває головну сторінку. Помилки реєстрації відображаються користувачу.

Моделювання процесу дозволило описати основні сценарії авторизації та реєстрації, передбачити обробку помилок і автоматичний перехід до особистого кабінету.

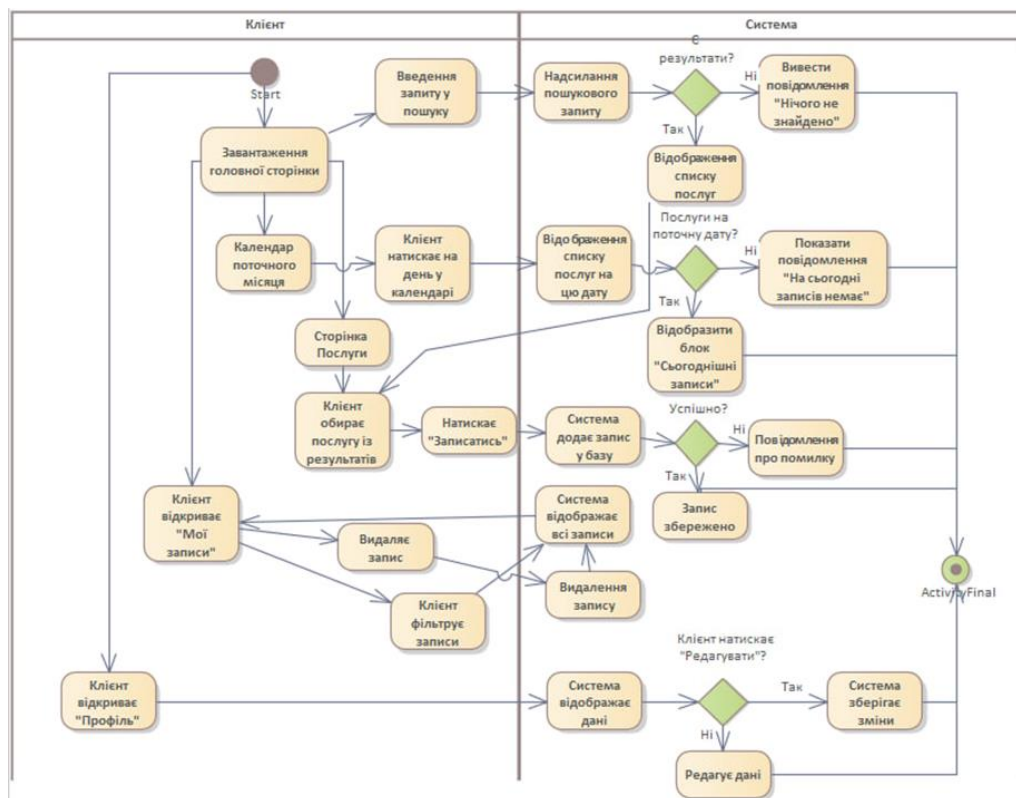


Рисунок 2.12 – Діаграма діяльності «Основні сценарії взаємодії клієнта з системою»

Джерело: розроблено автором на основі виконаних досліджень

Взаємодія користувача з системою починається з головної сторінки, де доступні основні функції: вибір дати в календарі, пошук послуг та навігація по розділах. Пошук та вибір дати дозволяють отримати перелік доступних послуг або

повідомлення про їх відсутність. Через меню користувач переходить до розділів «Послуги», «Профіль» та «Мої записи», де може переглядати, оформлювати або редагувати відповідні дані.

Для постачальника послуг після авторизації доступні інструменти управління каталогом послуг, розкладом та записами клієнтів.

Моделювання взаємодії забезпечило логічну послідовність дій, чітке розмежування ролей користувача та системи, що сприяє ефективному проектуванню інтерфейсу та покращенню користувацького досвіду.

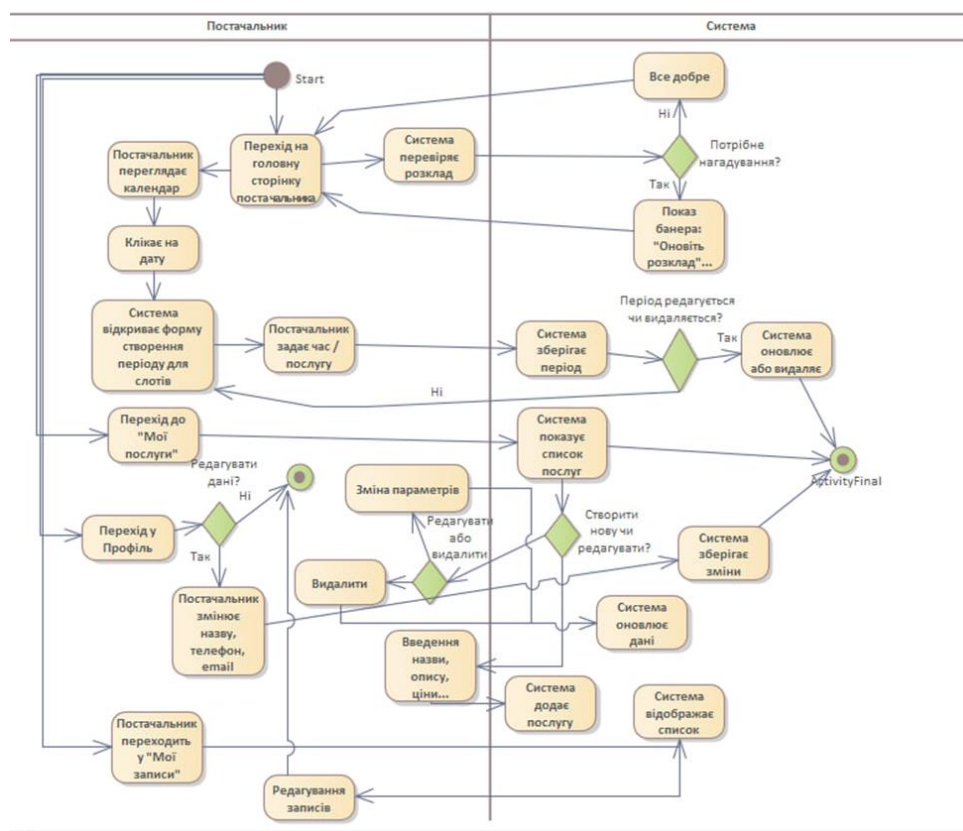


Рисунок 2.13 – Діаграма діяльності «Основні сценарії взаємодії постачальника з системою»

Джерело: розроблено автором на основі виконаних досліджень

Взаємодія постачальника з системою починається з головної сторінки особистого кабінету, де він може використовувати календар для керування розкладом, отримувати сповіщення та переходити до основних розділів через

навігаційне меню. За допомогою календаря постачальник встановлює робочі години та визначає доступність послуг для бронювання.

У розділі «Профіль» доступна можливість перегляду та редагування особистої й професійної інформації. У розділі «Мої послуги» постачальник може керувати наявними послугами, змінювати їхні характеристики, створювати нові або видаляти застарілі. У розділі «Записи клієнтів» надається доступ до бронювань, з можливістю їх підтвердження, скасування та фільтрації за різними параметрами.

Моделювання взаємодії дозволило систематизувати ключові процеси управління послугами та розкладом, врахувати основні сценарії дій постачальника та побудувати логічну структуру навігації в межах особистого кабінету.

2.3 Моделювання інформаційної системи

2.3.1 Моделювання поведінки системи

Задля реалізації моделювання інформаційної підсистеми було вирішено обрати об'єктний підхід, який допоможе зрозуміти складну систему за допомогою абстрагування від деталей. Під час моделювання було використано програмне середовище Enterprise Architect.

Діаграма прецедентів [10] (або діаграма варіантів використання) (англ. Use case diagram) — в UML, діаграма, на якій зображено відношення між акторами та прецедентами в системі. Було побудовано модель поведінки системи для підтримки користувачів системи у вигляді діаграми прецедентів у нотації UML, яка відображає доступні дії користувачів та відповідні варіанти використання (рис. А.2, Додаток А).

Діаграма послідовності (англ. sequence diagram) — різновид діаграми в UML. Діаграма послідовності відображає взаємодії об'єктів впорядкованих за часом.

На основі діаграми прецедентів можна впевнено переходити до побудови діаграм послідовності, які деталізують взаємодію між актором і системними компонентами для реалізації кожного прецеденту.

Розглянемо ключові сценарії взаємодії користувачів із системою, представлені на діаграмах послідовності, що наведені у додатку А.

"Процес автентифікації та реєстрації" демонструє послідовність взаємодій під час входу користувача в систему, включно з автоматичною перевіркою сесії, ручним введенням облікових даних та реєстрацією нового користувача. Діаграма відображає логіку обробки запитів, взаємодію з базою даних, перевірку інформації та формування відповідей системи. (рис. А.3, Додаток А).

"Взаємодія з профілями" ілюструє процес перегляду та редагування профілю користувача. Вона показує передачу запитів між компонентами системи, роботу сервісів, а також обмін даними з базою для отримання та оновлення інформації. (рис. А.4, Додаток А).

"Пошук" відображає послідовність дій при виконанні пошуку послуг, включаючи обробку пошукового запиту сервісом та взаємодію з базою даних для отримання релевантних результатів. (рис. А.5, Додаток А).

"Дії з бронюванням клієнта" ілюструє операції клієнта зі створення, перегляду та видалення бронювань, демонструючи обробку запитів сервісом бронювання та взаємодію з базою даних. (рис. А.6, Додаток А).

"Календар клієнта" відображає використання календаря для перегляду послуг на обрану дату та виконання операцій, таких як видалення послуги, із відповідним обміном даними між сервісами та базою. (рис. А.7, Додаток А).

"Календар постачальника" демонструє, як постачальник переглядає послуги, обирає дату та час, а також перевіряє доступність обраного часу через взаємодію з відповідними сервісами та базою даних. (рис. А.8, Додаток А).

"Послуги постачальника" ілюструє управління послугами постачальника — перегляд, створення, редагування та видалення, з обробкою запитів сервісами управління та оновленням бази даних. (рис. А.9, Додаток А).

"Бронювання постачальника" відображає процес перегляду та видалення бронювань, пов'язаних із послугами постачальника, включаючи взаємодію з відповідними сервісами та оновлення інформації у базі даних. (рис. А.10, Додаток А).

2.3.2 Моделювання структури системи

Для моделювання структури інформаційної системи використано UML-діаграму класів [11], які відображає організацію класів та зв'язки між ними. Діаграми класів є основою для побудови архітектури системи, оскільки дозволяють наочно представити ключові компоненти, їх властивості, операції та взаємодії.

Задля забезпечення прозорості і зрозумілості моделі, на різних діаграмах рекомендовано демонструвати:

- пакети класів та відношення між ними;
- підмножини класів із відповідними зв'язками;
- підмножини класів із деталізацією атрибутів та операцій.

У представлених діаграмах використано основні стереотипи класів:

- класи-сутності — моделюють дані та поведінку об'єктів із тривалим життєвим циклом, наприклад, Users, Client, Suppliers, Services, Supplier_schedules, Time_slots, Client_bookings;
- граничні класи — представляють інтерфейси користувача на високому рівні, такі як ClientRegistrationPage, LoginPage, ClientMainPage, SearchService тощо;

- класи керування (менеджери) — відповідають за послідовну поведінку системи, координацію подій та реалізацію логіки прецедентів, прикладами є `ClientService`, `BookingService`, `SupplierService`.

На діаграмах наведені атрибути класів (наприклад, `email`, `created_at`, `description`) та методи, які забезпечують функціональність (наприклад, `createService()`). Це дає змогу зрозуміти не лише структуру даних, а й поведінку об'єктів в системі.

Взаємозв'язки між класами відображають реалізацію бізнес-процесів, таких як реєстрація користувачів, пошук послуг, бронювання, управління профілями та розкладом. Діаграми класів є невід'ємною частиною проєктної документації та слугують базою для подальшого моделювання сценаріїв використання та розробки програмної реалізації.

На (рис. А.11, Додаток А) наведено відповідну діаграму класів системи, яка ілюструє описану структуру.

З метою забезпечення логічних структур та ефективного виконання функціональних вимог, було вирішено створити архітектуру системи, що враховує в себе окремі компоненти, кожен з яких відповідає за реалізацію певного підмножини вимог. На цьому етапі було здійснено трасування вимог до відповідних компонентів системи з метою перевірки повноти покриття та узгодженості проєктних рішень.

Було побудовано трасувальні діаграми, які поєднують між собою елементи вимог, прецедентів, елементів інтерфейсу та тест-кейсів (рис. А.12-15, Додаток А). Між елементами діаграми трасування проведені відповідні зв'язки

РОЗДІЛ 3 ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ

3.1 Інформаційне забезпечення

3.1.1 Загальна характеристика інформаційного забезпечення

Інформаційне забезпечення системи включає декілька взаємопов'язаних компонентів, кожен з яких виконує чітко визначені функції в межах загальної архітектури.

Серверна частина реалізована у вигляді Java-додатку, який забезпечує логіку обробки запитів, керування сесіями користувачів залежно від ролі, авторизацію та доступ до бази даних. Саме через цю частину відбувається перевірка даних, виконання умов бізнес-логіки, зберігання нових записів і зміна вже наявної інформації в базі даних. Java-додаток працює у тісній зв'язці з клієнтською частиною та забезпечує ефективну взаємодію між користувачем і сховищем даних.

Архітектура побудована за принципом багаторівневої клієнт-серверної моделі [12], що дозволяє розділити обробку інтерфейсу, бізнес-логіки та доступу до даних, підвищуючи масштабованість та гнучкість системи.

Клієнтська частина представлена вебінтерфейсом, створеним на базі HTML, CSS та JavaScript. [13] Розробка такого інтерфейсу надає користувачам зручний доступ до функціоналу системи, дозволяючи реєструватися, входити в систему, переглядати доступні послуги, здійснювати бронювання, а також переглядати та редагувати власний профіль. Вебінтерфейс побудований за принципами інтуїтивної взаємодії, що забезпечує простоту користування та мінімізує кількість помилок при введенні даних.

Центральною частиною інформаційного забезпечення є реляційна база даних PostgreSQL, яка відповідає за зберігання всіх основних даних системи. [14] Вона містить таблиці для користувачів, клієнтів, постачальників, послуг, бронювань, часових інтервалів, а також графіків роботи.

Використання PostgreSQL обґрунтоване її високою надійністю, підтримкою транзакцій, механізмів відновлення після збоїв, а також масштабованістю, що дозволяє ефективно працювати під значними навантаженнями. Крім того, PostgreSQL підтримує розширені типи даних і потужні засоби контролю цілісності даних. [15]

Таблиця users — універсальна таблиця з обліковими даними, де зберігається

- унікальний ідентифікатор користувача (UUID),
- ім'я користувача (username),
- пароль (password),
- роль (role), що визначає доступ до функціоналу (клієнт або постачальник).

Таблиця clients — таблиця, що містить розширену інформацію про користувачів у ролі клієнтів:

- ім'я та прізвище,
- електронна пошта,
- стать, вік,
- номер телефону,
- аватар,
- дата створення облікового запису.

Таблиця suppliers — таблиця з інформацією про постачальників послуг:

- назва бізнесу,
- короткий опис послуг,
- електронна пошта та номер телефону,
- адреса,

- аватар,
- дата створення профілю.

Таблиця `services` — перелік послуг, що пропонуються у системі:

- назва та опис послуги,
- ціна,
- тривалість надання (у хвилинах),
- постачальник, який надає послугу.

Таблиця `time_slots` — таблиця з доступними часовими інтервалами:

- час початку,
- час завершення,
- унікальний ідентифікатор слота.

Таблиця `client_bookings` — таблиця для фіксації бронювань, здійснених клієнтами:

- клієнт, що здійснив замовлення,
- обрана послуга,
- часовий слот,
- дата бронювання,
- дата створення запису.

Таблиця `supplier_schedules` — таблиця графіків роботи постачальників:

- ідентифікатор постачальника,
- послуга, яку він надає,
- доступний часовий слот,
- дата, коли послуга надається.

Всі таблиці мають логічні зв'язки між собою, реалізовані через зовнішні ключі, що гарантує цілісність даних і унеможливорює появу некоректних або “порожніх” записів. Для забезпечення унікальності записів використовується тип даних `UUID` [16], що генерується автоматично при кожному додаванні нового

запису. Це підвищує надійність та масштабованість системи, особливо у випадках розподілених баз даних чи масштабування.

Для зберігання даних використовується електронний тип носія – серверні SSD-диски, які забезпечують швидкий доступ до інформації та підтримують високі навантаження. Для підвищення відмовостійкості передбачено резервне копіювання даних, яке може виконуватися як локально, так і з використанням хмарних сервісів.

Вибір PostgreSQL як СКБД обумовлений її стабільністю, широкою функціональністю та потужною підтримкою складних транзакцій, що дозволяє гарантувати цілісність і надійність зберігання даних. PostgreSQL ідеально інтегрується з Java-додатком через відповідні JDBC-драйвери та ORM-фреймворки (Hibernate, Jakarta Persistence [17]), що спрощує управління даними, дозволяє гнучко розширювати функціонал і знижує ризики помилок під час розробки.

Завдяки такій архітектурі та вибору технологій система досягає високої продуктивності, масштабованості і надійності, що є критично важливим для комфортного користування і подальшого розвитку.

У системі реалізовано комплекс методів контролю інформації для забезпечення її безпеки, цілісності та надійності. Контроль доступу здійснюється через ролі користувачів (клієнт, постачальник) і обмеження доступу до сторінок за допомогою механізмів авторизації. Цілісність даних підтримується на рівні бази даних PostgreSQL через зовнішні ключі, унікальні ідентифікатори та обмеження (NOT NULL, UNIQUE). Валідність введених даних перевіряється на рівні серверної логіки до їх збереження, що дозволяє уникнути помилкової або шкідливої інформації. Журналювання змін і подій допомагає виявляти інциденти або збої, а автентичність сесій контролюється через токени з обмеженим часом дії.

З вимог до надійності і достовірності інформації можна відзначити такі:

- Надійність: система має забезпечувати безперервну роботу з мінімальними перебоями.
- Достовірність: система має забезпечити точність та актуальність даних, що зберігаються і обробляються.

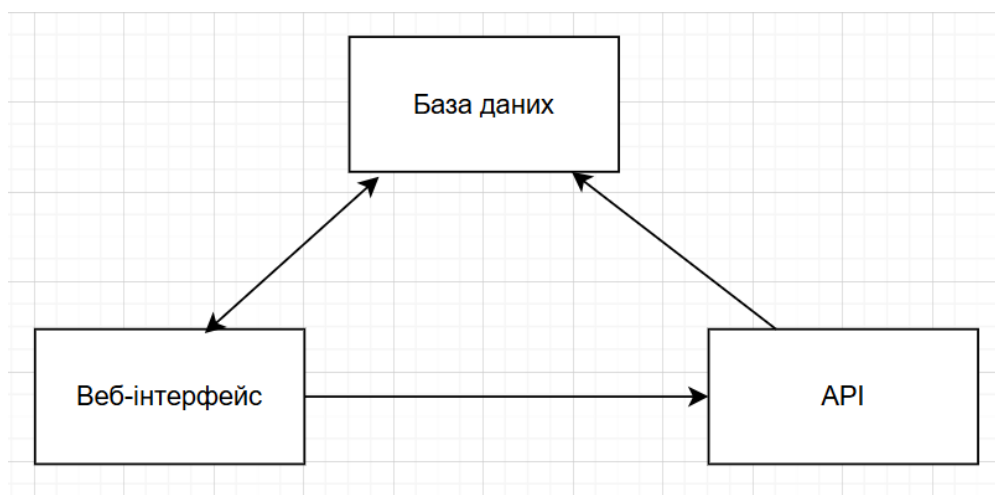


Рисунок. 3.1. – Загальна схема ІЗ

Джерело: розроблено автором на основі виконаних досліджень

3.1.2 Організація збору і передавання первинної інформації

Далі описано організацію збору і передавання первинної інформації такими критеріями: перелік джерел і носіїв інформації та опис загальних вимог.

