

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**  
**ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА**  
**ФРАНКА**  
**ФАКУЛЬТЕТ УПРАВЛІННЯ ФІНАНСАМИ ТА БІЗНЕСУ**

**Кафедра Цифрової Економіки та Бізнес-Аналітики**

**КУРСОВА РОБОТА**

з дисциплін професійної та практичної підготовки  
на тему:

**Інформаційна система для салону краси**

**спеціальність:** 051 “Економіка”  
(код та найменування спеціальності)  
**освітня програма:** “Інформаційні технології в бізнесі”  
(найменування спеціалізації)  
**освітня ступінь:** Бакалавр  
(бакалавр/магістр)

**Науковий керівник:**

к.ф.-м.н., доцент Депутат Б.Я.  
(науковий ступінь, посада, прізвище, ініціали)  
“ ” 2025 р.  
(підпис)

**Виконавець:**

студент(ка) групи УФЕ-31с  
Марич Ю.В.  
(прізвище, ініціали)  
“ ” 2025 р.  
(підпис)

**Загальна кількість балів** \_\_\_\_\_  
(підпис, ППІ членів комісії)  
\_\_\_\_\_  
(підпис, ППІ членів комісії)  
\_\_\_\_\_  
(підпис, ППІ членів комісії)

**ЛЬВІВ 2025**

## **ЗМІСТ**

<b>ВСТУП.....</b>	<b>3</b>
<b>РОЗДІЛ 1. АНАЛІЗ ВИМОГ ІНФОРМАЦІЙНОЇ СИСТЕМИ ДЛЯ САЛОНУ КРАСИ.....</b>	<b>5</b>
<b>1.1. Постановка завдання .....</b>	<b>5</b>
<b>1.2 Аналіз предметної області .....</b>	<b>7</b>
<b>1.3 Аналіз засобів реалізації (техніко-економічне обґрунтування вибору)     .....</b>	<b>10</b>
<b>РОЗДІЛ 2. РОЗРОБКА БАЗИ ДАНИХ.....</b>	<b>13</b>
<b>2.1. Опис моделі даних.....</b>	<b>13</b>
<b>2.2. Нормалізація відношень.....</b>	<b>15</b>
<b>2.3. Визначення типів даних .....</b>	<b>17</b>
<b>2.4. Обмеження цілісності даних .....</b>	<b>20</b>
<b>2.5. Реалізація SQL-скрипту.....</b>	<b>26</b>
<b>РОЗДІЛ 3. РОЗРОБКА ВЕБ-САЙТУ .....</b>	<b>32</b>
<b>3.1. Структура веб-сайту.....</b>	<b>32</b>
<b>3.3. Програмування серверної частини .....</b>	<b>37</b>
<b>3.4. Програмування клієнтської частини.....</b>	<b>37</b>
<b>3.5 Розміщення веб-сайту на локальному віртуальному середовищі .....</b>	<b>39</b>
<b>ВИСНОВКИ .....</b>	<b>41</b>
<b>СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ .....</b>	<b>43</b>
<b>ДОДАТКИ.....</b>	<b>44</b>

## ВСТУП

У сучасному суспільстві сфера обслуговування відіграє одну з провідних ролей у задоволенні щоденних потреб населення. Одним із важливих її сегментів є індустрія краси, зокрема діяльність салонів краси, які надають послуги з догляду за зовнішністю: перукарські, косметологічні, манікюрні, масажні та інші. З розвитком інформаційних технологій усе більше бізнесів прагнуть автоматизувати свої внутрішні процеси, щоб підвищити ефективність управління, покращити обслуговування клієнтів і забезпечити конкурентоспроможність на ринку.

Більшість салонів краси, особливо у великих містах, уже не можуть обійтись без сучасної інформаційної системи, яка дозволяє клієнтам зручно здійснювати онлайн-запис, а адміністраторам — швидко керувати розкладом працівників, обліковувати надані послуги та аналізувати фінансову звітність. Створення спеціалізованої інформаційної системи дає змогу вирішити низку проблем: уникнути плутанини в розкладі, зменшити кількість пропущених візитів, підвищити рівень персоналізованого обслуговування та сформувати клієнтську базу з історією замовлень.

Такі інструменти як веб-застосунки або CRM-системи стають дедалі більш доступними, і навіть малі бізнеси можуть дозволити собі впровадження ІТ-рішень. Тому актуальність теми курсової роботи визначається необхідністю цифровізації процесів у сфері надання послуг та зростаючим попитом на ІТ-продукти для малого й середнього бізнесу.

**Метою дослідження** є проаналізувати специфіку функціонування салону краси як суб'єкта господарювання та розробити інформаційну систему, яка дозволяє автоматизувати його ключові бізнес-процеси, а саме: управління клієнтською базою, працівниками, послугами, розкладом та записом, а також збір аналітичних даних.

**Завдання дослідження:**

- проаналізувати предметну область, її бізнес-процеси та типові функції;

- сформулювати технічне завдання на створення інформаційної системи;
- розробити логічну структуру бази даних з урахуванням цілісності даних;
- виконати нормалізацію таблиць та визначити типи даних;
- реалізувати SQL-структуру, що відображає реальні процеси салону;

**Об'єктом дослідження** є діяльність салону краси як підприємства, що надає послуги у сфері краси та догляду за зовнішністю. **Предметом** є автоматизація бізнес-процесів салону краси за допомогою засобів створення інформаційних систем, зокрема — проектування та реалізація бази даних.

**Практичне значення** отриманих результатів: Результати дослідження можуть бути використані для впровадження інформаційної системи у реальний салон краси. Розроблена структура бази даних є гнучкою, масштабованою і здатна обробляти велику кількість інформації про клієнтів, працівників, послуги, записи та оплату. Її впровадження забезпечить швидку взаємодію з клієнтами, контроль ефективності персоналу, гнучке керування ресурсами та прийняття управлінських рішень на основі зібраної статистики.

**Використане програмне забезпечення:** Для реалізації проєкту застосовувались середовище моделювання баз даних MySQL Workbench, локальний сервер XAMPP, а також мови HTML, CSS, JavaScript

**Структура роботи.** Курсова робота складається з трьох основних розділів. У першому розділі здійснено аналіз предметної області, сформульовано технічне завдання, обґрунтовано вибір інструментів та описано загальні вимоги до інформаційної системи для салону краси. У другому розділі виконано проєктування бази даних: визначено основні сутності, побудовано логічну модель, проведено нормалізацію, встановлено типи даних та реалізовано SQL-структуру таблиць. У третьому розділі подано реалізацію клієнтської частини інформаційної системи у вигляді веб-сайту: описано його структуру, створено макети сторінок, наведено приклади HTML- і CSS-коду, а також розглянуто можливості майбутньої інтеграції з серверною частиною. Робота завершується висновками, списком використаних джерел та додатками.

## РОЗДІЛ 1. АНАЛІЗ ВИМОГ ДО ІНФОРМАЦІЙНОЇ СИСТЕМИ ДЛЯ САЛОНУ КРАСИ

### 1.1. Постановка завдання

У сучасних умовах цифрової трансформації все більше підприємств прагнуть оптимізувати свою діяльність шляхом впровадження інформаційних систем. Однією з таких галузей є індустрія краси, зокрема салони краси, які надають широкий спектр послуг. Через велику кількість клієнтів, високий рівень конкуренції та потребу у високій якості обслуговування виникає необхідність у впровадженні автоматизованих рішень, які дозволяють покращити ефективність роботи, мінімізувати людський фактор та підвищити рівень сервісу.

Метою цієї курсової роботи є розробка інформаційної системи для салону краси, яка буде орієнтована на автоматизацію ключових бізнес-процесів: запис клієнтів, управління розкладом працівників, облік послуг і платежів, взаємодію з клієнтами через відгуки та перегляд статистики адміністратором. Розроблена система матиме як клієнтську, так і адміністративну частину.

Завдання, що вирішуються в межах даної роботи:

- Аналіз предметної області та аналогічних інформаційних систем;
- Постановка задачі та формування функціональних вимог;
- Проектування структури бази даних та взаємозв'язків між об'єктами;
- Створення веб-інтерфейсу для клієнта та адміністратора;
- Реалізація механізмів запису на послуги, обробки платежів і залишення відгуків;

Інформаційна система повинна забезпечити доступ для кількох категорій користувачів:

- 1) Клієнт — має можливість переглядати перелік послуг, записуватися онлайн, залишати відгуки, переглядати історію візитів;
- 2) Працівник (майстер) — бачить власний розклад, виконує послуги, отримує оцінки від клієнтів;

3) Адміністратор — управляє працівниками, графіками, послугами, переглядає записи, платежі та відгуки;

Очікуваним результатом впровадження такої системи є зменшення витрат часу на адміністративні операції, підвищення зручності для клієнтів, зростання рівня довіри до салону та підвищення конкурентоспроможності бізнесу на ринку послуг.

Створена база даних повинна дозволити реалізувати наступні технічні завдання (ТЗ):

- Авторизацію користувачів з поділом за ролями (адміністратор, працівник, клієнт);
- Перегляд переліку послуг з вказанням назви, ціни, тривалості та відповідального працівника;
- Пошук послуг за назвою або типом;
- Перегляд популярних послуг за певний період часу (на основі кількості записів);
- Перегляд клієнтів та їх історії записів, включно із загальною сумою витрачених коштів;
- Обрахунок середньої оцінки працівників за відгуками;
- Аналіз завантаженості працівників (кількість прийнятих записів за період);
- Формування списку клієнтів із певною кількістю записів або сумою платежів;
- Перегляд відгуків на послуги та працівників;
- Можливість прийому нового працівника до системи або видалення працівника;
- Виведення всіх записів за період з фільтрами за статусом, працівником чи клієнтом;
- Аналіз статистики платежів за способами оплати та по датах;
- Автоматичне оновлення рейтингу послуг або підняття цін для найпопулярніших;

- Аналіз демографії клієнтів за віком, статтю або місяцем народження;
- Формування зведеної статистики по категоріях послуг та завантаженості системи.

## 1.2 Аналіз предметної області

Салон краси — це комерційний заклад, який спеціалізується на наданні послуг з догляду за зовнішністю. Основні напрямки діяльності включають перукарські послуги, косметологію, масаж, манікюр, педикюр, нейл-арт, епіляцію та інші.

У традиційній моделі роботи клієнт телефонує або приходить особисто, щоб записатися. Адміністратор веде розклад вручну або в простій таблиці. Це створює ризик дублювання записів, плутанини в графіках, втрати контактів клієнтів або даних про попередні візити. Через це все частіше салони переходять до цифрових рішень.

На основі вивчення предметної області було визначено такі основні сутності, які повинні бути реалізовані в системі:

1. **Клієнт** — особа, яка має персональний профіль, може записуватися на послуги, залишати відгуки та переглядати історію візитів.
2. **Працівник** (майстер) — виконавець послуг, прив'язаний до певного графіку, має спеціалізацію та отримує оцінки від клієнтів.
3. **Послуга** — одиниця послуг салону з вказаними характеристиками: назва, опис, ціна, тривалість.
4. **Запис** (appointment) — зв'язок між клієнтом, послугою, працівником та обраним часом. Має статус (підтверджено, очікує, скасовано).
5. **Платіж** (payment) — сума, метод оплати, дата та відповідність до запису.
6. **Відгук** (review) — оцінка і коментар клієнта, прив'язаний до послуги та працівника.
7. **Адміністратор** — керує всіма аспектами функціонування системи: перегляд записів, управління працівниками, оновлення послуг.

Додатково враховуються такі особливості:

- Необхідність захисту персональних даних клієнтів (GDPR);
- Швидкий доступ до пошуку записів, історії, перегляду рейтингу працівників;
- Наявність простого інтерфейсу з адаптивною версткою для клієнтів.

Під час аналізу аналогічних систем у Європі та Україні виявлено, що найбільш функціональними є ті сервіси, які дозволяють швидкий онлайн-запис, формування автоматичного нагадування, гнучке управління графіками і повну інтеграцію з клієнтською частиною (відгуки, історія, бонуси).

У межах реалізації даної курсової роботи було створено веб-сайт із сторінками `index.html`, `pro-nas.html`, `posluhy.html`, `kontakty.html`, а також стилями `style.css` та мапою сайту `salon_site_map.xml`. Ці компоненти відображають реальні сторінки системи, з якими взаємодіє користувач. Також була розроблена структура бази даних на основі попереднього аналізу, з таблицями `Client`, `Employee`, `Service`, `Appointment`, `Payments`, `Reviews`, `Admin_users`, що охоплюють усі необхідні функції системи.

