ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені І.І.МЕЧНИКОВА
(повне найменування вищого навчального закладу)

Факультет математики, фізики та інформаційних технологій
(повне найменування інституту, назва факультету (відділення))

Кафедра математичного забезпечення комп'ютерних систем
(повна назва кафедри (предметної, циклової комісії))

Кваліфікаційна робота

на здобуття рівня вищої освіти «бакалавр»
(рівень вищої освіти)

Розподілена автоматизована система для мережі салонів краси
Automation of small business activities in the beauty industry

Виконала: студентка денної форми навчання
спеціальності 123 – Комп'ютерна інженерія
(шифр і назва напряму підготовки, спеціальності)
Освтня програма «Комп'ютерна інженерія»
(назва освітньої програми)
Батенко Аліна Ігорівна
(прізвище, ім'я, по-батькові)
Керівник к.ф-м.н., доцент Шпінарева І.М.
(науковий ступінь, вчене звання, прізвище та ініціали, **підпис**)
Рецензент Розновець О.І.
(науковий ступінь, вчене звання, прізвище та ініціали)

Рекомендовано до захисту:
Протокол засідання кафедри
№___ від «___»_____ 2024 р.

Завідувач кафедри
_____Євгеній МАЛАХОВ_
(підпис) (ім'я, прізвище)

Захищено на засіданні ЕК № ___
протокол № __ від «___» ____ 2024 р.
Оцінка _____/_____/_____
(за національною шкалою, шкалою ECTS, бали)
Голова ЕК
_____ _Алла КОБОЗЄВА_
(підпис) (ім'я, прізвище)

Одеса - 2024

АНОТАЦІЯ

Метою кваліфікаційні роботи є розробка розподіленої автоматизованої системи управління філіалами салону краси для оптимізації робочого процесу, автоматизації фінансового моніторингу та підвищення загальної ефективності управління системою «Call of Beauty», що працює як мобільний та веб-застосунок. Спроектована та побудована комп'ютерна мережа для філіалів салону. Система призначена для використання як працівниками салону краси, так і його клієнтами, з метою покращення якості обслуговування та підвищення ефективності управління бізнесом.

Для розробки системи обрані сучасні програмні засоби, зокрема мова програмування PHP із фреймворком Laravel, мова програмування Dart та фреймворк Flutter, що дозволяє створювати кросплатформені додатки з високою продуктивністю, а також СУБД PostgreSQL. Система будується за принципами трирівневої архітектури клієнт-сервер, забезпечуючи розподілене управління даними через VPN-з'єднання між філіалами.

У створеній системі виконується облік працівників салону, їх робочого графіку, облік клієнтів та наданих послуг, ведеться фінансовий облік доходів та витрат, надається статистика про ефективність роботи майстрів, а також використовуються рейтингова система для визначення популярності послуг.

Особливостями даного проекту є можливість гнучкого налаштування під конкретні потреби кожного салону, зручний інтерфейс для майстрів та клієнтів, а також високий рівень автоматизації бізнес-процесів. Система забезпечує миттєвий доступ до необхідної інформації та дозволяє ефективно управляти ресурсами салону.

Результатом кваліфікаційної роботи є розподілена система управління салоном краси «Call of Beauty» з інтуїтивно зрозумілим інтерфейсом та індивідуальним функціоналом для кожного користувача системи.

ABSTRACT

The purpose of this qualification work is to develop a distributed automated management system for beauty salon branches to optimize the workflow, automate financial monitoring, and improve the overall efficiency of managing the «Call of Beauty» system, which operates as both a mobile and web application. A computer network has been designed and built for the salon branches. The system is intended for use by both the beauty salon employees and its clients to improve service quality and increase business management efficiency.

Modern software tools were chosen for the development of the system, including the PHP programming language with the Laravel framework, the Dart programming language, and the Flutter framework, which allows the creation of high-performance cross-platform applications, as well as the PostgreSQL DBMS. The system is built on the principles of three-tier client-server architecture, providing distributed data management through VPN connections between branches.

The created system performs the accounting of salon employees, their work schedules, client management, and the services provided. It also handles financial accounting of income and expenses, provides statistics on the effectiveness of the work of the masters, and uses a rating system to determine the popularity of services.

The features of this project include the ability to flexibly customize according to the specific needs of each salon, a convenient interface for both masters and clients, and a high level of business process automation. The system ensures instant access to the necessary information and allows efficient management of the salon's resources.

The result of this qualification work is a distributed beauty salon management system «Call of Beauty» with an intuitive interface and personalized functionality for each system user.