У системі автоматизації взаємодії між клієнтами та постачальниками послуг збір і передавання первинної інформації організовано через вебінтерфейс та внутрішню обробку даних. Джерелами інформації є користувачі системи — клієнти та постачальники, які вводять дані через відповідні форми (реєстрація, бронювання, профіль, послуги тощо). Також використовуються внутрішні джерела — база даних та системні сервіси. Інформація надходить у режимі реального часу за допомогою захищених запитів, а її носієм є електронна форма (HTML, JSON). Загальні вимоги до організації збору і передавання інформації передбачають:

- Точність і своєчасність — система забезпечує збір достовірної інформації від користувачів (клієнтів і постачальників) у режимі реального часу, що дозволяє оперативно формувати та обробляти запити.

- Безперервність збору — усі ключові дії в системі (бронювання, оновлення профілю, додавання послуг) фіксуються негайно, без затримок, що гарантує актуальність доступних даних.
- Безпека — передавання інформації здійснюється захищеними каналами з використанням автентифікації та авторизації, а також передбачено резервне копіювання.
- Контроль якості — дані проходять перевірку на рівні бізнес-логіки (валідація формату, обов'язковість полів, унікальність), що мінімізує ризик помилок та забезпечує достовірність.
- Масштабованість — архітектура системи передбачає можливість розширення, обробки зростаючої кількості запитів та додавання нових джерел інформації без втрати продуктивності.

3.1.3 Побудова системи класифікації та кодування

У процесі розробки інформаційної системи автоматизації взаємодії між клієнтами та постачальниками послуг було впроваджено систему класифікації та кодування основних об'єктів даних. Метою цієї системи є забезпечення однозначної ідентифікації сутностей, структуризація даних, а також підтримка цілісності та масштабованості системи при зростанні обсягів інформації.

Для ідентифікації всіх ключових об'єктів, таких як користувачі, послуги, записи на бронювання тощо, використовується механізм універсального унікального ідентифікатора (UUID). Формат UUID становить 36 символів і має вигляд xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx, де x — це шістнадцятковий символ. Наприклад, один із можливих ідентифікаторів може мати значення 7f3b18ea-91b2-4a45-bc9e-2d8f7b7e6a21. Генерація UUID відбувається автоматично,

відповідно до стандарту UUIDv4, що гарантує унікальність кожного коду незалежно від джерела створення.

Окрім кодування, у системі реалізовано логічну класифікацію об'єктів на основі прикладних характеристик. Зокрема:

- Користувачі класифікуються за роллю, що визначає функціональні можливості та доступ до ресурсів системи. Основні ролі: CLIENT (клієнт) і SUPPLIER (постачальник).
- Записи на послуги класифікуються за статусом опрацювання, що дозволяє відстежувати етапи взаємодії. До основних статусів належать: WAITING (очікує підтвердження), CONFIRMED (підтверджено), CANCELLED (скасовано).
- Стать користувача також є об'єктом класифікації. Зберігається у вигляді текстового або символьного коду. Можливі значення: MALE — чоловіча стать, FEMALE — жіноча стать, OTHER — інше / не зазначено

Ці класифікації реалізовано у вигляді перелічуваних типів (enum) на рівні бізнес-логіки Java-додатку, що спрощує обробку та валідацію даних. У разі потреби їх також можна зберігати у вигляді довідників у базі даних для подальшої аналітики або налаштувань.

Використання UUID як унікальних кодів і системи логічної класифікації забезпечує гнучкість, масштабованість і безпеку інформаційної системи, а також полегшує інтеграцію з іншими компонентами або зовнішніми сервісами.

3.1.4 Структура інформаційних масивів

Було розроблено структуру основних інформаційних масивів, що використовуються в системі. Для кожного масиву представлено детальний опис у вигляді таблиці (табл. 3.1 – 3.8), де зазначено назву полів, типи даних, розміри та призначення кожного елемента.

Масив users (Користувачі)

1. Найменування масиву: Користувачі.
2. Ідентифікатор масиву: users.
3. Найменування носія інформації: сервер pgAdmin.
4. Максимальний об'єм масиву: без обмежень.
5. Довжина запису: залежить від довжини текстових полів.
6. Метод організації записів: послідовний.

Таблиця 3.1 – Масив користувачів

Найменування	Ідентифікатор у програмі	Умове позначення у формулах	Формат	Бізнес-правила				Логічні чи семантичні зв'язки
				Первинний/вторинний ключ	Умова на значення	Обов'язкове поле	Індексне поле	
Ідентифікатор	id	Iu	UUID	PK	-	Так	Так	-
Ім'я користувача	username	Un	varchar(30)	-	-	Так	Так	-
Пароль	password	Pw	text	-	-	Так	-	-
Роль	role	R	varchar(10)	-	-	Так	-	-

Джерело: розроблено автором на основі виконаних досліджень

Масив clients (Клієнти)

1. Найменування масиву: Клієнти.
2. Ідентифікатор масиву: clients.
3. Найменування носія інформації: сервер pgAdmin.
4. Максимальний об'єм масиву: без обмежень.
5. Довжина запису: приблизно 400 байтів (залежить від вмісту полів).
6. Метод організації записів: послідовний.

Таблиця 3.2. – Масив клієнтів

Найменування	Ідентифікатор у програмі	Умове позначення у формулах	Формат	Бізнес-правила				Логічні чи семантичні зв'язки
				Первинний/вторинний	Умова на значення	Обов'язкове поле	Індексне поле	
Ідентифікатор	id	Iu	UUID	PK	-	Так	Так	users
Ім'я	first_name	Fn	varchar(50)	-	-	Так	-	-
Прізвище	last_name	Ln	varchar(50)	-	-	Так	-	-
Електронна пошта	email	Em	varchar(255)	-	-	Так	Так	-
Стать	gender	G	varchar(6)	-	-	Hi	-	-
Вік	age	A	int	-	-	Hi	-	-
Телефон	phone_number	Pn	varchar(15)	-	-	Hi	-	-
Аватар	avatar	Av	bytea	-	-	Hi	-	-
Дата створення	created_at	Ca	timestamp	-	-	Так	-	-

Джерело: розроблено автором на основі виконаних досліджень

Масив suppliers (Постачальники послуг)

1. Найменування масиву: Постачальники послуг.
2. Ідентифікатор масиву: suppliers.
3. Найменування носія інформації: сервер pgAdmin.
4. Максимальний об'єм масиву: без обмежень.
5. Довжина запису: приблизно 500 байтів (залежить від вмісту полів).
6. Метод організації записів: послідовний.

Таблиця 3.3. – Масив Постачальники послуг

Найменування	Ідентифікатор у програмі	Умовне позначення у формулах	Формат	Бізнес-правила				Логічні чи семантичні зв'язки
				Первинний/вторинний	Умова на значення	Обов'язкове поле	Індексне поле	
Ідентифікатор	id	Iu	UUID	PK	-	Так	Так	users
Назва компанії	business_name	Bn	varchar(100)	-	-	Так	-	-
Опис послуг	service_description	Sd	varchar(255)	-	-	Hi	-	-
Електронна пошта	email	Em	varchar(255)	-	-	Так	Так	-
Телефон	phone_number	Pn	varchar(15)	-	-	Так	-	-
Адреса	address	Ad	varchar(255)	-	-	Hi	-	-
Аватар	avatar	Av	bytea	-	-	Hi	-	-
Дата створення	created_at	Ca	timestamp	-	-	Так	-	-

Джерело: розроблено автором на основі виконаних досліджень

Масив services (Послуги)

- Найменування масиву: Послуги.
- Ідентифікатор масиву: services.
- Найменування носія інформації: сервер pgAdmin.
- Максимальний об'єм масиву: без обмежень.
- Довжина запису: приблизно 400 байтів (залежить від вмісту полів).
- Метод організації записів: послідовний.

Таблиця 3.4. – Масив Послуги

Найменування	Ідентифікатор у програмі	Умовне позначення у формулах	Формат	Бізнес-правила				Логічні чи семантичні зв'язки
				Первинний/вторинний	Умова на значення	Обов'язкове поле	Індексне поле	
Ідентифікатор	id	Is	UUID	PK	-	Так	Так	-
Ідентифікатор постачальника	supplier_id	Sp	UUID	FK	Так	Так	-	suppliers
Назва послуги	name	N	varchar(100)	-	Так	Так	-	-
Опис	description	D	varchar(255)	-	-	Hi	-	-
Ціна	price	P	double precision	-	Так	Так	-	-
Тривалість (хвилин)	duration_minutes	T	int	-	Так	Так	-	-
Дата створення	created_at	Ca	timestamp	-	-	Так	-	-

Джерело: розроблено автором на основі виконаних досліджень

Масив time_slots (Часові слоти)

- Найменування масиву: Часові слоти.
- Ідентифікатор масиву: time_slots.
- Найменування носія інформації: сервер pgAdmin.
- Максимальний об'єм масиву: без обмежень.
- Довжина запису: приблизно 80 байтів (залежить від формату часу та реалізації UUID).
- Метод організації записів: послідовний.
- Ключі упорядкування: id (первинний ключ).

Таблиця 3.5 – Масив Time Slots (Часові слоти)

Найменування	Ідентифікатор у програмі	Умовне позначення у формулах	Формат	Бізнес-правила				Логічні чи семантичні зв'язки
				Первинний/вторинний	Умова на значення	Обов'язкове поле	Індексоване поле	
Ідентифікатор	id	Is	UUID	PK	Так	Так	Так	-
Час початку	start_time	St	TIME	-	-	Так	-	-
Час завершення	end_time	Et	TIME	-	-	Так	-	-

Джерело: розроблено автором на основі виконаних досліджень

Масив client_bookings (Бронювання клієнтів)

- Найменування масиву: Бронювання клієнтів.
- Ідентифікатор масиву: client_bookings.
- Найменування носія інформації: сервер pgAdmin.
- Максимальний об'єм масиву: без обмежень.
- Довжина запису: приблизно 160 байтів (залежить від формату UUID, дати та часу).
- Метод організації записів: послідовний.
- Ключі упорядкування: id (первинний ключ)

Таблиця 3.6 – Масив Client Bookings (Бронювання клієнтів)

Найменування	Ідентифікатор у програмі	Умове позначення у формулах	Формат	Бізнес-правила				Логічні чи семантичні зв'язки
				Первинний/вторинний	Умова на значення	Обов'язкове поле	Індексне поле	
Ідентифікатор	id	Is	UUID	P K	Ta k	Ta k	Ta k	-
Ідентифікатор клієнта	client_id	Cl	UUID	F K	-	Ta k	-	clients
Ідентифікатор послуги	service_id	S	UUID	F K	-	Ta k	-	service
Ідентифікатор слота	time_slot_id	Ts	UUID	F K	-	Ta k	-	time_slot s
Дата бронювання	booking_date	Bd	DATE	-	-	Ta k	-	-
Дата створення	created_at	Ca	TIMESTAMP	-	-	Ta k	-	-

Джерело: розроблено автором на основі виконаних досліджень

Масив supplier_schedules (Графіки постачальників)

- Найменування масиву: Графіки постачальників.
- Ідентифікатор масиву: supplier_schedules.
- Найменування носія інформації: сервер PostgreSQL.
- Максимальний об'єм масиву: без обмежень.
- Довжина запису: приблизно 160 байтів (залежить від формату UUID, дати та часу).
- Метод організації записів: послідовний.
- Ключі упорядкування: id (первинний ключ).

Таблиця 3.7 – Масив Supplier Schedules (Графіки постачальників)

Найменування	Ідентифікатор у програмі	Умове позначення у формулах	Формат	Бізнес-правила				Логічні чи семантичні зв'язки
				Первинний/вторинний	Умова на значення	Обов'язкове поле	Індексоване поле	
Ідентифікатор	id	Is	UUID	PK	Так	Так	Так	-
Ідентифікатор постачальника	supplier_id	Sp	UUID	FK	-	Так	-	suppliers
Ідентифікатор послуги	service_id	S	UUID	FK	-	Так	-	services
Ідентифікатор слота	time_slot_id	Ts	UUID	FK	-	Так	-	time_slots
Дата	date	D	DATE	-	-	Так	-	-
Дата створення	created_at	Ca	TIMESTAMP	-	-	Так	-	-

Джерело: розроблено автором на основі виконаних досліджень

3.1.5 Вибір СКБД

Вибір PostgreSQL як основної системи керування базами даних зумовлений насамперед її високим рівнем надійності, що є критично важливим для інформаційної системи, яка обробляє персональні дані користувачів, записи бронювань і розклади послуг. PostgreSQL підтримує ACID-транзакції, забезпечуючи цілісність та узгодженість даних навіть у разі збоїв або одночасного доступу багатьох користувачів. Її механізм журналювання (write-ahead logging)

гарантує відновлення даних після аварій, а підтримка резервного копіювання та реплікації дозволяє створити захищену та відмовостійку інфраструктуру.

Крім того, PostgreSQL має стабільну архітектуру та активно підтримується спільнотою розробників, що дає змогу швидко отримувати оновлення безпеки та дотримуватися кращих практик. Усе це робить PostgreSQL ідеальним вибором для систем, де пріоритетом є збереження даних, їхня безпечна обробка та мінімізація ризиків втрати. [15].

3.1.6 Інфологічна модель бази (сховища) даних

Було представлено інфологічну модель бази даних для веб-застосунку автоматизації бронювання послуг між клієнтом та постачальником, подану на рис.3.2. Базу даних нормалізовано відповідно до стандартних нормалізаційних форм. Інфологічна модель включає основні сутності, їх атрибути та зв'язки. Основними сутностями є Користувачі, Клієнти, Постачальники, Послуги, Розклад постачальників, Таймслоти, Бронювання клієнтів.

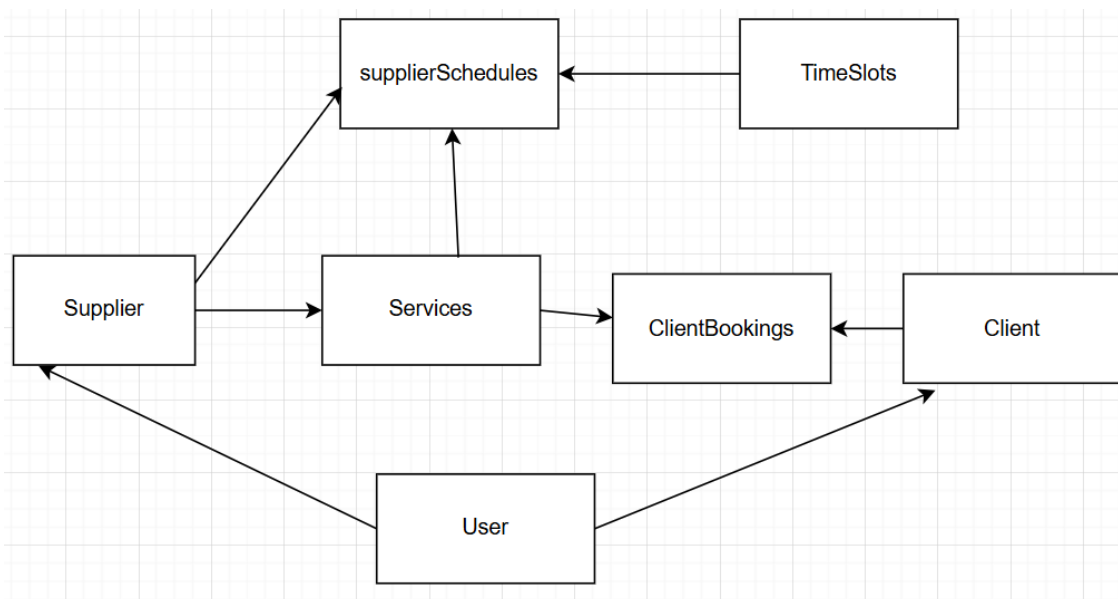


Рисунок. 3.2. – Інфологічна модель БД

Джерело: сформоване автором на основі виконаних досліджень

3.1.7 Даталогічна модель бази (сховища)даних

У цьому розділі представлено фізичну модель бази даних проєкту, розроблену в середовищі PostgreSQL. Фізична модель створена за допомогою інструменту DBeaver та відображає структуру основних таблиць системи бронювання: їх атрибути, типи даних, первинні та зовнішні ключі, а також логічні зв'язки між таблицями. На рисунку 3.3 показано ER-діаграму, яка ілюструє взаємозв'язки між об'єктами бази даних — користувачами, клієнтами, постачальниками послуг, записами на прийом, послугами, часовими слотами та розкладами.

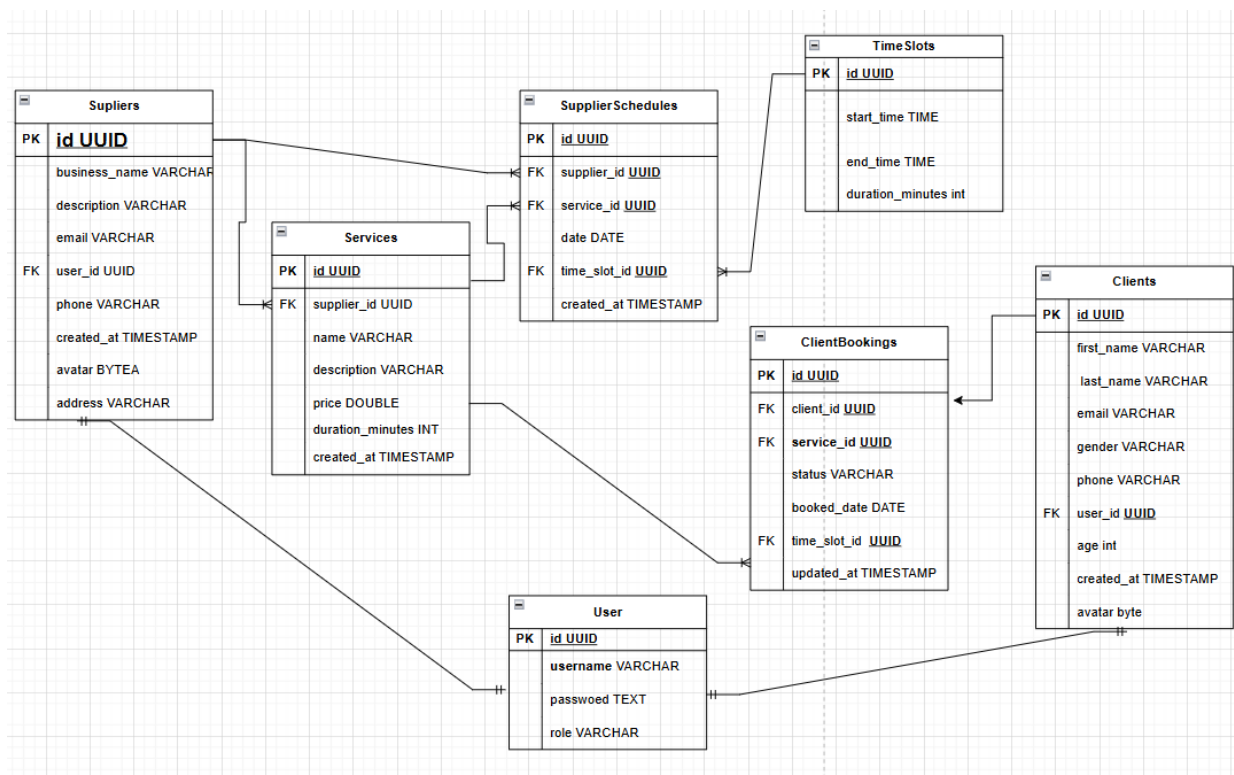


Рисунок. 3.3. – Даталогічна модель БД

Джерело: розроблено автором на основі виконаних досліджень

Фізична модель реалізується засобами мови опису даних (DDL) у системі PostgreSQL. Основу моделі становить таблиця users, що зберігає загальну

інформацію про всіх користувачів системи — логін, пароль і роль (CLIENT або SUPPLIER). Повний SQL-код створення цієї таблиці наведено в Додатку Б.1.

На основі users побудовано дві спеціалізовані таблиці: clients та suppliers. Вони містять додаткову інформацію, характерну для кожного типу користувача: ім'я, прізвище, вік, контактні дані для клієнтів, а також назву бізнесу, опис, адресу — для постачальників. Структури цих таблиць подано у Додатках Б.2 та Б.3 відповідно.