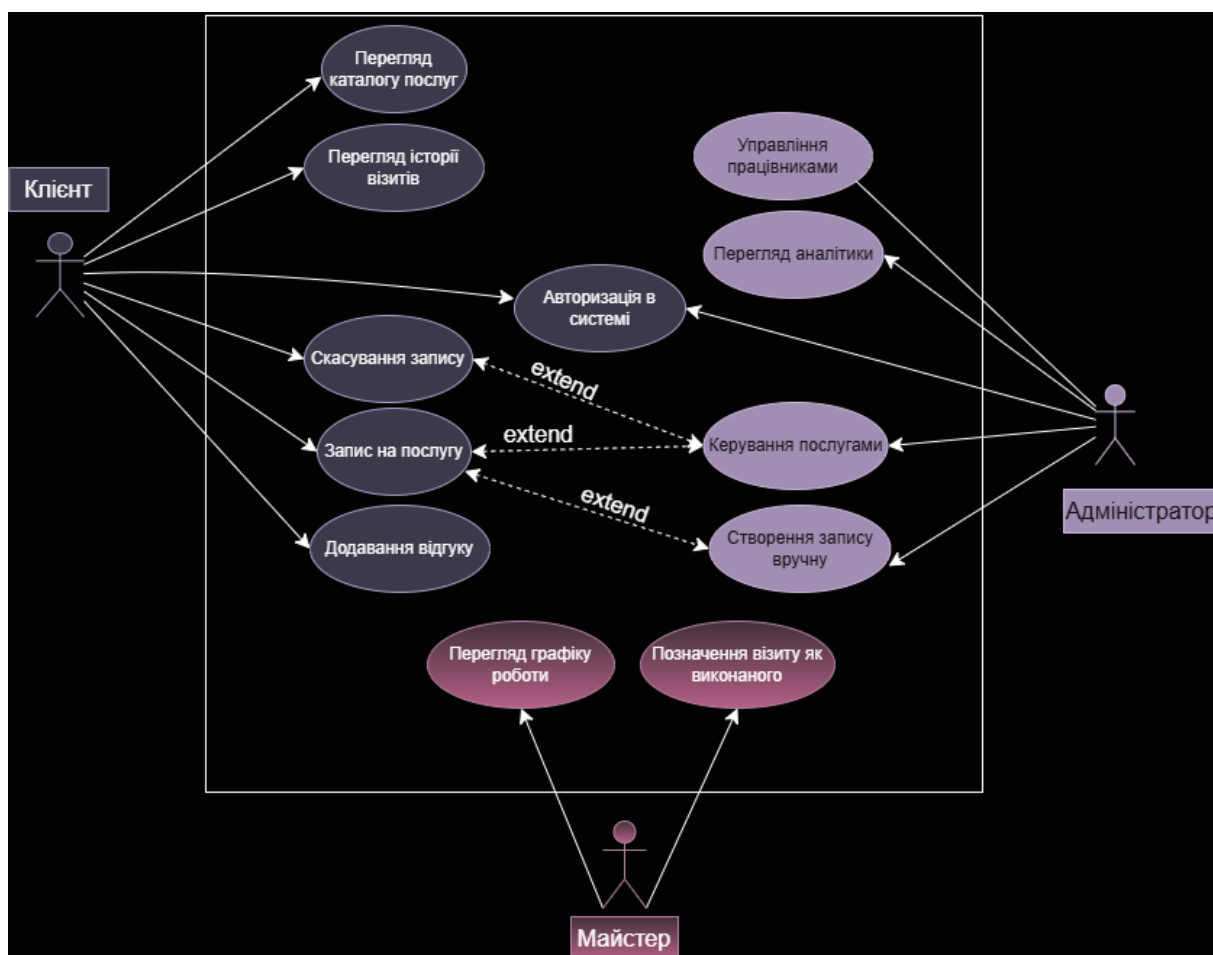
Завдяки цьому етапу сформовано чітке уявлення про функціональні модулі системи, а саме: модуль управління записами, модуль персоналу, модуль послуг, модуль статистики та модуль управління відгуками.

Для подальшого структурування можливостей і загроз, а також глибшого аналізу взаємодії користувачів із системою, доцільно представити діаграму варіантів використання (Use Case) та провести SWOT-аналіз, що дозволяє комплексно оцінити потенціал створюваної системи.

Діаграма є важливим інструментом для візуалізації сценаріїв взаємодії користувачів з інформаційною системою. Вона дає змогу представити, які саме функції доступні кожній ролі (клієнту, працівнику, адміністратору) та як ці ролі взаємодіють із ключовими модулями системи. Такий підхід дозволяє краще структурувати функціональні вимоги ще на етапі проектування та уникнути двозначностей у реалізації логіки системи.



У межах даного проекту діаграма допомагає зрозуміти, як користувач може здійснювати онлайн-запис, залишати відгуки, переглядати історію візитів, а також як адміністратор керує послугами, працівниками і переглядає статистику. Це особливо корисно при подальшій реалізації програмної частини, оскільки дозволяє чітко розмежувати обов'язки між компонентами системи та уникнути дублювання функцій.



**Рисунок 1.1 Use Case діаграма взаємодії користувачів із системою**

З метою подальшого аналізу доцільно також додати SWOT-аналіз, який дозволяє виявити сильні й слабкі сторони, а також потенційні можливості загрози при впровадженні інформаційної системи в реальних умовах.



**Рисунок 1.2. SWOT-аналіз**

Проведений SWOT-аналіз ще раз підтвердив наше технічне завдання. Як і зазначено вище, ІС дозволить автоматизувати більшість процесів салону краси, зробивши його більш конкурентоспроможним.

### **1.3 Аналіз засобів реалізації (техніко-економічне обґрунтування вибору)**

На основі проведеного аналізу можна виокремити ключові бізнес-процеси, які є характерними для роботи сучасного салону краси. Їх опис дозволяє точніше визначити функціональні потреби до розроблюваної інформаційної системи та спроектувати її архітектуру з урахуванням реальних вимог користувачів.

1. *Запис клієнта на послугу.* Цей процес починається з вибору клієнтом бажаної послуги, перегляду доступного графіка працівників і завершенням бронювання на вільний час. У системі зберігається інформація про клієнта, обрану послугу, дату, час і відповідального працівника. Важливою особливістю є можливість підтвердження або скасування запису.

2. *Надання послуги.* Після запису, клієнт у визначений час відвідує салон, де працівник виконує обрану послугу. У системі це фіксується як виконана процедура, що надалі може впливати на рейтинг працівника та накопичувану історію візитів клієнта.
3. *Оплата послуги.* Після отримання послуги клієнт проводить оплату. У системі зберігається інформація про суму, метод оплати (готівка, картка тощо) та дата. Це дозволяє вести фінансову статистику та формувати звіти.
4. *Збір відгуків.* Клієнт може залишити коментар або оцінку після отримання послуги. Це є частиною механізму зворотного зв'язку, який дозволяє контролювати якість роботи працівників та загальний імідж салону.
5. *Адміністрування системи.* Адміністратор має можливість переглядати всі записи, змінювати графіки працівників, додавати або видаляти послуги, формувати звіти щодо доходів, завантаженості персоналу та популярності послуг. Крім того, він може редагувати інформацію про клієнтів і блокувати дублікати.

У рамках реалізації інформаційної системи для салону краси було прийнято рішення використовувати стек технологій, який є водночас ефективним, доступним і широко застосовуваним у розробці веб-рішень.

Мова програмування та середовище:

Для реалізації бекенд-частини може бути використана мова PHP або Python, проте в контексті курсової роботи основна логіка реалізується через SQL-структури та HTML/CSS-розмітку. Базовий функціонал створено за допомогою HTML5 та CSS3, що дозволяє забезпечити адаптивність, кросбраузерність та естетичний інтерфейс.

Система управління базами даних:

MySQL обрано як основну СУБД через її популярність, простоту використання, безкоштовну ліцензію, високу сумісність із веб-інструментами та підтримку SQL-запитів будь-якого рівня складності. Вона дозволяє ефективно реалізувати таблиці, зв'язки між сутностями та механізми обробки даних.

Інструменти розробки:

- MySQL Workbench — для створення та адміністрування бази даних;
- Draw.io — для створення діаграм і моделювання структури системи;
- Visual Studio Code або Sublime Text — як редактори коду;
- Браузер Google Chrome — для перегляду результатів.

Переваги вибраного підходу:

- 1) Використання безкоштовних інструментів і відкритих стандартів;
- 2) Низький поріг входу для користувачів та розробників;
- 3) Простота підтримки та масштабування системи;
- 4) Можливість інтеграції з іншими веб-технологіями в майбутньому.

Таким чином, обрані засоби реалізації повністю відповідають поставленим цілям та забезпечують створення працездатної і зручної в користуванні інформаційної системи.

## РОЗДІЛ 2. РОЗРОБКА БАЗИ ДАНИХ

### 2.1. Опис моделі даних

Модель даних є ключовим елементом інформаційної системи, що забезпечує логічну структуровану організацію збереження інформації про всі об'єкти та взаємозв'язки в межах предметної області. Для реалізації інформаційної системи салону краси було створено реляційну модель, яка охоплює основні сутності, характерні для функціонування такого типу закладу.

До моделі включено наступні таблиці:

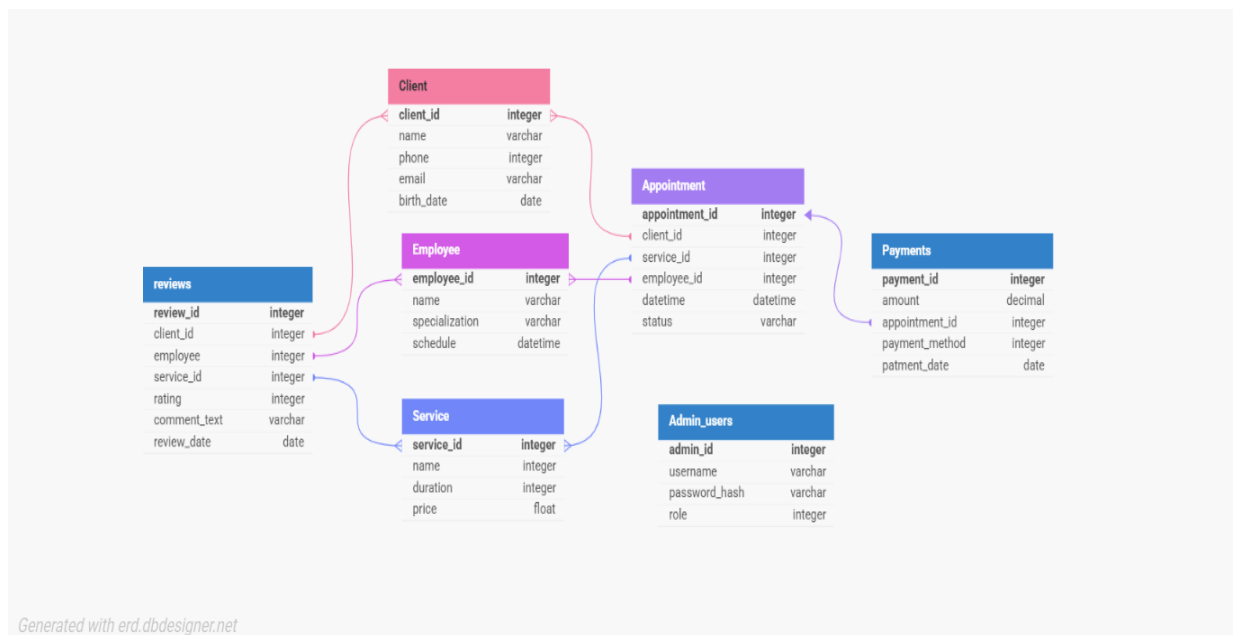
- 1) Client (Клієнт) — зберігає інформацію про зареєстрованих користувачів: ім'я, номер телефону, електронна пошта, дата народження. Це дозволяє ідентифікувати кожного клієнта та будувати персоніфіковану історію взаємодії з салоном.
- 2) Employee (Працівник) — містить дані про фахівців салону, включаючи ім'я, спеціалізацію та розклад роботи. Це дозволяє планувати надання послуг відповідно до доступності майстрів.
- 3) Service (Послуга) — зберігає інформацію про перелік доступних послуг, їхню тривалість та вартість. Це основа для формування інтерфейсу вибору послуг та розрахунку оплати.
- 4) Appointment (Запис) — фіксує факт бронювання клієнтом певної послуги на конкретну дату й час із зазначенням майстра. Поля містять зовнішні ключі на таблиці Client, Employee та Service. Статус запису (очікує, підтверджено, скасовано) дозволяє керувати процесом обслуговування.
- 5) Payments (Платежі) — містить дані про суму оплати, метод (готівка, картка тощо) та дату платежу. Зв'язок з таблицею Appointment дає змогу формувати фінансову аналітику.
- 6) Reviews (Відгуки) — забезпечує можливість клієнтам залишати оцінку та коментарі після отриманої послуги. Тут фіксуються ID клієнта, працівника, послуги, рейтинг і текстовий відгук.

- 7) Admin\_users (Адміністратори) — таблиця для авторизації користувачів з адміністративним доступом. Включає логін, хешований пароль та роль (адміністратор, менеджер тощо).

Усі таблиці пов'язані між собою за допомогою зовнішніх ключів, що дозволяє забезпечити цілісність та узгодженість даних. Наприклад, кожен запис у таблиці Payments відповідає конкретному запису в таблиці Appointment, а кожен Review пов'язаний із конкретною послугою, клієнтом та працівником.

Загальна модель даних побудована таким чином, щоб забезпечити масштабованість системи, легкість у виконанні SQL-запитів та підтримку широкого спектру звітності. Вона дозволяє ефективно реалізувати технічне завдання, описане у розділі 1.1, та є основою для створення повноцінного інтерфейсу користувача, а також подальшої інтеграції з іншими компонентами системи.

Для наочності взаємозв'язків між таблицями інформаційної системи розроблено ER-діаграму (Entity-Relationship diagram), яка демонструє структуру бази даних, типи зв'язків між сутностями та їх атрибути.



**Рисунок 2.1 – ER-діаграма моделі даних інформаційної системи салону краси**

Дана діаграма дозволяє швидко оцінити логіку побудови бази, переконатися у правильності зв'язків та забезпечує зручність подальшої реалізації запитів і звітів. Зокрема, вона наочно демонструє зв'язки між записами клієнтів, послугами, працівниками, відгуками, платежами та адміністративною частиною системи. Це значно спрощує процес підтримки та розвитку системи в майбутньому.

## **2.2. Нормалізація відношень**

Нормалізація є фундаментальним етапом проєктування реляційних баз даних, який дозволяє структурувати дані таким чином, щоб уникати надлишковості, зменшити дублювання інформації, забезпечити логічну узгодженість та полегшити подальше обслуговування й масштабування системи. В процесі проєктування інформаційної системи салону краси була проведена поетапна нормалізація всієї структури бази даних згідно з класичним підходом до нормалізації реляційних відношень.

Основна мета нормалізації — забезпечити таку організацію таблиць, при якій кожен факт або запис зберігається лише один раз, а всі повторювані або похідні дані — винесені у пов'язані таблиці. Це дозволяє уникнути так званих аномалій оновлення, вставки та видалення, що виникають у ненормалізованих структурах.

Процес нормалізації проводився поетапно, включаючи перші три основні нормальні форми (1НФ, 2НФ, 3НФ).

Перша нормальна форма (1НФ) передбачає, що всі атрибути мають бути атомарними, тобто містити лише одне значення в кожному полі. У системі салону краси всі таблиці було створено таким чином, щоб жодне поле не містило списків значень або повторюваних груп. Наприклад, у таблиці Client поля name, phone, email, birth\_date чітко розділені та зберігають однозначну інформацію.