ЗМІСТ

ВСТУП 5
1 ОГЛЯД ІСНУЮЧИХ АВТОМАТИЗОВАНИХ СИСТЕМ УПРАВЛІННЯ 7
2 ПОСТАНОВКА ЗАВДАННЯ 10
3 ПРОЕКТУВАННЯ СИСТЕМИ 15
3.1 Архітектура та шаблон проектування мережевої системи для 15
організації діяльності філіалів салону краси «Call of Beauty» 15
3.2 Інформаційна модель предметної області 17
3.3 Модель клієнтського застосунку 23
3.4 Вибір технологій для реалізації системи 26
4 ПРОЕКТУВАННЯ МЕРЕЖІ ФІЛІАЛІВ 28
4.1 Побудова структурно-логічної схеми комп'ютерної мережі 28
4.2 Вибір активного та пасивного обладнання для реалізації мережі 29
4.3 Моделювання мережі в середовищі Packet Tracer 32
5 ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ САЛОНІВ КРАСИ 37
5.1 Створення бази даних 37
5.2 Запити до бази даних для вирішення задач 39
5.3 Безпека інформаційної системи 49
5.4 Інструкція користувача 52
ВИСНОВКИ 62
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 64
ДОДАТОК А ER-діаграма для салонів краси «Call of Beauty» 66
ДОДАТОК Б Структурно-логічна схема мережі салонів краси 67
ДОДАТОК В Запити для створення бази даних 69
ДОДАТОК Г Створення функцій 72

ВСТУП

У теперішньому світі стрімко зростаюча конкуренція у сфері бізнесу потребує постійних змін. А впровадження інноваційних технологій є необхідним рішенням та головним рушієм для успішної діяльності підприємства салону краси. Одним із ключових напрямків цієї зміни є впровадження розподіленої автоматизованої системи управління, спрямованої на оптимізацію процесів та підвищення ефективності усіх аспектів діяльності філіалів салону краси.

Індустрія краси є однією з найбільш динамічних галузей в сучасному бізнесі. Зростання свідомості споживачів про здоров'я та красу в останні роки значно збільшило попит на косметичні послуги і продукти. Теперішні салони краси не обмежуються лише звичайними стрижками і зачісками, вони є багатофункціональними центрами, де клієнти можуть отримати комплексне обслуговування, включаючи манікюр, педикюр, косметичні процедури, масажі та спа-процедури. Таке розширення спектру послуг вимагає від салонів краси постійного удосконалення управлінських та технологічних підходів для задоволення різноманітних потреб клієнтів.

Зростаючий вплив соціальних медіа і трендів у сфері краси також викликає необхідність удосконалення маркетингових стратегій і взаємодії з потенційними клієнтами. Нинішні відвідувачі салонів краси активно використовують інтернет засоби для вибору послуг і діляться своїми враженнями про відвідування онлайн, що робить репутацію салону ще важливішою для його успіху.

Сучасний салон краси, щоб залишатись конкурентоспроможнім, повинен відповідати найвищим стандартам обслуговування, задовольняти запити своїх клієнтів і швидко реагувати на зміни у ринкових умовах. Ось чому існуючі традиційні методи управління, як от ручний облік графіків роботи, клієнтської бази, фінансових потоків, нерідко є недосконалими або надто часо-й ресурсномісткими.

Крім того, розвиток індустрії краси безпосередньо впливає на заняття трудових ресурсів та можливості для кар'єрного зросту в цій галузі. Створення інноваційних технологій управління та автоматизації дозволяє не лише підвищувати ефективність бізнес-процесів, але й створювати нові робочі місця та підвищувати кваліфікацію персоналу.

Метою роботи є розробка розподіленої автоматизованої системи управління філіалами салону краси для оптимізації робочого процесу, автоматизації фінансового моніторингу та підвищення загальної ефективності управління системою.

Така система повинна включати в себе планування робочих годин майстрів, облік клієнтів, ведення обліку доходів та витрат за періоди, аналіз ключових показників ефективності. Це забезпечить миттєвість і точність процесу управління ресурсами та дозволить керівництву зосередитися на стратегії та розвитку бізнесу.

Для досягнення мети потрібно розв’язати наступні задачі:

- проаналізувати предметну область та існуючі системи;
- виконати проектування бази даних;
- виконати проектування комп’ютерні мережі філіалів;
- обґрунтувати вибір засобів та технологій розробки;
- побудувати та захистити мережу в середовищі Packet Tracer;
- розробити автоматизовану систему управління салону краси;
- протестувати систему.

1 ОГЛЯД ІСНУЮЧИХ АВТОМАТИЗОВАНИХ СИСТЕМ УПРАВЛІННЯ

На ринку існує велике різноманіття CRM-систем, спеціально розроблених для салонів краси. Для прикладу можу привести BloknotApp, EasyWeek та Booksy. Кожна з цих систем має свої унікальні характеристики та переваги, проте всі вони мають свої особливості, що відповідають різним специфікам ведення бізнесу. Далі детально розглянуті переваги та недоліки обраних систем.

Таким чином одна з найпоширеніших CRM-систем, EasyWeek, є зручним онлайн-сервісом для управління записами в салоні краси, який дозволяє легко організувати роботу майстрів і забезпечує зручність для клієнтів при бронюванні послуг [1]. До ключових переваг можна віднести: онлайн запис, клієнти можуть самостійно записуватися на послуги через веб-сайт або мобільний додаток; календар для майстрів дозволяє відслідковувати та керувати розкладом роботи майстрів у режимі реального часу та автоматичні нагадування, які безперечно зручні для клієнтів. Але з іншого боку, виявлено низку недоліків, таких як обмежений функціонал для управління фінансами та недостатньо гнучкі інструменти для аналізу бізнес-процесів.