Таблиця services відображає інформацію про послуги, які надає кожен постачальник. Вона пов'язана зовнішнім ключем із таблицею suppliers і містить поля для назви послуги, опису, ціни та тривалості. SQL-код для створення наведено в Додатку Б.4.

Для моделювання доступних інтервалів часу використовується таблиця time_slots, структура якої наведена у Додатку Б.5.

Розклад роботи постачальників задається через таблицю supplier_schedules, де фіксується, яка послуга доступна в який день і в який часовий слот. Ця таблиця пов'язана з services, suppliers та time_slots, і створює унікальну комбінацію доступності. SQL-структуру розкладу див. у Додатку Б.6.

Фактичні записи клієнтів на послуги фіксуються в таблиці client_bookings, яка пов'язує клієнтів з конкретними розкладами. Це дозволяє контролювати кількість бронювань та запобігати дублюванню. SQL-код цієї таблиці наведено у Додатку Б.7.

Усі таблиці мають первинні ключі (переважно типу UUID), зовнішні ключі для зв'язків між сутностями, обмеження цілісності (унікальність, перевірки), а також логіку каскадного видалення (ON DELETE CASCADE), що дозволяє підтримувати цілісність бази при видаленні об'єктів.

Таким чином, фізична структура бази даних забезпечує повний опис основних бізнес-об'єктів системи бронювання та дозволяє ефективно реалізовувати ключові функції, пов'язані з реєстрацією користувачів, керуванням послугами, розкладом та бронюванням.

3.2 Технічне забезпечення

3.2.1 Загальні положення та схема автоматизації

Для забезпечення ефективного функціонування веб-застосунку було обґрунтовано організацію технічного забезпечення системи. До його складу входить комплекс технічних засобів (КТЗ), що охоплює серверне обладнання, користувацькі робочі станції, мережеву інфраструктуру та периферійні пристрої. Метою автоматизації є стабільна робота веб-застосунку, висока продуктивність і надійний захист даних.

Інформаційна система для автоматизації взаємодії між клієнтами та постачальниками індивідуальних послуг функціонуватиме на основі клієнт-серверної архітектури. Ядро системи (вебсервер, сервер додатків і база даних) розміщуватиметься на віддаленому хмарному сервері або в дата-центрі, що гарантує цілодобову доступність і масштабованість. Доступ до системи здійснюватиметься через вебінтерфейс у браузері користувача. Клієнти та постачальники зможуть користуватись функціоналом (реєстрація, перегляд розкладу, створення записів, керування послугами) з будь-якого пристрою, підключеного до Інтернету — комп'ютера, планшета або смартфона. Це забезпечує гнучкість і зручність без потреби встановлювати додаткове програмне забезпечення.

На загальній схемі автоматизації (рис. 3.4) наведено основні компоненти технічних засобів, що забезпечують функціонування системи.

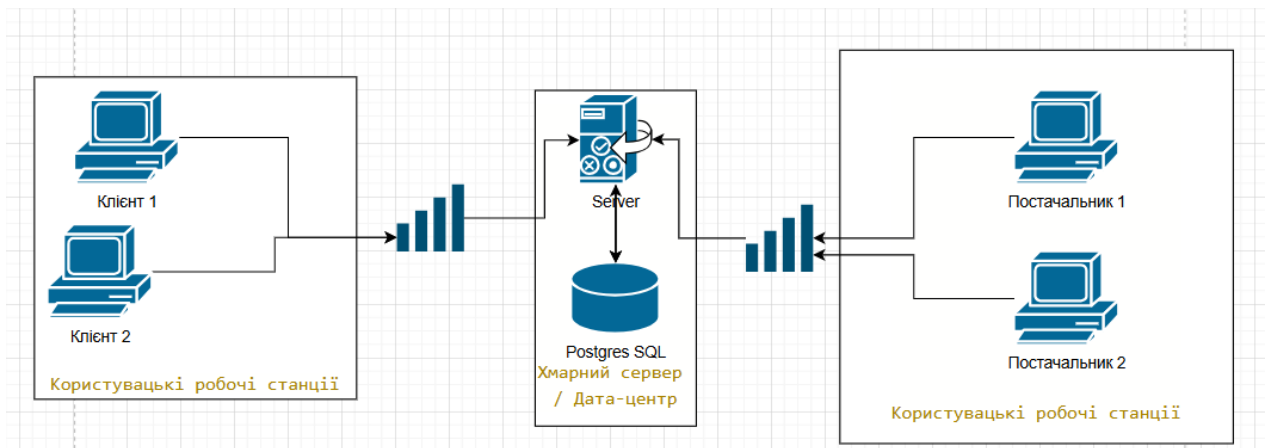


Рисунок. 3.4. – Загальна схема автоматизації

Джерело: розроблено автором на основі виконаних досліджень

3.2.2 Структура комплексу технічних засобів

Інформаційна система реалізується як вебзастосунок із клієнт-серверною архітектурою. Для її хостингу обрано сучасну хмарну платформу Railway, яка забезпечує гнучке масштабування, високу доступність та інтеграцію з популярними CI/CD-інструментами. Railway [18] підтримує автоматичне розгортання з GitHub та зберігання середовищних змінних, що спрощує DevOps-процеси.

Основним критерієм вибору Railway стала повноцінна підтримка середовища виконання Java, що є основною мовою розробки даної інформаційної системи. Railway також дозволяє безперешкодно запускати Spring Boot [20] застосунки, що значно спрощує процес розгортання серверної частини.

Railway надає такі базові технічні ресурси:

- автоматичне масштабування залежно від навантаження;
- ізольоване середовище для кожного сервісу;
- PostgreSQL як керована база даних;
- підтримка Node.js, Java, Python, Docker та інших середовищ;

- HTTPS за замовчуванням, TLS 1.3.

Мінімально необхідна кількість автоматизованих робочих місць (АРМ) для забезпечення повноцінного функціонування системи:

- 2 АРМ адміністратора для моніторингу та керування;
- 2 АРМ для роботи зі сторони клієнтів (реєстрація, перегляд послуг, запис);
- 2 АРМ для постачальників послуг (керування профілем, перегляд бронювань).

Обмін даними з іншими автоматизованими системами може бути реалізований за допомогою REST API, що дозволяє здійснювати інтеграцію з широким спектром зовнішніх сервісів. Такий підхід забезпечує масштабованість, гнучкість та адаптивність інформаційної системи до потреб користувачів і змін у бізнес-середовищі.

Предбачається можливість інтеграції з такими зовнішніми компонентами:

- банківські платіжні шлюзи — для обробки онлайн-оплат та проведення фінансових транзакцій;
- поштові сервіси — для надсилання автоматичних повідомлень користувачам (підтвердження реєстрації, нагадування про бронювання тощо);
- сервіси геолокації (наприклад, Google Maps API) — для визначення місця надання послуг та побудови маршрутів;
- інтеграція з іншими мікросервісами або внутрішніми інформаційними модулями — для розширення функціональних можливостей системи.

Для обміну даними планується використання захищеного HTTPS-з'єднання та JWT-автентифікації, що дозволяє забезпечити конфіденційність, цілісність і достовірність переданої інформації.

У межах розробки інформаційної системи значну увагу приділено питанню забезпечення захисту технічних засобів і даних, що обробляються. Оскільки застосунок розгортається у хмарному середовищі Railway, захист реалізується як на інфраструктурному рівні, так і на рівні прикладної логіки.

На інфраструктурному рівні хмарна платформа Railway надає набір вбудованих механізмів безпеки, які забезпечують ізоляцію мікросервісів, шифрування мережевого трафіку та контроль доступу до системи. Кожен сервіс виконується в ізольованому середовищі, що виключає можливість перехресного впливу між модулями. Передача даних здійснюється через зашифровані з'єднання з використанням сучасного протоколу TLS 1.3, що гарантує конфіденційність і цілісність інформації. Для захищеного керування доступом до хмарної інфраструктури використовуються SSH-ключі та автентифікація через GitHub OAuth. Також реалізовано обмеження доступу за IP-адресами, що дозволяє блокувати несанкціоновані підключення. Крім того, Railway підтримує системи журналювання та моніторингу, які забезпечують можливість виявлення аномальної активності та проведення безпекового аудиту.

На рівні прикладного програмного забезпечення реалізовано конфігурацію безпеки за допомогою фреймворку Spring Security. У системі використовується хешування паролів із застосуванням алгоритму BCrypt, що підвищує захист облікових записів користувачів у випадку витоку даних. Розмежування доступу до ресурсів реалізовано як на рівні HTTP-запитів (шляхом вказання дозволених маршрутів у конфігураційному класі SecurityConfig), так і на рівні методів за допомогою анотацій @PreAuthorize та @Secured, що забезпечує гнучке управління правами доступу. Вхід до системи здійснюється через окрему форму, з налаштуванням власного обробника успішної аутентифікації, що дозволяє реалізувати додаткову бізнес-логіку одразу після входу користувача.

У перспективі передбачається можливість впровадження додаткових засобів захисту, які підвищать рівень безпеки системи відповідно до сучасних вимог. Зокрема, можливе використання апаратного шифрування каналів зв'язку, впровадження фізичних засобів захисту (наприклад, ізольовані серверні кімнати, сейфи для мережевого обладнання), біометричних засобів ідентифікації або апаратних ключів доступу.

3.2.3 Опис автоматизованого робочого місця

У складі інформаційної системи автоматизовані робочі місця (АРМ) є ключовими елементами, що забезпечують користувачам ефективну роботу з інформацією. Вибір технічних засобів для АРМ обґрунтовується необхідністю відповідності вимогам системи, а також забезпеченням високої продуктивності і зручності роботи.

Для реалізації поставлених завдань було визначено технічні характеристики комп'ютерного обладнання, які включають процесори класу Intel Core i7 або AMD Ryzen 7, що здатні забезпечити багатозадачність та швидке виконання складних обчислень. Обсяг оперативної пам'яті встановлено не менше 16 ГБ, що дозволяє працювати з великими обсягами даних без уповільнень. Для зберігання інформації використовується твердотільний накопичувач (SSD) ємністю від 512 ГБ, що сприяє прискоренню доступу до файлів та програм.

Периферійне обладнання підбиралося з урахуванням вимог до зручності та комунікації. Основними пристроями є монітори з IPS-матрицею і роздільною здатністю не нижче 2560×1440 пікселів, що забезпечує якісне відображення інформації. Для комфортної роботи використовуються ергономічні клавіатури та миші з підсвічуванням, а також веб-камери з високою роздільною здатністю і гарнітури з активним шумозаглушенням для організації відеозв'язку.

Кількість АРМ у системі визначається чисельністю персоналу, який взаємодіє з інформаційною системою. Кожне робоче місце підключається до корпоративної мережі через швидкісний інтерфейс Ethernet з підтримкою стандарту Gigabit Ethernet, що гарантує стабільний та швидкий обмін даними.

Виконання вимог безпеки технічних засобів передбачає захист від фізичних впливів, таких як механічні пошкодження та температурні перепади, а також використання сучасних методів запобігання несанкціонованому доступу. Це досягається шляхом встановлення обладнання у захищені шафи, використання

апаратного шифрування каналів зв'язку, а також систем аутентифікації користувачів, які можуть включати біометричні засоби ідентифікації.

Загалом, вибрана структура комплексу технічних засобів забезпечує надійне і ефективне функціонування інформаційної системи, відповідає вимогам нефункціональних параметрів і створює комфортні умови для роботи персоналу.

3.3 Програмне забезпечення

3.3.1 Структура програмного забезпечення

У цьому розділі представлена структура програмного забезпечення (ПЗ) інформаційної системи, яка включає системне програмне забезпечення, прикладне програмне забезпечення та програмну документацію.

Загальна структура програмного забезпечення інформаційної системи представлена на (рис. 3.5).

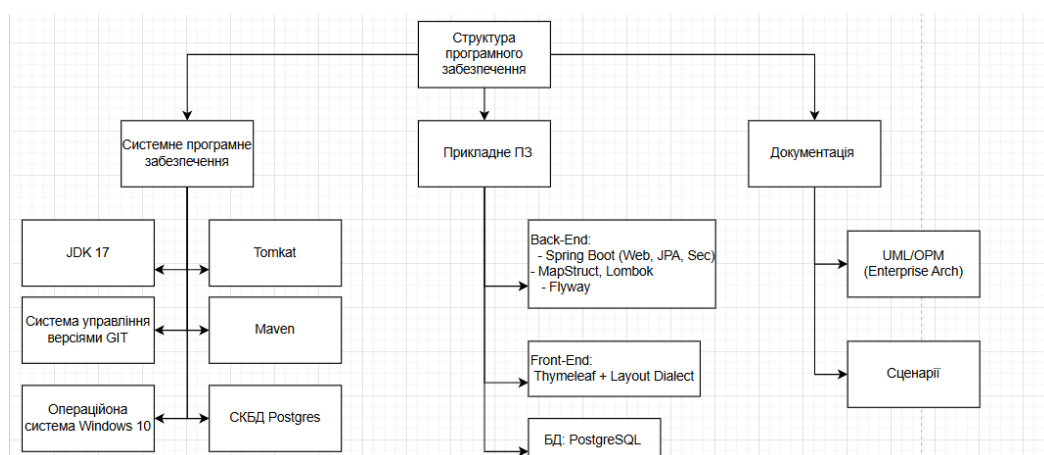


Рисунок 3.5 - Структура програмного забезпечення

Джерело: розроблено автором на основі виконаних досліджень

Прикладне, або спеціалізоване ПЗ вимагає створення текстових сценаріїв для кожного варіанта використання – щонайменше один, "нормальний", а за необхідності – ще один або кілька альтернативних, що описують відхилення від нього.

1. Вхід до облікового запису

Користувач відкриває головну сторінку → вибирає роль (Клієнт / Постачальник) → переходить на сторінку авторизації → вводить логін і пароль → система перенаправляє на головну сторінку відповідно до ролі.

Альтернатива: при неправильних даних виводиться повідомлення про помилку → користувач повторно вводить дані.

2. Реєстрація користувача

Користувач натискає «Зареєструватись» → вибирає роль → переходить на сторінку реєстрації → вводить необхідні дані → натискає «Зареєструватись» → система створює обліковий запис і перенаправляє на головну сторінку.

3. Редагування профілю

Користувач натискає «Профіль» у хедері → переходить на сторінку → редагує дані → натискає «Зберегти» → зміни зберігаються.

4. Створення нової послуги (постачальник)

Постачальник натискає «Мої послуги» → натискає «Створити» → заповнює форму → натискає «Створити» → послуга зберігається.

5. Видалення послуги (постачальник)

Постачальник натискає «Мої послуги» → натискає «Редагувати» → натискає «Видалити» → підтверджує → послуга видаляється.

6. Редагування послуги (постачальник)

Постачальник натискає «Мої послуги» → натискає «Редагувати» → змінює дані → натискає «Зберегти» → зміни зберігаються.

7. Перегляд записів постачальника

Постачальник натискає «Записи» у хедері → система відображає список усіх записів клієнтів (дата, час, послуга).

8. Запис клієнта на послугу

Клієнт натискає «Послуги» → обирає послугу → натискає «Записатись» → обирає дату й час → підтверджує → запис створюється.

9. Перегляд записів клієнта

Клієнт натискає «Мої записи» → система виводить список активних, майбутніх і минулих записів.

3.3.2 Системне програмне забезпечення

Для забезпечення функціонування комп'ютера та комп'ютерних мереж у межах розробки інформаційної системи було обґрунтовано вибір наступного комплексу системного програмного забезпечення:

1. Операційна система

Як основну операційну систему для серверів та робочих станцій обрано Microsoft Windows. Це рішення забезпечує високу сумісність із більшістю прикладного програмного забезпечення, підтримку широкого спектра апаратного забезпечення та стабільну роботу в мережевому середовищі.

2. RAD-засоби розробки проєкту

Для моделювання баз даних та підготовки графічної частини роботи використано наступні інструменти:

- Draw.io — для створення діаграм, схем і структур;
- Enterprise Architect — для розробки моделей системи;
- pgAdmin — для управління базами даних PostgreSQL у зручному графічному середовищі.

3. Основа для сайту

Для створення вебінтерфейсу системи використовуються класичні вебтехнології:

- HTML, CSS — для розмітки й стилізації сторінок;
- JavaScript — для інтерактивності клієнтської частини;

4. Сервісні програми

До складу допоміжного ПЗ для обслуговування операційної системи входять:

- CCleaner — для очищення системи та оптимізації її роботи;
- Avast [20] — для забезпечення базового антивірусного захисту.

5. Програми технічного обслуговування

Для діагностики й тестування апаратної частини системи використовується:

AIDA64 — програма для моніторингу стану апаратного забезпечення, тестування продуктивності та стабільності.

6. Програми машинної графіки

Для створення й редагування графічного контенту використовуються:

Figma — онлайн-інструмент для UI/UX-дизайну;

7. Інструментальне програмне забезпечення

Для написання й редагування коду обрано:

IntelliJ IDEA 2024.2.0.2 (Ultimate Edition) — багатофункціональне середовище розробки з підтримкою розширень, підсвічування синтаксису та інтеграцією з системами контролю версій.

8. Система контролю версій

Для керування версіями програмного коду використовується:

- Git — децентралізована система контролю версій, що забезпечує ефективне збереження змін у коді, спільну роботу в команді та відновлення попередніх станів проекту;

- GitHub — онлайн-сервіс для розміщення Git-репозиторіїв, що полегшує віддалену співпрацю, ведення історії змін і впровадження pull request-ів.

3.3.3 Прикладне програмне забезпечення

Прикладне програмне забезпечення — це програми, які безпосередньо реалізують функції інформаційної системи та забезпечують зручну взаємодію з користувачем. До його складу входять компоненти для роботи з базою даних, побудови вебінтерфейсу, обробки запитів та відображення інформації.

У межах створення інформаційної системи було реалізовано наступні компоненти прикладного програмного забезпечення:

1. Основна програма системи (проблемно-орієнтоване ПЗ)

Це головна частина системи, розроблена спеціально під потреби проєкту. Вона складається з двох частин:

- Серверна частина (бекенд) — реалізована за допомогою фреймворку Spring Boot. Вона відповідає за обробку запитів від користувача, перевірку даних, реалізацію бізнес-логіки та взаємодію з базою даних.
- Клієнтська частина (фронтенд) — створена з використанням HTML, CSS та JavaScript. Вона забезпечує інтерфейс користувача: форми введення, кнопки, сторінки з інформацією тощо.

2. Програми для роботи з базою даних

- Spring Data JPA — засіб для зручної роботи з базою даних через Java-код. Дозволяє швидко виконувати операції створення, читання, оновлення та видалення записів (CRUD).
- Liquibase або Flyway [21] — інструменти для керування змінами в структурі бази даних. Забезпечують контроль за міграціями бази та зберігають історію змін.

3.3.4 Програмна документація

Програмна документація охоплює ключові складові, необхідні для повного опису веб-додатку, та забезпечує всебічну підтримку як для користувачів, так і для технічного персоналу. Вона гарантує ефективну, стабільну та безпечну роботу інформаційної системи, спрямованої на автоматизацію процесів взаємодії між постачальниками послуг і клієнтами. Система дозволяє створювати, редагувати та зберігати дані, записуватись на послуги, керувати графіком доступності та організовувати зручну комунікацію між сторонами.

У Додатку В наведено приклади програмного коду, розробленого в межах створення веб-додатку.

Формуляр програмного забезпечення

Назва: Розроблення веб-застосунку для автоматизації взаємодії між постачальниками послуг та клієнтами

Розробник: Ляхович Ігор

Версія: 1.0

Склад програмного забезпечення: IntelliJ IDEA, PostgreSQL, pgAdmin, Figma, GitHub

Інформація про експлуатацію:

- Періодичні оновлення функціоналу.
- Підтримка з боку розробника.
- Підготовка до інтеграції з CRM, платіжними системами, аналітикою.