Друга нормальна форма (2НФ) вимагає, щоб усі неключові атрибути таблиці залежали від повного первинного ключа, а не від його частини. Це

особливо важливо для таблиць, які мають складений ключ. У випадку нашої бази даних усі таблиці мають прості первинні ключі, але, наприклад, у таблиці Appointment, яка містить посилання на client\_id, employee\_id та service\_id, атрибути datetime та status залежать від конкретного запису як цілісної сутності, а не від окремої частини.

Третя нормальна форма (3НФ) усуває транзитивні залежності, коли неключовий атрибут залежить не напряму від первинного ключа, а через інший неключовий атрибут. Наприклад, у таблиці Service зберігається лише інформація про назву послуги, тривалість та ціну. Вона винесена окремо й не дублюється в таблиці Appointment. Завдяки цьому будь-яка зміна вартості послуги не потребує змін у кількох місцях — достатньо оновити її в єдиному джерелі.

Переваги застосування нормалізації у цій системі:

- Економія місця зберігання. Завдяки розділенню таблиць зменшується дублювання однієї й тієї ж інформації в різних записах;
- Логічна ясність структури. Кожна таблиця чітко відображає лише один об'єкт предметної області;
- Масштабованість. У майбутньому легко можна додати нові поля, зв'язки або модулі, не порушуючи існуючу структуру;
- Узгодженість і актуальність даних. Зміни в одному місці автоматично відображаються у всіх залежних запитах;
- Підтримка звітності. Нормалізовані таблиці полегшують створення агрегованих SQL-запитів, аналітики та візуалізацій.

Прикладом ефективної нормалізації є також таблиця Reviews, яка містить лише ключі до клієнта, працівника і послуги, та відповідні поля з оцінкою й текстом коментаря. Завдяки цьому один клієнт може залишити кілька відгуків до різних послуг або працівників, а відгуки можуть бути швидко знайдені за будь-яким критерієм — без дублювання зайвої інформації.

Ще одним прикладом є таблиця Payments, де кожна транзакція чітко прив'язана до конкретного запису Appointment, що дозволяє аналізувати



фінансову статистику, визначати популярні методи оплати та формувати звіти за період.

У результаті застосованої нормалізації структура бази даних відповідає сучасним вимогам до реляційного проєктування і забезпечує міцну основу для розвитку системи. Всі сутності чітко поділені, а логіка взаємодії між ними збережена, що робить базу легкою для підтримки, оновлення й масштабування., нормалізована модель забезпечує логічну та ефективну основу для збереження даних, що дозволяє системі працювати стабільно, точно та надійно навіть при збільшенні обсягу інформації та кількості користувачів.

### 2.3. Визначення типів даних

Правильне визначення типів даних є критично важливим етапом фізичного проєктування бази даних, оскільки воно визначає ефективність зберігання, швидкість обробки інформації, а також точність та надійність функціонування інформаційної системи. Типи даних призначаються відповідно до природи зберезуваних значень у кожному полі таблиць.

У процесі створення структури бази даних для салону краси було використано типи даних, які забезпечують оптимальний баланс між обсягом пам'яті, продуктивністю та гнучкістю запитів.

Нижче наведено загальну характеристику обраних типів для основних таблиць:

#### *Лістинг 2.1*

Таблиця Client:

```
client_id - INT AUTO_INCREMENT PRIMARY KEY (унікальний  
ідентифікатор клієнта)  
name - VARCHAR(100) (повне ім'я клієнта)  
phone - BIGINT (номер телефону у міжнародному форматі)  
email - VARCHAR(100) (електронна пошта)  
birth_date - DATE (дата народження)
```

Таблиця Employee:

employee\_id - INT AUTO\_INCREMENT PRIMARY KEY

name - VARCHAR(100)

specialization - VARCHAR(100) (спеціалізація - перукар, масажист тощо)

schedule - DATETIME (час виходу на зміну або початку роботи)

Таблиця Service:

service\_id - INT AUTO\_INCREMENT PRIMARY KEY

name - VARCHAR(100)

duration - INT (тривалість у хвилинах)

price - DECIMAL(10,2) (вартість послуги)

Таблиця Appointment:

appointment\_id - INT AUTO\_INCREMENT PRIMARY KEY

client\_id - INT (зовнішній ключ на Client)

service\_id - INT (зовнішній ключ на Service)

employee\_id - INT (зовнішній ключ на Employee)

datetime - DATETIME (дата та час запису)

status - ENUM('Pending', 'Confirmed', 'Cancelled')

Таблиця Payments:

payment\_id - INT AUTO\_INCREMENT PRIMARY KEY

amount - DECIMAL(10,2) (сума платежу)

appointment\_id - INT (зовнішній ключ на Appointment)

payment\_method - TINYINT (1 - готівка, 2 - картка, 3 - онлайн тощо)

payment\_date - DATE

Таблиця Reviews:

review\_id - INT AUTO\_INCREMENT PRIMARY KEY

client\_id - INT

employee - INT

service\_id - INT

```
rating - TINYINT (від 1 до 5)
comment_text - TEXT (текстовий відгук)
review_date - DATE
```

Таблиця Admin\_users:

```
user_id - INT AUTO_INCREMENT PRIMARY KEY
username - VARCHAR(50)
password_hash - VARCHAR(255)
role - TINYINT (1 - адміністратор, 2 - менеджер, 3 - персонал)
```

Загальні принципи вибору типів даних:

- Для числових ідентифікаторів використовуються типи INT або BIGINT в залежності від передбачуваного обсягу даних.
- Для текстових полів – VARCHAR із оптимальною довжиною, що враховує середній розмір значення.
- Для фінансових полів – DECIMAL, що дозволяє уникнути похибок при обчисленнях з копійками.
- Для дат – DATE або DATETIME, залежно від необхідності врахування часу доби.
- Для переліків значень – ENUM або TINYINT, що дозволяє зручно кодувати статуси чи типи.
- Завдяки коректному вибору типів даних досягається висока швидкість обробки SQL-запитів, зменшується ризик помилок, полегшується масштабування та підтримка бази даних у довготривалій перспективі.

На основі визначених типів даних була сформована відповідна ER-діаграма (Entity-Relationship diagram), згенерована за допомогою середовища розробки MySQL Workbench. Вона не лише демонструє структуру реляційної моделі, але й дозволяє візуально оцінити типи даних, встановлені для кожного поля в таблицях, включаючи числові, символічні, часові й логічні. Завдяки підсвічуванню типів даних у кожному полі, ця діаграма є не лише інструментом візуалізації логіки проектування, а й засобом перевірки коректності типізації та

побудови структур у фізичній реалізації бази. Такий підхід полегшує супровід, дає змогу швидко виявляти потенційні конфлікти у схемі та сприяє її подальшому масштабуванню.

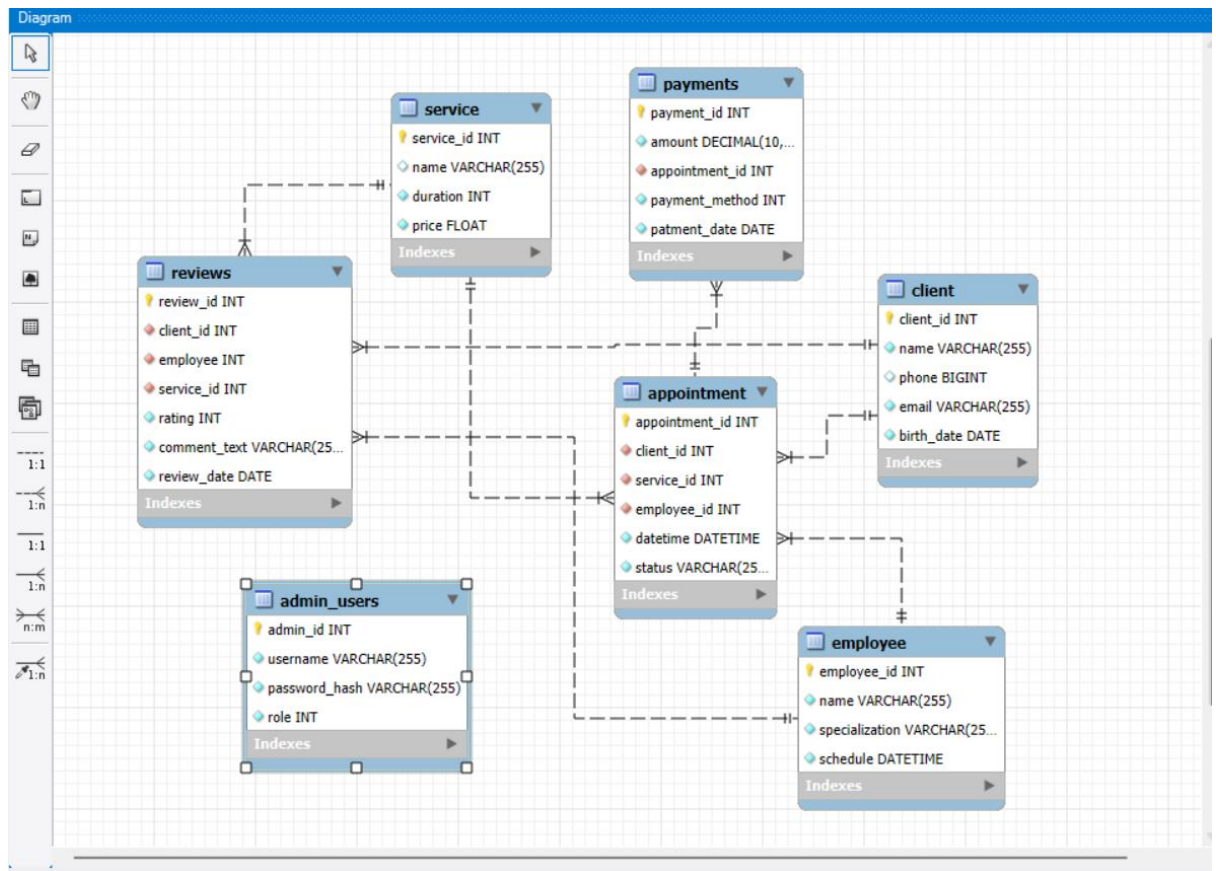


Рисунок 2.2. ER-діаграма

## 2.4. Обмеження цілісності даних

Обмеження цілісності в реляційній базі даних — це сукупність правил, які забезпечують достовірність, логічну узгодженість і захист від помилок при введенні, оновленні або видаленні даних. У процесі створення інформаційної системи для салону краси було реалізовано основні типи обмежень, які забезпечують структурну надійність бази даних.

Основні типи обмежень:

1. PRIMARY KEY — гарантує унікальність кожного запису в таблиці. Наприклад, у кожній таблиці (Client, Employee, Appointment тощо) первинні ключі (\*\_id) мають тип INT AUTO\_INCREMENT, що забезпечує автоматичну генерацію унікального ідентифікатора.

2. NOT NULL — вказує, що поле обов'язкове до заповнення. Це правило застосоване для таких полів, як name, phone, email, birth\_date, schedule, rating тощо. Воно запобігає виникненню неповних або помилкових записів.
3. UNIQUE — гарантує, що значення у полі не буде дублюватися в інших записах. Наприклад, у таблиці Admin\_users обмеження UNIQUE накладено на поля username та email, що виключає можливість створення облікових записів з однаковим логіном.
4. FOREIGN KEY — реалізує зовнішні зв'язки між таблицями, забезпечуючи збереження послідовності та узгодженості між сутностями.

Наприклад:

- Appointment.client\_id → Client.client\_id
- Appointment.employee\_id → Employee.employee\_id
- Payments.appointment\_id → Appointment.appointment\_id
- Reviews.service\_id → Service.service\_id

Ці ключі захищають базу від введення неіснуючих ID у зв'язках.

5. ENUM або CHECK — для логічних обмежень, наприклад, поле status в таблиці Appointment може набувати значень 'Pending', 'Confirmed', 'Cancelled'.
6. DEFAULT — за потреби можна додавати значення за замовчуванням, наприклад, для статусу нового запису чи дати створення.

На рисунках нижче подано фрагменти SQL-коду створення таблиць, у яких реалізовані ключові обмеження цілісності.

```
CREATE TABLE IF NOT EXISTS `Admin_users` (
  `admin_id` int AUTO_INCREMENT NOT NULL UNIQUE,
  `username` varchar(255) NOT NULL UNIQUE,
  `password_hash` varchar(255) NOT NULL,
  `role` int NOT NULL,
  PRIMARY KEY (`admin_id`)
);
```

**Рисунок 2.3 – Створення таблиці Admin\_users з обмеженнями PRIMARY KEY, UNIQUE, NOT NULL**

На цьому скріншоті видно використання:

1. PRIMARY KEY для поля admin\_id;
2. UNIQUE для username;
3. NOT NULL для обов'язкових полів (ім'я користувача, хеш паролю, роль).

