

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»**

Навчально-науковий інститут атомної та теплової енергетики

Кафедра цифрових технологій в енергетиці

**КУРСОВА РОБОТА**

з дисципліни « Системи баз даних »  
(назва дисципліни)

на тему: «WEB-застосунок для ведення замовлень у барбершопі»

Студента групи ТР-12  
зі спеціальності 122 – «Комп’ютерні науки»

Крилова М.А.

(Прізвище ініціали)

Керівник доцент, к.е.н., доцент Сегеда І.В.

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Керівник Софієнко А. Ю.

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Кількість балів: \_\_\_\_\_ Оцінка: \_\_\_\_\_

Члени комісії

\_\_\_\_\_  
(підпис)

\_\_\_\_\_  
(вчене звання, науковий ступінь, прізвище та ініціали)

\_\_\_\_\_  
(підпис)

\_\_\_\_\_  
(вчене звання, науковий ступінь, прізвище та ініціали)

\_\_\_\_\_  
(підпис)

\_\_\_\_\_  
(вчене звання, науковий ступінь, прізвище та ініціали)

Київ - 2023 рік

**Національний технічний університет України  
«Київський політехнічний інститут ім. Ігоря Сікорського»**

**НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ АТОМНОЇ ТА ТЕПЛОВОЇ ЕНЕРГЕТИКИ**

**Кафедра ЦИФРОВИХ ТЕХНОЛОГІЙ В ЕНЕРГЕТИЦІ**

Рівень вищої освіти перший (бакалаврський)

За освітньою програмою «Цифрові технології в енергетиці»

Спеціальності 122 Комп'ютерні науки

**З А В Д А Н Н Я  
НА КУРСОВУ РОБОТУ СТУДЕНТУ**

Крильову Микиті Анатолійовичу

(прізвище, ім'я, по батькові)

1. Тема роботи « WEB-застосунок для ведення замовлень у барбершопі»

керівник курсової роботи - Софієнко Антон Юрійович

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

2. Строк подання студентом роботи 23 грудня 2023 року

3. Вихідні дані до проекту (роботи): середовище розробки – мова JavaScript, інтерфейс користувача – HTML/CSS, сервер – Express.js

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Введення, проектування, розробка, висновки.

**КАЛЕНДАРНИЙ ПЛАН**

№ з/п	Назва етапів виконання дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1.	Вибір та затвердження теми роботи	10.09.2023	
2.	Вивчення предметної області	20.09.2023	
3.	Проектування бази даних	30.09.2023	
4.	Розробка та заповнення бази даних	25.10.2023	
5.	Написання запитів відповідно до завдання	12.11.2023	
6.	Розробка візуалізації програмного продукту	30.11.2023	
7.	Оформлення пояснювальної записки	01.12.2023-20.12.2023	

Студент

( підпис )

Керівник курсової роботи

( підпис )

**Крильов М.А.**

(прізвище та ініціали)

**Софієнко А. Ю.**

(прізвище та ініціали)

## АНОТАЦІЯ

Ця робота має на меті розробку ефективної системи для автоматизації процесів управління та обслуговування клієнтів у сфері барбершопів. Дослідження включає глибокий аналіз потреб ринку, розробку бази даних та веб-інтерфейсу, який забезпечує інтуїтивне управління записами, замовленнями та персональною інформацією клієнтів. Особлива увага приділяється інтеграції функціональності знижок і акцій, що сприяє підвищенню лояльності клієнтів. Робота включає детальну документацію процесів розробки, використання сучасних веб-технологій, та аналіз ефективності впровадження системи.

Для досягнення поставленої мети необхідно виконати наступні завдання:

- виконати проєктування бази даних та програмного забезпечення;
- виконати розробку бази даних і програмного забезпечення;

В результаті виконання курсової була розроблена база даних для зберігання даних записів барбершопу, а також програмне забезпечення для візуалізації та взаємодії з базою даних.

Пояснювальна записка на курсовий проєкт включає 47 сторінок, ілюстрована 26 рисунками, містить 1 таблицю та використовує 10 джерел літератури.

## ANNOTATION

This thesis aims to develop an effective system for automating management and customer service processes in barbershops. The research includes a deep analysis of market needs, development of a database, and a web interface that provides intuitive management of bookings, orders, and client personal information. Special attention is given to integrating discount and promotion functionalities, enhancing customer loyalty. The work includes detailed documentation of development processes, use of modern web technologies, and analysis of the system implementation effectiveness.

The objectives include designing the database and software, and developing the database and software.

The outcome of this thesis is the development of a database for storing barbershop booking records and software for visualizing and interacting with this database.

The explanatory note for the course project includes 47 pages, is illustrated with 26 figures, contains 1 table, and references 10 sources of literature.

## ЗМІСТ

ВСТУП.....	6
1. АНАЛІЗ ВИМОГ.....	8
1.1 Постановка завдання.....	8
1.2 Розробка концептуальної моделі.....	10
1.3 Діаграма прецедентів.....	12
1.4 Аналіз засобів реалізації.....	14
2. РОЗРОБКА БАЗИ ДАНИХ.....	17
2.1 Опис моделі даних.....	17
2.2 Створення збережених процедур.....	19
2.3 Побудова представлень та курсорів.....	25
2.4 Створення тригера та курсора.....	28
3. ІНТЕРФЕЙС КОРИСТУВАЧА.....	31
3.1 Вибір засобу реалізації.....	31
3.2 Робота користувача з системою.....	33
3.3 Взаємодія адміністратора з системою.....	34
3.4 Взаємодія клієнта з системою.....	36
ВИСНОВКИ.....	42
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	43
ДОДАТОК А.....	44

## ВСТУП

В сучасному світі, де конкуренція у сфері послуг краси та догляду за собою постійно зростає, наявність ефективної та добре структурованої системи управління клієнтською базою стає ключовим фактором успіху для барбершопу. Відповідно до специфіки барбершопу як закладу, який спеціалізується на чоловічих стрижках, моделюванні борід і вусів, необхідно розробити систему, яка б враховувала унікальні потреби та побажання клієнтів.

Розробка спеціалізованої бази даних для барбершопу не лише сприяє автоматизації бізнес-процесів, але й значно підвищує ефективність взаємодії з клієнтами. Саме тому темою цієї курсової роботи було обрано розробку Web-додатку для автоматизації роботи барбершопу.

Метою дослідження є створення інтегрованої бази даних для барбершопу, що дозволить вести облік клієнтів, видів стрижок, їх вартості, а також забезпечувати ефективне управління інформацією про послуги та знижки для постійних клієнтів. Завданнями дослідження є:

- Аналіз потреб бізнес-процесів барбершопу.
- Розробка структури бази даних для задоволення цих потреб.
- Реалізація механізмів запису та відслідковування історії обслуговувань клієнтів.
- Впровадження системи знижок та лояльності для постійних клієнтів.

Предмет дослідження охоплює процеси збору, обробки, та зберігання інформації в контексті управління барбершопом, в той час як об'єкт дослідження представляє інформаційну систему управління. Проблематика дослідження охоплює виклики, пов'язані з ефективним проектуванням бази даних, розробкою функціоналу для управління клієнтською базою, історією замовлень, та розробкою програм лояльності. Сприяння оптимізації робочих процесів, підвищення якості

обслуговування, та зростання ефективності бізнесу стають важливими цілями розробки.

У рамках курсової роботи, проблемні питання, які будуть розглядатися, є ключовими для розробки ефективної бази даних для барбершопу.

По-перше, значну увагу буде приділено ефективному проектуванню бази даних, яка має бути спеціально адаптована до унікальних потреб та вимог сфери барбершопу. Це включає оптимізацію структури даних, забезпечуючи швидкий доступ до інформації та легкість управління даними.

Розробка функціоналу для управління клієнтською базою та історією замовлень є другим важливим аспектом дослідження. Важливо створити інтуїтивно зрозумілі та ефективні інструменти для ведення записів про клієнтів, їх замовлення та переваги, що дозволить барбершопу вести історію взаємодії з кожним клієнтом, підвищуючи рівень персоналізованого обслуговування.

Третім важливим аспектом є створення механізмів для програм лояльності та знижок. Розробка систем, які дозволять винагороджувати постійних клієнтів та стимулювати нових відвідувачів, є критичною для підвищення задоволеності клієнтів та сприяння їхній лояльності.

У роботі буде використане сучасне програмне забезпечення для розробки та тестування бази даних, а структура роботи включатиме теоретичну частину, практичну розробку, тестування та аналіз отриманих результатів.

# 1. АНАЛІЗ ВИМОГ

Аналіз вимог є фундаментальним кроком у процесі розробки будь-якого програмного продукту. Цей етап передбачає глибоке занурення в потреби і очікування користувачів, а також визначення технічних і бізнес-параметрів проекту. У цьому контексті, аналіз вимог для веб-застосунку барбершопу слугує не тільки для виявлення ключових характеристик і функціоналу, які має включати система, але й для встановлення чітких напрямків розробки, які забезпечать створення ефективного і зручного у використанні продукту.

На цьому етапі особливу увагу приділяється детальному опису кожної вимоги, її важливості та впливу на кінцевого користувача. Аналіз вимог охоплює не тільки технічні аспекти, такі як вибір мови програмування, баз даних, інструментів розробки, але й досліджує бізнес-процеси, логіку роботи застосунку та його взаємодію з користувачами. Такий підхід дозволяє забезпечити, що розроблений продукт буде не тільки технологічно ефективним, але й максимально відповідатиме потребам і очікуванням його кінцевих користувачів.

## 1.1 Постановка завдання

У сучасному світі, де час є найціннішим ресурсом, ефективність і зручність в управлінні повсякденними справами стають важливим пріоритетом. Ця необхідність породжує потребу в розробці веб-застосунку для барбершопу, спрямованого на спрощення процесу ведення замовлень та планування візитів.

Завданням данної курсової роботи була розробка WEB застосунку для ведення замовлень барбершопу, що дозволить користувачам переглядати доступних барберів, послуги, ціни, і робити онлайн-запис на стрижку або інші послуги. Сайт має бути інтуїтивно зрозумілим і зручним у використанні.