Призначення програми:

- Реєстрація та управління обліковими записами користувачів (клієнтів і постачальників)
- Формування розкладу послуг і доступного часу
- Запис клієнтів на послуги
- Зручне редагування, пошук і фільтрація доступних послуг

- Візуалізація розкладу в календарі
- Обмін повідомленнями, сповіщення та нагадування

Області використання

- Онлайн-запис до перукарів, майстрів манікюру, клінінгових сервісів тощо

- Організація розкладу роботи постачальників послуг
- Автоматизація бронювання та взаємодії з клієнтами
- Розширення цифрової присутності для малого бізнесу

Системні вимоги

- Працює в сучасному веббраузері (Chrome, Firefox, Edge, Safari)
- Не потребує потужного обладнання на боці користувача
- Серверна частина потребує середовище з підтримкою Java (Spring Boot) та бази даних PostgreSQL

3.3 Результати реалізації інформаційної системи

Для створення веб-додатку «Розроблення веб-застосунку для автоматизації взаємодії між постачальниками послуг та клієнтами» було обрано триярусну архітектуру. Такий вибір обумовлений низкою вагомих переваг.

По-перше, триярусна архітектура забезпечує масштабованість системи, що є важливим для подальшого розвитку додатку. Розділення системи на три рівні — презентаційний, логічний та рівень даних — дозволяє розробляти та вдосконалювати кожен із них незалежно, що значно спрощує процес розробки та подальшої підтримки.

По-друге, така архітектура дає змогу ефективно управляти системою та її компонентами. Кожен із рівнів може реалізовуватись окремо, що забезпечує гнучкість внесення змін і оновлень без суттєвого впливу на інші частини системи.

Крім того, триярусна архітектура чітко розділяє відповідальності між різними складовими системи. Презентаційний рівень відповідає за взаємодію з користувачем та відображення даних, логічний рівень — за реалізацію бізнес-логіки, а рівень даних — за збереження та доступ до інформації.

Розділення на рівні також спрощує тестування та супровід програмного забезпечення, оскільки дає змогу проводити окреме тестування компонентів, що значно полегшує виявлення і усунення помилок. Це сприяє підвищенню надійності та якості системи в цілому.

Для реалізації веб-додатку були використані технології Spring Boot, Spring Data JPA, Thymeleaf та база даних PostgreSQL, що забезпечує надійну та ефективну роботу з даними, а також зручний інтерфейс для користувачів.

Веб-додаток є мінімально життєздатним продуктом (MVP — minimum viable product), оскільки реалізує мінімально необхідний функціонал для ефективної взаємодії між постачальниками послуг та клієнтами, що дозволяє розпочати його експлуатацію та подальший розвиток.

Після запуску додатку, користувач має змогу вибрати свою роль (рис. 3.6)

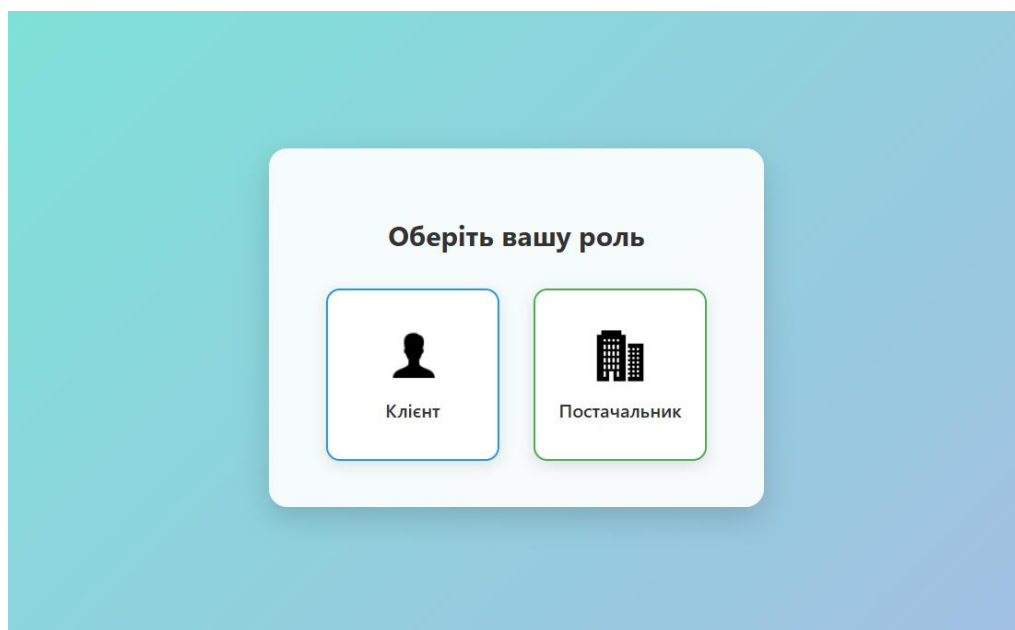
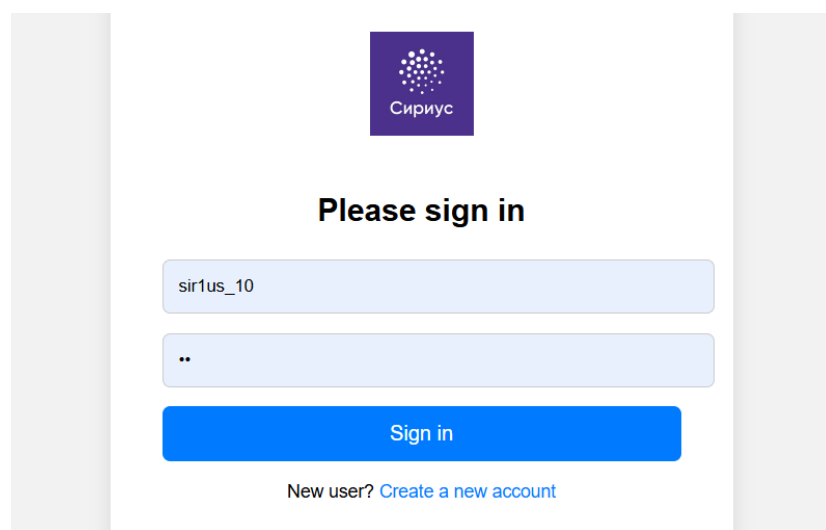


Рисунок 3.6 – Сторінка вибору ролі

Джерело: розроблено автором на основі виконаних досліджень

Після вибору ролі він потрапляє на сторінку входу(рис. 3.7)



Сириус

Please sign in

sir1us_10

..

Sign in

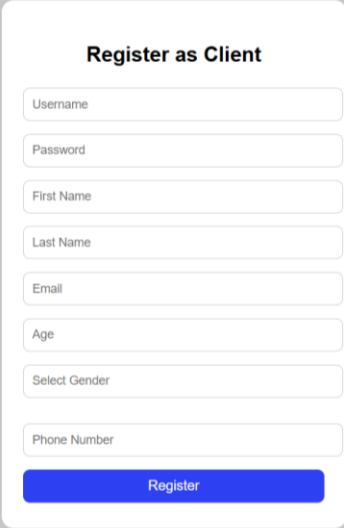
New user? [Create a new account](#)

Рисунок 3.7 – Сторінка входу в систему

Джерело: розроблено автором на основі виконаних досліджень

Користувач на цій сторінці може авторизуватися в системі, на головну сторінку(рис. 3.10 або рис. 3.14), в залежності від ролі,також є можливість перейти на сторінку реєстрації.

Форма реєстрації клієнта вимагає введення імені користувача, пароля, імені, прізвища, електронної пошти, віку, статі та номера телефону. (рис. 3.8) Після успішного заповнення даних і натискання кнопки "Register", користувач автоматично перенаправляється на сторінку входу. (рис. 3.7)

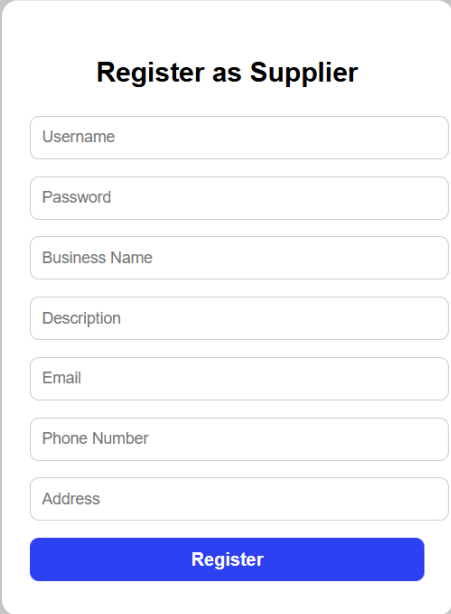


The image shows a 'Register as Client' form. It is a white rectangular box with rounded corners centered on a gray background. The title 'Register as Client' is at the top in bold. Below the title are eight input fields: 'Username', 'Password', 'First Name', 'Last Name', 'Email', 'Age', 'Select Gender' (a dropdown menu), and 'Phone Number'. At the bottom of the form is a blue button with the text 'Register' in white.

Рисунок 3.8 – Сторінка для реєстрації клієнта.

Джерело: розроблено автором на основі виконаних досліджень

Форма реєстрації постачальника вимагає введення імені користувача, пароля, назви бізнесу, опису, електронної пошти, номера телефону та адреси. (рис. 3.9) Після заповнення форми й натискання кнопки " Register ", у разі успіху, користувач автоматично перенаправляється на сторінку входу. (рис. 3.7)



The image shows a 'Register as Supplier' form. It is a white rectangular box with rounded corners centered on a gray background. The title 'Register as Supplier' is at the top in bold. Below the title are seven input fields: 'Username', 'Password', 'Business Name', 'Description', 'Email', 'Phone Number', and 'Address'. At the bottom of the form is a blue button with the text 'Register' in white.

Рисунок 3.9 – Сторінка для реєстрації постачальника.

Джерело: розроблено автором на основі виконаних досліджень

Для забезпечення зручної навігації між сторінками в залежності від ролі користувача в системі були створені окремі header та footer для кожної ролі.

Для користувачів з роллю постачальник послуг у верхній частині сторінок (header) передбачено такі елементи меню:

- Головна сторінка
- Профіль постачальника
- Всі послуги
- Мої записи

Для користувачів з роллю клієнт у відповідному header розміщено наступні пункти меню:

- Головна сторінка
- Профіль клієнта
- Послуги
- Записи

Таке розділення дозволяє адаптувати інтерфейс під потреби кожної ролі, підвищуючи зручність використання веб-додатку та покращуючи користувацький досвід.

На головній сторінці постачальника розміщено блок нагадувань, у якому відображаються важливі повідомлення для постачальника послуг, зокрема нагадування про необхідність оновлення розкладу своїх послуг на найближчі дні.

Також реалізовано інтерактивний календар, який у майбутньому дозволить постачальнику встановлювати конкретний час на надання послуги. Для цього достатньо буде обрати потрібну дату на календарі та вказати відповідні часові проміжки за допомогою інтуїтивно зрозумілого інтерфейсу. (рис. 3.10)

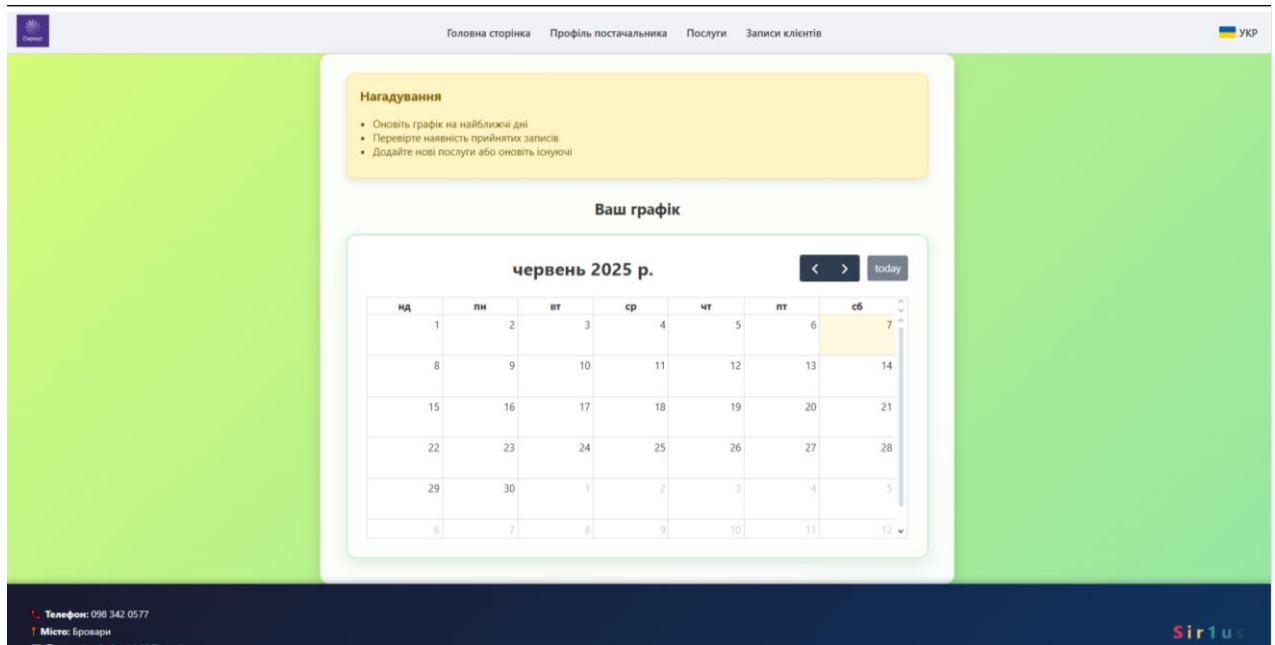


Рисунок 3.10 – Головна сторінка для постачальника

Джерело: розроблено автором на основі виконаних досліджень

На сторінці “Профіль постачальника” відображаються дані профілю постачальника з можливістю їх редагування. У найближчих оновленнях планується додати функціонал завантаження власних аватарок. Також реалізовано блок, де будуть представлені статистичні дані, що відображають результати діяльності компанії. (рис. 3.11)



Рисунок 3.11 – Профіль постачальника

Джерело: розроблено автором на основі виконаних досліджень

На сторінці “Послуг постачальника” представлено перелік його власних послуг з можливістю їх редагування та видалення, а також функціонал для створення нової послуги. (рис. 3.12)

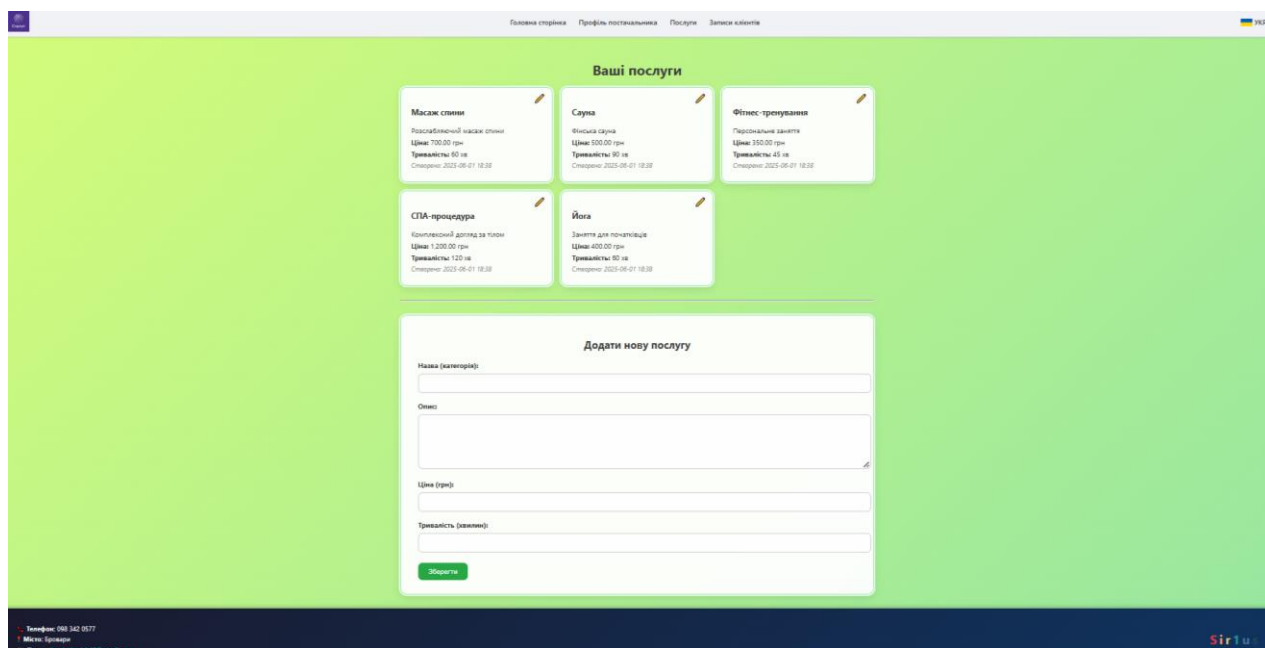


Рисунок 3.12 – Послуги постачальника

Джерело: розроблено автором на основі виконаних досліджень

На сторінці «Записи клієнтів» відображається список усіх записів із можливістю їх фільтрації. (рис. 3.13)

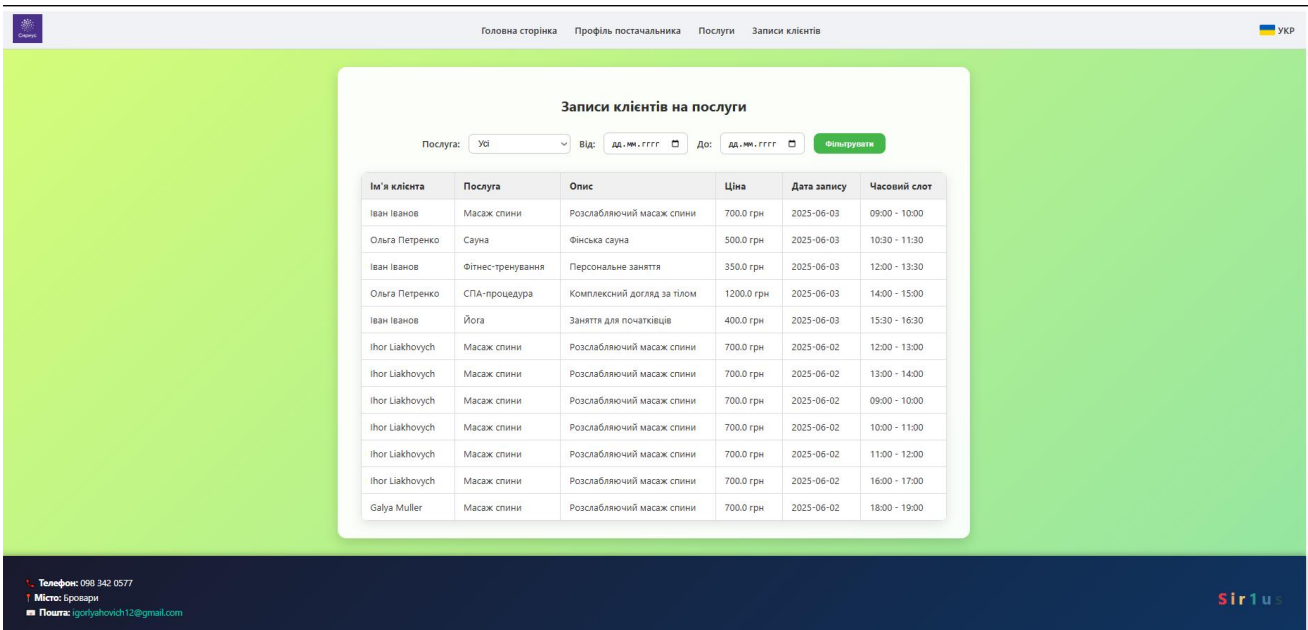


Рисунок 3.13 – Записи клієнтів до постачальника

Джерело: розроблено автором на основі виконаних досліджень

На головній сторінці клієнта розміщена, пошукова форма ,для пошуку даних ,про послуги ,та календар який повинен в майбутньому виводити дані, про записи клієнта цього дня . (рис. 3.14)