BloknotApp також популярна CRM-система, яка опирається на широкий спектр функцій для управління бізнесом [2]. Тут фінансовий облік доходів і витрат, генерація фінансових звітів є неодмінною перевагою серед конкурентів. Проте, BloknotApp також має свої недоліки, зокрема, обмежену гнучкість налаштувань для великих мереж салонів краси.

Booksy є лідером серед система для управління салонами краси, що відзначається високим рівнем автоматизації і широким набором функцій [3], наприклад, таких як:

1) висока клієнтоорієнтованість, тобто можливість записуватися на послуги через мобільний додаток або веб-сайт з будь-якої точки світу;

2) детальне планування розкладу роботи майстрів та управління записами за допомогою потужних інструментів;

3) розширені можливості для проведення маркетингових кампаній, включаючи розсилки, акції та програми лояльності;

4) детальні фінансові звіти, контроль прибутків та витрат;

5) аналітика бізнесу та аналіз ефективності роботи майстрів.

Booksy об’єднує в собі всі позитивні аспекти вище перелічених систем, але має найсуттєвіший недолік – дороговартісність та обмежені можливості адаптації до потреб конкретного салону краси та не завжди забезпечують повний контроль над усіма аспектами бізнесу.

В результаті аналізу предметної області салону краси в системі виявлено, що жодна з них не повністю задовольняє потреби нашої мережі. На основі цього прийнято рішення розробити власну систему під назвою «Call of Beauty». Планується, що ця система автоматизації для салонів краси матиме ряд переваг, які зроблять її кращим вибором для малих підприємств у цій галузі.

Передбачено, що система «Call of Beauty» буде безкоштовною, матиме гнучку архітектуру, що дозволить легко налаштовувати її під конкретні потреби кожного салону. Адміністраторам надано можливість додавати нових майстрів та клієнтів, редагувати послуги та типи послуг, керувати філіалами салону та налаштовувати звітність.

Крім того, передбачається, що система матиме зручний інтерфейс для майстрів та клієнтів, який сприятиме покращенню їх досвіду користування. Майстри матимуть змогу переглядати свій календар записів, змінювати статуси записів та додавати нові, а також переглядати статистику наданих послуг. Клієнти, у свою чергу, зможуть зареєструватися, переглядати інформацію про майстрів та послуги, залишати відгуки та редагувати свої облікові записи [4].

У табл. 1.1 наведено порівняльну характеристику різних CRM-систем, включаючи систему для салонів «Call of Beauty».

Також для впровадження розподіленої системи в мережі салонів краси важливо забезпечити ефективну та безпечну взаємодію між філіалами. В цьому контексті використання віртуальних приватних мереж є оптимальним рішенням. VPN дозволяє створити захищений тунель для передачі даних між

філіалами, що забезпечує конфіденційність і цілісність інформації. Це особливо важливо для бізнесу, де обмін даними включає особисту інформацію клієнтів, фінансові транзакції та інші чутливі дані.

Таблиця 1.1 – Порівняльна характеристика CRM-систем

	BloknotApp	CleverBox	Booksy	Call of Beauty
Вартість	від 15$ на місяць	від 45$ на місяць	Від 35 $	Безкоштовна
Ведення календаря	+	+	+	+
Користувачі	Адміністратори та майстри	Адміністратори та майстри	Адміністратори, майстри та клієнти	Адміністратори, майстри та клієнти
Ведення бази клієнтів	+	+	+	+
Статичні звіти	+	+	+	+
Контроль фінансів	+	+	+	+
IP-телефонія	-	+	-	-
Нагадування	+	+	+	-

Завдяки розподіленій системі, кожен філіал салону краси може працювати як незалежний вузол, який взаємодіє з центральним сервером для синхронізації даних та управління бізнес-процесами. Такий підхід дозволяє керівництву мати централізований контроль над усіма філіалами, відстежувати ефективність роботи, управляти фінансами та приймати стратегічні рішення на основі актуальних даних.

Отже, впровадження розподіленої автоматизованої системи управління мережею салонів краси не тільки підвищує ефективність внутрішніх процесів, але й забезпечує високу надійність, масштабованість та безпеку, що є ключовими факторами успіху в умовах сучасного конкурентного ринку.

2 ПОСТАНОВКА ЗАВДАННЯ

При розробці розподіленої автоматизованої системи для мережі салонів краси необхідно виконати такі завдання.

1) Виконати проектування комп'ютерної мережі для філіалів салону краси, які розташовані в різних містах України та забезпечити захист мережі.

2) Основні задачі, що повинна виконувати система салону краси:

- реєстрація клієнтів;
- ведення календарю записів на послуги;
- управління персоналом салону краси;
- управління клієнтами салону краси;
- ведення доходів та витрат;
- перегляд звітності роботи салону;
- створення відгуків.