Основні функції, що має виконувати розроблена програма:

- Реєстрація/авторизація клієнтів: Клієнти можуть створювати обліковий запис, увійти в систему, редагувати свої профілі.
- Перегляд барберів і Послуг: Показати інформацію про барберів, їх спеціалізацію, робочі години, та послуги, які вони надають.
- Календар записів: Календар, де клієнти можуть бачити доступні часові слоти та записуватися на вільний час.
- Управління записами: Клієнти можуть переглядати, редагувати або скасовувати свої записи.
- Пошук за послугами: Фільтрація барберів та їх доступності за обраними послугами.
- Адміністрування: Панель для адміністраторів для управління записами, послугами, барберами, та відгуками.
- Звітність: Створення звітів про записи, доходи та активність клієнтів для власників барбершопу.
- Технічні Вимоги
- Мова програмування: Використовувати технології для фронтенду (наприклад, HTML, React, Vue.js) та бекенду (наприклад, Node.js, Ruby on Rails).
- База даних: Вибрати надійну базу даних, яка підтримує транзакції та має хорошу підтримку скалярності (наприклад, PostgreSQL, MySQL).
- Адаптивний дизайн: Сайт має бути оптимізований для роботи на різних пристроях, включаючи мобільні телефони та планшети.
- Вимоги до інтерфейсу
- Простота та інтуїтивність: Інтерфейс має бути простим та інтуїтивно зрозумілим для всіх категорій користувачів.

## 1.2 Розробка концептуальної моделі

Концептуальна модель бази даних для системи управління барбершоп "Barbershop". Основна мета цієї моделі - забезпечення ефективного зберігання, доступу та обробки інформації, необхідної для повсякденного функціонування та управління барбершопом.

У рамках розробки бази даних для Web-застосунку барбершопу було сформовано наступну структуру даних, яка відповідає технічному завданню та оптимізує процеси управління та обслуговування клієнтів.

Таблиця `barber_categories`: ця сутність слугує для класифікації барберів за різними категоріями. Кожен запис у цій таблиці складається з унікального ідентифікатора категорії (`category_id`), назви категорії (`category_name`), та націнки (`markup`). Таблиця `barbers` асоціюється з цією таблицею через зовнішній ключ `category_id`, що дозволяє зв'язати барберів з їх відповідними категоріями.

Таблиця `barbers`: центральна таблиця для зберігання інформації про барберів, включаючи особисті та контактні дані. Ця таблиця зв'язується з `barber_categories` через `category_id` та з `appointments` через `barber_id`, формуючи основу для управління персоналом і їх розкладами.

Таблиця `services` та `service_prices`: `services` описує перелік послуг, які пропонуються, тоді як `service_prices` відслідковує ціноутворення на ці послуги. Обидві таблиці асоційовані через `service_id`, що дозволяє динамічно управляти цінами на послуги.

Таблиця `clients`: містить детальну інформацію про клієнтів, включаючи дані про кількість відвідувань та отримані знижки. Вона взаємодіє з `appointments` через `client_id`, що відіграє ключову роль у відстеженні відвідувань клієнтів та їхнього взаємодії з послугами.

Таблиця `appointments`: життєво важлива для організації робочого процесу барбершопу, оскільки зберігає всю інформацію про записи клієнтів, включаючи дату,

час, статус, відповідного барбера та клієнта. Ця таблиця формує зв'язки між clients, barbers, та services.

Таблиця coupons: забезпечує зберігання даних про різноманітні купони на знижки, які можуть використовуватися клієнтами для економії на послугах.

Таблиця administrators: містить інформацію про адміністраторів барбершопу, ключових учасників в управлінні різними аспектами діяльності бізнесу.

Ця структура бази даних забезпечує комплексний підхід до обробки та управління інформацією, що є критично важливим для ефективного функціонування барбершопу. На основі створеної концептуальної моделі було сформовано бізнес правила до бази даних.:

- Номер телефону клієнта повинен починатись з 380.
- Імейл повинен містити символ @
- Клієнт може мати одне або більше записів, але записів не можуть
- бути призначені для більше, ніж одного клієнта.
- Кожен записів повинен бути пов'язаний з з барбером, до якого зроблено запис.
- Записи можуть мати різні статуси, такі як "заплановано", "виконано" або "скасовано".
- У кожного клієнта може бути дисконтна карта, яка надає знижки на товари або послуги.
- У кожного клієнта повинні бути збережені його персональні дані, такі як його повне ім'я, електронна адреса, номер телефону.

### 1.3 Діаграма прецедентів

Проаналізувавши вимоги до програмного забезпечення вдалося сформулювати основний перелік функцій, що має виконувати програма. Варіанти використання і функції що виконуються програмою можна в повній мірі продемонструвати за допомогою рисунку 1.1 – діаграми прецедентів.

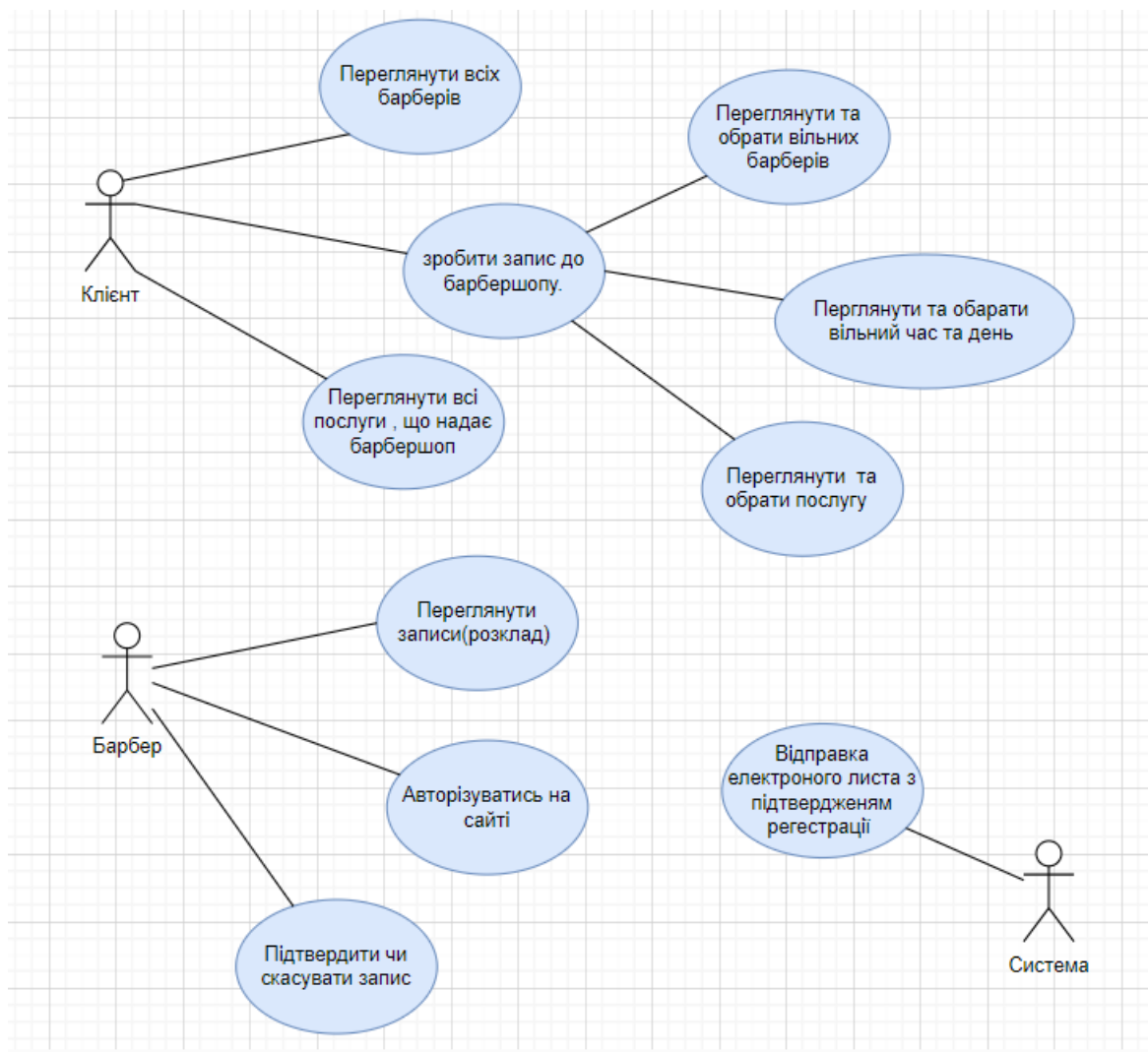


Рисунок 1.1 – Діаграма прецедентів

Надалі наведемо опис основних прецедентів на діаграмі:

- Переглянути всіх барберів: клієнт має можливість переглядати список усіх барберів, що працюють у барбершопі.
- Зробити запис до барбершопу: клієнт може забронювати час для візиту до барбершопу.
- Переглянути всі послуги, що надає барбершоп: клієнт може переглянути перелік послуг, які пропонує барбершоп.
- Переглянути та обрати вільних барберів: клієнт може вибрати барбера зі списку тих, хто доступний у вибраний час.
- Переглянути та обрати вільний час та день: клієнт може переглянути та обрати вільну дату та час для запису.
- Переглянути та обрати послугу: клієнт може вибрати одну або декілька послуг з переліку.
- Авторизуватися на сайті: барбер може увійти в систему для доступу до свого розкладу та записів.
- Переглянути записи (розклад): барбер може переглянути свій розклад на певний день.
- Підтвердити чи скасувати запис: система обробляє запити на підтвердження або скасування записів.
- Відправка електронного листа з підтвердженням реєстрації: після реєстрації або зміни запису система автоматично відправляє клієнту електронний лист з підтвердженням.

Ця діаграма допомагає визначити функціональні вимоги до системи та є основою для подальшої розробки системи бронювання. Вона також визначає ключові взаємодії між користувачами та системою.

## 1.4 Аналіз засобів реалізації

База даних це спосіб зберігання різних даних в одному місці. Усередині бази можуть зберігатися різні дані: фото, текст, музика, числа, код, посилання, ціни тощо. Коли говорять про бази даних, найчастіше мають на увазі табличні бази даних – ті, де інформація зберігається у різних таблицях. Приклад табличної бази — MySQL. Вона багато вміє, до неї написано багато документації та правил, тому починають зазвичай із неї. Щоб керувати даними в базі, наприклад, додавати нові записи, видаляти старі або щось шукати, використовують спеціальну мову запитів до бази — SQL. Ці запити опрацьовує СУБД — система управління базами даних. Це як двигун для сайтів - він виконує запити, працює з базою і надає результати.

MySQL — це потужна система керування базами даних, яка забезпечує ефективну роботу з великими обсягами інформації і підтримує широкий спектр функціональних можливостей. Вона є відмінним вибором для розробників і адміністраторів баз даних завдяки своїй надійності, швидкості обробки даних та гнучкості.