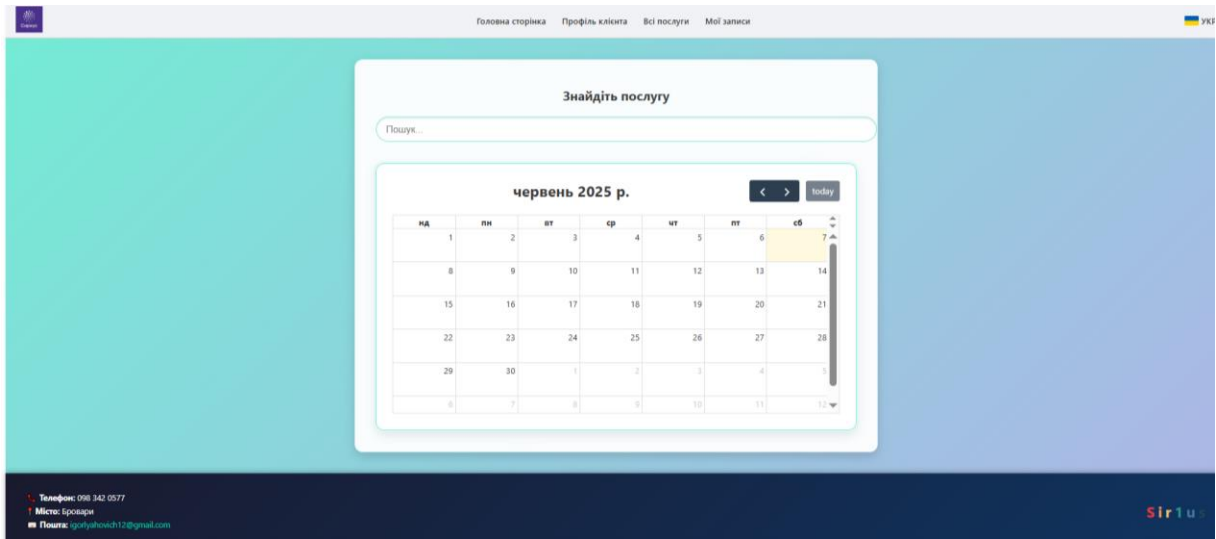


Рисунок 3.14 – Головна сторінка для клієнта

Джерело: розроблено автором на основі виконаних досліджень

На сторінці профілю клієнта передбачена можливість ,редагування даних в профілі ,та перегляд найпопулярніших послуг, які бронюють клієнти. (рис. 3.15)

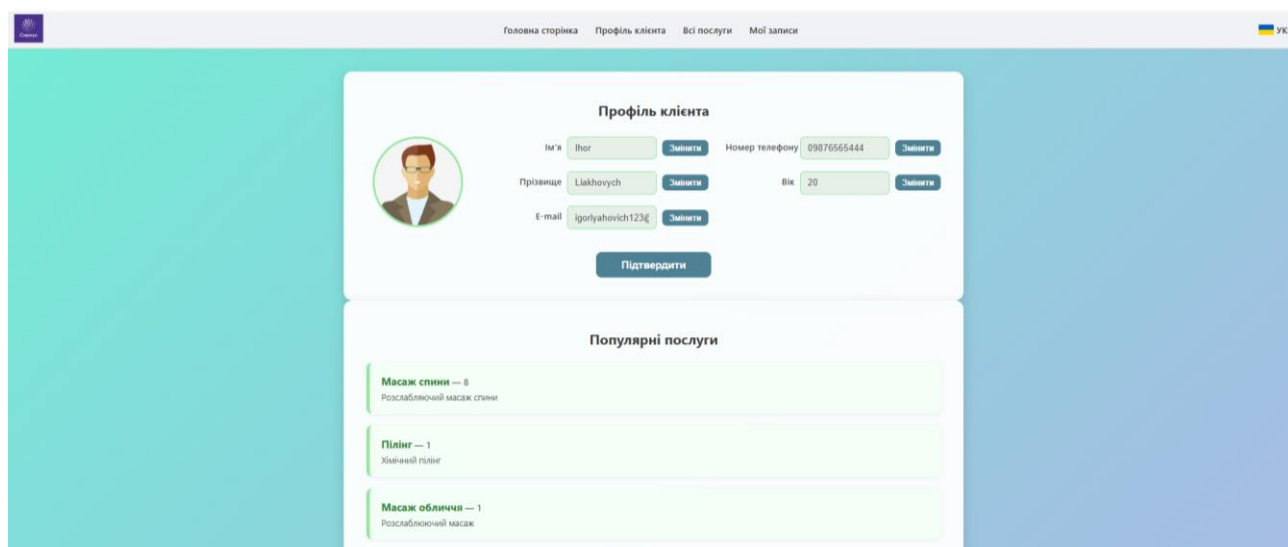


Рисунок 3.15 – Профіль клієнта

Джерело: розроблено автором на основі виконаних досліджень

На сторінці всіх послуг, є можливість переглядати всі послуги та бронювати потрібну ,вибравши можливу дату та час . (рис. 3.16)

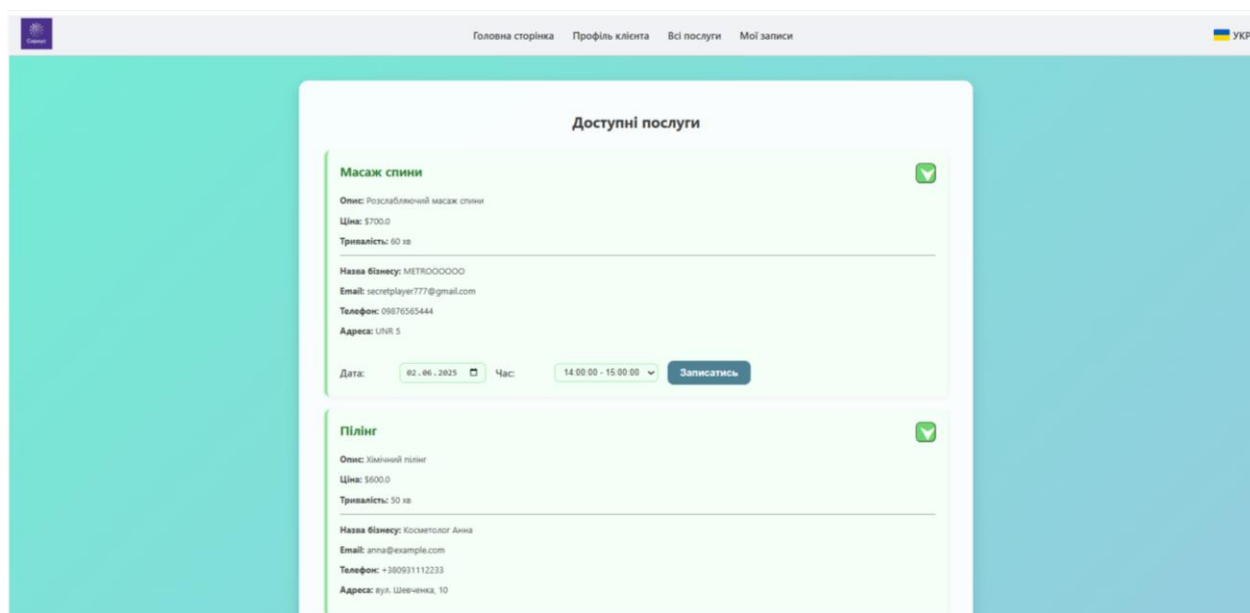


Рисунок 3.16 – Сторінка всіх послуг, для клієнта

Джерело: розроблено автором на основі виконаних досліджень

На сторінці ,мої бронювання, клієнт може переглянути всі свої бронювання, фільтрувати їх та видалити ,якщо потрібно(рис. 3.17)

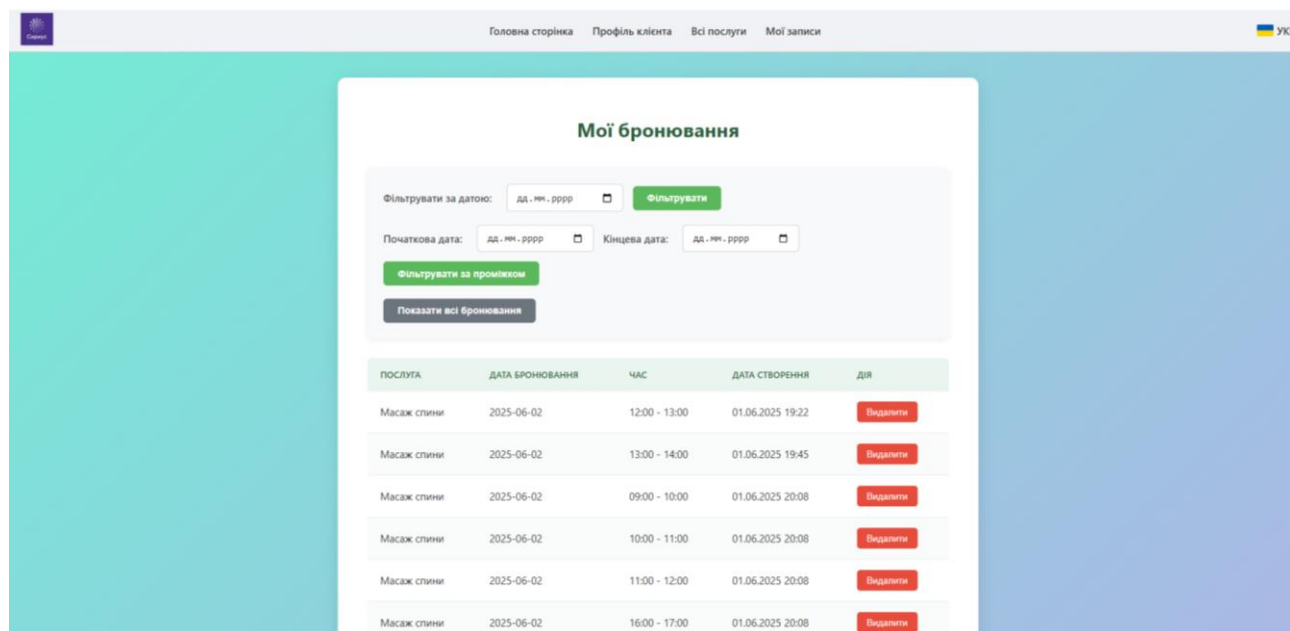


Рисунок 3.17 – Сторінка мої бронювання

Джерело: розроблено автором на основі виконаних досліджень

Розроблений веб-додаток для автоматизації взаємодії між постачальниками послуг та клієнтами має суттєву практичну цінність. Використання триярусної архітектури забезпечує масштабованість, гнучкість і надійність системи, що є важливими факторами для її подальшого розвитку та адаптації під змінні вимоги користувачів. Розділення системи на презентаційний, логічний та рівень даних дозволяє незалежно вдосконалювати кожен із компонентів, що значно підвищує ефективність супроводу та розширення функціоналу.

З економічної точки зору, впровадження розробленого додатку сприятиме автоматизації ключових бізнес-процесів, зменшенню трудових витрат на обробку замовлень і бронювання, а також мінімізації помилок, що виникають у процесі взаємодії між клієнтами та постачальниками послуг. Це, у свою чергу, підвищить продуктивність роботи підприємств малого і середнього бізнесу, які надають індивідуальні послуги, і дозволить їм ефективніше управляти ресурсами.

На даному етапі веб-додаток представлений як мінімально життєздатний продукт (MVP), що реалізує основний набір функцій для початкової експлуатації. Для виходу на ринок та масштабування системи необхідно провести комплексне тестування з реальними користувачами, оптимізувати інтерфейс та інтегрувати додаткові сервіси, зокрема платіжні шлюзи. Наявність окремих меню та адаптованих інтерфейсів для різних ролей користувачів (постачальник, клієнт) забезпечує зручність і персоналізацію роботи з додатком.

Перспективи подальшого розвитку включають:

- Розширення функціоналу інтерактивного календаря для гнучкого управління графіком надання послуг;
- Інтеграцію з платіжними системами для автоматизації процесу оплати;
- Розробку мобільної версії або додатку для збільшення доступності сервісу;
- Впровадження аналітичних інструментів для моніторингу діяльності та прийняття управлінських рішень;
- Використання технологій машинного навчання для прогнозування попиту та оптимізації ресурсів.

Таким чином, розроблений веб-додаток є базою для створення конкурентоспроможної системи управління послугами з високим потенціалом для комерційного впровадження і подальшого масштабування.

ВИСНОВКИ

У сучасних умовах стрімкого розвитку цифрових технологій особливої актуальності набуває автоматизація процесів взаємодії між постачальниками послуг та клієнтами. У багатьох сферах обслуговування спостерігається потреба в зручних, ефективних і прозорих інструментах для організації запису на послуги, управління розкладом, комунікації та обробки запитів. Водночас значна частина сервісів все ще функціонує за застарілими моделями, що призводить до неефективності, втрати клієнтів і труднощів в управлінні даними. Ці виклики стали мотивацією для розроблення інтегрованого веб-додатку, який забезпечує централізовану платформу для взаємодії між клієнтами та постачальниками послуг, автоматизує основні бізнес-процеси та підвищує рівень сервісу.

У ході виконання бакалаврської дипломної роботи було успішно розроблено інформаційну систему, що відповідає поставленим завданням та цілям. Основною метою роботи було створення програмного продукту, який забезпечує автоматизацію ключових бізнес-процесів, підвищення ефективності обробки даних та покращення взаємодії користувачів із системою.

У першому розділі було здійснено глибокий аналіз предметної області, визначено актуальність проблематики, описано стан сфери індивідуальних послуг в Україні, наявні сайти, їхні переваги та недоліки. Було сформульовано об'єкт, предмет, мету й завдання дослідження.

Другий розділ присвячено аналізу функціональних та нефункціональних вимог до системи, а також їх формалізації за допомогою специфікацій. Для моделювання структури, поведінки та логіки системи були використані засоби SysML у середовищі Enterprise Architect. Зокрема, у розділі наведено інформаційну модель, діаграму діяльності, діаграму прецедентів, діаграму послідовності, діаграму класів та діаграму трасування, що надали повне та формалізоване уявлення про систему і її функціональні взаємозв'язки.

У третьому розділі детально розглянуто інформаційне, технічне та програмне забезпечення системи. Розроблено інфологічну та даталогічну моделі бази даних, обґрунтовано вибір мови програмування, бази даних та HTML для створення додатку. Крім того, продемонстровано структуру реалізованого сайту, його функціональні компоненти та сценарії взаємодії користувачів.

Фінальним етапом стала практична реалізація системи на основі розроблених моделей і специфікацій. Результатом є адаптивний, безпечний і функціональний сайт, який може бути використаний для реального використання в малому та середньому бізнесі.

Таким чином, було досягнуто мету бакалаврської роботи – розроблено та реалізовано автоматизовану інформаційну систему для ефективного управління веб-додатком для автоматизації взаємодії між клієнтами та постачальниками послуг. Реалізовано повний цикл створення системи: від аналізу предметної області до практичного створення. Запропоноване рішення може бути використане для реального впровадження або подальшого розвитку в комерційній чи державній сфері.

Отже, виконана бакалаврська робота дозволила реалізувати повноцінний цикл створення сучасної веб-орієнтованої інформаційної системи. Розроблений програмний продукт має потенціал до подальшого розвитку та комерційного використання. В результаті виконання проєкту було досягнуто поставленої мети, вирішено всі завдання, а також здобуто цінний досвід роботи над комплексним програмним рішенням.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Fielding Dissertation: CHAPTER 5: Representational State Transfer (REST). Home / UC Irvine Donald Bren School of Information & Computer Sciences. URL: https://ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm (date of access: 25.05.2025).
2. Fowler M. Patterns of Enterprise Application Architecture. Addison-Wesley Professional, 2002. 560 p.
3. ЦИФРОВІЗАЦІЯ ЕКОНОМІКИ: НОВІ МОЖЛИВОСТІ ТА ПЕРСПЕКТИВИ / М В Руденко / Економіка та держава №11 2018 стр 61-65. Журнал Економіка та держава / наукове фахове видання України з економіки. URL: <http://www.economy.in.ua/?op=1&z=4191&i=10> (дата звернення: 25.05.2025).
4. Booksy - hair stylists, barbers, beauticians... book appointments online! / Booksy. URL: https://booksy.com/en-us/#ba_s=seo (date of access: 25.05.2025).
5. Fresha. Top Salon Software / Salon Management Software / Best Salon Booking Software / Spa Software / Salon Scheduling Software / Top 10 salon software / Top 10 barber software. Fresha / Instantly book salons and spas nearby. URL: <https://www.fresha.com/for-business> (date of access: 25.05.2025).
6. Хмарні технології в освіті: застосування та вплив / Wezom. ІТ-компанія повного циклу розробки програмних продуктів WEZOM / Київ, Україна. URL: <https://wezom.com.ua/ua/blog/rol-ta-vpliv-hmarnih-tehnologiy-v-osviti> (дата звернення: 25.05.2025).
7. Що таке функціональні вимоги: приклади, визначення, повний посібник / Visure Solutions, visuresolutions.com/uk/блог/функціональні-вимоги. Останній перегляд 12 трав. 2025.

8. Визначення нефункціональних вимог / Онлайн-курси від компанії QATestLab / Головна сторінка. URL: <https://training.qatestlab.com/blog/technical-articles/non-functional-requirements-examples-types-approaches/> (дата звернення: 25.05.2025).
9. GeeksforGeeks. Activity Diagrams / Unified Modeling Language (UML) / GeeksforGeeks. URL: <https://www.geeksforgeeks.org/unified-modeling-language-uml-activity-diagrams/> (date of access: 25.05.2025).
10. Моралес Дж. In-depth knowledge of UML use case diagram / with tutorial. MindOnMap / Free Mind Mapping Tool to Draw Ideas Easily Online. URL: <https://www.mindonmap.com/uk/blog/what-is-a-uml-use-case-diagram/> (date of access: 06.06.2025).
11. UML class diagram tutorial / Ideal Modeling & Diagramming Tool for Agile Team Collaboration. URL: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/> (date of access: 25.05.2025).
12. IBM. What Is Three-Tier Architecture? / IBM. IBM / United States. URL: <https://www.ibm.com/think/topics/three-tier-architecture> (date of access: 25.05.2025).
13. GeeksforGeeks. Learn Web Development Basics with HTML CSS and JavaScript / GeeksforGeeks. URL: <https://www.geeksforgeeks.org/javascript/learn-web-development-basics-with-html-css-and-javascript/> (date of access: 25.05.2025).
14. PostgreSQL. PostgreSQL. URL: <https://www.postgresql.org> (date of access: 07.06.2025).
15. PostgreSQL що це і як застосовується в різних напрямках / FoxmindEd. URL: <https://foxminded.ua/postgresql-shcho-tse/> (дата звернення: 25.05.2025).
16. UUID type / PostgreSQL Documentation. URL: <https://www.postgresql.org/docs/current/datatype-uuid.html> (date of access: 25.05.2025).