В результаті аналізу предметної області салону краси в системі передбачено три типи користувачів. Кожен з типів має свої можливості та вимоги до системи:

1) клієнт – може залишати інформацію про себе – (ПІБ, номер для зворотного зв’язку), може переглядати список усіх доступних майстрів та послуги, що вони надають, залишати відгуки про салон та записатись на обрану послугу до конкретного майстра у вільний час;

2) майстер салону краси – може переглядати потрібну, дозволену інформацію: історія записів клієнтів, календар поточних записів, переглядати інформацію про послуги, додавати новий запис та змінити його статус;

3) адміністратор – може переглядати всю інформацію в системі, відповідає за управління даними робітників, клієнтів та переглядає всю можливу звітність (доходи за певний період, кількість записів клієнтів на процедури, ефективність, статистика відмінених записів), формує графіки доходів та витрат за певний період. Також виконує додавання нових клієнтів, робітників, послуг.

У табл. 2.1 наведено перелік задач для кожного з користувачів із зазначенням вхідної та вихідної інформації.

Таблиця 2.1 – Задачі користувачів

Номер	Задача	Вхідні дані	Вихідні дані
Адміністратор			
A1	Додати майстра	ПІБ робітника, номер телефону, заробітна плата, дата народження, стать, посада робітника	Новий робітник з табельним номером
A2	Видалити майстра	Табельний номер	Відсутні
A3	Редагувати майстра	Табельний номер, ПІБ робітника, посада, заробітна плата, стать, дата народження майстра	Оновлена інформація про робітника
A4	Додавання нового клієнта	Табельний номер ,ПІБ, номер телефону, пошта, дата народження, емейл, тип клієнта, стать клієнта	Новий клієнт з табельним номером
A5	Видалити клієнта	Табельний номер	Відсутні
A6	Редагувати клієнта	Табельний номер ,ПІБ, номер телефону, пошта, дата народження, емейл, тип, стать клієнта	Оновлена інформація про робітника
A7	Додати нову послугу	Табельний номер, назва, ціна, витрати, тривалість, опис, тип послуги	Нова послуга з табельним номером
A8	Видалити послугу	Табельний номер	Відсутні

Продовження таблиці 2.1

Номер	Задача	Вхідні дані	Вихідні дані
Адміністратор			
A9	Редагувати послугу	Табельний номер, назва, ціна, витрати, тривалість, опис, тип	Оновлена інформація про послугу
A10	Додати новий тип послуги	Табельний номер, назва	Новий тип послуги
A11	Видалити тип послуги	Табельний номер	Відсутні
A12	Додати новий філіал салону	Табельний номер, місто	Новий філіал
A13	Видалити філіал салону	Табельний номер	Відсутні
A14	Видаляти записи клієнтів	Табельний номер	Відсутні
A15	Переглядати звітності про доходи та розходи (За період або взагалі)	Послуга, вартість, витрати, дати	Дохід від певної послуги
A16	Переглядати звітності про ефективність майстрів за прибутком за період (місяць)	Ім’я майстра, дати,кількість виконаних процедур	Рейтинг ефективності айстрів
A17	Переглядати звітності про відмінені клієнтом записи	Послуга зі статусом Відмінено	Число виконаних записів
A18	Переглядати інформацію про популярність послуг	Послуга, кількість записів на цю послугу	Рейтинг послуг

Продовження таблиці 2.1

Номер	Задача	Вхідні дані	Вихідні дані
Адміністратор			
A19	Перегляд графіку роботи майстрів	Майстер, статус запису – В очікуванні	Дата запису, назва послуги, час запису, ім'я клієнта
Майстер салону краси			
М1	Переглядати свій календар записів	Майстер, статус запису – В очікуванні	Дата запису, назва послуги, час запису, ім'я клієнта
М2	Змінити статус запису	Майстер, запис, статус запису – Послуга надана	Оновлена інформація про запис
М3	Додавати нові записи	Дата запису, назва послуги, час запису, ім'я клієнта	Новий запис
М4	Переглянути кількість наданих послуг за певний проміжок часу	Послуги, кількість записів на послугу, період часу	Рейтинг послуг
Клієнт			
К1	Зареєструватись	Логін, пароль, ПІБ, номер телефону, дата народження	Новий клієнт
К2	Переглядати інформацію про майстрів	І’мя майстра	ФІО, вік, послуги,що надає, салон
К3	Переглядати відгуки		Коментар, дата, логін клієнта, оцінка

Продовження таблиці 2.1

Номер	Задача	Вхідні дані	Вихідні дані
Клієнт			
К4	Залишити відгук (За умови, що клієнт є у таблиці записів)	Коментар, дата, логін клієнта, оцінка	Новий відгук
К5	Переглядати інформацію про послуги	Тип послуги	Назва, ціна, тривалість, опис
К6	Переглядати інформацію про можливі дати запису на послугу	Тип послуги, певний період часу	Назва, ціна, тривалість, опис
К7	Редагувати свій обліковий запис	Нові ПІБ, дата народження, номер телефону	Оновлена інформація про клієнта
К8	Записатись на процедуру	ПІБ майстра, дата, процедура, ПІБ клієнта	Новий запис з табельним номером

3 ПРОЕКТУВАННЯ СИСТЕМИ

3.1 Архітектура та шаблон проектування мережевої системи для організації діяльності філіалів салону краси «Call of Beauty»

Для розв'язання поставлених задач серед багатьох видів системних архітектур обрано триланкову архітектуру. У сукупності трирівневі архітектури – це моделі програмування, які дозволяють розподіляти функціональні можливості додатків між трьома незалежними системами, як правило:

– клієнтські компоненти, що працюють на локальних робочих станціях (рівень представлення даних – перший рівень);

– сервер додатків (рівень прикладного компоненту – другий рівень);

– сервер бази даних (рівень управління ресурсами – третій рівень).