MySQL підтримує всі основні функції, які можна очікувати від сучасної СУБД:

- Робота з різноманітними запитами та високий рівень оптимізації SQL-запитів.
- Підтримка різних типів зв'язків між таблицями.
- Зберігання даних у різноманітних форматах.
- Можливість створення складних запитів та обробки транзакцій.
- Розширені функції, включаючи тригери, процедури, керування кешем тощо.
- Стабільність та Надійність

MySQL відома своєю стабільністю і надійністю. Вона використовується у великих проектах і системах, де важлива висока продуктивність і масштабованість. Система забезпечує стабільну роботу навіть при великому навантаженні та у складних умовах, включаючи:

- Обробку великих обсягів даних.
- Високу конкурентність доступу до даних.
- Роботу на потужному серверному обладнанні.
- Підтримка різних мов програмування

MySQL сумісна з багатьма мовами програмування, включаючи:

- Python,
- PHP,
- Java,
- C++,
- C#,
- Ruby,
- Perl.

MySQL є основою для багатьох відомих веб-додатків, CMS (систем управління контентом) та інших сервісів, наприклад:

WordPress, Joomla, Drupal, Facebook, Twitter, YouTube.

Ці характеристики роблять MySQL однією з найпопулярніших систем управління базами даних, яка підходить як для маленьких проектів, так і для великих корпоративних систем.

Порівняння альтернативних баз даних наведено в таблиці 1.1

Таблиця 1.1 – Порівняння альтернативних баз даних до MySQL

База Даних	Підтримка ОС	Реплікація та Синхронізація	Масштабованість	Підтримка Транзакцій	Тип Ліцензії	Мінімальний Розмір	Особливості	Компанія Розробник
PostgreSQL	Linux, Windows, macOS, BSD, Solaris	Synchronous, Asynchronous	Горизонтальна та Вертикальна	Так (ACID)	Відкритий код (Ліцензія PostgreSQL)	Залежить від розгортання	Extensive, JSONB, HStore, GIS	PostgreSQL Global Development Group
MariaDB	Linux, Windows, macOS	Master-slave, Galera cluster	Горизонтальна та Вертикальна	Так (ACID)	Відкритий код (GPLv2)	Залежить від розгортання	JSON, GIS, Temporal Tables	MariaDB Corporation Ab, MariaDB Foundation
MySQL	Linux, Windows, macOS, Solaris	Master-slave, master-master	Горизонтальна та Вертикальна	Так (ACID)	Відкритий код (GPLv2)	Залежить від розгортання	JSON, GIS	Oracle Corporation
Oracle Database	Windows, Linux, Solaris, HP-UX, AIX	Active Data Guard, GoldenGate	Вертикальна	Так (ACID)	Приватна	Потребує значних ресурсів	PL/SQL, Java, XML DB, JSON	Oracle Corporation
Amazon RDS	Керована послуга, абстракція ОС	Multi-AZ, Read Replicas	Горизонтальна та Вертикальна	Так (ACID)	Комерційна	Залежить від розміру інстанції	Managed, Automated backups, Monitoring	Amazon Web Services



## **2. РОЗРОБКА БАЗИ ДАНИХ**

У рамках даного розділу здійснюється глибоке дослідження та аналіз процесу розробки бази даних для веб-застосунку барбершопу. Ключовим аспектом є ретельне проектування структури бази даних, яке передбачає визначення оптимальної схеми для ефективного зберігання, доступу та обробки інформації. Особливу увагу приділено процесам нормалізації, які спрямовані на досягнення третьої нормальної форми. Це забезпечує мінімізацію редундантності даних і підвищення ефективності бази даних. Ретельно спланована структура бази даних є фундаментом для створення надійного і масштабованого веб-застосунку, який здатен задовольнити потреби користувачів та управлінські вимоги бізнесу.

Додатково в цьому розділі представлено візуальне відображення логічної структури бази даних, що дає змогу оцінити повноту та комплексність взаємозв'язків між різними елементами даних. Визначення та реалізація збережених процедур, тригерів та курсорів, що відіграють ключову роль у маніпулюванні та обробці даних, є важливими кроками у забезпеченні гнучкості та ефективності роботи системи. Це не лише підвищує продуктивність запитів до бази даних, але й забезпечує високий рівень інтеграції та автоматизації в управлінні бізнес-процесами барбершопу.

### **2.1 Опис моделі даних**

В процесі проектування було виконано ключові етапи нормалізації для досягнення третьої нормальної форми, що забезпечує ефективне зберігання даних без надмірної редундантності. Зокрема, були створені проміжні таблиці для реалізації зв'язків "багато до багатьох", виділені відповідні атрибути в окремі таблиці та встановлені первинні та зовнішні ключі для забезпечення цілісності даних.

На рисунку 2.1 представлено візуальне відображення логічної структури бази даних, що включає схему зв'язків між таблицями та атрибутами, що дозволяє краще уявити структуру та взаємозв'язки в базі даних.

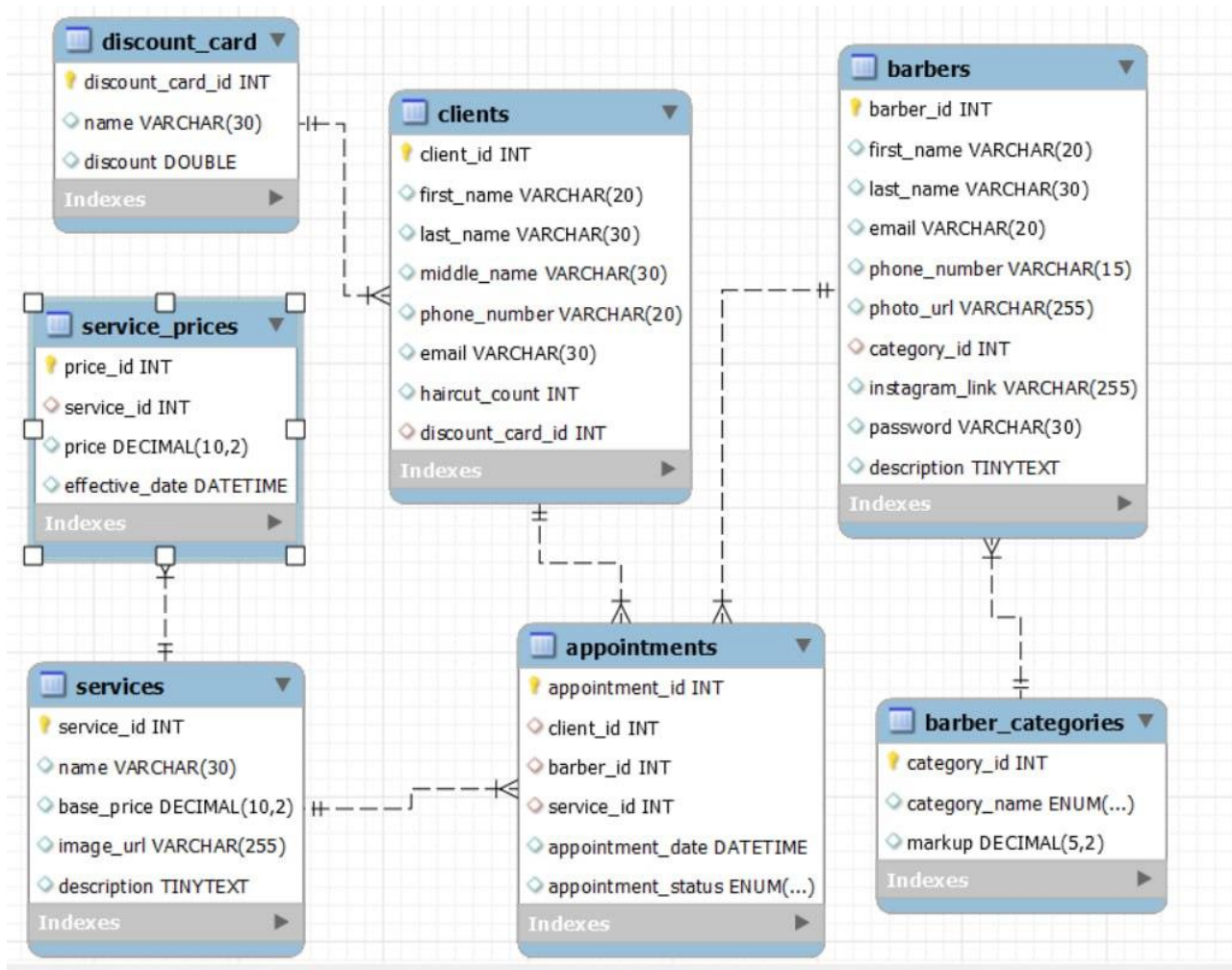


Рисунок 2.1 – Модель бази даних

Розглянемо детальніше структуру та функції ключових таблиць бази даних, які є фундаментом для ефективного управління інформаційною системою барбершопу:

## 2.2 Створення збережених процедур

У процесі розробки комплексної бази даних для веб-застосунку барбершопу, особливе місце займає створення та імплементація збережених процедур. Ці процедури є критичними компонентами, що сприяють досягненню високого рівня функціональності, ефективності та автоматизації в обробці даних. Вони виконують важливу роль у забезпеченні гнучкості системи, дозволяючи виконувати складні операції з даними за допомогою визначених інструкцій, вбудованих безпосередньо у саму базу даних. Цей підхід значно підвищує продуктивність застосунку, знижуючи навантаження на сервер та оптимізуючи взаємодію з даними.

У наступних розділах буде детально представлений опис розроблених процедур, кожна з яких розроблена з метою вирішення конкретних задач в контексті управління барбершопом. Від відображення цін за категоріями барберів до планування графіків прийомів, від автоматичного оновлення статусів записів до створення звітів і аналізу даних - кожна процедура сприяє збільшенню ефективності бізнес-процесів і поліпшенню користувацького досвіду. Опис цих процедур дасть змогу глибше зрозуміти їхній функціонал і роль у загальній архітектурі бази даних.

На рисунку 2.2 представлено о процедуру `GetPricesByCategory`, що була створена для реалізації та відображення ціни для конкретної категорії барбера.

```
DELIMITER //
```

```
CREATE PROCEDURE GetPricesByCategory(IN barberCategory VARCHAR(255))  
BEGIN  
    SELECT  
        s.service_id,  
        s.name,  
        s.description,  
  
        (s.base_price * (1 + bc.markup / 100)) AS price_with_markup  
    FROM  
        services s  
    JOIN  
        barber_categories bc ON bc.category_name = barberCategory;  
END //
```

```
DELIMITER ;
```

Рисунок 2.2 – «Процедура GetPricesByCategory»

Процедура приймає назву категорії барбера (barberCategory) та повертає ціни на послуги для цієї категорії. Вона виконує JOIN між таблицями services та barber\_categories та обчислює ціну з урахуванням націнки, визначеної для категорії.