17. Jakarta Persistence (JPA) / IntelliJ IDEA. IntelliJ IDEA Help. URL: <https://www.jetbrains.com/help/idea/jakarta-persistence-jpa.html#share-facet-settings> (date of access: 25.05.2025).
18. Railway. Railway. URL: <https://railway.com> (date of access: 25.05.2025).
19. spring boot – про програмування / Про Програмування – ABout IT And Programming. URL: <https://abitap.com/1-2-spring-boot/> (дата звернення: 25.05.2025).
20. Avast / Завантажте Free Antivirus і VPN / Просто й безкоштовно. Avast / Завантажте Free Antivirus і VPN / Просто й безкоштовно. URL: <https://www.avast.ua/index#pc> (дата звернення: 25.05.2025).
21. Автоматизоване проєктування інформаційних систем / навч. посіб. / О.О. Денісова. – К. / КНЕУ, 2011. – 412 с.
22. Проєктування баз і сховищ даних / Навч.-метод. посіб. для самост. вивч. дисц. / Ситник Н.В., Краснюк М.Т. — К. / КНЕУ, 2005. — 264 с.
23. Документація для роботи з Enterprise Architect [Електронний ресурс] / Режим доступу до ресурсу: <https://sparxsystems.com/>
24. Beauty Pro. Beauty Salon Software / Hair Salon and Spa Management System / Beauty Pro. URL: <https://beautyprosoftware.com> (date of access: 25.05.2025).
25. Побудова динамічних вебзастосунків за допомогою MVC / Стаття з блогу IT-школи Hillel. Корисні матеріали / Статті та новини IT-індустрії / Комп'ютерна школа Hillel. URL: <https://blog.ithillel.ua/articles/building-dynamic-web-applications-using-mvc> (дата звернення: 25.05.2025)

ДОДАТКИ

Додаток А

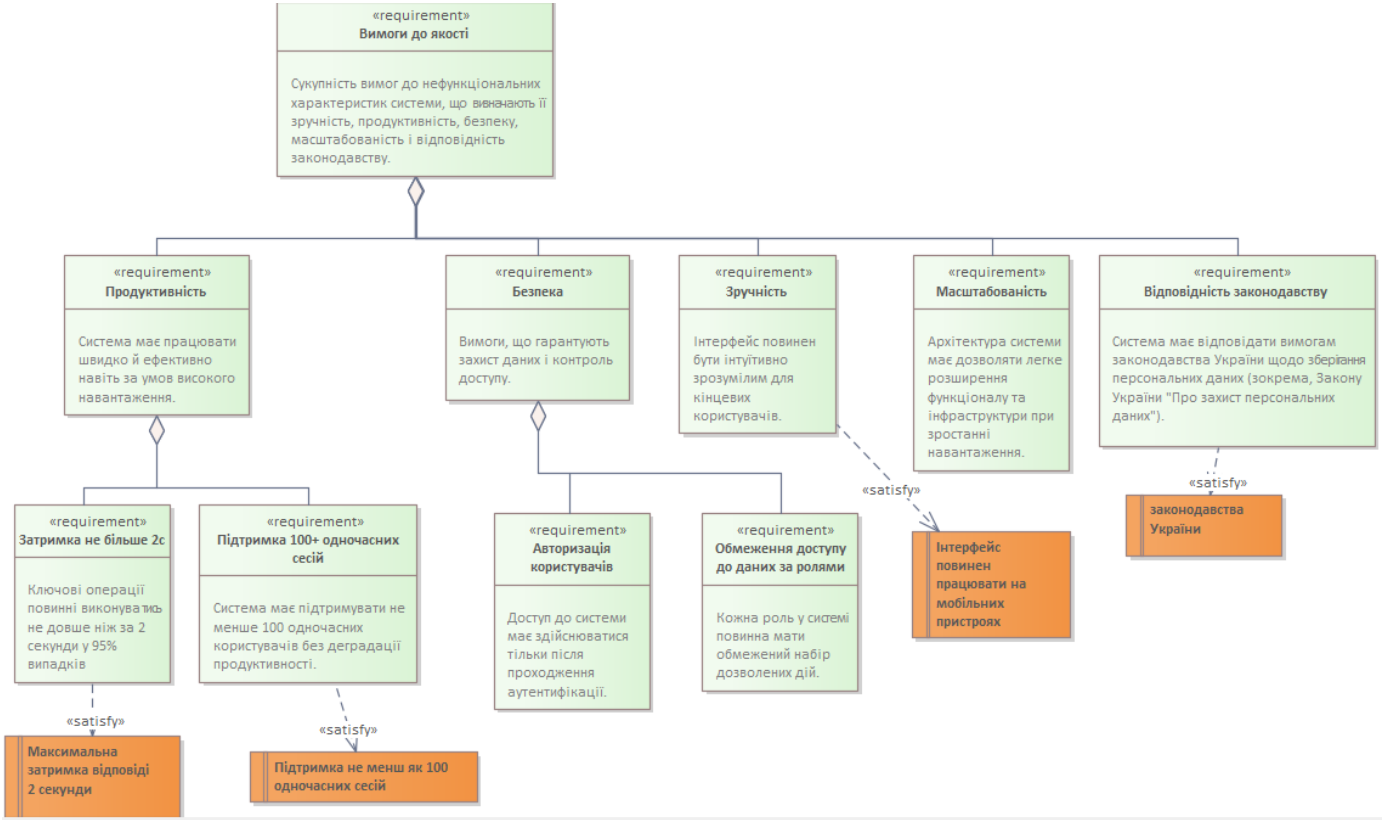


Рисунок А.1 – Діаграма нефункціональних вимог

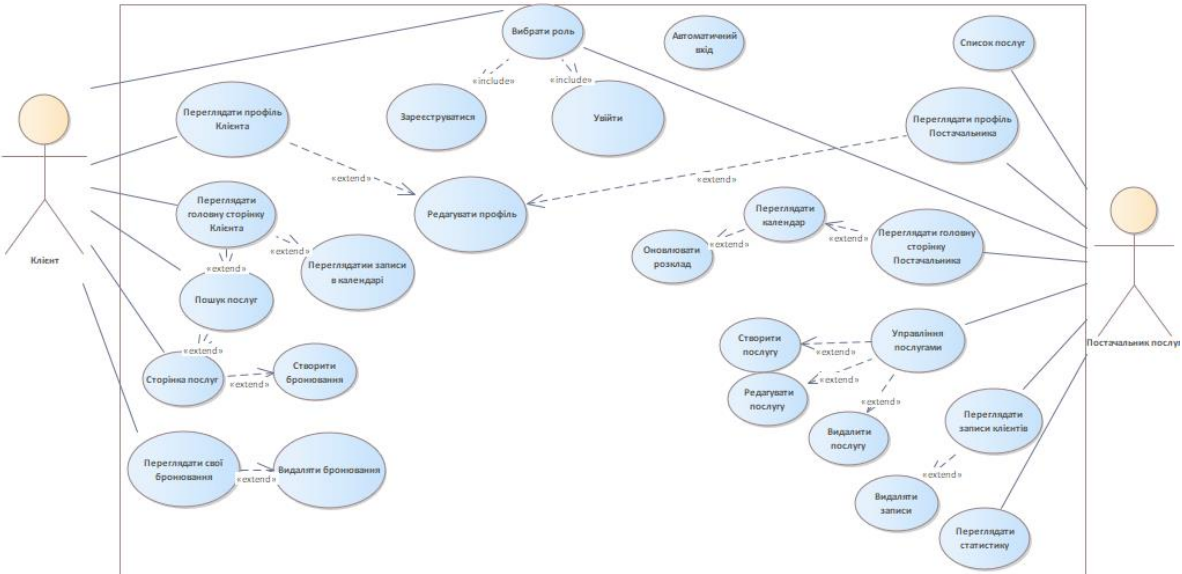


Рисунок А.2 – Діаграма прецендентів

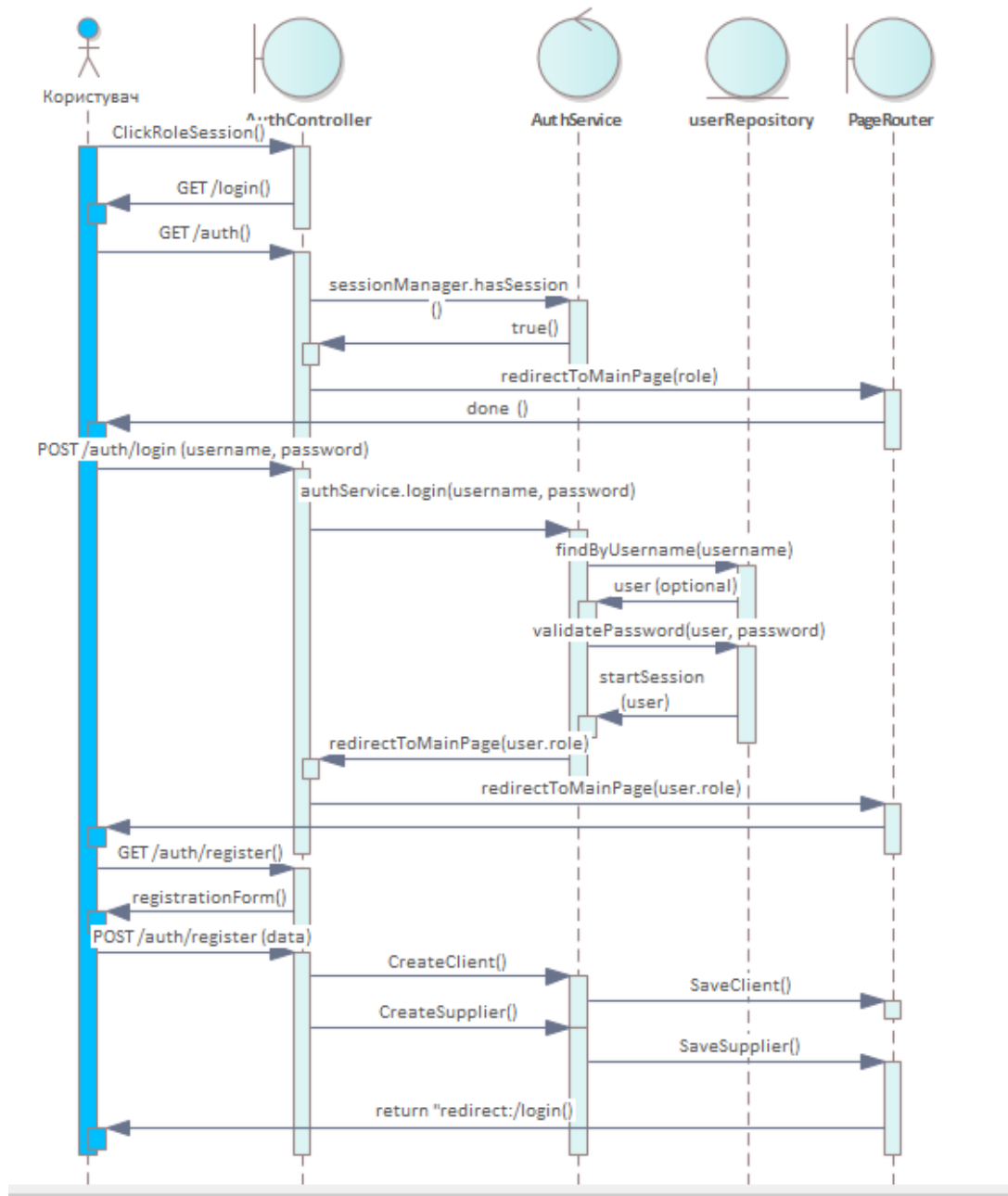


Рисунок А.3 – Діаграма послідовності: Процес автентифікації та реєстрації

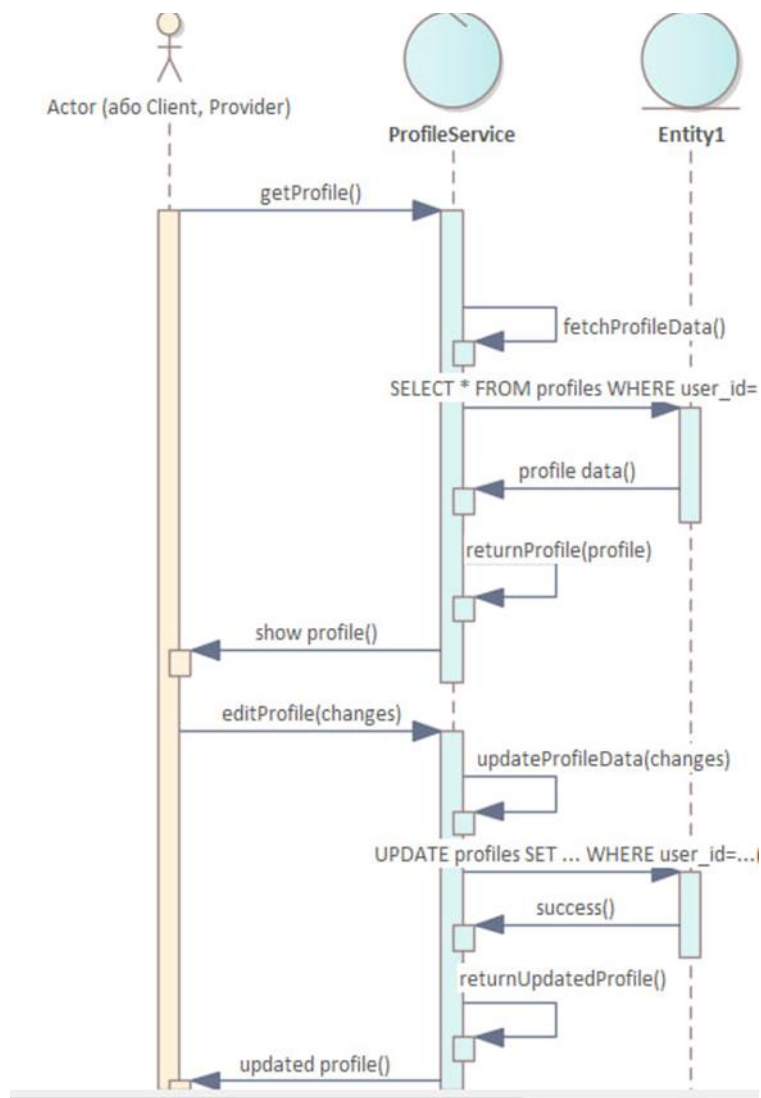


Рисунок А.4. Діаграма послідовності: Взаємодія з профілями.

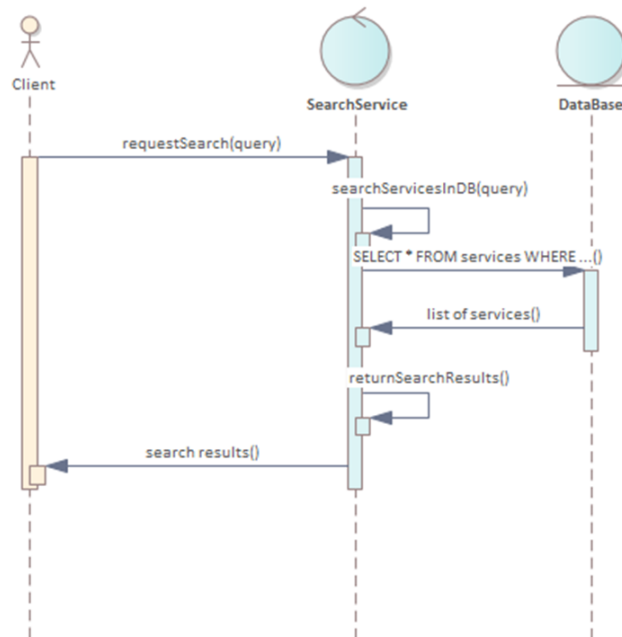


Рисунок А.5. Діаграма послідовності: Пошук.