Взаємодія між ними відбувається наступним чином: клієнт надсилає запит на перегляд сторінки який обробляється сервером додатків, після чого сервер додатків за потреби звертається до серверу бази даних та надсилає відповідь у вигляді запитуваної сторінки клієнту. Схема цієї взаємодії наведена на рисунку 3.1.

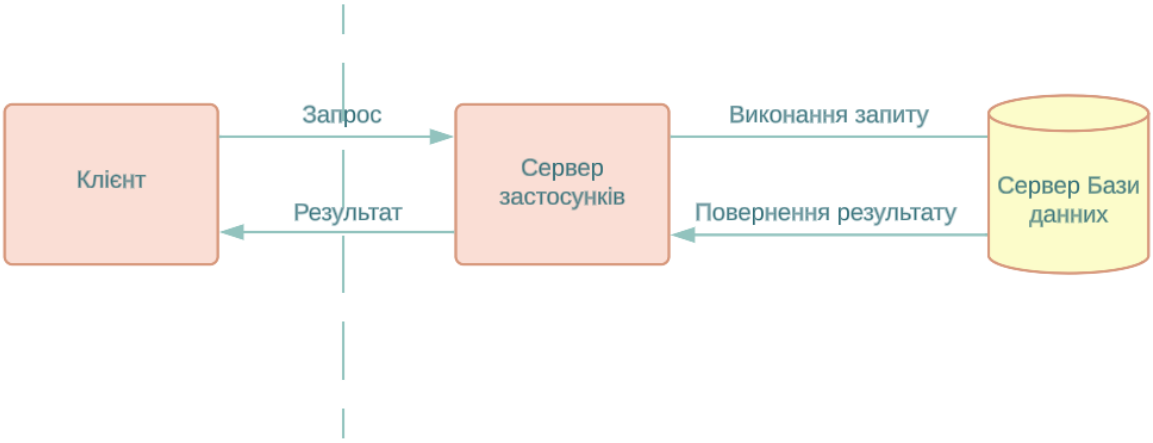


Рисунок 3.1 – Схема триланкової архітектури

В якості шаблону проектування обрано MVC (Model, View, Controller) [5]. Модель представляє інформацію (дані) програми та правила маніпулювання цими даними. Моделі в основному використовуються для керування

правилами взаємодії з відповідною таблицею бази даних. Представлення представляють інтерфейс користувача програми. Представлення – це часто файли HTML із вбудованим кодом PHP, які виконують завдання, пов’язані виключно з представленням даних. Представлення виконують роботу з надання даних у веб-браузер або інший інструмент, який використовується для надсилання запитів із програми. Контролери забезпечують потік між моделями та представленнями. Контролери відповідають за обробку вхідних запитів від веб-браузера, опитування моделей щодо даних і передачу цих даних у представлення для представлення. Схему взаємодії між моделлю, контролером та представленням наведено на рисунку 3.2.