Це гарантує, що кожен адміністратор має унікальний логін і визначену роль.

```
CREATE TABLE IF NOT EXISTS `reviews` (
  `review_id` int AUTO_INCREMENT NOT NULL UNIQUE,
  `client_id` int NOT NULL,
  `employee` int NOT NULL,
  `service_id` int NOT NULL,
  `rating` int NOT NULL,
  `comment_text` varchar(255) NOT NULL,
  `review_date` date NOT NULL,
  PRIMARY KEY (`review_id`)
);
```

**Рисунок 2.4 Створення таблиці Reviews з обмеженнями PRIMARY KEY, NOT NULL**

Ця таблиця містить поля з обов'язковим заповненням (NOT NULL) для кожного з відгуків. Поле review\_id є первинним ключем (PRIMARY KEY) і

автоінкрементується. Це дозволяє чітко відслідковувати кожен відгук і гарантує його унікальність.

```
CREATE TABLE IF NOT EXISTS `Payments` (  
    `payment_id` int AUTO_INCREMENT NOT NULL UNIQUE,  
    `amount` decimal(10,0) NOT NULL,  
    `appointment_id` int NOT NULL UNIQUE,  
    `payment_method` int NOT NULL,  
    `payment_date` date NOT NULL,  
    PRIMARY KEY (`payment_id`)  
);
```

**Рисунок 2.5 Створення таблиці Payments з обмеженнями PRIMARY KEY, NOT NULL**

Скріншот демонструє визначення PRIMARY KEY для payment\_id та обов'язкових полів для суми, типу платежу і дати. Застосування типу DECIMAL для amount дозволяє уникнути похибок з округленням фінансових значень.

```
CREATE TABLE IF NOT EXISTS `Client` (  
    `client_id` int AUTO_INCREMENT NOT NULL UNIQUE,  
    `name` varchar(255) NOT NULL,  
    `phone` int NOT NULL,  
    `email` varchar(255) NOT NULL,  
    `birth_date` date NOT NULL,  
    PRIMARY KEY (`client_id`)  
);
```

**Рисунок 2.6 Створення таблиці Client з обмеженнями PRIMARY KEY, NOT NULL**

У таблиці клієнтів передбачено поля name, phone, email та birth\_date, які мають бути заповнені. Всі клієнти мають унікальний client\_id, що дозволяє підтримувати логічні зв'язки з іншими таблицями.

```
CREATE TABLE IF NOT EXISTS `Appointment` (  
    `appointment_id` int AUTO_INCREMENT NOT NULL UNIQUE,  
    `client_id` int NOT NULL,  
    `service_id` int NOT NULL,  
    `employee_id` int NOT NULL,  
    `datetime` datetime NOT NULL,  
    `status` varchar(255) NOT NULL,  
    PRIMARY KEY (`appointment_id`)  
);
```

**Рисунок 2.7 Створення таблиці Appointment з обмеженнями PRIMARY KEY, NOT NULL**

У цій таблиці зберігаються записи про зустрічі. Поля, що відповідають за зовнішні ключі (client\_id, service\_id, employee\_id), є обов'язковими (NOT NULL). Це дозволяє системі уникати помилок при створенні записів без вказання виконавця чи послуги.

```
CREATE TABLE IF NOT EXISTS `Service` (  
    `service_id` int AUTO_INCREMENT NOT NULL UNIQUE,  
    `name` int NOT NULL,  
    `duration` int NOT NULL,  
    `price` float NOT NULL,  
    PRIMARY KEY (`service_id`)  
);
```

**Рисунок 2.8 Створення таблиці Service з обмеженнями PRIMARY KEY, NOT NULL**



Сервіси також мають ідентифікатор `service_id` з автоматичною генерацією, а також поля `name`, `duration`, `price`, які не можуть бути порожніми. Поле `price` визначено як `FLOAT` для точного зберігання вартості.

```
CREATE TABLE IF NOT EXISTS `Employee` (
  `employee_id` int AUTO_INCREMENT NOT NULL UNIQUE,
  `name` varchar(255) NOT NULL,
  `specialization` varchar(255) NOT NULL,
  `schedule` datetime NOT NULL,
  PRIMARY KEY (`employee_id`)
);
```

**Рисунок 2.9 Створення таблиці Employee з обмеженнями PRIMARY KEY, NOT NULL**

В таблиці працівників усі ключові поля є обов'язковими, що виключає появу некоректних записів. Поле `schedule` дозволяє точно визначити робочий графік кожного співробітника.

```
ALTER TABLE `Appointment` ADD CONSTRAINT `Appointment_fk1` FOREIGN KEY (`client_id`) REFERENCES `Client`(`client_id`);

ALTER TABLE `Appointment` ADD CONSTRAINT `Appointment_fk2` FOREIGN KEY (`service_id`) REFERENCES `Service`(`service_id`);

ALTER TABLE `Appointment` ADD CONSTRAINT `Appointment_fk3` FOREIGN KEY (`employee_id`) REFERENCES `Employee`(`employee_id`);
ALTER TABLE `Payments` ADD CONSTRAINT `Payments_fk2` FOREIGN KEY (`appointment_id`) REFERENCES `Appointment`(`appointment_id`);

ALTER TABLE `reviews` ADD CONSTRAINT `reviews_fk1` FOREIGN KEY (`client_id`) REFERENCES `Client`(`client_id`);

ALTER TABLE `reviews` ADD CONSTRAINT `reviews_fk2` FOREIGN KEY (`employee`) REFERENCES `Employee`(`employee_id`);

ALTER TABLE `reviews` ADD CONSTRAINT `reviews_fk3` FOREIGN KEY (`service_id`) REFERENCES `Service`(`service_id`);
```

**Рисунок 2.10 Додавання зовнішніх ключів через ALTER TABLE**

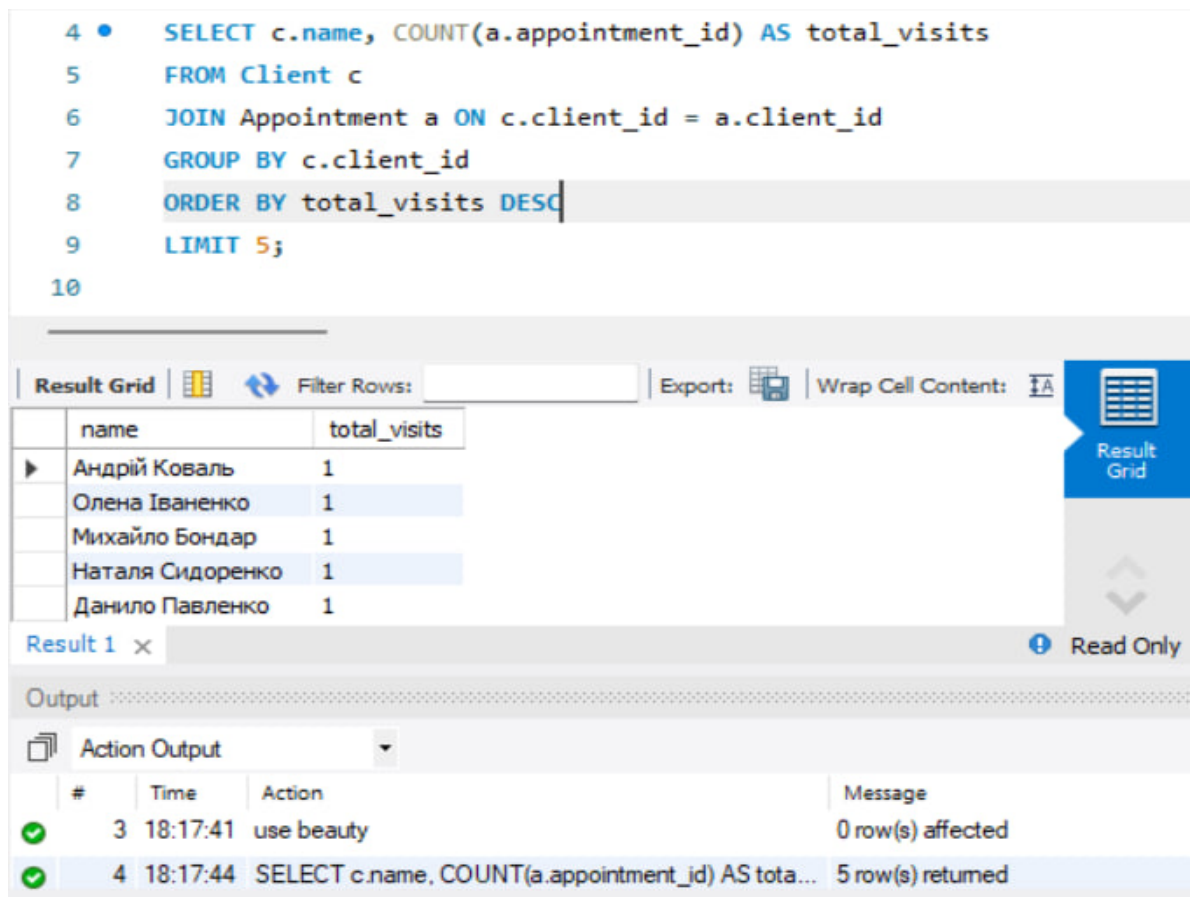
На цьому скріншоті представлено додавання обмежень цілісності типу `FOREIGN KEY` через команди `ALTER TABLE`. Вони забезпечують точність, узгодженість та надійність даних. Використання відповідних обмежень цілісності даних може допомогти уникнути помилок, полегшити роботу з даними та покращити загальну продуктивність бази даних.

SQL-структура представлена у Додатку А.

## 2.5. Реалізація SQL-скрипту

У процесі реалізації бази даних для інформаційної системи салону краси було створено низку SQL-запитів, що демонструють гнучкість і потужність запитів у мові SQL. Ці запити не тільки дозволяють здійснювати глибоку аналітику даних, але й є прикладом ефективного використання структурованої інформації. Нижче наведено серію SQL-запитів, реалізованих у середовищі MySQL Workbench, з поясненнями щодо їхньої мети та інтерпретації отриманих результатів.

### 1. Топ-5 клієнтів за кількістю записів



```

4 • SELECT c.name, COUNT(a.appointment_id) AS total_visits
5 FROM Client c
6 JOIN Appointment a ON c.client_id = a.client_id
7 GROUP BY c.client_id
8 ORDER BY total_visits DESC
9 LIMIT 5;
10

```

name	total_visits
Андрій Коваль	1
Олена Іваненко	1
Михайло Бондар	1
Наталя Сидоренко	1
Данило Павленко	1

Result 1 x Read Only

Output

Action Output

#	Time	Action	Message
3	18:17:41	use beauty	0 row(s) affected
4	18:17:44	SELECT c.name, COUNT(a.appointment_id) AS tota...	5 row(s) returned

**Рисунок 2.11 SQL-запит та результат для топ-5 клієнтів за кількістю записів.**

Цей запит визначає клієнтів, які найчастіше відвідували салон. В результаті можна побачити список імен клієнтів з кількістю їх візитів.

## 2. Середній вік зареєстрованих клієнтів салону краси

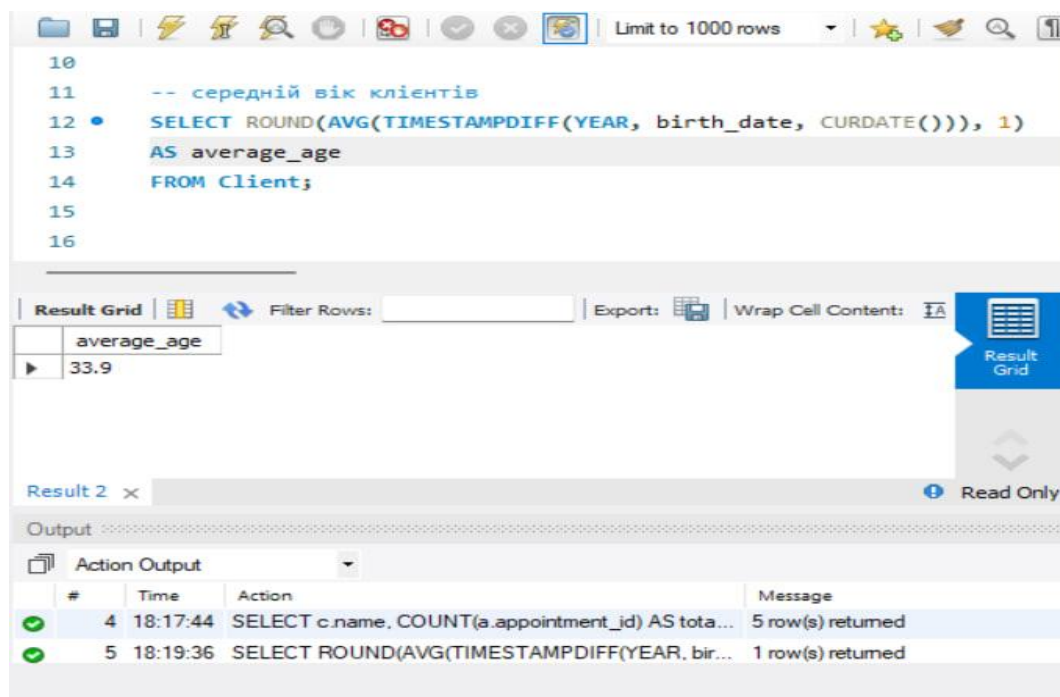


Рисунок 2.12 Визначення середнього віку клієнтів.

Обчислює середній вік клієнтів салону.

## 3. Кількість записів на кожну послугу

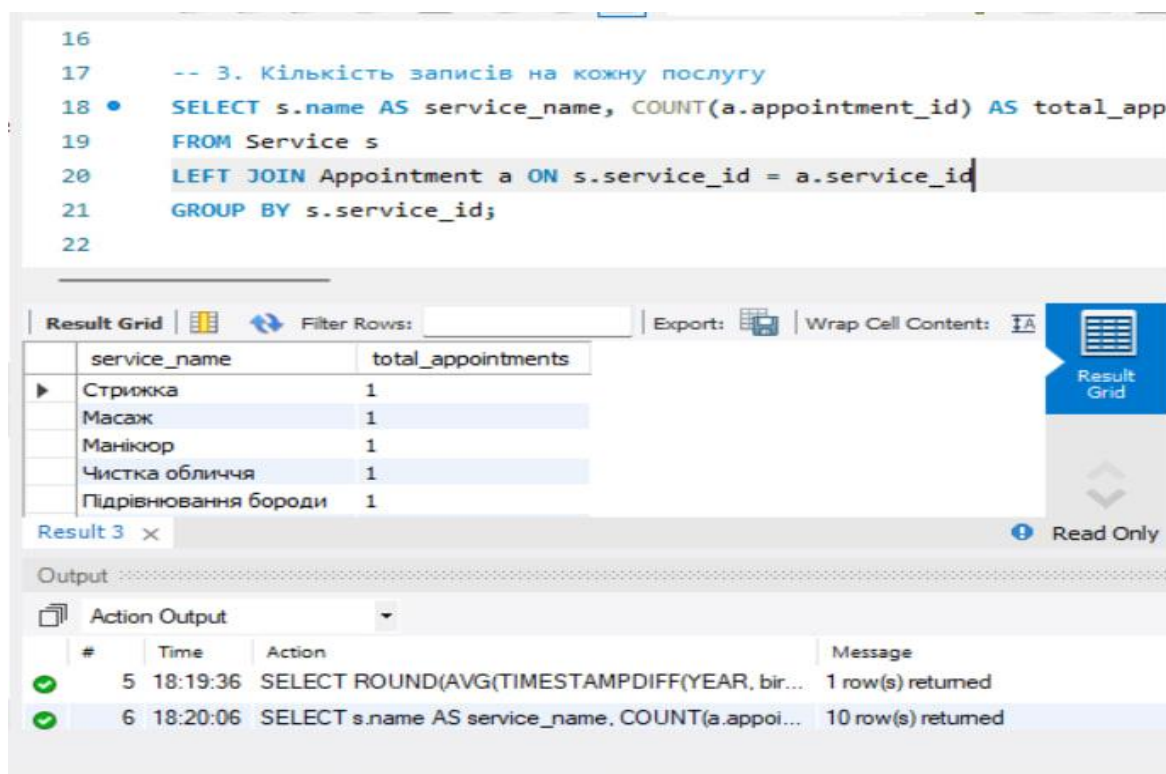


Рисунок 2.13 Кількість записів на кожну з послуг салону.