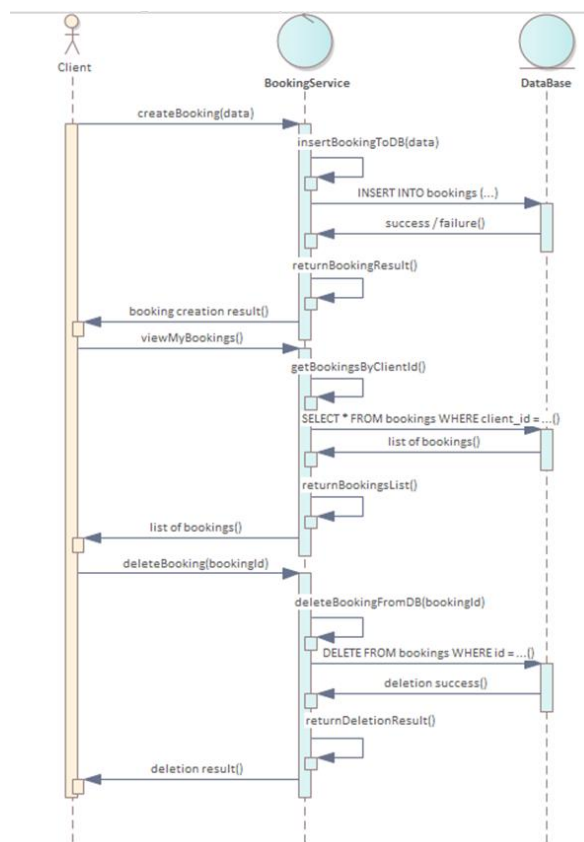


Рисунок А.6. Діаграма послідовності: Дії з бронюванням клієнта

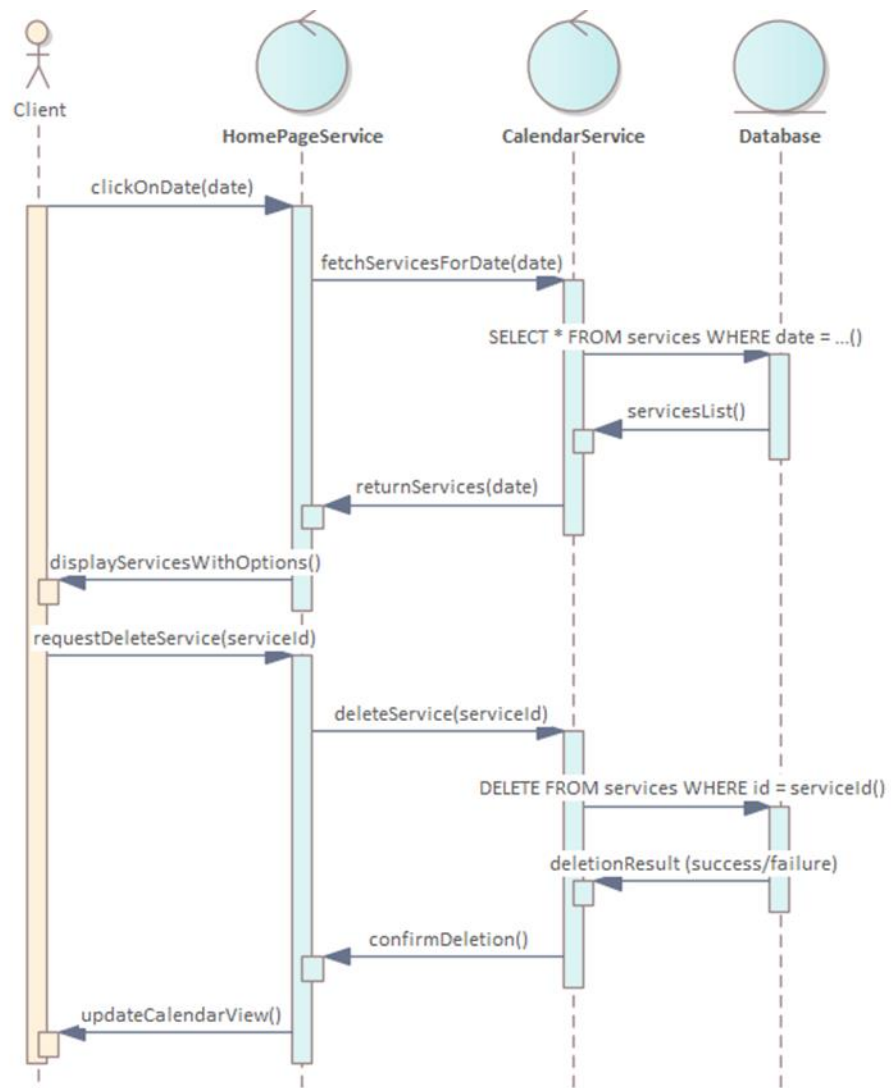


Рисунок А.7. Діаграма послідовності: Календар клієнта

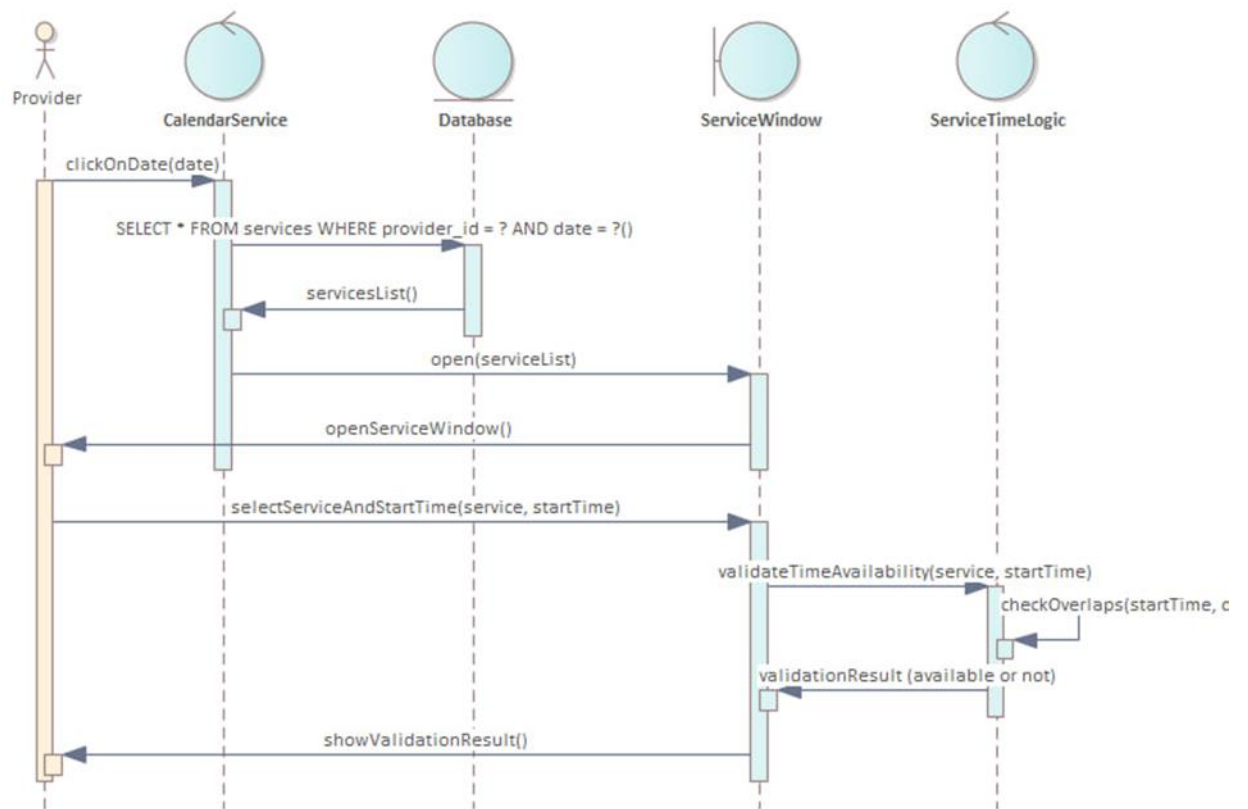


Рисунок А.8. Діаграма послідовності: Календар постачальника

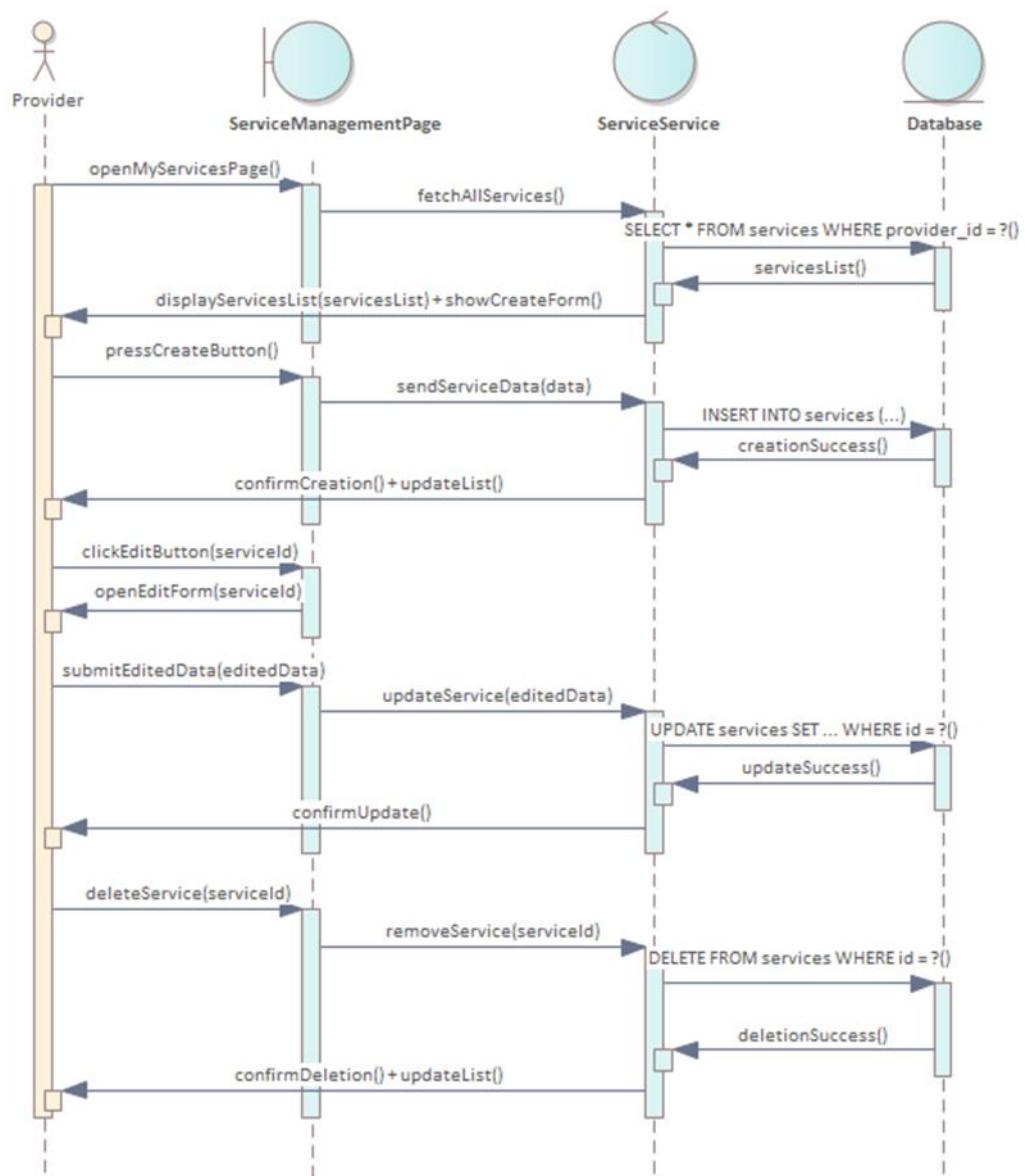


Рисунок А.9. Діаграма послідовності: Послуги постачальника

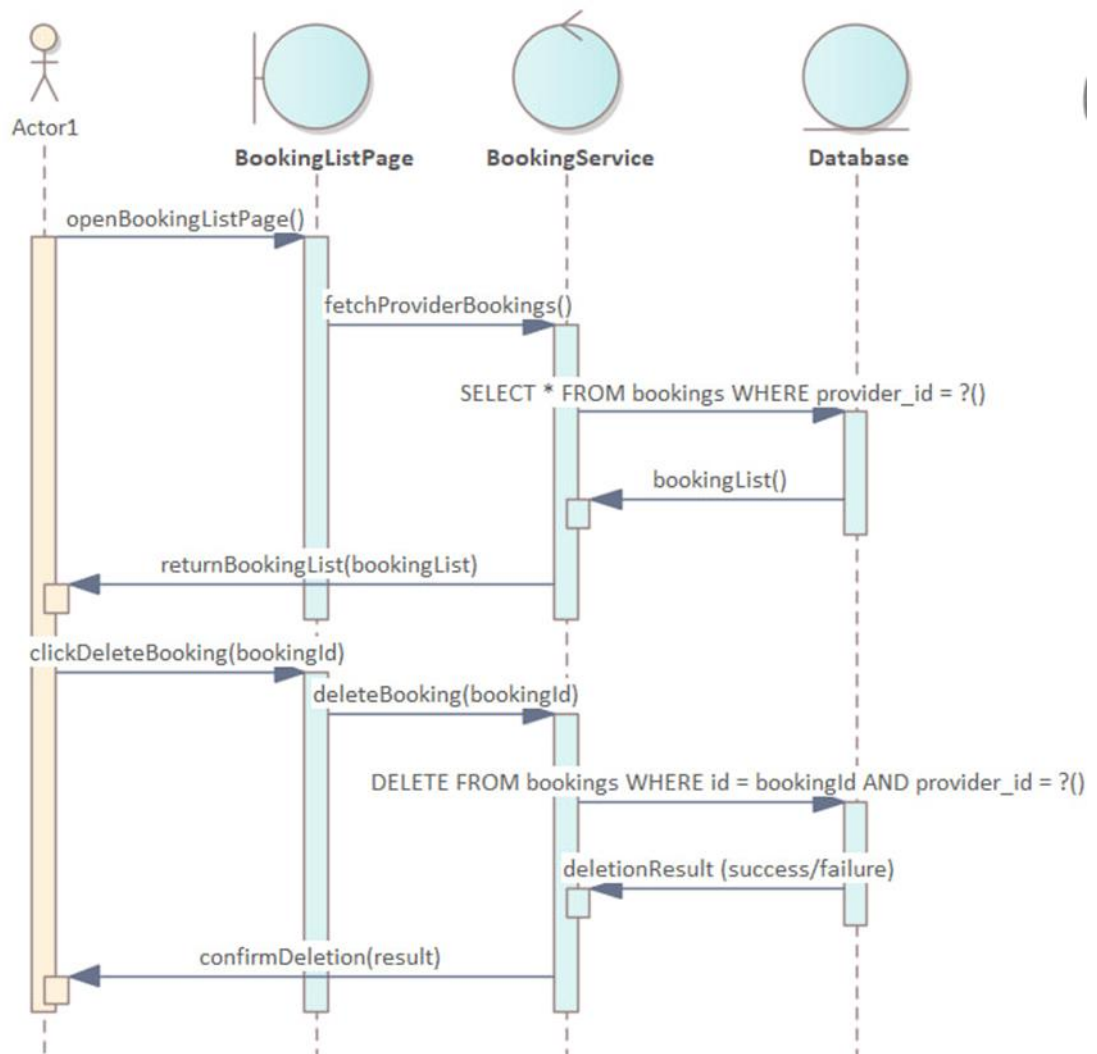


Рисунок А.10. Діаграма послідовності: Послуги постачальника

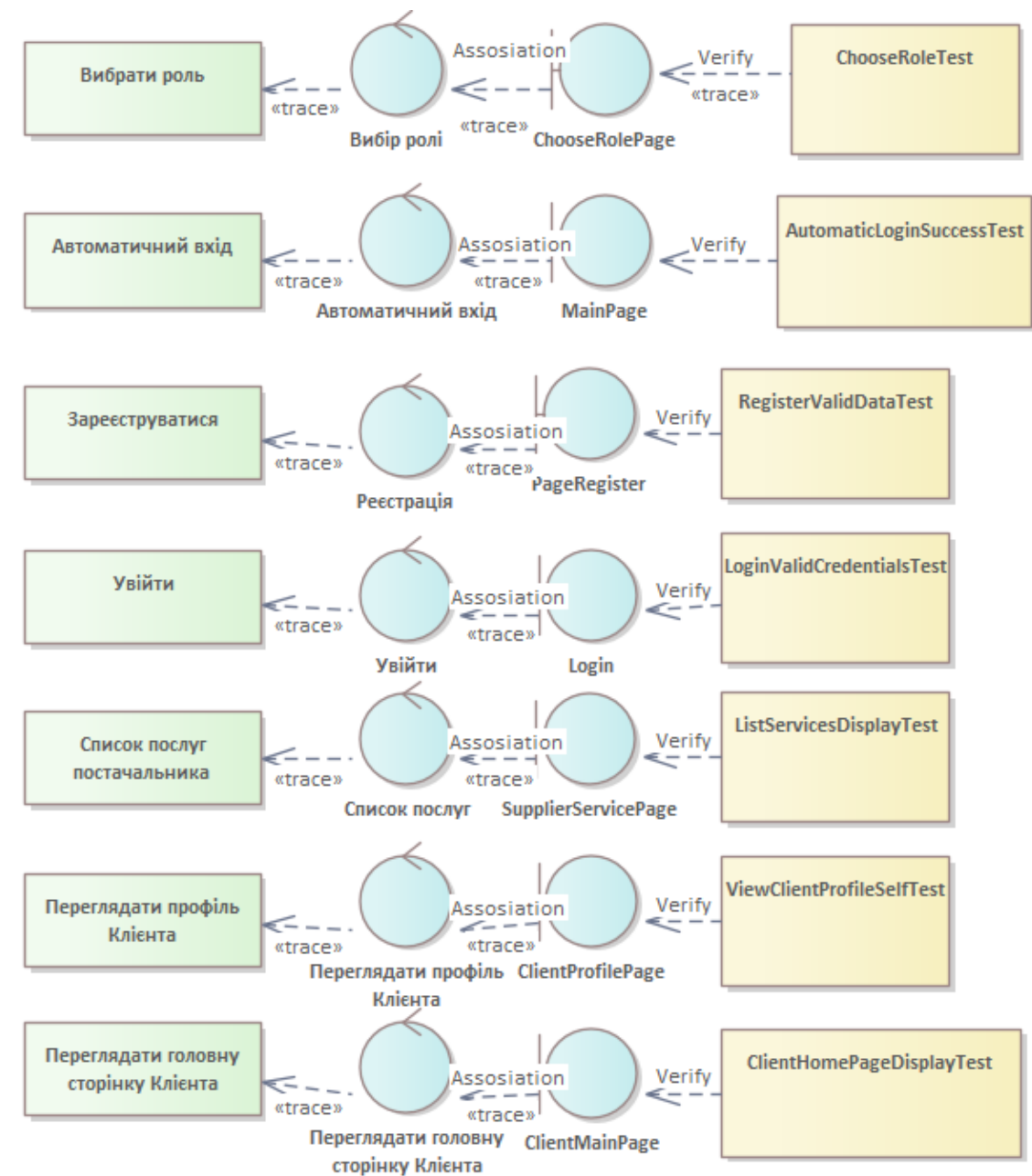


Рисунок А.12. Діаграма трасування

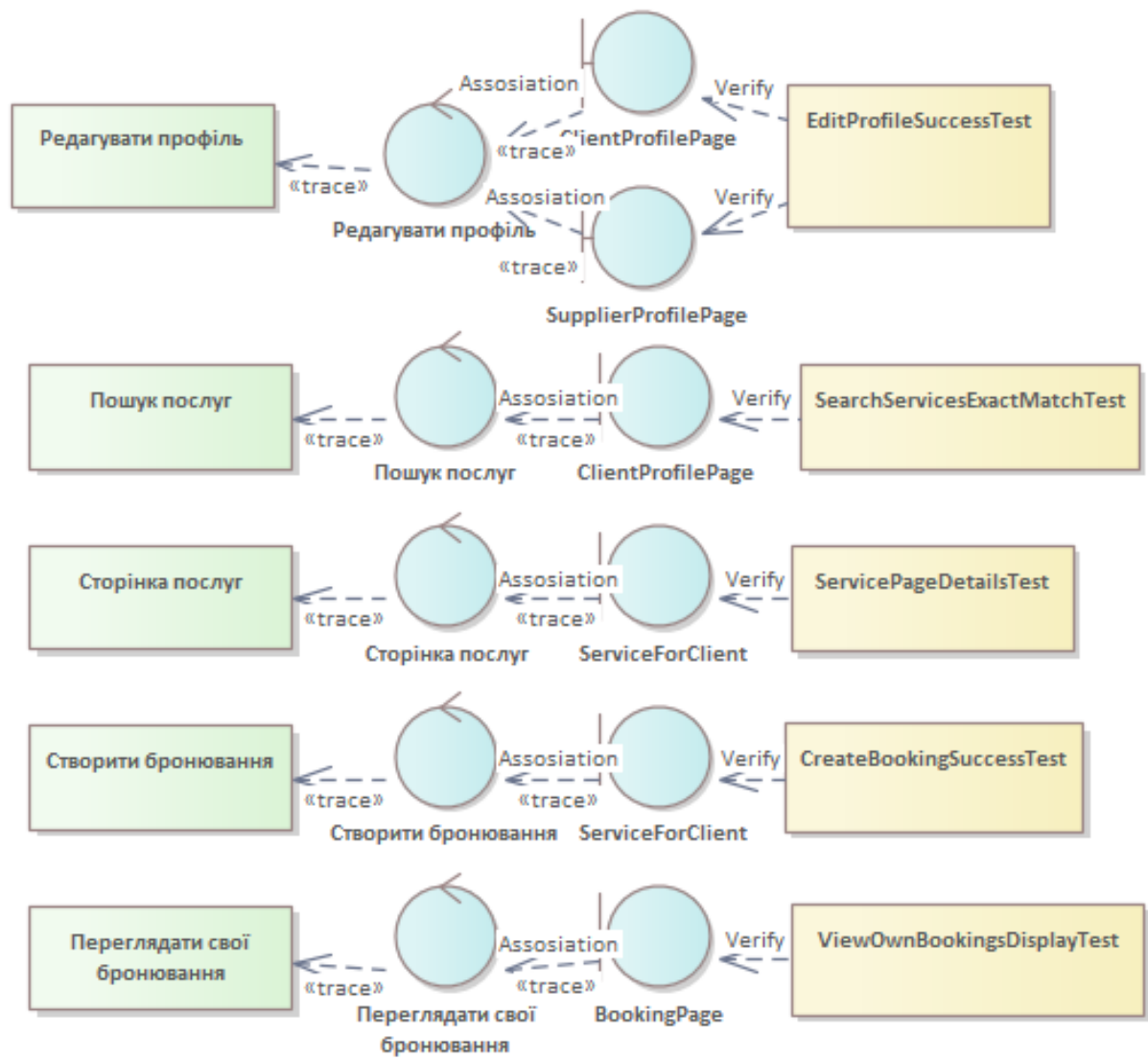


Рисунок А.13. Діаграма трасування

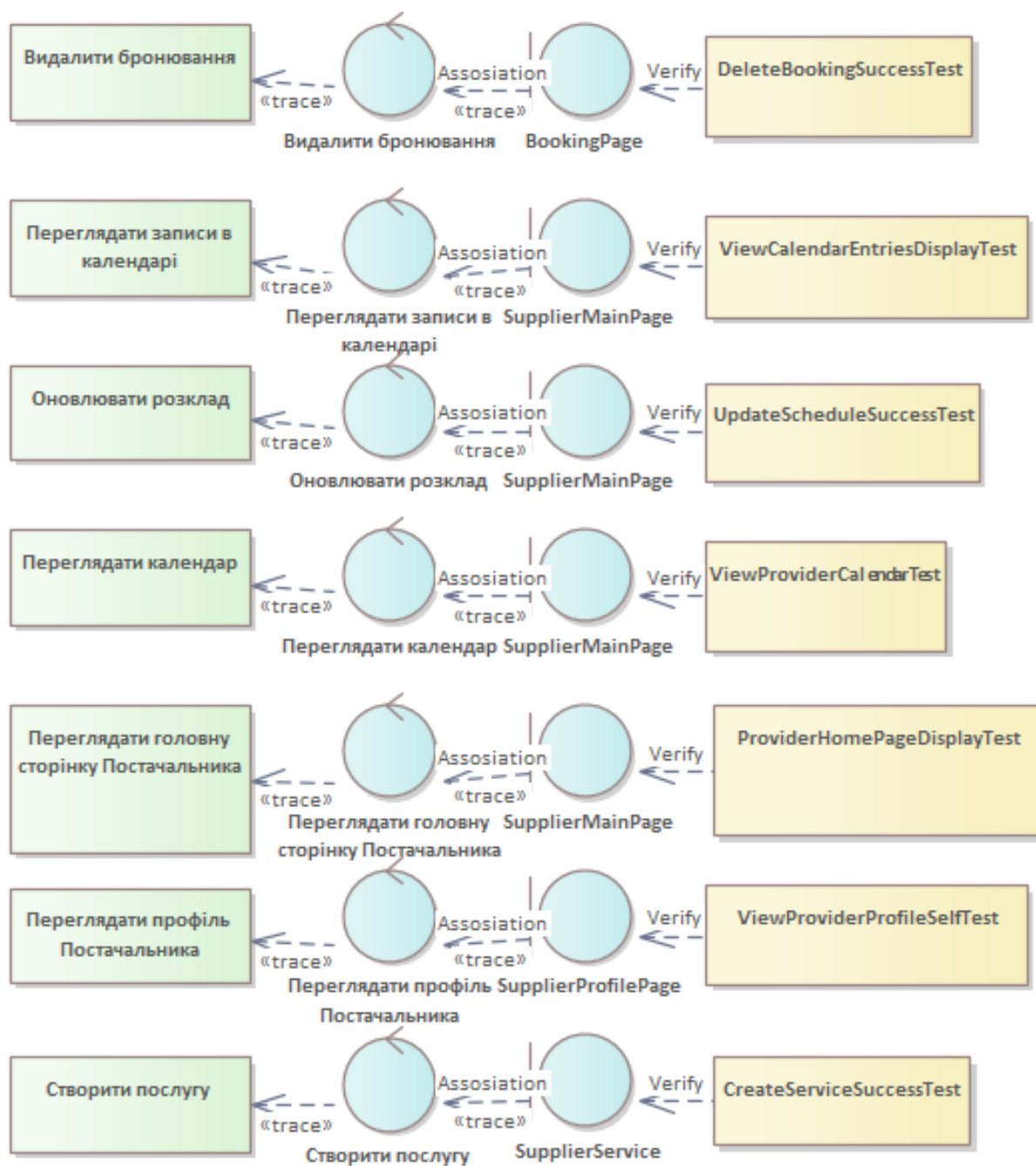


Рисунок А.14. Діаграма трасування

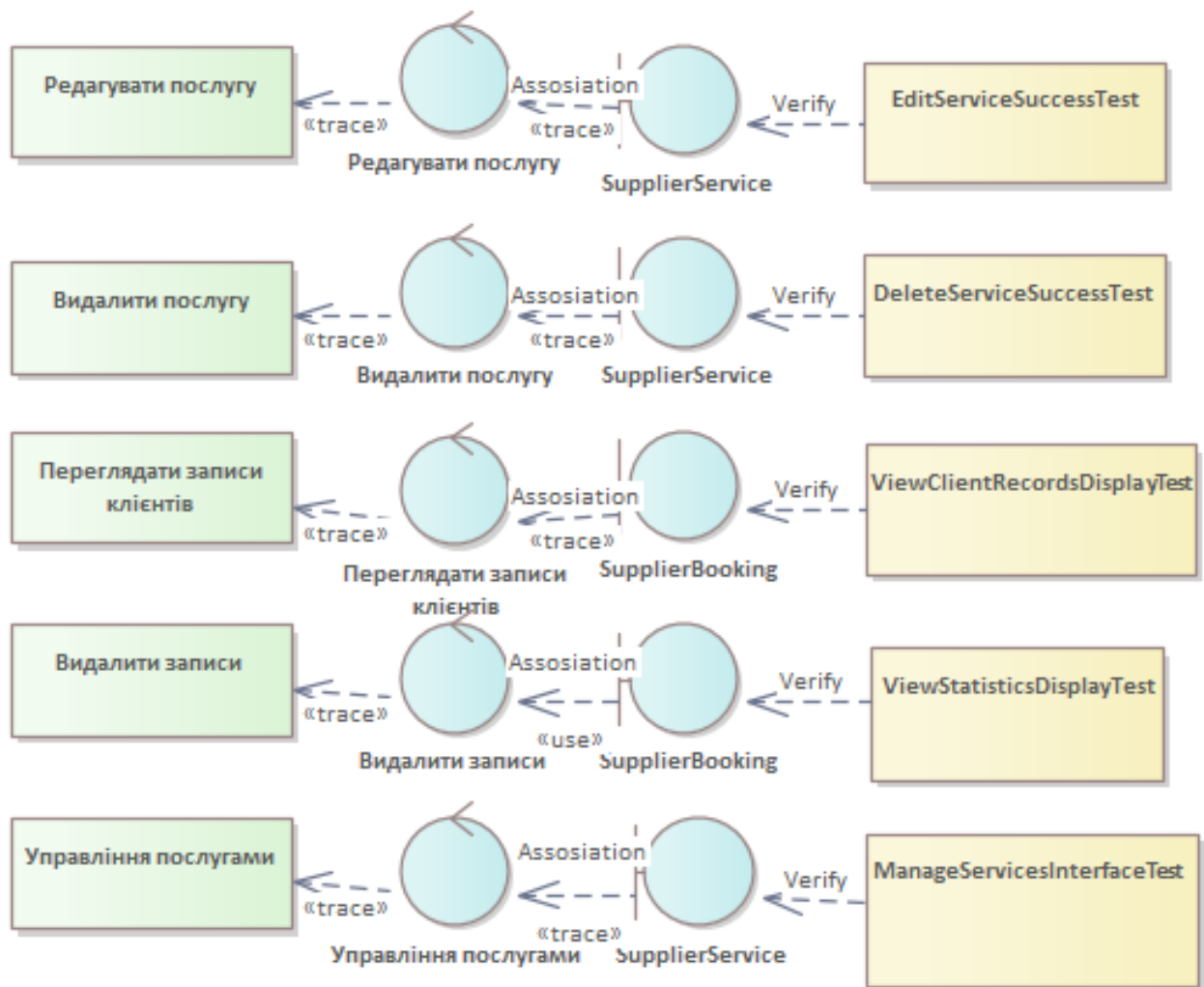


Рисунок А.15. Діаграма трасування

```
CREATE TABLE users (  
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),  
    username VARCHAR(255) UNIQUE NOT NULL,  
    password VARCHAR(255) NOT NULL,  
    role VARCHAR(20) CHECK (role IN ('CLIENT', 'SUPPLIER')) NOT NULL);
```

Б.1. Створення таблиці users

```
CREATE TABLE clients (  
    id UUID PRIMARY KEY,  
    first_name VARCHAR(255),  
    last_name VARCHAR(255),  
    age INTEGER,  
    gender VARCHAR(10),  
    email VARCHAR(255) UNIQUE,  
    phone VARCHAR(20),  
    avatar_url TEXT,  
    FOREIGN KEY (id) REFERENCES users(id) ON DELETE CASCADE  
);
```

Б.2. Створення таблиці clients

```
CREATE TABLE suppliers (  
    id UUID PRIMARY KEY,  
    business_name VARCHAR(255),  
    description TEXT,  
    email VARCHAR(255) UNIQUE,  
    phone VARCHAR(20),  
    address TEXT,  
    avatar_url TEXT,  
    FOREIGN KEY (id) REFERENCES users(id) ON DELETE CASCADE  
);
```

Б.3. Створення таблиці suppliers

```
CREATE TABLE services (  
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),  
    supplier_id UUID NOT NULL,  
    name VARCHAR(255) NOT NULL,  
    description TEXT,  
    price NUMERIC(10, 2),  
    duration_minutes INTEGER,  
    FOREIGN KEY (supplier_id) REFERENCES suppliers(id) ON DELETE CASCADE  
);
```

Б.4. Створення таблиці services

```
CREATE TABLE time_slots (  
    id SERIAL PRIMARY KEY,  
    start_time TIME NOT NULL,  
    end_time TIME NOT NULL  
);
```

Б.5. Створення таблиці time_slots

```
CREATE TABLE supplier_schedules (  
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),  
    supplier_id UUID NOT NULL,  
    service_id UUID NOT NULL,  
    date DATE NOT NULL,  
    time_slot_id INTEGER NOT NULL,  
    FOREIGN KEY (supplier_id) REFERENCES suppliers(id) ON DELETE CASCADE,  
    FOREIGN KEY (service_id) REFERENCES services(id) ON DELETE CASCADE,  
    FOREIGN KEY (time_slot_id) REFERENCES time_slots(id),  
    UNIQUE (supplier_id, service_id, date, time_slot_id));
```

Б.6. Створення таблиці supplier_schedules

```
CREATE TABLE client_bookings (  
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),  
    client_id UUID NOT NULL,  
    schedule_id UUID NOT NULL,  
    booking_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
    FOREIGN KEY (client_id) REFERENCES clients(id) ON DELETE CASCADE,  
    FOREIGN KEY (schedule_id) REFERENCES supplier_schedules(id) ON DELETE CASCADE,  
    UNIQUE (client_id, schedule_id)  
);
```

Б.7. Створення таблиці client_bookings

Фрагмент коду

Spring Data JPA фрагмент:

```
package ua.edu.kneu.serviceflow.entity;
import jakarta.persistence.*;
import lombok.AllArgsConstructor;
import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;
import java.time.LocalDateTime;
import java.util.UUID;
@Entity
@Table(name = "services")
@Getter
@Setter
@NoArgsConstructor
@AllArgsConstructor
public class Services {
    @Id
    @GeneratedValue
    @Column(name = "id")
    private UUID id;
    @ManyToOne
    @JoinColumn(name = "supplier_id", nullable = false)
    private Supplier supplier;
    @Column(nullable = false, length = 100)
    private String name;
    @Column(length = 255)
    private String description;
    @Column(nullable = false)
    private Double price;
    @Column(name = "duration_minutes", nullable = false)
    private Integer durationMinutes;
    @Column(name = "created_at", nullable = false, updatable = false)
    private LocalDateTime createdAt;
    @PrePersist
    public void prePersist() {
        this.createdAt = LocalDateTime.now();
    }
}
```

Spring Data JPA репозиторій фрагмент:

```
package ua.edu.kneu.serviceflow.dao;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;
import org.springframework.stereotype.Repository;
import ua.edu.kneu.serviceflow.entity.Services;
import ua.edu.kneu.serviceflow.entity.Supplier;
```

```

import java.util.List;
import java.util.UUID;
@Repository
public interface ServicesRepository extends JpaRepository<Services, UUID> {
    List<Services> findBySupplier(Supplier supplier);
    @Query(value = ""
SELECT s.id, s.name, s.description, COUNT(cb.id) AS booking_count
FROM services s
LEFT JOIN client_bookings cb ON s.id = cb.service_id
GROUP BY s.id, s.name, s.description
ORDER BY booking_count DESC
LIMIT :limit
"", nativeQuery = true)
    List<Object[]> findTopPopularServicesByBookings(@Param("limit") int limit);
}

```

Фрагмент класу-обгортки для API-відповідей або запитів, пов'язаних із послугами:

```

package ua.edu.kneu.serviceflow.dto;
import lombok.*;
import java.time.LocalDateTime;
import java.util.UUID;
@Getter
@Setter
@NoArgsConstructor
@AllArgsConstructor
@Builder
public class ServicesDto {
    private UUID id;
    private UUID supplierId;
    private String name;
    private String description;
    private Double price;
    private Integer durationMinutes;
    private LocalDateTime createdAt;
}

```

Фрагмент сервісних класів у Spring Framework:

```

package ua.edu.kneu.serviceflow.service;
import lombok.RequiredArgsConstructor;
import org.springframework.stereotype.Service;
import ua.edu.kneu.serviceflow.dao.ClientBookingRepository;
import ua.edu.kneu.serviceflow.dao.ClientRepository;
import ua.edu.kneu.serviceflow.dao.ServicesRepository;
import ua.edu.kneu.serviceflow.dao.TimeSlotRepository;
import ua.edu.kneu.serviceflow.dto.ClientBookingDto;
import ua.edu.kneu.serviceflow.entity.Client;
import ua.edu.kneu.serviceflow.entity.ClientBooking;
import ua.edu.kneu.serviceflow.entity.Services;
import ua.edu.kneu.serviceflow.entity.TimeSlot;

```

```

import java.time.LocalDate;
import java.util.List;
import java.util.UUID;
@Service
@RequiredArgsConstructor
public class BookingService {
    private final ClientRepository clientRepository;
    private final ServicesRepository servicesRepository;
    private final TimeSlotRepository timeSlotRepository;
    private final ClientBookingRepository clientBookingRepository;
    public void createBooking(String username, UUID serviceId, LocalDate date, UUID timeSlotId) {
        Client client = clientRepository.findByUsername(username)
            .orElseThrow(() -> new RuntimeException("Клієнта не знайдено"));
        Services service = servicesRepository.findById(serviceId)
            .orElseThrow(() -> new RuntimeException("Послугу не знайдено"));
        TimeSlot timeSlot = timeSlotRepository.findById(timeSlotId)
            .orElseThrow(() -> new RuntimeException("Таймслот не знайдено"));
        boolean isTaken = clientBookingRepository.existsByServiceAndDateAndSlot(serviceId, date,
timeSlotId);
        if (isTaken) {
            throw new IllegalStateException("Цей слот уже зайнятий.");
        }
        ClientBooking booking = new ClientBooking();
        booking.setClient(client);
        booking.setService(service);
        booking.setTimeSlot(timeSlot);
        booking.setBookingDate(date);
        clientBookingRepository.save(booking);
    }
    public List<ClientBooking> getBookingsByClient(UUID clientId) {
        return clientBookingRepository.findByClientId(clientId);
    }
    public List<ClientBooking> getBookingsByDate(UUID clientId, LocalDate date) {