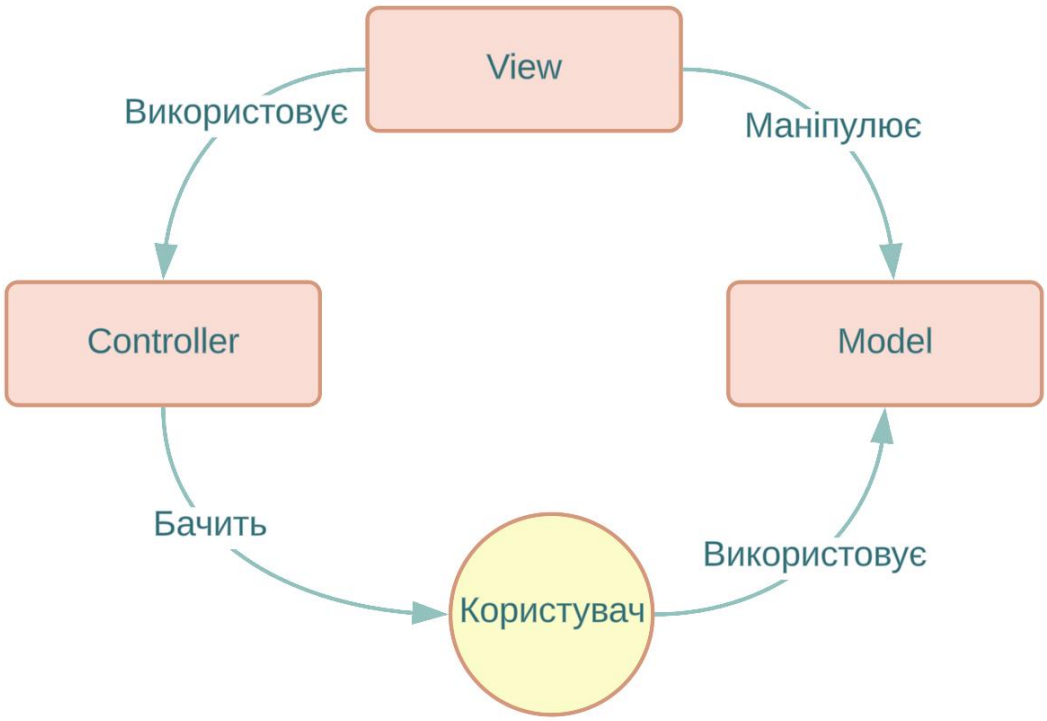


Рисунок 3.2 – Схема взаємодії MVC

Цей шаблон передбачає, що користувач бачить інтерфейс(View) і якось з ним взаємодіє, ця взаємодія передається до контролеру (Controller), що змінює модель(Model), яка в свою чергу змінює користувацький інтерфейс.

3.2 Інформаційна модель предметної області

Сутності та їх атрибути з описом обмежень, що потрібні для розв’язання поставлених задач, наведені в таблиці 3.1.

Таблиця 3.1 – Опис сутностей

Атрибут	Призначення атрибута	Обмеження цілісності
user_profiles (Профілі користувачів)		
id	Ідентифікатор профілю користувача	Не пусте значення, первинний ключ
user_id	Ідентифікатор користувача	Не пусте значення , зовнішній ключ для зв’язку з Users (id)
first_name	Ім’я користувача системи	Не пусте значення, довжина більша за 1
last_name	Прізвище користувача	Не пусте значення, довжина більша за 1
phone	Номер телефону	Не пусте значення, будь-яка комбінація цифр (0-9), дефісів (-), круглих дужок ((і)), косих рисок (/) та пробільних символів
birth_date	Дата народження	Не пусте значення, не може перевищувати 100 років, та робітники мають бути старше 21
created_at	Дата створення профілю	Не пусте значення, зберігає дату та час (до секунди) без вказівки часової зони
updated_at	Дата останніх змін профілю	Не пусте значення, зберігає дату та час (до секунди) без вказівки часової зони

Продовження таблиці 3.1

Атрибут	Призначення атрибута	Обмеження цілісності
users (Користувачі системи)		
id	Ідентифікатор користувача системи	Не пусте значення , первинний ключ
email	Пошта користувача	Не пусте значення, унікальне значення
password	Пароль користувача системи	Не пусте значення, мінімум з 8 символів
created_at	Дата створення профілю	Не пусте значення, зберігає дату та час (до секунди) без вказівки часової зони
updated_at	Дата останніх змін профілю	Не пусте значення, зберігає дату та час (до секунди) без вказівки часової зони
customer_types (Типи клієнтів)		
id	Ідентифікатор типу клієнта	Не пусте значення , первинний ключ
name	Назва типу	Не пусте значення, довжина більша за 1
discount	Знижка	Не пусте значення, додатне
Customers (Клієнти системи)		
Id	Ідентифікатор клієнтів	Не пусте значення , первинний ключ
user_id	Ідентифікатор користувача	Не пусте значення , зовнішній ключ для зв’язку з Users (id)

Продовження таблиці 3.1

Атрибут	Призначення атрибута	Обмеження цілісності
customer_type_id	Ідентифікатор типу клієнта	Не пусте значення , зовнішній ключ зв’язок customer_typesя(id)
created_at	Дата створення профілю	Зберігає дату та час (до секунд) без вказівки часової зони
updated_at	Дата останніх змін профілю	Зберігає дату та час (до секунд) без вказівки часової зони
Salons (Філіали салонів)		
Id	Ідентифікатор філіалів салонів	Не пусте значення , первинний ключ
city	Місто	Не пусте значення, довжина більша за 1
address	Адреса	Не пусте значення, довжина більша за 1
Employees (Працівники салону)		
Id	Ідентифікатор працівників	Не пусте значення , первинний ключ
user_id	Ідентифікатор користувача	Не пусте значення , зовнішній ключ для зв’язку з Users (id)
working_hours	Часи роботи	Не пусте значення
salon_id	Ідентифікатор салону	Не пусте значення , зовнішній ключ для зв’язку з Salon (id)
salary	Ставка	Не пусте значення, додатне
created_at	Дата створення профілю	Зберігає дату та час (до секунд) без вказівки часової зони

Продовження таблиці 3.1

Атрибут	Призначення атрибута	Обмеження цілісності
updated_at	Дата останніх змін профілю	Зберігає дату та час (до секунд) без вказівки часової зони
service_types (Типи послуг)		
id	Ідентифікатор типу послуг	Не пусте значення , первинний ключ
name	Назва типу	Не пусте значення, довжина більша за 1
description	Опис типу послуги	
Services (Послуги)		
id	Ідентифікатор послуг	Не пусте значення , первинний ключ
service_type_id	Ідентифікатор типу послуг	Не пусте значення , зовнішній ключ зв’язок з service_types (id)
name	Назва послуги	Не пусте значення, довжина більша за 1
description	Опис типу послуги	
price	Ціна послуги	Не пусте значення, додатне
expenses	Затрати на надання послуги	Не пусте значення, додатне
duration	Тривалість проведення процедури	Не пусте значення, повинно відповідати формату часу години : хвилини : секунди
customer_reviews (Відгуки клієнтів)		
id	Ідентифікатор відгуків	Не пусте значення, первинний ключ