Для відображення вільних чахсович слотів до певного барбера було створено процедуру GetAvaibleTimes, реалізацію якої продемонстровано на рисунку 2.3.

```

DELIMITER //

CREATE PROCEDURE GetAvailableTimes(barberID INT, appointmentDate DATE)
BEGIN
    DECLARE startTime TIME DEFAULT '10:00:00';
    DECLARE endTime TIME DEFAULT '22:00:00';
    DECLARE currentTime TIME;
    CREATE TEMPORARY TABLE IF NOT EXISTS AllTimes (timeSlot TIME);

    SET currentTime = startTime;
    WHILE currentTime < endTime DO
        INSERT INTO AllTimes (timeSlot) VALUES (currentTime);
        SET currentTime = ADDTIME(currentTime, '01:00:00');
    END WHILE;

    SELECT a.timeSlot
    FROM AllTimes a
    LEFT JOIN appointments ap ON a.timeSlot = TIME(ap.appointment_date)
        AND ap.barber_id = barberID
        AND DATE(ap.appointment_date) = appointmentDate
    WHERE ap.appointment_id IS NULL;
    DROP TEMPORARY TABLE IF EXISTS AllTimes;
END //
DELIMITER ;

```

Рисунок 2.3 – Процедура «GetAvaibleTimes»

Процедура приймає два параметра – ідентифікатор певного барбера (barberID) та дату яку вибрали (appointmentDate). Вона створює тимчасову таблицю з усіма можливими часовими слотами протягом робочого дня та потім вибирає ті часові слоти, які не перетинаються з уже запланованими записами.

На рисунку 2.4 зображено процедуру GetAppointmentsForBarberOnDate. Ця процедура повертає у вигляді таблиць всі заплановані дати до певного барбера.

```

CREATE PROCEDURE GetAppointmentsForBarberOnDate(IN barberID INT, IN appointmentDate DATE)
BEGIN
    SELECT
        a.appointment_date AS Time,
        c.first_name AS FirstName,
        c.last_name AS LastName,
        c.phone_number AS PhoneNumber,
        c.email AS Email,
        s.name AS ServiceName
    FROM
        appointments a
    JOIN
        clients c ON a.client_id = c.client_id
    JOIN
        services s ON a.service_id = s.service_id
    WHERE
        a.barber_id = barberID AND
        DATE(a.appointment_date) = appointmentDate AND
        a.appointment_status = 'scheduled'
    ORDER BY
        a.appointment_date;
END //

DELIMITER ;

```

Рисунок 2.4 – Процедура «GetAppointmentsForBarberOnDate»

Ця процедура повертає всі заплановані записи на певну дату (appointmentDate) для певного барбера (barberID). Вона об'єднує таблиці appointments, clients та services для отримання детальної інформації про кожен запис.

На рисунку 2.5 продемонстровано процедуру UpdateAppointmentStatus, що представляє собою ключову компоненту системи управління записами в барбершопу, що дозволяє з легкістю керувати статусами запланованих зустрічей. Ця процедура була спеціально розроблена для забезпечення гнучкості та ефективності при взаємодії з базою даних барберів, що включає в себе широкий спектр послуг та численних фахівців. код процедури.

```

CREATE PROCEDURE UpdateAppointmentStatus(IN appointmentID INT, IN newStatus
ENUM('scheduled', 'completed', 'cancelled'))
BEGIN
    UPDATE appointments
    SET appointment_status = newStatus
    WHERE appointment_id = appointmentID;
END //

DELIMITER ;

```

Рисунок 2.5 – Процедура «UpdateAppointmentStatus»

Функціональність процедури орієнтована на оновлення поля `appointment_status` в таблиці `appointments`, яка зберігає інформацію про всі заплановані зустрічі. Важливістю цього механізму є його здатність до реагування на динамічні зміни в графіку роботи барбершопу, такі як скасування, перенесення часу зустрічі, або підтвердження завершення надання послуги.

Процедура `CreateClientAndAppointment` - це гнучкий та автоматизований інструмент, що інтегрований у систему управління барбершопу, спрямований на оптимізацію процесу запису клієнтів на послуги. На рисунку 2.6 продемонстровано вищеописану процедуру.

```

DELIMITER //
CREATE PROCEDURE CreateClientAndAppointment(
    IN p_first_name VARCHAR(255),

    IN p_phone_number VARCHAR(20),
    IN p_email VARCHAR(255),
    IN p_barber_id INT,
    IN p_service_id INT,
    IN p_appointment_date DATETIME,
    IN p_appointment_status ENUM('scheduled', 'completed', 'cancelled')
)
BEGIN
    DECLARE v_client_id INT;

    SELECT client_id INTO v_client_id FROM clients WHERE phone_number = p_phone_number LIMIT 1;

    IF v_client_id IS NULL THEN
        INSERT INTO clients (first_name, phone_number, email)
        VALUES (p_first_name, p_phone_number, p_email);

        SET v_client_id = LAST_INSERT_ID(); -- Get the ID of the newly inserted client
    END IF;

    INSERT INTO appointments (client_id, barber_id, service_id, appointment_date, appointment_status)
    VALUES (v_client_id, p_barber_id, p_service_id, p_appointment_date, p_appointment_status);
END //

DELIMITER ;

```

Рисунок 2.6 - Процедура «CreateClientandApointment»

Ця процедура вирішує дві ключові задачі: реєстрацію нових клієнтів та негайне планування їхніх візитів.

- **Верифікація Існуючих Клієнтів:** Процедура починається з перевірки наявності клієнта у базі даних, використовуючи номер телефону як унікальний ідентифікатор, вона шукає існуючі записи у таблиці `clients`. Це дозволяє уникнути дублювання інформації та забезпечує консистентність даних.
- **Реєстрація Нових Клієнтів:** Якщо клієнт не знайдений у базі даних, процедура автоматично реєструє нового клієнта, додаючи його основну інформацію, таку як ім'я, номер телефону та електронна адреса, до таблиці `clients`. Це покращує користувацький досвід, забезпечуючи швидке та зручне обслуговування нових клієнтів.
- **Планування Записів:** Незалежно від того, чи є клієнт новим, чи існуючим, процедура продовжує процес, створюючи запис на послугу. Вона використовує передані параметри, такі як ID барбера (`barber_id`), ID послуги (`service_id`), дату та час візиту (`appointment_date`), та статус запису (`appointment_status`), для створення детального запису в таблиці `appointments`.

Процедура `GetAvailableBarbersForDay` є витонченим інструментом у системі управління барбершопом, призначеним для полегшення процесу планування та оптимізації розподілу робочого навантаження серед персоналу. Ця процедура ефективно вирішує проблему ідентифікації доступних барберів для конкретної дати, спираючись на дані запланованих записів. На рисунку 2.7 продемонстровано код процедури.

Процедура повертає список барберів, які доступні для запису на обрану дату (`selectedDate`). Вона використовує тимчасову таблицю для визначення вільних часових слотів та перевіряє які барбери не мають запланованих записів на цю дату. Для ефективного визначення вільних часових інтервалів використовується тимчасова таблиця, яка містить усі можливі часові слоти протягом робочого дня. Це забезпечує динамічний та гнучкий підхід до планування.



```

DELIMITER //
CREATE PROCEDURE GetAvailableBarbersForDay(IN selectedDate DATE)
BEGIN
    DECLARE startTime TIME DEFAULT '10:00:00';
    DECLARE endTime TIME DEFAULT '22:00:00';
    DROP TEMPORARY TABLE IF EXISTS AllDaySlots;
    CREATE TEMPORARY TABLE AllDaySlots (timeSlot TIME);
    WHILE startTime < endTime DO
        INSERT INTO AllDaySlots (timeSlot) VALUES (startTime);
        SET startTime = ADDTIME(startTime, '01:00:00');
    END WHILE;
    SELECT
        b.barber_id,
        b.first_name,
        bc.category_name,
        b.photo_url
    FROM barbers b
    JOIN barber_categories bc ON b.category_id = bc.category_id
    WHERE
        b.barber_id NOT IN (
            SELECT a.barber_id
            FROM appointments a
            WHERE DATE(a.appointment_date) = selectedDate
            AND a.appointment_status = 'scheduled'
            AND TIME(a.appointment_date) BETWEEN '10:00:00' AND '22:00:00'
        )
    GROUP BY b.barber_id;
    DROP TEMPORARY TABLE IF EXISTS AllDaySlots;
END //

```

Рисунок 2.7 – Процедура «GetAvailableBarbersForDay»

## 2.3 Побудова представлень та курсорів

Представлення — це віртуальні таблиці, які надають можливість структурованого і безпечного доступу до даних. Вони уможливають оптимізацію запитів і можуть слугувати для забезпечення рівня безпеки, приховуючи складні запити за простим інтерфейсом.

В даній курсовій роботі було створено 2 представлення для досягнення поставлених цілей та коректного відображення інформації в застосунку. На малюнках 2.8-2.9. буде наведено ці представлення з детальним описом.

Представлення `services_with_markup` розроблено для показу цін на послуги з урахуванням націнки, що залежить від категорії барбера.