Цей запит використовується для визначення популярності послуг.

#### 4. Загальна сума платежів по кожному клієнту

The screenshot shows a SQL query in a window titled 'SQL File 4\*'. The query is as follows:

```

23
24  -- 4. Загальна сума платежів по кожному клієнту
25  • SELECT c.name, SUM(p.amount) AS total_spent
26    FROM Client c
27   JOIN Appointment a ON c.client_id = a.client_id
28   JOIN Payments p ON a.appointment_id = p.appointment_id
29   GROUP BY c.client_id;

```

Below the query, the 'Result Grid' displays the following data:

	name	total_spent
▶	Андрій Коваль	250
	Олена Іваненко	500
	Михайло Бондар	200
	Наталя Сидоренко	400
	Данило Павленко	150

The 'Output' pane shows the execution log:

#	Time	Action	Message
6	18:20:06	SELECT s.name AS service_name, COUNT(a.appoi...	10 row(s) returned
7	18:21:05	SELECT c.name, SUM(p.amount) AS total_spent FR...	10 row(s) returned

**Рисунок 2.14 SQL-запит із підрахунком загальної суми витрат по клієнтах.**

Цей запит дозволяє побачити, скільки кожен клієнт витратив у салоні.

#### 5. Середній рейтинг кожного працівника

The screenshot shows a SQL query in a window titled 'SQL File 4\*'. The query is as follows:

```

32
33  -- 5. Середній рейтинг кожного працівника
34  • SELECT e.name, ROUND(AVG(r.rating), 2) AS avg_rating
35    FROM Employee e
36   JOIN Reviews r ON e.employee_id = r.employee
37   GROUP BY e.employee_id;
38

```

Below the query, the 'Result Grid' displays the following data:

	name	avg_rating
▶	Ірина Ковальчук	5.00
	Віталій Сорока	4.00
	Ганна Романюк	3.00
	Сергій Дяченко	2.00
	Катерина Шевченко	5.00

The 'Output' pane shows the execution log:

#	Time	Action	Message
7	18:21:05	SELECT c.name, SUM(p.amount) AS total_spent FR...	10 row(s) returned
8	18:21:27	SELECT e.name, ROUND(AVG(r.rating), 2) AS avg_...	5 row(s) returned

**Рисунок 2.15 Середній рейтинг працівників за відгуками.**



Запит дозволяє аналізувати якість роботи персоналу за оцінками клієнтів.

## 6. Послуги дорожчі за 300 грн

The screenshot shows a SQL query editor with the following code:

```

38
39  -- 6. Послуги дорожчі за 300 грн
40  • SELECT name, price
41    FROM Service
42    WHERE price > 300;
43
44  -- 7. Кількість клієнтів за місяцями народження

```

Below the editor is the 'Result Grid' showing the results of the query:

	name	price
▶	Масаж	500
	Чистка обличчя	400
	Фарбування волосся	600
	Глибокий масаж	700
	Нейл-арт	350

The 'Output' pane shows the execution log:

#	Time	Action	Message
8	18:21:27	SELECT e.name, ROUND(AVG(r.rating), 2) AS avg_...	5 row(s) returned
9	18:21:55	SELECT name, price FROM Service WHERE price ...	5 row(s) returned

**Рисунок 2.16 Вибірка послуг з ціною вище 300 грн.**

Фільтрує та виводить усі послуги, які коштують понад 300 грн.

## 7. Кількість клієнтів за місяцями народження

The screenshot shows a SQL query editor with the following code:

```

44  -- 7. Кількість клієнтів за місяцями народження
45  • SELECT MONTH(birth_date) AS birth_month, COUNT(*) AS clients_count
46    FROM Client
47    GROUP BY birth_month
48    ORDER BY birth_month;
49
50

```

Below the editor is the 'Result Grid' showing the results of the query:

	birth_month	clients_count
▶	2	2
	3	2
	4	2
	5	2
	6	2

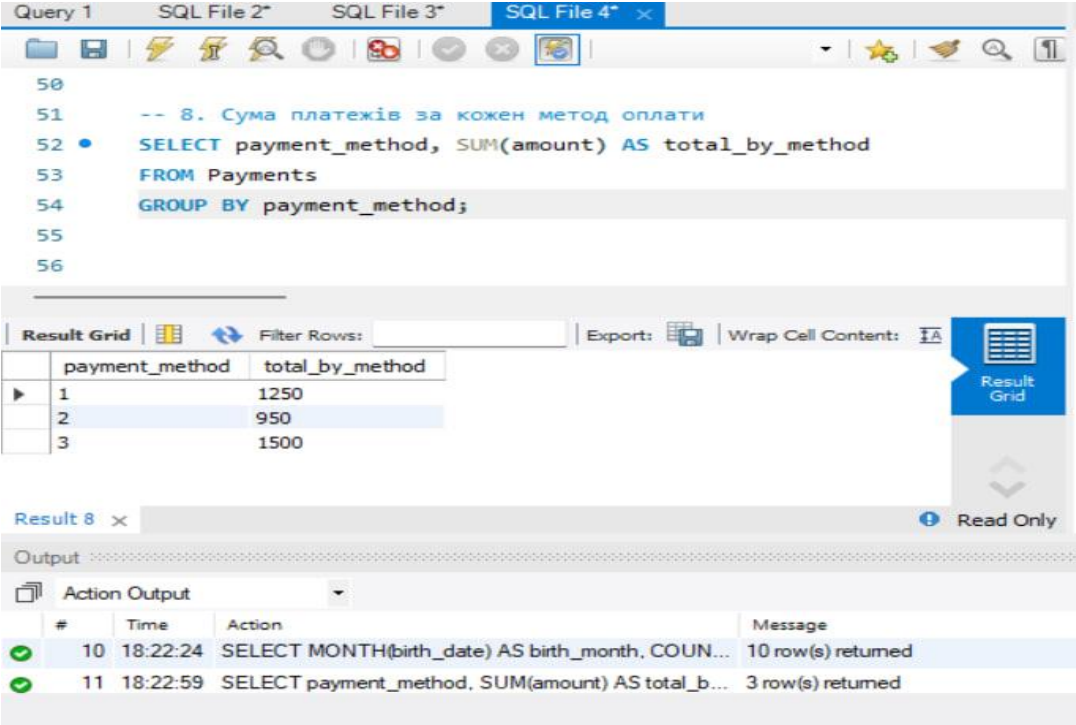
The 'Output' pane shows the execution log:

#	Time	Action	Message
9	18:21:55	SELECT name, price FROM Service WHERE price ...	5 row(s) returned
10	18:22:24	SELECT MONTH(birth_date) AS birth_month, COUN...	10 row(s) returned

**Рисунок 2.17 Розподіл клієнтів за місяцем народження.**

Аналізує розподіл клієнтів за місяцями народження, що може бути корисно для акцій чи знижок до дня народження.

#### 8. Сума платежів за кожен метод оплати



The screenshot shows a SQL Developer window with a query titled "8. Сума платежів за кожен метод оплати". The query is as follows:

```
-- 8. Сума платежів за кожен метод оплати
SELECT payment_method, SUM(amount) AS total_by_method
FROM Payments
GROUP BY payment_method;
```

The result grid displays the following data:

payment_method	total_by_method
1	1250
2	950
3	1500

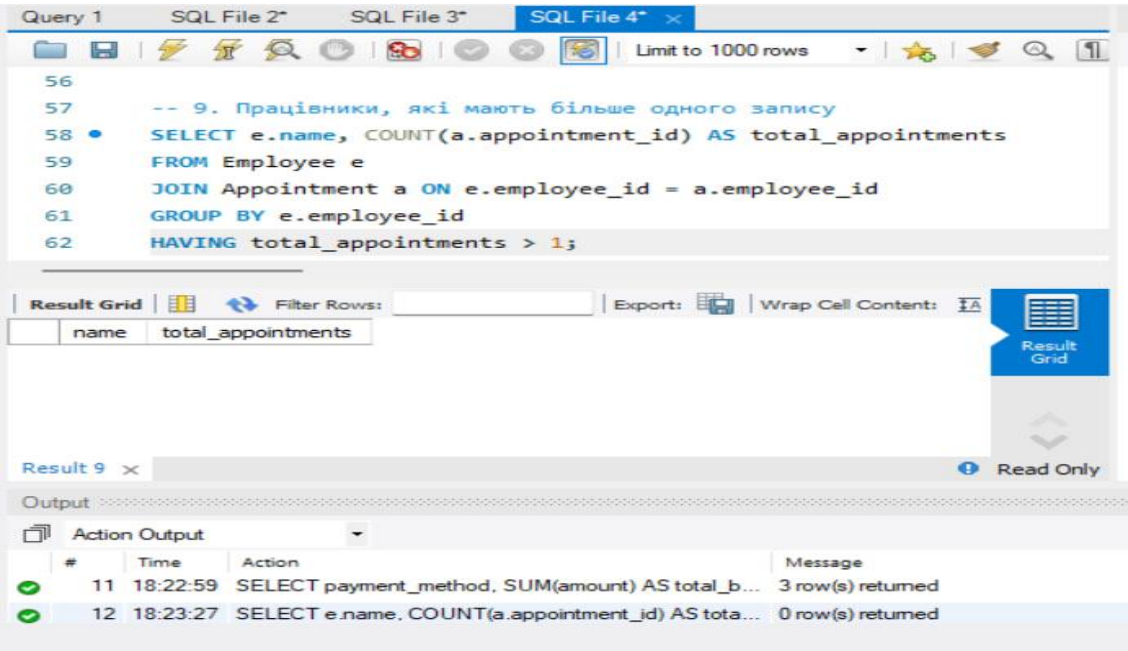
The bottom panel shows the "Output" window with the following messages:

#	Time	Action	Message
10	18:22:24	SELECT MONTH(birth_date) AS birth_month, COUN...	10 row(s) returned
11	18:22:59	SELECT payment_method, SUM(amount) AS total_b...	3 row(s) returned

**Рисунок 2.18 Суми за методами оплати.**

Дозволяє виявити популярність методів оплати: готівка, картка чи інші.

#### 9. Працівники, які мають більше одного запису



The screenshot shows a SQL Developer window with a query titled "9. Працівники, які мають більше одного запису". The query is as follows:

```
-- 9. Працівники, які мають більше одного запису
SELECT e.name, COUNT(a.appointment_id) AS total_appointments
FROM Employee e
JOIN Appointment a ON e.employee_id = a.employee_id
GROUP BY e.employee_id
HAVING total_appointments > 1;
```

The result grid displays the following data:

name	total_appointments
------	--------------------

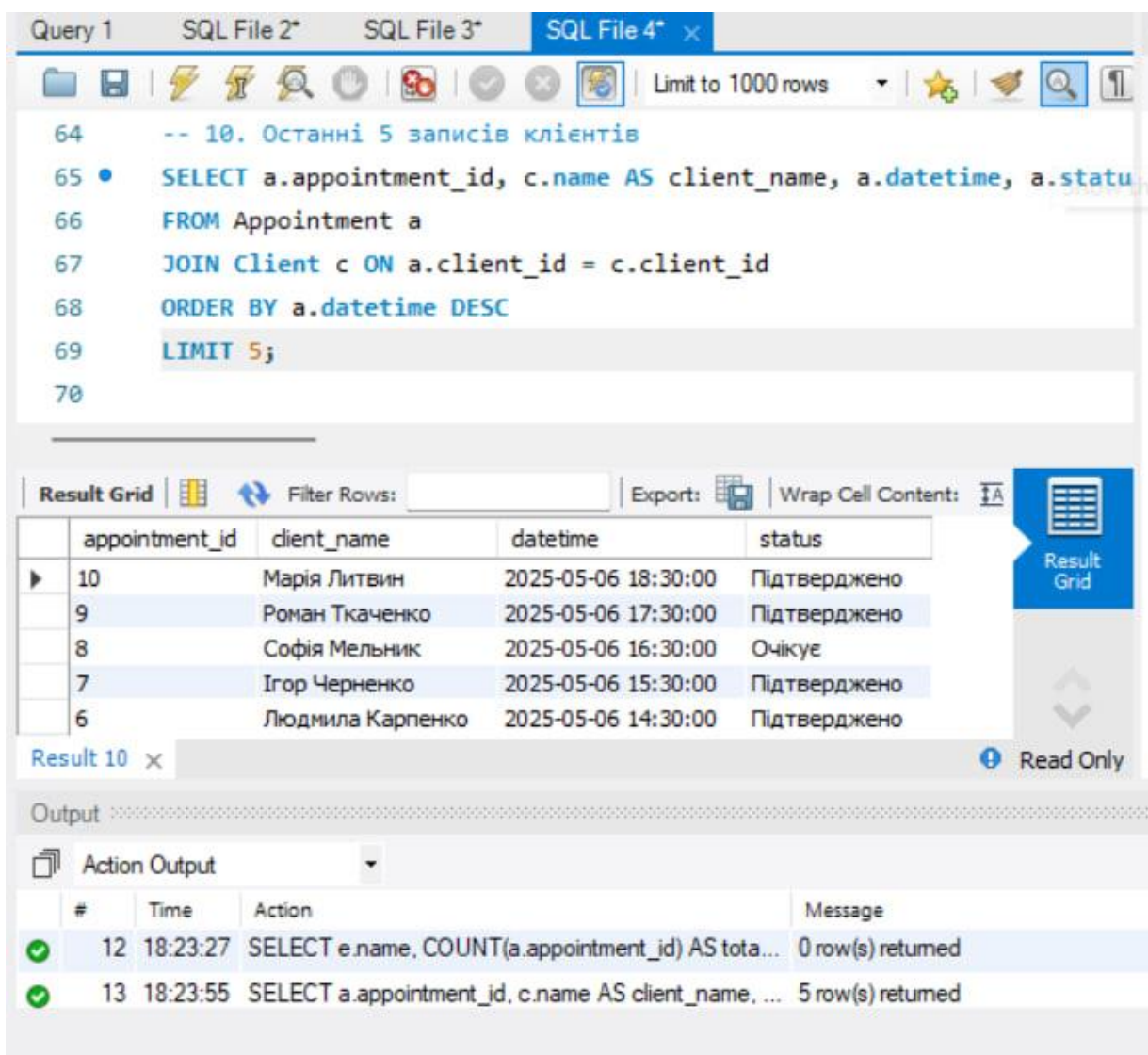
The bottom panel shows the "Output" window with the following messages:

#	Time	Action	Message
11	18:22:59	SELECT payment_method, SUM(amount) AS total_b...	3 row(s) returned
12	18:23:27	SELECT e.name, COUNT(a.appointment_id) AS tota...	0 row(s) returned

**Рисунок 2.19 Працівники з двома і більше записами.**

Цей запит показує, які працівники користуються найбільшим попитом.