Продовження таблиці 3.1

Атрибут	Призначення атрибута	Обмеження цілісності
customer_id	Ідентифікатор клієнта	Не пусте значення, зовнішній ключ зв’язок з customer (id)
skills (Типи послуг)		
employee_id	Ідентифікатор працівника	Не пусте значення, зовнішній ключ зв’язок з employee (id)
service_id	Ідентифікатор типу послуг	Не пусте значення, зовнішній ключ зв’язок з service (id)
appointments (Записи на послуги)		
id	Ідентифікатор записів на послугу	Не пусте значення, первинний ключ
customer_id	Ідентифікатор клієнта	Не пусте значення, зовнішній ключ зв’язок з customer (id
employee_id	Ідентифікатор працівника	Не пусте значення, зовнішній ключ зв’язок з employee (id)
service_id	Ідентифікатор типу послуг	Не пусте значення, зовнішній ключ зв’язок з service (id)
appointment_datetime	Дата запису на процедуру	Не пусте значення, зберігає дату та час (до секунд) без вказівки часової зони

Формалізація зв’язків між сутностями, описаними в табл. 3.1.

Між цими сутностями наявні два типи зв’язків:

– один-до-багатьох;

– один-до-одного.

1) Зв'язок між користувачами та профілем користувача. Один користувач може мати лише один профіль, так як і за одним профілем закріплений один користувач системи. Отримуємо зв’язок один до одного

безумовний з обох боків. Для формалізації зв’язків додаємо ідентифікатор користувача, як зовнішній ключ до таблиці профілів.

2) Зв'язок між користувачами та робітниками. Користувач може буди записаний в таблиці робітників лише один раз, так само як і робітник є лише одним користувачем системи. Отримуємо зв’язок один до одного безумовний з обох боків. Для формалізації зв’язків додаємо ідентифікатор користувача, як зовнішній ключ до таблиці робітників.

3) Зв'язок між користувачами та клієнтами. Користувач може буди записаний в таблиці клієнтів лише один раз, так само як і робітник є лише одним користувачем системи. Отримуємо зв’язок один до одного безумовний з обох боків. Для формалізації зв’язків додаємо ідентифікатор користувача, як зовнішній ключ до таблиці клієнтів.

4) Зв'язок між філіалом салону та робітниками. Один філіал має певну кількість робітників, в той же час один працівник відноситься лише до одного філіалу. Таким чином, отримуємо зв’язок один до багатьох безумовний з обох боків. Для формалізації зв’язків додаємо ідентифікатор салону, як зовнішній ключ для таблиці робітників.

5) Зв'язок між клієнтом та відгуком. Один клієнт може залишити багато відгуків, але відгук залишається закріпленим лише за одним клієнтом. Отже, отримуємо зв’язок один до багатьох безумовний з обох боків. Для формалізації зв’язків додаємо ідентифікатор клієнта, як зовнішній ключ для таблиці відгуків.

6) Зв'язок між типом клієнта та клієнтом. До одного типу можуть відноситись багато клієнтів, в той самий час як клієнт одночасно може мати лише один тип. Отримуємо зв’язок один до багатьох безумовний з обох боків. Для формалізації зв’язків додаємо ідентифікатор типу клієнта, як зовнішній ключ для таблиці клієнта.

7) Зв'язок між типом послуги та послугою. Один тип послуги має безліч видів послуг, але послуга відноситься лише до одного типу. Таким чином, отримуємо зв’язок один до багатьох безумовний з обох боків. Для формалізації зв’язків додаємо ідентифікатор типу послуги, як зовнішній ключ для таблиці послуг.

8) Зв'язок між послугами та здібностями. Одна послуга може бути здібністю для багатьох, в той час як одна здібність відноситься лише до однієї послуги. Отримуємо зв’язок один до багатьох безумовний з обох боків. Для формалізації зв’язків додаємо ідентифікатор послуги, як зовнішній ключ для таблиці здібностей.

9) Зв'язок між працівниками та здібностями. Одну послугу можуть надавати багато майстрів, в той час як одна здібність відноситься лише до одного майстра. Таким чином, отримуємо зв’язок один до багатьох безумовний з обох боків. Для формалізації зв’язків додаємо ідентифікатор працівника, як зовнішній ключ для таблиці здібностей.

10) Зв'язок між працівником та записами. У працівника є багато записів, в той час як запис відноситься лише до одного працівника. Отримуємо зв’язок один до багатьох безумовний з обох боків. Для формалізації зв’язків додаємо ідентифікатор працівника, як зовнішній ключ для таблиці записів.

11) Зв'язок між клієнтами та записами. Клієнт може мати багато записів, в той час як запис відноситься лише до одного клієнта. Отримуємо зв’язок один до багатьох безумовний з обох боків. Для формалізації зв’язків додаємо ідентифікатор клієнта, як зовнішній ключ для таблиці записів.

12) Зв'язок між послугами та записами. Існує безліч записів на одну послугу, при тому, що записи закраплені за однією послугою. Отримуємо зв’язок один до багатьох безумовний з обох боків. Для формалізації зв’язків додаємо ідентифікатор клієнта, як зовнішній ключ для таблиці записів.