Цей запит використовує CROSS JOIN для нарахування націнки на базову ціну послуги відповідно до категорії барбера. Представлення дозволяє вивести різні ціни для кожної категорії барбера.

```
CREATE VIEW services_with_markup AS
SELECT
    s.image_url,
    s.name,
    s.description,

    MAX(CASE WHEN bc.category_name = 'trainee' THEN s.base_price * (1 + bc.markup / 100) ELSE NULL END) AS trainee_price,
    MAX(CASE WHEN bc.category_name = 'barber' THEN s.base_price * (1 + bc.markup / 100) ELSE NULL END) AS barber_price,
    MAX(CASE WHEN bc.category_name = 'senior_barber' THEN s.base_price * (1 + bc.markup / 100) ELSE NULL END) AS senior_barber_price,
    MAX(CASE WHEN bc.category_name = 'expert' THEN s.base_price * (1 + bc.markup / 100) ELSE NULL END) AS expert_price
FROM
    services s
CROSS JOIN
    barber_categories bc
GROUP BY
    s.service_id, s.name, s.description, s.image_url, s.base_price;
```

Рисунок 2.8 – Представлення для ціни сервісу в залежності від категорії барбера

Представлення barber\_details створено для забезпечення швидкого доступу до детальної інформації про барберів, їх фото, імена, категорії та опис послуг. На рисунку 2.9 відображено використання процедури.

image_url	name	description	trainee_price	barber_price
url_to_image_standard.jpg	Standard Haircut	A standard haircut for a fresh look.	110.00000000	120.00000000
url_to_image_deluxe.jpg	Deluxe Haircut	A deluxe haircut with additional styling services.	220.00000000	240.00000000
./img/iconService/iconService (1).png	Сервіс 1	Опис сервісу 1	12.08900000	13.18800000
./img/iconService/iconService (2).png	Сервіс 2	Опис сервісу 2	14.28900000	15.58800000
./img/iconService/iconService (3).png	Сервіс 3	Опис сервісу 3	16.48900000	17.98800000
./img/iconService/iconService (4).png	Сервіс 4	Опис сервісу 4	17.58900000	19.18800000

Рисунок 2.9 – Результат використання процедури «services\_with\_markup»

```

CREATE VIEW barber_details AS
SELECT
    b.barber_id ,
    b.photo_url,

    b.first_name,

    bc.category_name,
    b.description,

    b.instagram_link
FROM barbers b
JOIN barber_categories bc ON b.category_id = bc.category_id;

```

Рисунок 2.9 Представлення інформації про барбера

У рамках розробки інформаційної системи для барбершопу було здійснено ключовий крок у вдосконаленні управління даними: створено інтегроване представлення барберів шляхом об'єднання інформації з двох основних таблиць - barbers та barber\_categories. Це представлення не тільки підвищує ефективність доступу до даних, але й забезпечує більш зручне та гнучке використання інформації в повсякденних операціях барбершопу. Результат роботи представлення продемонстровано на рисунку 2.10.

barber_id	photo_url	first_name	category_name	description	instagram_link
1	NULL	Alex	trainee	NULL	NULL
5	./img/BarberPage/backgrBarber1.png	Ім'я1	trainee	NULL	NULL
9	./img/BarberPage/backgrBarber5.png	Ім'я5	trainee	NULL	NULL
2	NULL	Chris	barber	NULL	NULL
6	./img/BarberPage/backgrBarber2.png	Ім'я2	barber	NULL	NULL
3	NULL	Jordan	senior_barber	NULL	NULL
7	./img/BarberPage/backgrBarber3.png	Ім'я3	senior_barber	NULL	NULL
4	NULL	Taylor	expert	NULL	NULL

Рисунок 2.10 – Результат використання процедури «barber\_details»

## 2.4 Створення тригера та курсора

У цій роботі особливу увагу приділено використанню тригера, який є ключовим компонентом у процесі автоматизації та ефективності системи управління базою даних барбершопу. Тригери в базі даних виконують важливу роль, автоматично ініціюючи певні дії у відповідь на конкретні події, що значно підвищує продуктивність та надійність системи.

Тригер `AfterAppointmentUpdate` представляє собою важливу частину системи управління барбершопом, що виконує критично важливу роль у автоматизації обробки даних про зустрічі клієнтів. Цей тригер активується після кожного оновлення запису в таблиці `appointments` і виконує ряд дій, заснованих на статусі зустрічі. Основна функція тригера полягає в тому, щоб реагувати на зміну статусу зустрічі на `'completed'`, що вказує на успішне завершення запланованої стрижки або іншої послуги.

На рисунку 2.11 відображено тригер «`AfterAppointmentUpdate`».

```
CREATE TRIGGER AfterAppointmentUpdate
AFTER UPDATE ON appointments
FOR EACH ROW
BEGIN
    DECLARE haircut_count_val INT DEFAULT 0;

    IF NEW.appointment_status = 'completed' THEN

        UPDATE clients SET haircut_count = haircut_count + 1
        WHERE client_id = NEW.client_id;

        SELECT haircut_count INTO haircut_count_val FROM clients WHERE client_id = NEW.client_id;

        IF haircut_count_val = 3 THEN
            UPDATE clients SET discount = 3 WHERE client_id = NEW.client_id;
        END IF;
    END IF;
END //

DELIMITER ;
```

Рисунок 2.11 – Тригера «`AfterAppointmentUpdate`»

Оновлення haircut\_count, коли статус зустрічі оновлюється до 'completed', тригер автоматично збільшує лічильник стрижок (haircut\_count) у таблиці clients для відповідного клієнта. Це забезпечує точне відстеження кількості послуг, наданих клієнту. Після оновлення лічильника стрижок, тригер перевіряє, чи досягнуто клієнтом певного порогу в даному випадку трьох стрижок, що може бути умовою для надання знижки. Ця функція важлива для впровадження програм лояльності та стимулювання постійних клієнтів.

Використання тригерів, таких як AfterAppointmentUpdate, демонструє прогресивний підхід до управління даними в сучасних інформаційних системах. Автоматизація рутинних процесів забезпечує високий рівень точності даних, підвищує ефективність робочих процесів та покращує загальний досвід користувача.

Для підвищення функціональності та зручності використання бази даних барбершопу була розроблена процедура ListAppointmentsForDate, продемонстрована на рисунку 2.12.

```
DELIMITER //
CREATE PROCEDURE ListAppointmentsForDate(IN inputDate DATE)
BEGIN
    DECLARE done INT DEFAULT FALSE;
    DECLARE a_appointment_id INT;
    DECLARE a_client_id INT;
    DECLARE a_barber_id INT;
    DECLARE a_service_id INT;
    DECLARE a_appointment_date DATETIME;
    DECLARE a_appointment_status VARCHAR(255);
    DECLARE curAppointments CURSOR FOR
        SELECT appointment_id, client_id, barber_id, service_id, appointment_date, appointment_status
        FROM appointments
        WHERE DATE(appointment_date) = inputDate;
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
    OPEN curAppointments;
    read_loop: LOOP
        FETCH curAppointments INTO a_appointment_id, a_client_id, a_barber_id, a_service_id, a_appointment_date, a_appointment_status;

        IF done THEN
            LEAVE read_loop;
        END IF;
        SELECT a_appointment_id, a_client_id, a_barber_id, a_service_id, a_appointment_date, a_appointment_status;
    END LOOP;
    CLOSE curAppointments;
END //
DELIMITER ;
```

Рисунок 2.12 – Процедура та курсор «ListAppointmentsForDate»

Ця процедура є витонченим інструментом у системі управління записами, дозволяючи адміністраторам та персоналу барбершопу ефективно відстежувати та аналізувати всі заплановані зустрічі на конкретну дату. Вона використовує курсор в MySQL для послідовного перебору та обробки кожного запису з таблиці appointments, що відповідає заданій даті. Ця процедура забезпечує систематичний підхід до відображення та управління розкладом, що є особливо важливим у контексті ефективного управління часом та ресурсами барбершопу.

Застосування курсорів у даній процедурі дозволяє точно та послідовно обробляти великі обсяги даних, забезпечуючи оптимальну продуктивність. При роботі з курсором curAppointments, відбувається послідовне перебирання кожного рядка в наборі результатів запиту.

Використання інструкції FETCH в рамках циклу курсора curAppointments дозволяє послідовно переходити до наступного рядка у наборі результатів, зберігаючи отримані дані у відповідні змінні. Кожен запис обробляється індивідуально, що дозволяє глибше аналізувати специфіку кожної зустрічі, вивчати їх характеристики та визначати тенденції. Таким чином, курсор curAppointments є не лише інструментом для читання даних, а й могутнім механізмом для їх аналізу, що забезпечує високу ефективність управління інформаційними ресурсами веб-застосунку.

### 3. ІНТЕРФЕЙС КОРИСТУВАЧА

У цьому розділі буде приділено увагу розробці та оптимізації інтерфейсу користувача для веб-застосунку барбершопу, що є ключовим аспектом забезпечення зручності та ефективності взаємодії користувачів із системою. Вибір технологій для реалізації цього інтерфейсу є вирішальним у створенні інтуїтивно зрозумілого, доступного та відповідного для користувачів середовища. Використання SQL для маніпулювання даними, Node.js для асинхронної обробки запитів, MySQL для надійного зберігання даних, Visual Studio для розробки та дебагінгу, а також HTML, CSS та JavaScript для створення користувацького інтерфейсу, забезпечує створення міцної та ефективної платформи для взаємодії з користувачами.

Цей розділ також детально висвітлює функціонал інтерфейсу користувача, розглядаючи різні сценарії взаємодії, включно з навігацією по сайту, переглядом доступних барберів та послуг, процесом запису на прийом, а також взаємодією адміністратора з системою. Ми демонструємо, як користувачі можуть легко обирати послуги, бронювати час, реєструватися та отримувати інформацію про свої записи, включно з підтвердженнями через електронну пошту. Всі ці аспекти важливі для створення приємного та зручного досвіду для користувачів, що сприяє підвищенню лояльності клієнтів та ефективності бізнес-операцій.

#### 3.1 Вибір засобу реалізації

SQL є ключовим для ефективної роботи з реляційними базами даних, забезпечуючи гнучкість і точність у запитах та управлінні даними. В контексті бази даних барбершопу, де потрібно обробляти інформацію про категорії барберів, купони, послуги, клієнтів та записи на прийоми, SQL дозволяє структурувати ці дані і забезпечує швидкий доступ до них, що є важливим для забезпечення ефективної взаємодії між різними частинами бази даних.

Використання Node.js в базі даних дозволяє забезпечити більш гнучку та ефективну обробку одночасних запитів, що є критичним для онлайн-запису на послуги. Асинхронність Node.js дозволяє серверу не блокуватися під час обробки окремих запитів, забезпечуючи високу продуктивність і реактивність веб-додатку.

MySQL є оптимальним вибором для зберігання великої кількості даних, забезпечуючи стабільність та надійність. У контексті бази даних барбершопу, де необхідно зберігати інформацію про клієнтів, послуги та записи, MySQL забезпечує ефективність обробки даних та безпеку. Також він добре інтегрується з SQL та Node.js, створюючи ефективну систему.

Visual Studio Code спрощує процес розробки та тестування, підвищуючи якість коду та ефективність реалізації функціоналу. Інтегровані інструменти сприяють швидкому виявленню та усуненню помилок, підвищуючи надійність системи.

Використання HTML, CSS та JavaScript є необхідним для створення зручного та привабливого користувацького інтерфейсу. Вони дозволяють створювати інтерфейс, що є функціональним .



## 3.2 Робота користувача з системою

Інтерфейс користувача у веб-застосунку барбершопу є інтуїтивно зрозумілим та зручним, розробленим для забезпечення легкості в навігації та доступності необхідної інформації. Коли користувач вперше відвідує сайт, він потрапляє на головну сторінку, де має можливість обрати різні опції, такі як перегляд барберів чи послуг. Ця сторінка є візуальним входом у веб-застосунок, демонструючи ключові аспекти послуг та надаючи загальне уявлення про барбершоп.

При першому вході на сайт користувач потрапляє на головну сторінку, що показана на рисунку 3.1.