#### 10. Останні 5 записів клієнтів



The screenshot displays a SQL IDE interface with a query editor and a results grid. The query is as follows:

```

64  -- 10. Останні 5 записів клієнтів
65  •  SELECT a.appointment_id, c.name AS client_name, a.datetime, a.status
66      FROM Appointment a
67      JOIN Client c ON a.client_id = c.client_id
68      ORDER BY a.datetime DESC
69      LIMIT 5;
70

```

The results grid shows the following data:

appointment_id	client_name	datetime	status
10	Марія Литвин	2025-05-06 18:30:00	Підтверджено
9	Роман Ткаченко	2025-05-06 17:30:00	Підтверджено
8	Софія Мельник	2025-05-06 16:30:00	Очікує
7	Ігор Черненко	2025-05-06 15:30:00	Підтверджено
6	Людмила Карпенко	2025-05-06 14:30:00	Підтверджено

The output section shows the execution log:

#	Time	Action	Message
12	18:23:27	SELECT e.name, COUNT(a.appointment_id) AS tota...	0 row(s) returned
13	18:23:55	SELECT a.appointment_id, c.name AS client_name, ...	5 row(s) returned

**Рисунок 2.20 П'ять останніх записів до салону.**

Це дозволяє вивести останні дії клієнтів для швидкого моніторингу.

## РОЗДІЛ 3. РОЗРОБКА ВЕБ-САЙТУ

### 3.1. Структура веб-сайту

Структура веб-сайту є фундаментальною складовою, яка визначає логіку побудови інтерфейсу, взаємозв'язки між сторінками та зручність користування для кінцевого споживача. Для інформаційної системи салону краси було спроектовано чітку, інтуїтивно зрозумілу структуру, яка охоплює всі основні функціональні компоненти, необхідні для ефективної взаємодії користувача з платформою.

Основна мета структури — забезпечити зручну навігацію, швидкий доступ до ключових розділів, а також створити логічну й естетичну послідовність сторінок. При побудові архітектури сайту враховано поділ користувачів на клієнтів, працівників та адміністраторів, що дозволило розмежувати функціональні зони для кожної категорії.

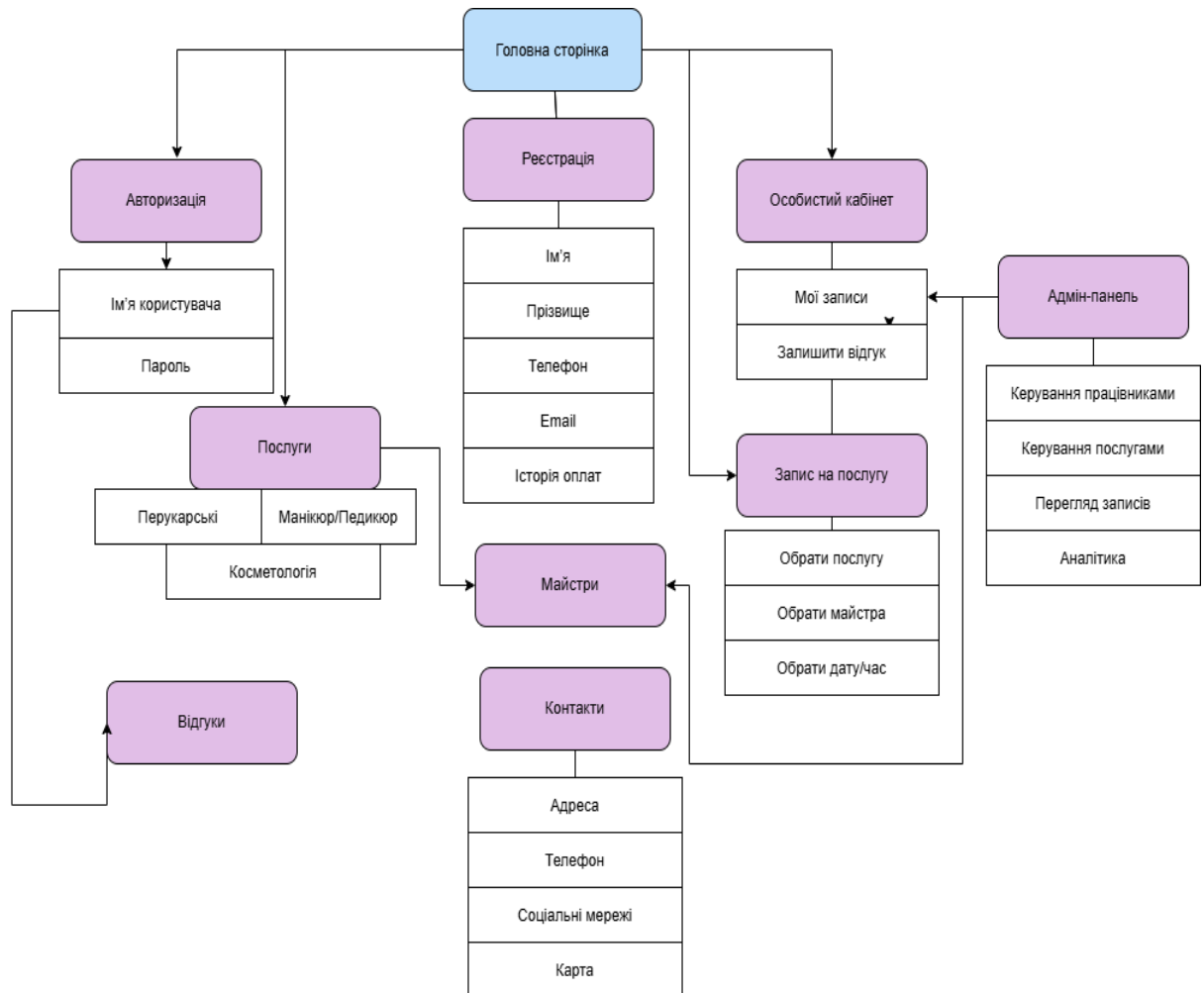
До головних розділів веб-сайту належать:

- Головна сторінка — привітальна зона, яка знайомить відвідувача з брендом, філософією салону та основними перевагами.
- Послуги — список доступних процедур, згрупованих за категоріями: перукарські, манікюр/педикюр, косметологія.
- Майстри — перелік працівників з вказанням спеціалізацій.
- Відгуки — розділ, де клієнти можуть залишити враження після візиту.
- Контакти — включає адресу, номер телефону, карту, соціальні мережі.
- Реєстрація та авторизація — сторінки створення облікового запису та входу в систему.
- Особистий кабінет — доступний після авторизації, містить можливість перегляду записів, залишення відгуків.
- Онлайн-запис — форма для швидкої реєстрації на послугу з вибором дати, спеціаліста та процедури.



- Адмін-панель — інтерфейс управління для адміністратора, який дозволяє керувати майстрами, послугами, переглядати аналітику та відгуки.

Вся ця структура була візуалізована у вигляді мапи сайту, яка була створена за допомогою інструменту «draw.io» й дозволяє графічно оцінити всі зв'язки між сторінками.

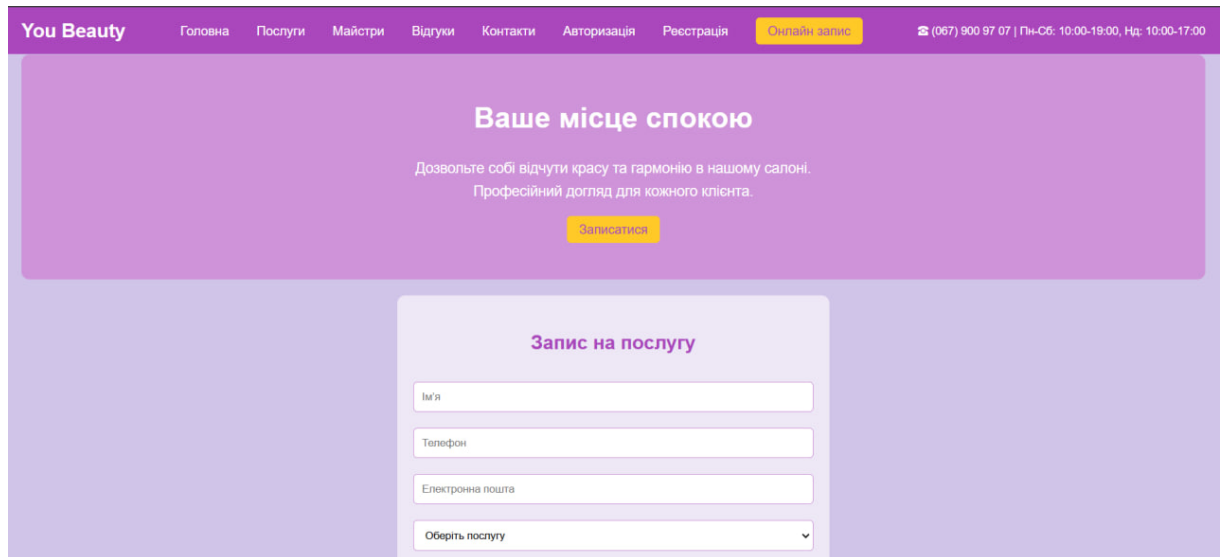


**Рисунок 3.1** Мапа сайту інформаційної системи салону краси

Мапа демонструє розгалуження функціоналу від головної сторінки, поділ на публічну частину сайту (доступну всім користувачам) та персоніфіковану частину (кабінет користувача, адмін-панель). Це рішення забезпечує зручну логіку взаємодії та дозволяє в майбутньому легко масштабувати веб-сайт, додаючи нові модулі та функції.

### 3.2. Макети сторінок веб-сайту

У поточній версії веб-сайт реалізовано як макет, що демонструє зовнішній вигляд і логіку розташування елементів. Хоча функціонал ще не повністю завершений, візуальна частина сайту дозволяє оцінити майбутню зручність користування.



The screenshot displays the home page of a website titled 'You Beauty'. The header is a dark purple bar containing a navigation menu with links: 'Головна', 'Послуги', 'Майстри', 'Відгуки', 'Контакти', 'Авторизація', 'Регістрація', and a highlighted 'Онлайн запис' button. On the right side of the header, contact information is provided: a phone number '(067) 900 97 07', operating hours 'Пн-Сб: 10:00-19:00, Нд: 10:00-17:00', and a location pin icon. The main content area has a light purple background. At the top, a large purple banner contains the heading 'Ваше місце спокою' (Your place of peace), followed by the text 'Дозвольте собі відчувати красу та гармонію в нашому салоні. Професійний догляд для кожного клієнта.' and a yellow 'Записатися' (Book) button. Below this, a white box titled 'Запис на послугу' (Book a service) contains a form with four input fields: 'Ім'я' (Name), 'Телефон', 'Електронна пошта' (Email), and a dropdown menu labeled 'Оберіть послугу' (Select service).

**Рисунок 3.2** Головна сторінка з формою онлайн-запису

Центральним елементом головної сторінки є слоган «Ваше місце спокою», що візуально підкреслює концепцію салону як простору турботи, комфорту та індивідуального підходу до кожного клієнта.

Нижче розміщено форму онлайн-запису, яка дозволяє зручно залишити заявку на обрану послугу. Вона містить поля для введення імені, телефону, електронної пошти та випадаючий список із доступними процедурами. Такий елемент взаємодії спрощує комунікацію між клієнтом і адміністрацією салону, а також дозволяє заощадити час на запис.

Інтерфейс сторінки оформлено в спокійних фіолетових тонах, що сприяє формуванню атмосфери затишку. Верхня навігаційна панель забезпечує швидкий перехід до всіх ключових розділів сайту, зокрема до послуг, майстрів, контактної інформації та форми авторизації. Розташування елементів продумане так, щоб користувач легко орієнтувався у структурі сайту незалежно від типу пристрою.

**You Beauty** Головна Послуги Майстри Відгуки Контакти (067) 900 97 07 | Пн-Сб: 10:00-19:00, Нд: 10:00-17:00

Авторизація Реєстрація **Онлайн запис**

### Запис на послугу

Ім'я

Телефон

Електронна пошта

Оберіть послугу

дд . мм . рррр -- : --

**Записатися**

**Рисунок 3.3 Адаптивна версія сторінки із розширеною формою онлайн-запису**

Друга ілюстрація демонструє адаптивність веб-сайту, яка дозволяє зручно працювати з ним на пристроях з різним розширенням екрана. Макет автоматично підлаштовується до ширини вікна, забезпечуючи коректне відображення полів форми. У розширеній формі додано поле для вибору дати і часу, що дозволяє користувачеві заздалегідь визначити бажаний період для візиту.