        return clientBookingRepository.findByClientIdAndBookingDate(clientId, date);
    }
    public List<ClientBooking> getBookingsBetweenDates(UUID clientId, LocalDate startDate,
LocalDate endDate) {
        return clientBookingRepository.findByClientIdAndBookingDateBetween(clientId, startDate,
endDate);
    }
    public void deleteBooking(UUID bookingId) {
        clientBookingRepository.deleteById(bookingId);
    }
    public List<ClientBookingDto> getBookingsForClientByDate(UUID clientId, LocalDate date) {
        List<ClientBooking> bookings = getBookingsByDate(clientId, date);
        return bookings.stream()
            .map(booking -> {
                ClientBookingDto dto = new ClientBookingDto();
                dto.setId(booking.getId());
                dto.setServiceName(booking.getService().getName());
                dto.setSupplierName(booking.getService().getSupplier().getName());
            });
    }
}

```

```

        dto.setStartTime(booking.getTimeSlot().getStartTime());
        dto.setEndTime(booking.getTimeSlot().getEndTime());
        return dto;
    })
    .toList();
}
}

```

Фрагмент Spring MVC-контролер для CRUD-операцій:

```

package ua.edu.kneu.serviceflow.controller;
import lombok.RequiredArgsConstructor;
import org.springframework.security.access.prepost.PreAuthorize;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.*;
import ua.edu.kneu.serviceflow.dto.SupplierDto;
import ua.edu.kneu.serviceflow.entity.Services;
import ua.edu.kneu.serviceflow.service.ServicesService;
import ua.edu.kneu.serviceflow.service.SupplierServiceProfile;
import ua.edu.kneu.serviceflow.service.SupplierService;
import java.security.Principal;
import java.util.List;
import java.util.UUID;
@Controller
@RequestMapping("/service")
@PreAuthorize("hasRole('SUPPLIER')")
@RequiredArgsConstructor
public class ServicesController {
    private final ServicesService servicesService;
    private final SupplierServiceProfile supplierServiceProfile; SupplierServiceProfile
    private final SupplierService supplierService;
    @GetMapping("/supplier/services")
    public String getServicesAndForm(Model model, Principal principal) {
        UUID supplierId = supplierServiceProfile.getSupplierIdByUsername(principal.getName());
        SupplierDto supplierDto = supplierServiceProfile.getSupplierById(supplierId);
        model.addAttribute("supplier", supplierDto);
        List<Services> services = servicesService.getServicesForCurrentSupplier();
        model.addAttribute("services", services);
        model.addAttribute("service", new Services());
        return "service_list";
    }
    @GetMapping("/supplier/services/create")
    public String showCreateForm(Model model, Principal principal) {
        UUID supplierId = supplierServiceProfile.getSupplierIdByUsername(principal.getName());
        SupplierDto supplierDto = supplierServiceProfile.getSupplierById(supplierId);
        model.addAttribute("supplier", supplierDto);
        model.addAttribute("service", new Services());
        return "service_list";
    }
    @PostMapping("/supplier/services/create")
    public String createService(@ModelAttribute("service") Services service) {

```

```

        servicesService.createService(service);
        return "redirect:/service/supplier/services";
    }
    @GetMapping("/supplier/services/edit/{id}")
    public String showEditForm(@PathVariable UUID id, Model model, Principal principal) {
        UUID supplierId = supplierServiceProfile.getSupplierIdByUsername(principal.getName());
        SupplierDto supplierDto = supplierServiceProfile.getSupplierById(supplierId);
        model.addAttribute("supplier", supplierDto); // Додайте supplier до моделі
        List<Services> services = servicesService.getServicesForCurrentSupplier();
        Services service = servicesService.getServiceById(id);
        model.addAttribute("services", services);
        model.addAttribute("service", service);
        return "service_list";
    }
    @PostMapping("/supplier/services/edit/{id}")
    public String updateService(@PathVariable UUID id, @ModelAttribute("service") Services
updatedService) {
        servicesService.updateService(id, updatedService);
        return "redirect:/service/supplier/services";
    }
    @GetMapping("/supplier/services/delete/{id}")
    public String deleteService(@PathVariable UUID id) {
        servicesService.deleteService(id);
        return "redirect:/service/supplier/services";
    }
}

```

Фрагмент Spring Security Configuration:

```

package ua.edu.kneu.serviceflow.config;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.authentication.AuthenticationManager;
import
org.springframework.security.config.annotation.authentication.configuration.AuthenticationConfigurat
ion;
import org.springframework.security.config.annotation.method.configuration.EnableMethodSecurity;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.security.web.SecurityFilterChain;
import lombok.RequiredArgsConstructor;
import ua.edu.kneu.serviceflow.controller.CustomAuthenticationSuccessHandler;
@Configuration
@EnableWebSecurity
@RequiredArgsConstructor
@EnableMethodSecurity(prePostEnabled = true)
public class SecurityConfig {
    private final CustomAuthenticationSuccessHandler customAuthenticationSuccessHandler;
    @Bean
    public PasswordEncoder passwordEncoder() {

```

```

        return new BCryptPasswordEncoder();
    }
    @Bean
    public AuthenticationManager authenticationManager(AuthenticationConfiguration
authenticationConfiguration) throws Exception {
        return authenticationConfiguration.getAuthenticationManager();
    }
    @Bean
    public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {
        return http
            .csrf(csrf -> csrf.disable())
            .authorizeHttpRequests(auth -> auth
                .requestMatchers(
                    "/login", "/role-selection", "/select-role",
                    "/api/auth/register/**", "/css/**", "/js/**", "/images/**", "/logo/**", "/error"
                ).permitAll()
                .anyRequest().authenticated()
            )
            .formLogin(form -> form
                .loginPage("/login")
                .loginProcessingUrl("/api/auth/login")
                .successHandler(customAuthenticationSuccessHandler)
                .failureUrl("/login?error=true")
                .permitAll()
            )
            .logout(logout -> logout
                .logoutUrl("/logout")
                .logoutSuccessUrl("/login?logout=true")
                .permitAll()
            )
            .build();
    }
}

```



Звіт подібності

метадані

Назва організації

Kyiv National Economic University named after Vadym Hetman KNEU

Заголовок

Розробка веб-додатку для автоматизації взаємодії між постачальниками послуг та клієнтами

Автор

Науковий керівник / Експерт

Ляхович Ігор Юрійович Васильєва Л. В.

підрозділ

кафедра інформаційних систем в економіці

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.



КП 1

25

Довжина фрази для коефіцієнта подібності 2



КП 2

12384

Кількість слів



КЦ

98267

Кількість символів

Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв		4
Інтервали		0
Мікропробіли		13
Білі знаки		0
Парафрази (SmartMarks)		41

Подібності за списком джерел

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Копію тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз

Копію тексту

ПОРЯДКОВИЙ НОМЕР	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	https://ir.kneu.edu.ua/server/api/core/bitstreams/1b35cb3a-b293-4677-9674-83983838c914/content	49 0.40 %
2	https://ir.kneu.edu.ua/bitstreams/066dbca0-6afa-4cf4-a181-c3e88a2aab6f/download	47 0.38 %
3	https://kneu.edu.ua/userfiles/Faculty_of_Information_Systems_and_Technology/kaf%20ise/Metod_rekom_en_6_050101.doc	26 0.21 %
4	https://ir.kneu.edu.ua/server/api/core/bitstreams/add9ce43-9c83-4412-bccd-41d87f874e83/content	21 0.17 %