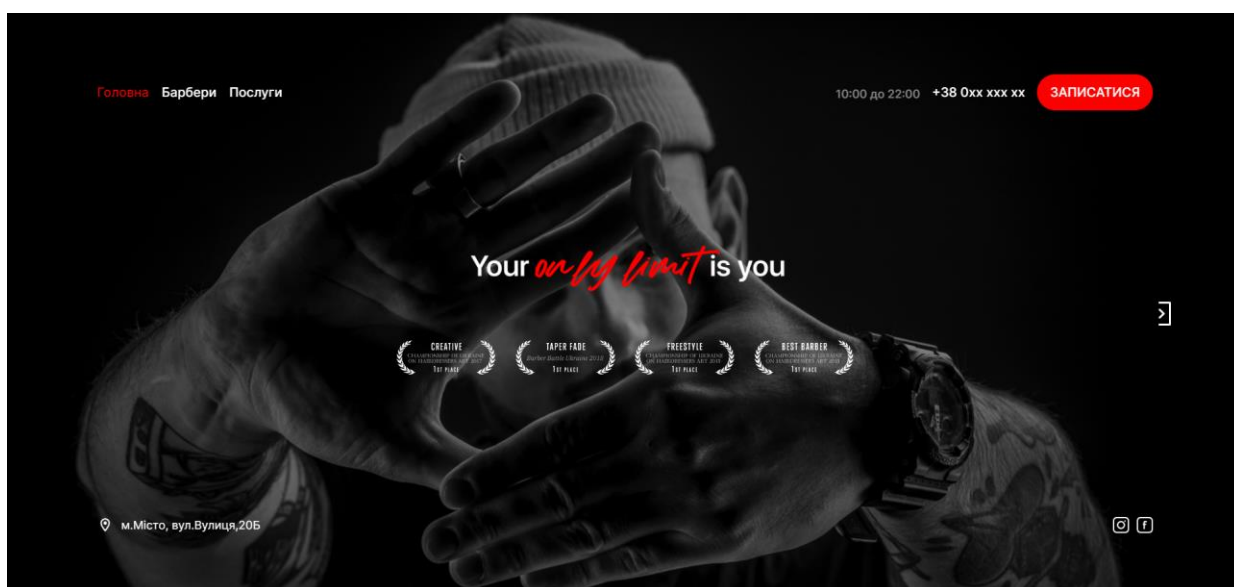


Рисунок 3.1 – Головна сторінка проекту

На ній користувач може вибрати яку сторінку переглянути далі, наприклад Барбери чи Послуги .

Для більш детальної інформації користувачі можуть перейти на сторінку "Барбери", де представлений перелік усіх доступних барберів. Ця сторінка допомагає відвідувачам сайту ознайомитися з кваліфікацією та спеціалізацією кожного барбера, а також переглянути їхні фотографії та іншу інформацію, цей функціонал наведено на рисунку 3.2.



Рисунок 3.2 – Сторінка барберів

Далі, на сторінці "Послуги", користувачі можуть ознайомитися з різноманітними послугами, які пропонує барбершоп, включно з описами та цінами.. Цей функціонал наведено на рисунку 3.3.

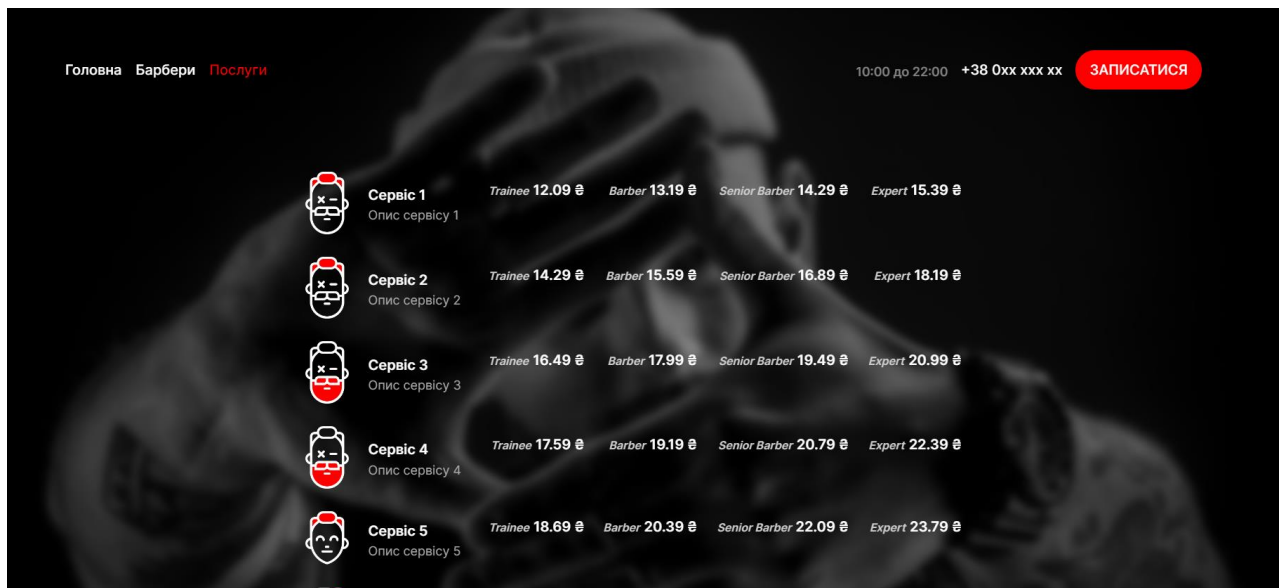


Рисунок 3.3 – Сторінка Послуги

### 3.3 Взаємодія адміністратора з системою

Для взаємодії адміністратора з системою, спочатку потрібна авторизація для доступу до адміністративної частини веб-застосунку. Після входу в систему, адміністратор має можливість управління записами, включаючи вибір дати та часу

для призначень, а також підтвердження чи скасування запланованих візитів.

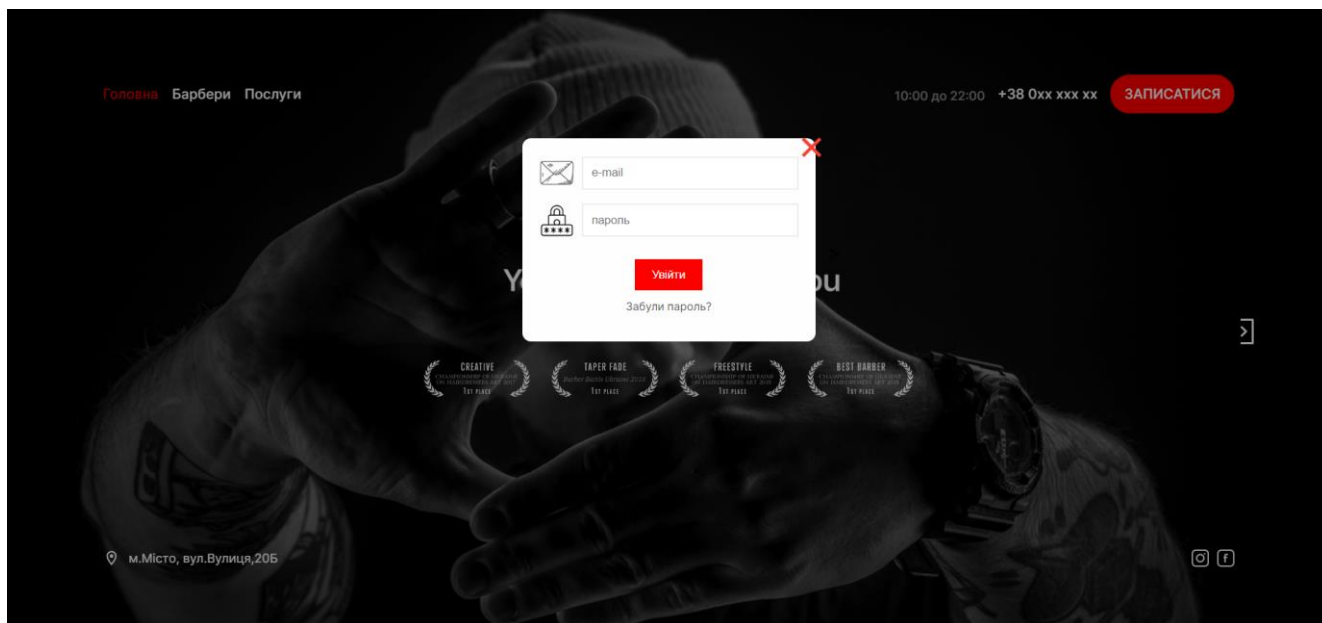


Рисунок 3.4 – Форма входу

Після входу в систему адміністратора, веб-застосунок перенаправляє на спеціалізовану сторінку для управління записами барбера, яка ілюструється на рисунку 3.5. Ця сторінка є центральним місцем для адміністрування робочих процесів барбершопу, де адміністратори мають можливість виконувати низку важливих функцій, спрямованих на координацію роботи барберів та планування їхнього графіка.

На цій сторінці адміністратор може переглядати всі заплановані записи, а також вносити необхідні зміни в розклад. Він має змогу обрати конкретну дату та час для нових записів, враховуючи наявні вільні слоти в робочому календарі барбера. Це дозволяє оптимізувати навантаження на персонал та забезпечити раціональне розподілення робочого часу.

Рисунок 3.5 – Сторінка барбера

### 3.4 Взаємодія клієнта з системою

Клієнти сайту мають змогу здійснити онлайн-запис на послуги через зручний інтерфейс. Натискання на кнопку "Записатись" відкриває вікно, де користувач може обрати барбера чи вибрати зручну дату, що продемонстровано на рисунку 3.6. Після вибору майстра та послуг, клієнт заповнює контактні дані та отримує підтвердження запису через електронний лист. Цей процес забезпечує зручність та простоту взаємодії з веб-застосунком, сприяючи позитивному досвіду користувачів та ефективності роботи барбершопу.