**You Beauty** Головна Послуги Майстри Відгуки Контакти Авторизація Реєстрація **Онлайн запис** (067) 900 97 07 | Пн-Сб: 10:00-19:00, Нд: 10:00-17:00

### Наші послуги

#### Перукарські послуги

- Стрижка - 30 хв, 400 грн
- Фарбування - 90 хв, 1200 грн
- Укладка - 45 хв, 500 грн

#### Манікюр/Педикюр

- Манікюр - 45 хв, 300 грн
- Педикюр - 60 хв, 400 грн
- Нарощування нігтів - 120 хв, 800 грн

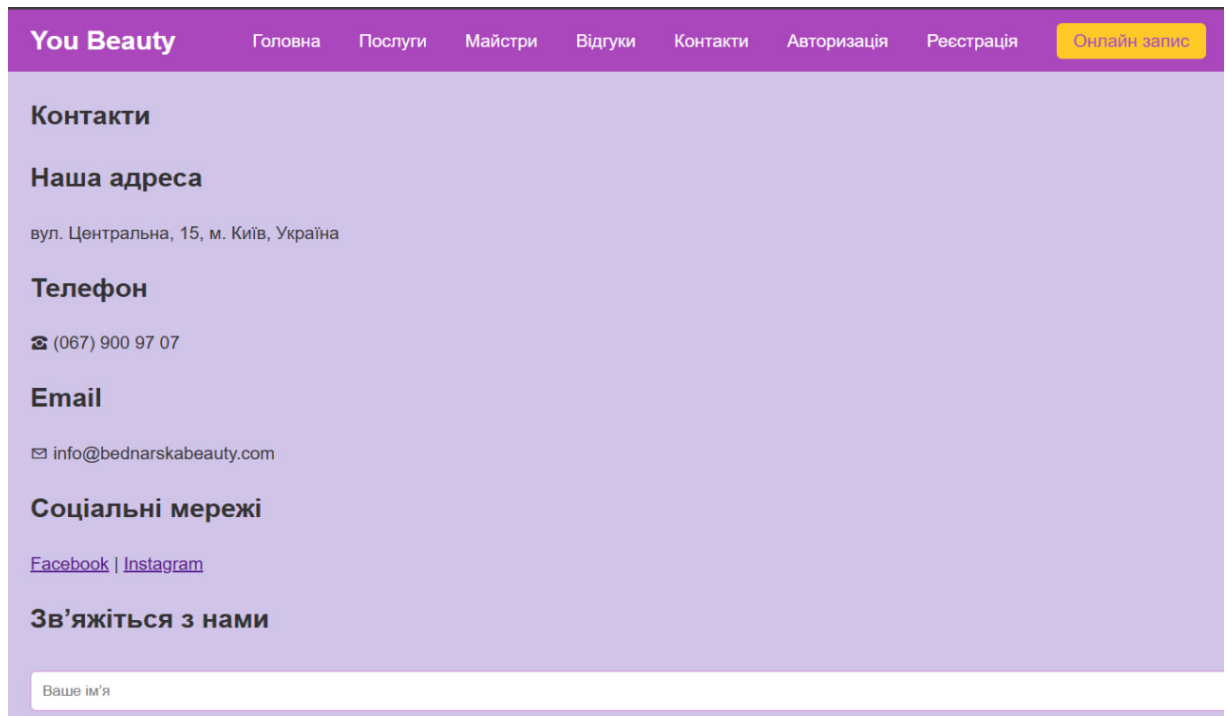
#### Косметологія

- Чистка обличчя - 60 хв, 600 грн
- Масаж обличчя - 30 хв, 400 грн
- Пілінг - 45 хв, 500 грн

**Обрати майстра**

**Рисунок 3.4 Сторінка "Послуги" з категоріями та описами процедур**

Сторінка "Послуги" містить детальний перелік процедур, поділених на три основні категорії: перукарські послуги, манікюр/педикюр та косметологія. Біля кожної послуги вказано її тривалість та вартість, що дозволяє користувачам зручно порівнювати варіанти. Кнопка "Обрати майстра" спрямована на полегшення навігації до подальшого етапу взаємодії з сайтом.



**Рисунок 3.5 Сторінка "Контакти" з контактною інформацією і формою зв'язку**

На сторінці "Контакти" представлено всю необхідну інформацію для зв'язку із салоном: адреса, телефон, електронна пошта, посилання на соціальні мережі. Також присутня форма зворотного зв'язку, яка дає змогу користувачеві залишити повідомлення без необхідності переходити на поштовий клієнт.

Загалом, макети сторінок демонструють логіку роботи майбутнього повноцінного сайту, який буде реалізований із використанням бази даних, серверної логіки та JavaScript-інтерактиву. Поточна версія є візуальною моделлю інтерфейсу, придатною для оцінки дизайну та зручності користування.

### 3.3. Програмування серверної частини

У даному проєкті серверна частина ще не реалізована, однак веб-сайт спроектовано з урахуванням її можливого підключення. Усі дані, які вводяться через форми (наприклад, форма онлайн-запису), вже відповідають структурі бази даних, описаної у розділі 2.

Завдяки правильному моделюванню логіки зберігання інформації (таблиці Client, Appointment, Service, Reviews) можлива повна інтеграція сайту з серверною частиною. У подальшому планується використання однієї з серверних мов (наприклад, PHP або Node.js) для обробки введених даних, перевірки їх достовірності, взаємодії з базою даних та відправлення відповідей клієнту.

Основними задачами серверної частини будуть:

- Обробка даних форм (запис клієнтів, відгуки);
- Реалізація авторизації/реєстрації;
- Зв'язок з базою даних через SQL-запити;
- Динамічна генерація інформації на сторінках.

Таким чином, інформаційна система вже підготовлена до майбутнього доопрацювання та переходу від демонстраційного макету до повноцінного динамічного сайту з backend-функціоналом.

### 3.4. Програмування клієнтської частини

Клієнтська частина веб-сайту реалізована з використанням мов HTML5 та CSS3. Головною метою є створення візуально привабливого, зручного та адаптивного інтерфейсу для взаємодії користувача з системою. Уся логіка клієнтської частини побудована на основі статичних сторінок, які мають послідовну структуру та об'єднані єдиним стилем.

Сайт складається з окремих HTML-файлів, зокрема: index.html (головна сторінка), posluhy.html, kontakty.html, pro-nas.html, які зв'язані між собою через верхнє меню навігації.

Головна сторінка містить форму запису на послугу, що реалізована наступним чином:

### *Лістинг 3.1*

```
html
<form>
  <input type="text" placeholder="Ім'я" required>
  <input type="tel" placeholder="Телефон" required>
  <input type="email" placeholder="Електронна пошта" required>
  <select>
    <option value="">Оберіть послугу</option>
    <option value="haircut">Стрижка</option>
    <option value="massage">Масаж</option>
    <option value="manicure">Манікюр</option>
  </select>
  <input type="datetime-local" required>
  <button type="submit">Записатися</button>
</form>
```

Цей код відповідає макету сайту, де форма міститься у блоці <section id="booking">, що стилізований окремо. Усі стилі оформлено в зовнішньому файлі style.css, який забезпечує кольорову палітру, читабельність і адаптивність.

Приклад стилів для форми:

### *Лістинг 3.2*

```
css
#booking {
  background-color: #EDE7F6;
  padding: 20px;
```

```

border-radius: 10px;
max-width: 500px;
margin: 0 auto;
}
form input, form select {
margin: 10px 0;
padding: 10px;
border: 1px solid #CE93D8;
border-radius: 5px;
}
form button {
background-color: #FFCA28;
color: #AB47BC;
padding: 10px;
border: none;
border-radius: 5px;
cursor: pointer;
}

```

Окрім стилізації форм, у CSS-файлі окремо оформлено навігаційне меню, секції hero з головним слоганом, футер, адаптивність для мобільних пристроїв.

Завдяки правильному поділу HTML та CSS структура сайту є гнучкою для подальшого розширення: можна легко інтегрувати JavaScript, підключити бекенд або додати нові сторінки.

Повні HTML та CSS-файли наведено у Додатках.

### **3.5 Розміщення веб-сайту на локальному віртуальному середовищі**

Після завершення етапу верстки веб-сайт було розміщено на локальному сервері для перевірки працездатності, зручності навігації та адаптивності макету. Для цього використовувалося віртуальне середовище Visual Studio Code у

поєднанні з розширенням Live Server, яке дозволяє запускати локальний вебсервер і переглядати HTML-сторінки в браузері у режимі реального часу.

Такий підхід має низку переваг:

- Швидкий запуск сайту без необхідності налаштовувати зовнішній хостинг;
- Автоматичне оновлення сторінки при збереженні змін у коді;
- Можливість тестування адаптивності, структури та коректності стилів;
- Зручність у налагодженні навіть для початківців.

Щоб запускати сайт локально, необхідно:

1. Встановити Visual Studio Code;
2. Встановити розширення Live Server із Marketplace;
3. Відкрити папку з HTML-файлами;
4. Відкрити один із HTML-файлів і натиснути «Go Live» (у нижній частині VS Code).

Після цього в браузері автоматично відкриється сторінка з локальним посиланням на кшталт `http://127.0.0.1:5500/index.html`, де можна переглядати сайт у дії.

У разі подальшого розширення функціоналу та реалізації серверної логіки сайт можна буде розгорнути на більш складному локальному серверному середовищі — наприклад, за допомогою XAMPP (для PHP) або Node.js + Express (для JavaScript).

Також у перспективі можливе розміщення проєкту на онлайн-хостингу:

- Для статичної версії — через GitHub Pages;
- Для динамічної версії — на хмарних платформах, таких як Firebase, Vercel, Netlify або Heroku.

Таким чином, навіть на етапі макету проєкт вже демонструє готовність до подальшої інтеграції та публікації в Інтернеті.



## ВИСНОВКИ

У процесі виконання курсової роботи на тему «Інформаційна система для салону краси» було детально проаналізовано особливості функціонування закладів у сфері індустрії краси, вивчено бізнес-процеси, що підлягають автоматизації, та розроблено модель інформаційної системи, яка дозволяє оптимізувати управління клієнтами, послугами, працівниками, записами й фінансами.

На першому етапі дослідження було проведено аналіз предметної області. Встановлено, що основні проблеми традиційної моделі функціонування салону — це плутанина в розкладах, відсутність централізованого обліку, людський фактор та низький рівень персоналізованого сервісу. На основі цього було визначено основні категорії користувачів системи (клієнт, працівник, адміністратор), сформовано технічне завдання та створено загальну структуру системи.

У другому розділі реалізовано детальне проєктування бази даних. Було визначено основні сутності (Client, Employee, Service, Appointment, Payments, Reviews, Admin\_users) та побудовано зв'язки між ними за допомогою зовнішніх ключів. Було виконано поетапну нормалізацію структури до третьої нормальної форми, що дозволило уникнути надлишковості та забезпечити логічну узгодженість збереження даних. Особливу увагу приділено визначенню типів даних для кожного поля таблиць, що забезпечує точність і оптимальну продуктивність. Крім того, було впроваджено обмеження цілісності, які захищають базу даних від помилок введення та забезпечують коректність усіх операцій.

Окрему частину роботи становить реалізація SQL-структури та створення прикладних SQL-запитів для виконання реальних завдань: аналізу рейтингу працівників, підрахунку платежів, визначення популярних послуг, перегляду статистики клієнтів тощо. Запити, реалізовані в середовищі MySQL Workbench,

демонструють аналітичні можливості системи, необхідні для прийняття управлінських рішень.

У третьому розділі було представлено веб-інтерфейс інформаційної системи. Розроблено макет сайту з кількох HTML-сторінок і єдиним стилістичним оформленням у CSS. Було створено форму онлайн-запису, сторінки послуг, контактів, а також реалізовано адаптивність інтерфейсу. Візуальна частина сайту повністю відповідає структурі бази даних і підготовлена до майбутнього підключення серверної частини. Для локального тестування макету використано інструмент Live Server у середовищі Visual Studio Code.

У результаті розроблено гнучку, масштабовану модель інформаційної системи, яка охоплює ключові функціональні потреби салону краси. Реалізований функціонал дозволяє автоматизувати процеси обліку клієнтів і записів, керування працівниками та послугами, збирання статистичних даних та обробку платежів.

Практичне значення проєкту полягає в можливості його подальшого впровадження у реальний бізнес. Система здатна забезпечити вищий рівень обслуговування клієнтів, оптимізувати внутрішні процеси, підвищити ефективність управління та конкурентоспроможність закладу. За рахунок використання відкритих технологій (HTML, CSS, SQL, MySQL Workbench) реалізація такого рішення є доступною навіть для малого бізнесу, що робить розроблену систему практично цінною та перспективною для подальшого розвитку.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Буряк, В. Я. Проектування баз даних: навчальний посібник. – Львів: ЛНУ ім. І. Франка, 2019. – 212 с.
2. Гринченко, С. М. Інформаційні системи і технології в економіці: навч. посібник. – К.: Центр учбової літератури, 2021. – 328 с.
3. Бейліс, А. HTML та CSS. Розробка та дизайн веб-сайтів. – К.: Видавництво «Фабула», 2020. – 512 с.
4. Шевчук, О. С. MySQL для початківців. Практичний посібник. – Харків: Ранок, 2021. – 176 с.
5. W3Schools. HTML, CSS, JavaScript Tutorials [Електронний ресурс]. – Режим доступу: <https://www.w3schools.com>
6. MDN Web Docs. HTML reference [Електронний ресурс]. – Режим доступу: <https://developer.mozilla.org/en-US/docs/Web/HTML>
7. Stack Overflow. Questions and solutions related to MySQL and HTML forms [Електронний ресурс]. – Режим доступу: <https://stackoverflow.com>
8. Visual Studio Code documentation [Електронний ресурс]. – Режим доступу: <https://code.visualstudio.com/docs>
9. MySQL Documentation [Електронний ресурс]. – Режим доступу: <https://dev.mysql.com/doc/>
10. Глушаков, В. В. Інформаційні технології в управлінні підприємством: навч. посіб. – К.: НАУ, 2020. – 252 с.