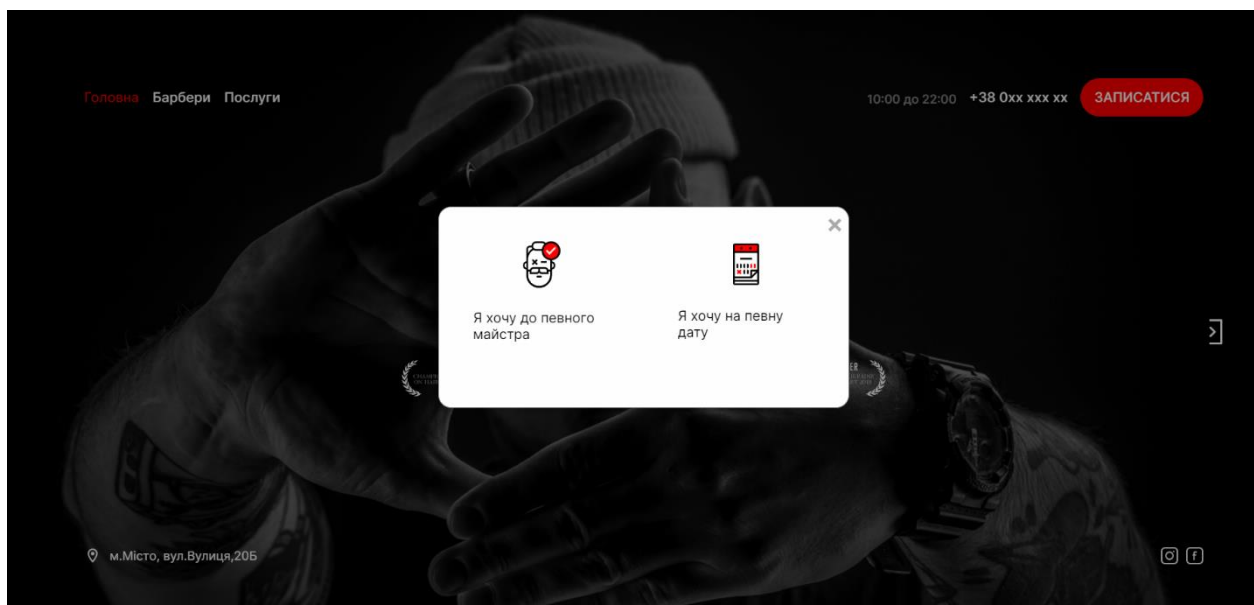


Рисунок 3.6 – Вікно вибору запису

Коли користувач обирає опцію "Записатись за датою" на веб-застосунку барбершопу, він переходить на спеціально розроблену сторінку, що демонструється на рисунку 3.7. Ця сторінка є зручним інструментом для планування візиту в барбершоп, оскільки вона дозволяє користувачам обирати відповідну дату для їхнього запису на послуги.

На цій сторінці презентований календар, який дозволяє користувачам легко орієнтуватися у доступних датах і вибирати найбільш зручний час для візиту. Користувачам пропонується переглядати дні та часи, коли барбери вільні, що дозволяє спланувати візит відповідно до їхнього особистого графіку. Ця функція є особливо корисною для клієнтів, чий розклад є змінним або непередбачуваним, оскільки вона дає можливість вибирати вільні слоти заздалегідь.

### Виберіть дату запису

December 2023

→

Sun	Mon	Tue	Wed	Thu	Fri	Sat
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

#### Деталі запису

Дата

Послуги

Фахівець

Час

До оплати

скинути


далі

Рисунок 3.7 – Сторінка запису за датою

Після вибору підходящої дати, користувачі направляються до наступного кроку, де їм пропонується обрати конкретного барбера та послугу. Це додатково спрощує процес запису, оскільки клієнтам не потрібно шукати вільних майстрів вручну – вони мають можливість переглядати доступних фахівців у вибраний час.

### Виберіть майстар


Ми маємо фахівців чотири рівня. Кожен має свою ціну. Якщо ви берете спеціаліста іншого рівня, ціна вказана.



Ім'я5

Chris

Барбер



Ім'я2

Jordan

Старший барбер

#### Деталі запису

Дата

Послуги

Фахівець

Час

До оплати

скинути

далі

Рисунок 3.8 – Сторінка запису за датою, де можна вибрати майстра

Після вибору відповідного барбера на веб-застосунку барбершопу, користувачу відкривається можливість ознайомлення з переліком послуг, які

пропонує обраний фахівець. Цей етап процесу бронювання детально ілюстрований на рисунку 3.9. На цій сторінці, користувачі мають змогу переглянути широкий спектр послуг, які пропонує барбершоп, зокрема стрижки, укладки, гоління та інші процедури, які виконує обраний майстер.

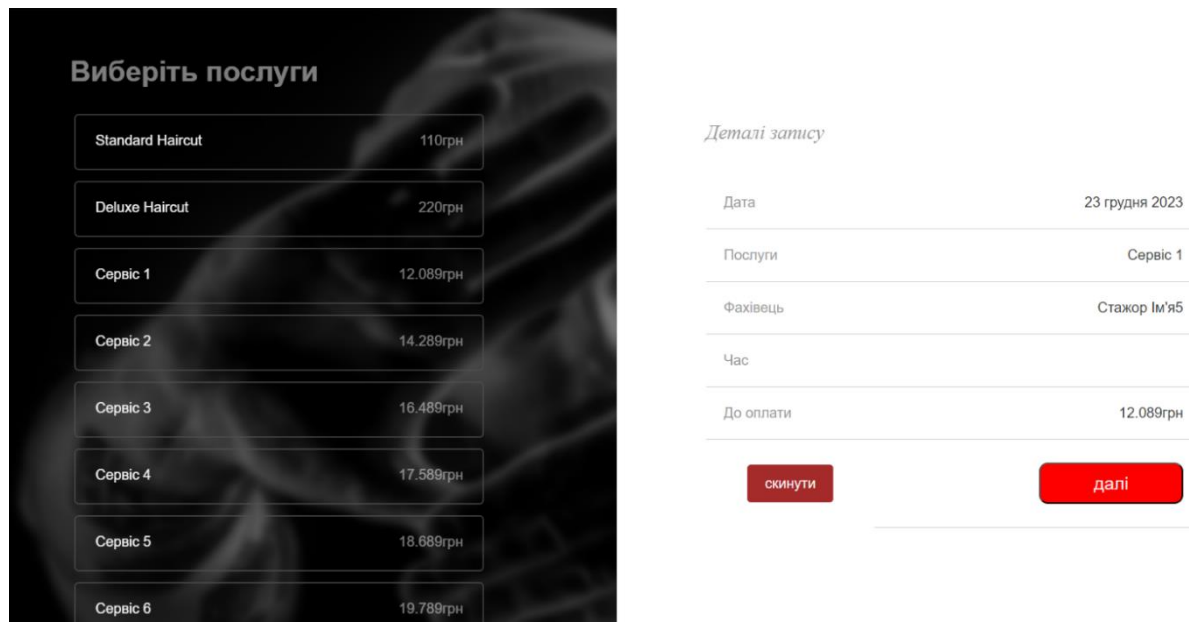
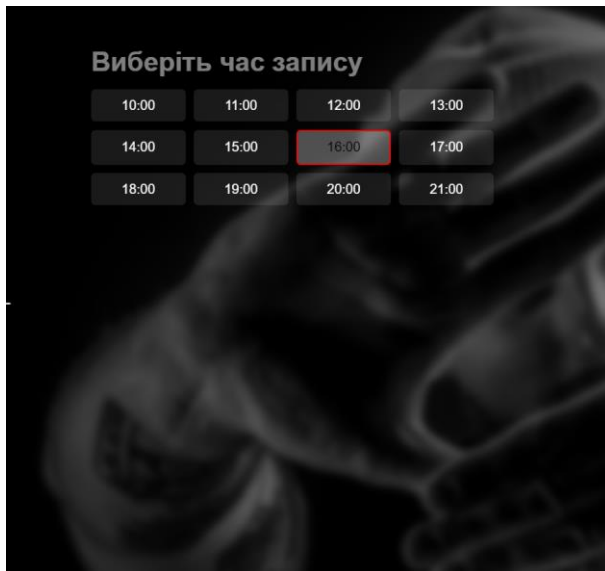


Рисунок 3.9 – Демонстрація послуг

Після вибору барбера та послуг, наступним важливим кроком для користувача є вибір часу запису. Ця процедура є ключовою в процесі бронювання послуг у веб-застосунку барбершопу і відображена на рисунку 3.10. Інтерфейс дозволяє користувачам зручно переглядати доступні часові слоти та обирати найбільш підходящий час для візиту. Це забезпечує гнучкість планування та допомагає уникнути конфліктів у розкладі, як для клієнтів, так і для барберів.

Після визначення дати та часу візиту, користувачі направляються до наступного етапу процесу бронювання - заповнення контактної інформації. Цей етап, представлений на рисунках 3.10 та 3.11, є критично важливим для забезпечення зв'язку між барбершопом та клієнтом. Тут користувачам пропонується ввести основні контактні дані, такі як ім'я, номер телефону та електронна адреса. Ця інформація використовується для підтвердження запису, а також для можливості зв'язку з клієнтом у разі необхідності зміни або уточнення деталей запису.



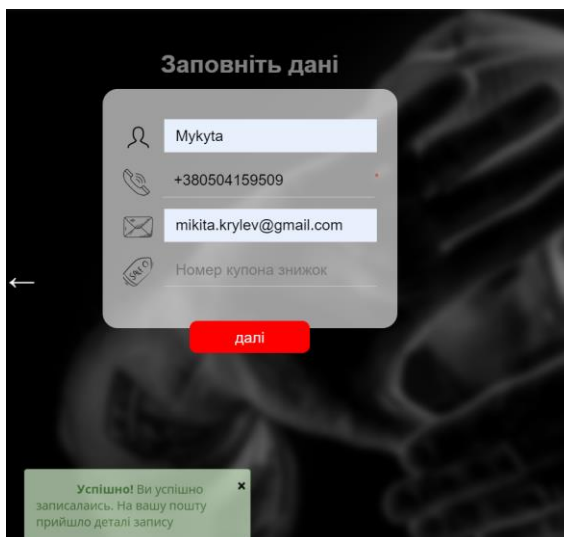
#### Деталі запису

Дата	23 грудня 2023
Послуги	Сервіс 1
Фахівець	Стажор Ім'я5
Час	16:00
До оплати	12.089грн

скинути

далі

Рисунок 3.10 – Вибір часу



#### Деталі запису

Дата	23 грудня 2023
Послуги	Сервіс 1
Фахівець	Стажор Ім'я5
Час	16:00
До оплати	12.089грн

скинути

Рисунок 3.11 – Реєстрація користувача

Після того, як користувач завершить процес реєстрації, ключовим моментом є отримання підтвердження. Це важливий крок, що забезпечує впевненість у успішному завершенні реєстрації. На електронну пошту, яку вказав користувач, буде надіслано спеціалізований електронний лист. Цей лист не тільки підтверджує реєстрацію, але й містить важливі деталі, які користувач повинен знати.

У цьому листі користувач знайде всі необхідні інструкції та інформацію про наступні кроки. Він включає деталі запису, такі як дата, час запису.