## ДОДАТКИ

### *Додаток А* *SQL-скрипт*

```
CREATE TABLE IF NOT EXISTS `Client` (  
    `client_id` int AUTO_INCREMENT NOT NULL UNIQUE,  
    `name` varchar(255) NOT NULL,  
    `phone` int NOT NULL,  
    `email` varchar(255) NOT NULL,  
    `birth_date` date NOT NULL,  
    PRIMARY KEY (`client_id`));  
  
CREATE TABLE IF NOT EXISTS `Employee` (  
    `employee_id` int AUTO_INCREMENT NOT NULL UNIQUE,  
    `name` varchar(255) NOT NULL,  
    `specialization` varchar(255) NOT NULL,  
    `schedule` datetime NOT NULL,  
    PRIMARY KEY (`employee_id`));  
  
CREATE TABLE IF NOT EXISTS `Service` (  
    `service_id` int AUTO_INCREMENT NOT NULL UNIQUE,  
    `name` int NOT NULL,  
    `duration` int NOT NULL,  
    `price` float NOT NULL,  
    PRIMARY KEY (`service_id`));  
  
CREATE TABLE IF NOT EXISTS `Appointment` (  
    `appointment_id` int AUTO_INCREMENT NOT NULL UNIQUE,  
    `client_id` int NOT NULL,  
    `service_id` int NOT NULL,  
    `employee_id` int NOT NULL,
```

```
`datetime` datetime NOT NULL,  
`status` varchar(255) NOT NULL,  
PRIMARY KEY (`appointment_id`));
```

```
CREATE TABLE IF NOT EXISTS `Payments` (  
  `payment_id` int AUTO_INCREMENT NOT NULL UNIQUE,  
  `amount` decimal(10,0) NOT NULL,  
  `appointment_id` int NOT NULL UNIQUE,  
  `payment_method` int NOT NULL,  
  `patment_date` date NOT NULL,  
  PRIMARY KEY (`payment_id`));
```

```
CREATE TABLE IF NOT EXISTS `Admin_users` (  
  `admin_id` int AUTO_INCREMENT NOT NULL UNIQUE,  
  `username` varchar(255) NOT NULL UNIQUE,  
  `password_hash` varchar(255) NOT NULL,  
  `role` int NOT NULL,  
  PRIMARY KEY (`admin_id`));
```

```
CREATE TABLE IF NOT EXISTS `reviews` (  
  `review_id` int AUTO_INCREMENT NOT NULL UNIQUE,  
  `client_id` int NOT NULL,  
  `employee` int NOT NULL,  
  `service_id` int NOT NULL,  
  `rating` int NOT NULL,  
  `comment_text` varchar(255) NOT NULL,  
  `review_date` date NOT NULL,  
  PRIMARY KEY (`review_id`));
```

```
ALTER TABLE `Appointment` ADD CONSTRAINT `Appointment_fk1` FOREIGN  
KEY (`client_id`) REFERENCES `Client`(`client_id`);
```

```
ALTER TABLE `Appointment` ADD CONSTRAINT `Appointment_fk2` FOREIGN  
KEY (`service_id`) REFERENCES `Service`(`service_id`);
```

```

ALTER TABLE `Appointment` ADD CONSTRAINT `Appointment_fk3` FOREIGN
KEY (`employee_id`) REFERENCES `Employee`(`employee_id`);

ALTER TABLE `Payments` ADD CONSTRAINT `Payments_fk2` FOREIGN KEY
(`appointment_id`) REFERENCES `Appointment`(`appointment_id`);

ALTER TABLE `reviews` ADD CONSTRAINT `reviews_fk1` FOREIGN KEY
(`client_id`) REFERENCES `Client`(`client_id`);

ALTER TABLE `reviews` ADD CONSTRAINT `reviews_fk2` FOREIGN KEY
(`employee`) REFERENCES `Employee`(`employee_id`);

ALTER TABLE `reviews` ADD CONSTRAINT `reviews_fk3` FOREIGN KEY
(`service_id`) REFERENCES `Service`(`service_id`);

```

## ***Додаток Б***

### ***Index.html***

```

<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>You Beauty</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <header>
        <div class="logo">You Beauty</div>
        <nav>
            <a href="index.html">Головна</a>
            <a href="posluhy.html">Послуги</a>
            <a href="masters.html">Майстри</a>
            <a href="reviews.html">Відгуки</a>
            <a href="kontakty.html">Контакти</a>
            <a href="auth.html">Авторизація</a>
            <a href="register.html">Реєстрація</a>
            <a href="#booking" class="btn">Онлайн запис</a>

```

```
</nav>

<div class="contact-info">
    <span>(067) 900 97 07</span> | <span>Пн-Сб: 10:00-
19:00, Нд: 10:00-17:00</span>
</div>

</header>

<main>
    <section class="hero">
        <h1>Ваше місце спокою</h1>

        <p>Дозвольте собі відчувати красу та гармонію в нашому
салоні. Професійний догляд для кожного клієнта.</p>

        <a href="#booking" class="btn">Записатися</a>
    </section>

    <section id="booking">
        <h2>Запис на послугу</h2>

        <form>
            <input type="text" placeholder="Ім'я" required>
            <input type="tel" placeholder="Телефон" required>
            <input type="email" placeholder="Електронна пошта"
required>

            <select>
                <option value="">Оберіть послугу</option>
                <option value="haircut">Стрижка</option>
                <option value="massage">Масаж</option>
                <option value="manicure">Манікюр</option>
            </select>

            <input type="datetime-local" required>
            <button type="submit">Записатися</button>
        </form>
    </section>

</main>

<footer>

    <a href="#booking" class="btn">Онлайн запис</a>
```

```
<div class="social">
    <a href="#">f</a>
    <a href="#">@</a>
</div>
</footer>
</body>
</html>
```

## ***Додаток В***

### ***Kontakty.html***

```
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Контакти - You Beauty</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <header>
        <div class="logo">You Beauty</div>
        <nav>
            <a href="index.html">Головна</a>
            <a href="posluhy.html">Послуги</a>
            <a href="masters.html">Майстри</a>
            <a href="reviews.html">Відгуки</a>
            <a href="kontakty.html">Контакти</a>
            <a href="auth.html">Авторизація</a>
            <a href="register.html">Реєстрація</a>
            <a href="#booking" class="btn">Онлайн запис</a>
        </nav>
```



```
<div class="contact-info">
    <span>(067) 900 97 07</span> | <span>Пн-Сб: 10:00-
19:00, Нд: 10:00-17:00</span>
</div>
</header>
<main>
    <section class="contacts">
        <h1>Контакти</h1>
        <div class="contact-details">
            <h2>Наша адреса</h2>
            <p>вул. Центральна, 15, м. Київ, Україна</p>
            <h2>Телефон</h2>
            <p> (067) 900 97 07</p>
            <h2>Email</h2>
            <p> info@bednarskabeauty.com</p>
            <h2>Соціальні мережі</h2>
            <p>
                <a href="#">Facebook</a> |
                <a href="#">Instagram</a>
            </p>
        </div>
        <div class="contact-form">
            <h2>Зв'яжіться з нами</h2>
            <form>
                <input type="text" placeholder="Ваше ім'я"
required>
                <input type="email" placeholder="Електронна
пошта" required>
                <textarea placeholder="Ваше повідомлення"
rows="5" required></textarea>
                <button type="submit">Надіслати</button>
            </form>
        </div>
```

```

        <div class="map">
            <h2>Ми на карті</h2>
            <p>[Тут буде карта, наприклад, Google Maps]</p>
        </div>
    </section>
</main>
<footer>
    <a href="#booking" class="btn">Онлайн запис</a>
    <div class="social">
        <a href="#">f</a>
        <a href="#">@</a>
    </div>
</footer>
</body>
</html>

```

*Додаток Г*  
*Posluhy.html*

```

<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Послуги - You Beauty</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <header>
        <div class="logo">You Beauty</div>
        <nav>
            <a href="index.html">Головна</a>

```

```
<a href="posluhy.html">Послуги</a>
<a href="masters.html">Майстри</a>
<a href="reviews.html">Відгуки</a>
<a href="kontakty.html">Контакти</a>
<a href="auth.html">Авторизація</a>
<a href="register.html">Реєстрація</a>
<a href="#booking" class="btn">Онлайн запис</a>

</nav>

<div class="contact-info">
    <span> (067) 900 97 07</span> | <span>Пн-Сб: 10:00-
19:00, Нд: 10:00-17:00</span>
</div>

</header>

<main>

    <section class="services">
        <h1>Наші послуги</h1>
        <div class="service-category">
            <h2>Перукарські послуги</h2>
            <ul>
                <li>Стрижка - 30 хв, 400 грн</li>
                <li>Фарбування - 90 хв, 1200 грн</li>
                <li>Укладка - 45 хв, 500 грн</li>
            </ul>
        </div>
        <div class="service-category">
            <h2>Манікюр/Педикюр</h2>
            <ul>
                <li>Манікюр - 45 хв, 300 грн</li>
                <li>Педикюр - 60 хв, 400 грн</li>
                <li>Нарощування нігтів - 120 хв, 800 грн</li>
            </ul>
        </div>
    </div>
```

```

        <div class="service-category">
            <h2>Косметологія</h2>
            <ul>
                <li>Чистка обличчя - 60 хв, 600 грн</li>
                <li>Масаж обличчя - 30 хв, 400 грн</li>
                <li>Пілінг - 45 хв, 500 грн</li>
            </ul>
        </div>
        <a href="masters.html" class="btn">Обрати майстра</a>
    </section>
</main>
<footer>
    <a href="#booking" class="btn">Онлайн запис</a>
    <div class="social">
        <a href="#">f</a>
        <a href="#">@</a>
    </div>
</footer>
</body>
</html>

```

*Додаток Д*  
*Pro-nas.html*

```

<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Про нас - You Beauty</title>
    <link rel="stylesheet" href="style.css">
</head>

```

```
<body>

  <header>

    <div class="logo">You Beauty</div>

    <nav>

      <a href="index.html">Головна</a>
      <a href="posluhy.html">Послуги</a>
      <a href="masters.html">Майстри</a>
      <a href="reviews.html">Відгуки</a>
      <a href="kontakty.html">Контакти</a>
      <a href="auth.html">Авторизація</a>
      <a href="register.html">Реєстрація</a>
      <a href="#booking" class="btn">Онлайн запис</a>

    </nav>

    <div class="contact-info">

      <span> (067) 900 97 07</span> | <span>Пн-Сб: 10:00-
19:00, Нд: 10:00-17:00</span>

    </div>

  </header>

  <main>

    <section class="about">

      <h1>Про нас</h1>

      <p>Ласкаво просимо до Bednarska Beauty – вашого місця
для краси та гармонії! Ми – команда професіоналів, які прагнуть
підкреслити вашу природну красу та подарувати вам відчуття
впевненості.</p>

      <h2>Наша місія</h2>

      <p>Наша місія – створювати комфортну атмосферу, де
кожен клієнт відчуває себе особливим. Ми використовуємо лише
якісні матеріали та сучасні техніки, щоб забезпечити вам найкращий
результат.</p>

      <h2>Наша команда</h2>

      <p>У нас працюють досвідчені майстри, які постійно
вдосконалюють свої навички. Від перукарів до косметологів – кожен
із нас готовий зробити ваш візит незабутнім.</p>

    </section>

  </main>

</body>
```

```

        <a href="kontakty.html" class="btn">Зв'яжіться з
нами</a>

    </section>
</main>
<footer>
    <a href="#booking" class="btn">Онлайн запис</a>
    <div class="social">
        <a href="#">f</a>
        <a href="#">@</a>
    </div>
</footer>
</body>
</html>

```

## Додаток E

### CSS стилі

```

/* Загальні стилі */
body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    background-color: #D1C4E9; /* М'який фіолетовий фон */
    color: #333;
    line-height: 1.6;}

/* Header */
header {
    background-color: #AB47BC; /* Темно-фіолетовий для шапки */
    color: #FFF;
    padding: 10px 20px;
    display: flex;
    justify-content: space-between;

```

```
    align-items: center;
    position: sticky;
    top: 0;
    z-index: 100;}

.logo {
    font-size: 24px;
    font-weight: bold;}

nav a {
    color: #FFF;
    text-decoration: none;
    margin: 0 15px;
    font-size: 16px;}

nav a:hover {
    color: #FFCA28; /* Жовтий акцент при наведенні */}

.btn {
    background-color: #FFCA28; /* Жовтий для кнопок */
    color: #AB47BC; /* Темно-фіолетовий текст на кнопках */
    padding: 8px 15px;
    border: none;
    border-radius: 5px;
    text-decoration: none;
    cursor: pointer;}

.btn:hover {
    background-color: #FFB300; /* Темніший жовтий при наведенні
*/}

.contact-info {
```

```
        font-size: 14px;}

/* Main */
main {
    padding: 20px;}

.hero {
    text-align: center;
    padding: 50px 20px;
    background-color: #CE93D8; /* Світліший фіолетовий для
геройської секції */
    color: #FFF;
    margin-bottom: 20px;
    border-radius: 10px;}

.hero h1 {
    font-size: 36px;
    margin: 0 0 20px;}

.hero p {
    font-size: 18px;
    max-width: 600px;
    margin: 0 auto 20px;}

/* Форма запису */
#booking {
    background-color: #EDE7F6; /* Дуже світлий фіолетовий для
форми */
    padding: 20px;
    border-radius: 10px;
    max-width: 500px;
    margin: 0 auto;}

#booking h2 {
```



```
        color: #AB47BC;
        text-align: center;}

form {
    display: flex;
    flex-direction: column;}

form input, form select {
    margin: 10px 0;
    padding: 10px;
    border: 1px solid #CE93D8;
    border-radius: 5px;}

form button {
    background-color: #FFCA28;
    color: #AB47BC;
    padding: 10px;
    border: none;
    border-radius: 5px;
    cursor: pointer;}

form button:hover {
    background-color: #FFB300;}

/* Footer */
footer {
    background-color: #AB47BC;
    color: #FFF;
    text-align: center;
    padding: 10px 1;
    margin-top: 20px;}
```

```
.social a {
    color: #FFCA28;
    margin: 0 10px;
    text-decoration: none;
    font-size: 20px;}

.social a:hover {
    color: #FFF;}

/* Адаптивність */
@media (max-width: 600px) {
    header {
        flex-direction: column;
        text-align: center;
    }
    nav a {
        margin: 5px 0;
    }
    .hero h1 {
        font-size: 24px;
    }
    .hero p {
        font-size: 14px;
    }
}
```