Окрім того, лист може містити корисні посилання на додаткові ресурси або інструкції. Наприклад, можуть бути посилання на веб-сайт для отримання додаткової інформації, інструкції щодо зміни або ануляції запису, а також контактні дані для зв'язку у випадку виникнення запитань чи проблем.

Для більш наочного представлення, як може виглядати такий лист, користувачі можуть звернутися до рисунку 3.12. Цей рисунок є прикладом стандартного електронного листа, який надсилається після реєстрації. Він включає всі ключові елементи: заголовок, вступ, основне тіло листа з деталями та інструкціями, а також заключну частину з подякою та контактною інформацією.



Рисунок 3.12 – Приклад електронного листа

Цей підхід забезпечує не тільки прозорість і ясність процесу, але й допомагає користувачам відчувати себе більш впевнено після реєстрації, маючи всю необхідну інформацію відразу під рукою.

## ВИСНОВКИ

У курсовій роботі було поставлено ціль розробити програму, яка спрощує процес управління записами клієнтів в барбершопі. Завдання включали аналіз існуючих рішень, проектування інтерфейсу та архітектури програми, реалізацію функціоналу для запису, зміни та видалення записів, а також тестування програми.

В процесі розробки було створено зручний і інтуїтивно зрозумілий інтерфейс користувача. Програма була реалізована з використанням сучасних технологій, що забезпечує її стабільність та швидкість роботи.

Основним функціоналом програми є керування записами клієнтів, включаючи створення, редагування, перегляд і видалення записів. Також програма надає можливість відстежувати історію візитів клієнта та планувати робочий графік майстрів.

В майбутньому планується розширення функціоналу програми, включно з інтеграцією з іншими системами управління бізнесом, та додавання нових можливостей, таких як онлайн-бронювання та автоматизоване нагадування клієнтам про записи.

Розроблена програма ефективно вирішує задачу управління записами в барбершопі, забезпечуючи зручність та ефективність роботи як для адміністраторів, так і для майстрів. Це створює значні переваги для бізнесу, сприяючи кращому обслуговуванню клієнтів і оптимізації робочих процесів.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Casciaro M., Mammino L. Node.js Design Patterns. – San Francisco: O'Reilly Media, 2020. – 352 с.
2. Haverbeke M. Eloquent JavaScript: A Modern Introduction to Programming. – San Francisco: No Starch Press, 2018. – 472 с.
3. Kruckenberg M., Pipes J. Pro MySQL. – New York: Apress, 2005. – 768 с.
4. Duckett J. HTML & CSS: Design and Build Websites. – Indianapolis: Wiley, 2011. – 490 с.
5. Simpson K. You Don't Know JS (серія). – Sebastopol: O'Reilly Media, 2015. – Різна кількість сторінок в залежності від книги серії.
6. Young A., Meck B., Cantelon M. Node.js in Action. – Manning Publications, 2017. – 384 с.
7. Budd A. CSS Mastery: Advanced Web Standards Solutions. – New York: Apress, 2016. – 350 с.
8. Nixon R. Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5. – Sebastopol: O'Reilly Media, 2018. – 720 с.
9. Документація по Node.js URL: <https://nodejsdev.ru/guides/webdraftt>
10. Документація за MySQL URL: <https://nodejsdev.ru/guides/webdraftt>
11. Flanagan D. JavaScript: The Definitive Guide. – Sebastopol: O'Reilly Media, 2020. – 706 с.
12. Документація по HTML5 URL: <https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/HTML5>
13. Документація за CSS3 URL: <https://developer.mozilla.org/en-US/docs/Web/CSS/CSS3>
14. Eich B. JavaScript: Up and Running. – Sebastopol: O'Reilly Media, 2017. – 250 с.

## **ДОДАТОК А**

SQL-скрипти

**НТУУ «КПІ ім. Ігоря Сікорського»**

**НИ ІАТЕ ЦТЕ ТР-12**

Листів 5

```
drop database barbershop;
create database barbershop;
use barbershop;
CREATE TABLE barber_categories (
    category_id INT AUTO_INCREMENT PRIMARY KEY,
    category_name ENUM('trainee', 'barber', 'senior_barber', 'expert'),
    markup DECIMAL(5, 2)
);
CREATE TABLE discount_card (
    discount_card_id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(30),
    discount double
);
CREATE TABLE barbers (
    barber_id INT AUTO_INCREMENT PRIMARY KEY,
    first_name VARCHAR(20),
    last_name VARCHAR(30),
    email VARCHAR(20),
    phone_number VARCHAR(15),
    photo_url VARCHAR(255),
    category_id INT,
    instagram_link varchar(255),
    password VARCHAR(30),
    description TINYTEXT,
    FOREIGN KEY (category_id) REFERENCES barber_categories(category_id)
);
CREATE TABLE services (
    service_id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(30),
```

```

    base_price DECIMAL(10, 2),
    image_url VARCHAR(255),
    description TINYTEXT
);

CREATE TABLE service_prices (
    price_id INT AUTO_INCREMENT PRIMARY KEY,
    service_id INT,
    price DECIMAL(10, 2),
    effective_date DATETIME,
    FOREIGN KEY (service_id) REFERENCES services(service_id)
);

```

```

CREATE TABLE clients (
    client_id INT AUTO_INCREMENT PRIMARY KEY,
    first_name VARCHAR(20),
    last_name VARCHAR(30),
    middle_name VARCHAR(30),
    phone_number VARCHAR(20),
    email VARCHAR(30),
    haircut_count INT DEFAULT 0,
    discount_card_id INT,
    FOREIGN KEY (discount_card_id) REFERENCES
discount_card(discount_card_id)
);

```

```

CREATE TABLE appointments (
    appointment_id INT AUTO_INCREMENT PRIMARY KEY,
    client_id INT,
    barber_id INT,
    service_id INT,
    appointment_date DATETIME,

```

```

appointment_status ENUM('scheduled', 'completed', 'cancelled'),
FOREIGN KEY (client_id) REFERENCES clients(client_id),
FOREIGN KEY (barber_id) REFERENCES barbers(barber_id),
FOREIGN KEY (service_id) REFERENCES services(service_id)
);

CREATE TABLE administrators (
    admin_id INT AUTO_INCREMENT PRIMARY KEY,
    first_name VARCHAR(255),
    last_name VARCHAR(255),
    email VARCHAR(255),
    phone_number VARCHAR(20)
);

use barbershop;

-- Вставка даних до таблиці barber_categories
INSERT INTO barber_categories (category_name, markup) VALUES
('trainee', 10.00),
('barber', 20.00),
('senior_barber', 30.00),
('expert', 40.00);

INSERT INTO clients (first_name, last_name, middle_name, phone_number, email)
VALUES
('John', 'Doe', 'K', '555-111-2222', 'john.doe@example.com'),
('Jane', 'Doe', 'L', '555-333-4444', 'jane.doe@example.com');

INSERT INTO administrators (first_name, last_name, email, phone_number) VALUES
('Sam', 'Admin', 'sam.admin@example.com', '555-555-5555');

INSERT INTO barbers (first_name, last_name, email, phone_number, photo_url,
category_id,description,password)

```

## VALUES

```
(('Ім\`я1', 'Прізвище1', 'email1@example.com', '1234567890',
'./img/BarberPage/backgrBarber1.png', 1, "Опис барбера", 'password1'),
('Ім\`я2', 'Прізвище2', 'email2@example.com', '2345678901',
'./img/BarberPage/backgrBarber2.png', 2, "Опис барбера", 'password2'),
('Ім\`я3', 'Прізвище3', 'email3@example.com', '3456789012',
'./img/BarberPage/backgrBarber3.png', 3, "Опис барбера", 'password3'),
('Ім\`я4', 'Прізвище4', 'email4@example.com', '4567890123',
'./img/BarberPage/backgrBarber4.png', 4, "Опис барбера", 'password4'),
('Ім\`я5', 'Прізвище5', 'email5@example.com', '5678901234',
'./img/BarberPage/backgrBarber5.png', 1, "Опис барбера", 'password5'));
```

```
select *from barbers;
```

```
INSERT INTO services (name, base_price, image_url, description) VALUES
('Сервіс 1', 10.99, './img/iconService/iconService (1).png', 'Опис сервісу 1'),
('Сервіс 2', 12.99, './img/iconService/iconService (2).png', 'Опис сервісу 2'),
('Сервіс 3', 14.99, './img/iconService/iconService (3).png', 'Опис сервісу 3'),
('Сервіс 4', 15.99, './img/iconService/iconService (4).png', 'Опис сервісу 4'),
('Сервіс 5', 16.99, './img/iconService/iconService (5).png', 'Опис сервісу 5'),
('Сервіс 6', 17.99, './img/iconService/iconService (6).png', 'Опис сервісу 6'),
('Сервіс 7', 18.99, './img/iconService/iconService (7).png', 'Опис сервісу 7'),
('Сервіс 8', 19.99, './img/iconService/iconService (8).png', 'Опис сервісу 8'),
('Сервіс 9', 20.99, './img/iconService/iconService (9).png', 'Опис сервісу 9'),
('Сервіс 10', 21.99, './img/iconService/iconService (10).png', 'Опис сервісу 10'),
('Сервіс 11', 22.99, './img/iconService/iconService (11).png', 'Опис сервісу 11'),
('Сервіс 12', 23.99, './img/iconService/iconService (12).png', 'Опис сервісу 12'),
('Сервіс 13', 24.99, './img/iconService/iconService (13).png', 'Опис сервісу 13'),
('Сервіс 14', 25.99, './img/iconService/iconService (14).png', 'Опис сервісу 14');
SELECT *FROM discount_card;
```



```
INSERT INTO appointments (client_id, barber_id, service_id, appointment_date,
appointment_status) VALUES
(1, 1, 1, '2023-12-25 10:00:00', 'scheduled'),
(1, 1, 1, '2023-12-25 11:00:00', 'scheduled'),
(1, 1, 1, '2023-12-25 12:00:00', 'scheduled'),
(1, 1, 1, '2023-12-25 13:00:00', 'scheduled'),
(1, 1, 1, '2023-12-25 14:00:00', 'scheduled'),
(1, 1, 1, '2023-12-25 15:00:00', 'scheduled'),
(1, 1, 1, '2023-12-25 16:00:00', 'scheduled'),
(1, 1, 1, '2023-12-25 17:00:00', 'scheduled'),
(1, 1, 1, '2023-12-25 18:00:00', 'scheduled'),
(1, 1, 1, '2023-12-25 19:00:00', 'scheduled'),
(1, 1, 1, '2023-12-25 20:00:00', 'scheduled'),
(1, 1, 1, '2023-12-25 21:00:00', 'scheduled');
```