ER-діаграма отриманої в результаті формалізації зв’язків наведена у додатку А. Для створення ER-діаграми використовувалося Lucid.

3.3 Модель клієнтського застосунку

Ієрархія екранних форм – це схема, що наглядно відображає етапи відтворення сторінок клієнтського інтерфейсу. Для зручності розуміння, через велику кількість задач користувачів ієрархія екранних форм поділена за частини за типами користувачів салону.

Користувачі системи «Call of Beauty» користуються веб-додатком, мобільним для таких операційних систем, як Android, IOS, а також комп’ютерним додатком для Windows, Android, MacOS.

Будь-який користувач спочатку повинен пройти аутентифікацію та авторизацію. Далі кожному користувачу доступні свої можливості. Ієрархія екранних форм клієнта зображена на рис. 3.3.

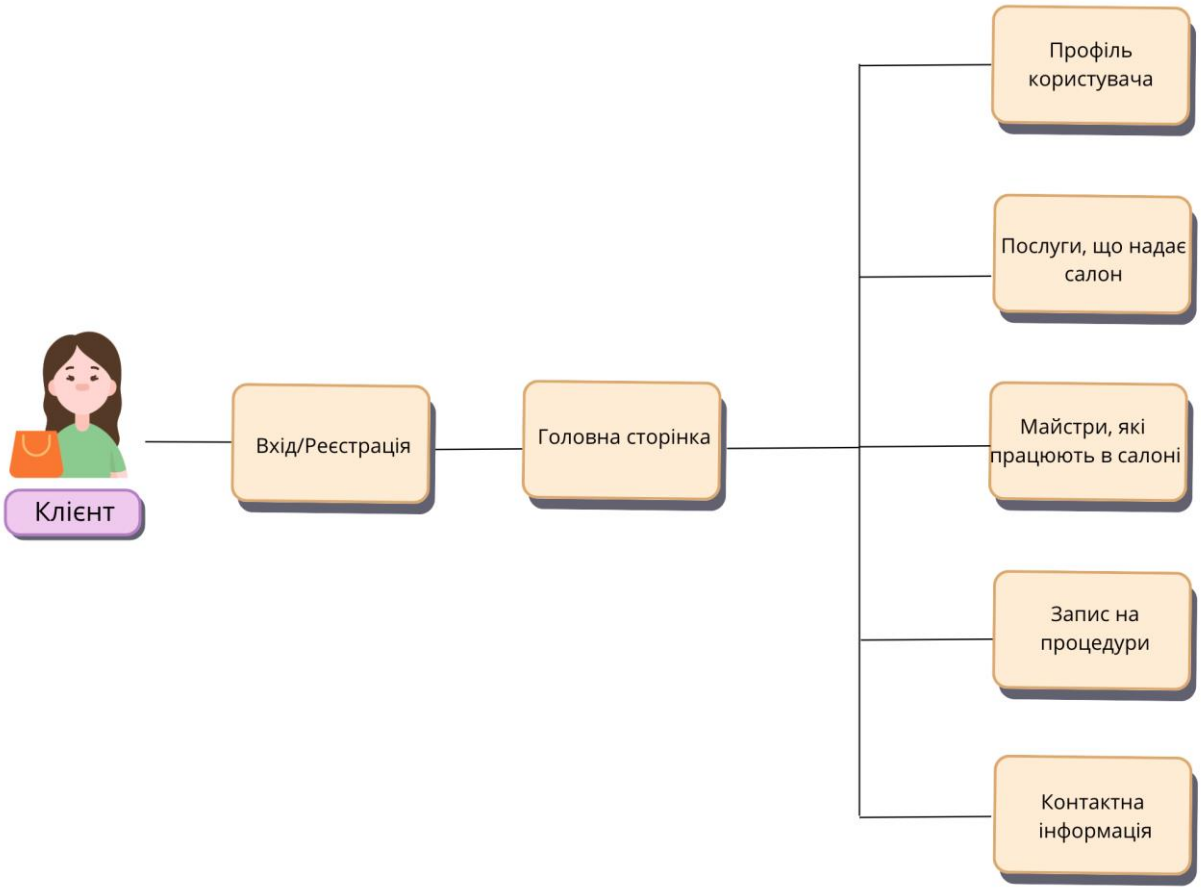


Рисунок 3.3 – Ієрархія форм підсистеми користувача Клієнт

Після входу в систему клієнт бачить головну сторінку з інформацію про філіали салону. А далі перед користувачем постає вибір, він може за потреби управляти інформацію в своєму профілі, тобто редагувати всі наявні поля. Переглядати всі послуги, що пропонує салон, ознайомитись з усіма майстрами та їхніми можливостями. І звичайно є можливість записатися на обрану клієнтом послугу у вільні часи роботи працівників. Також клієнту надається контактна інформація для зв’язку з салоном.

Можливості майстра такі самі як у клієнтів, за винятком сторінки записів на процедури, де працівник сам може записати до себе на послуги та змінити статус запису. Ієрархія екранних форм майстра зображена на рис. 3.4.

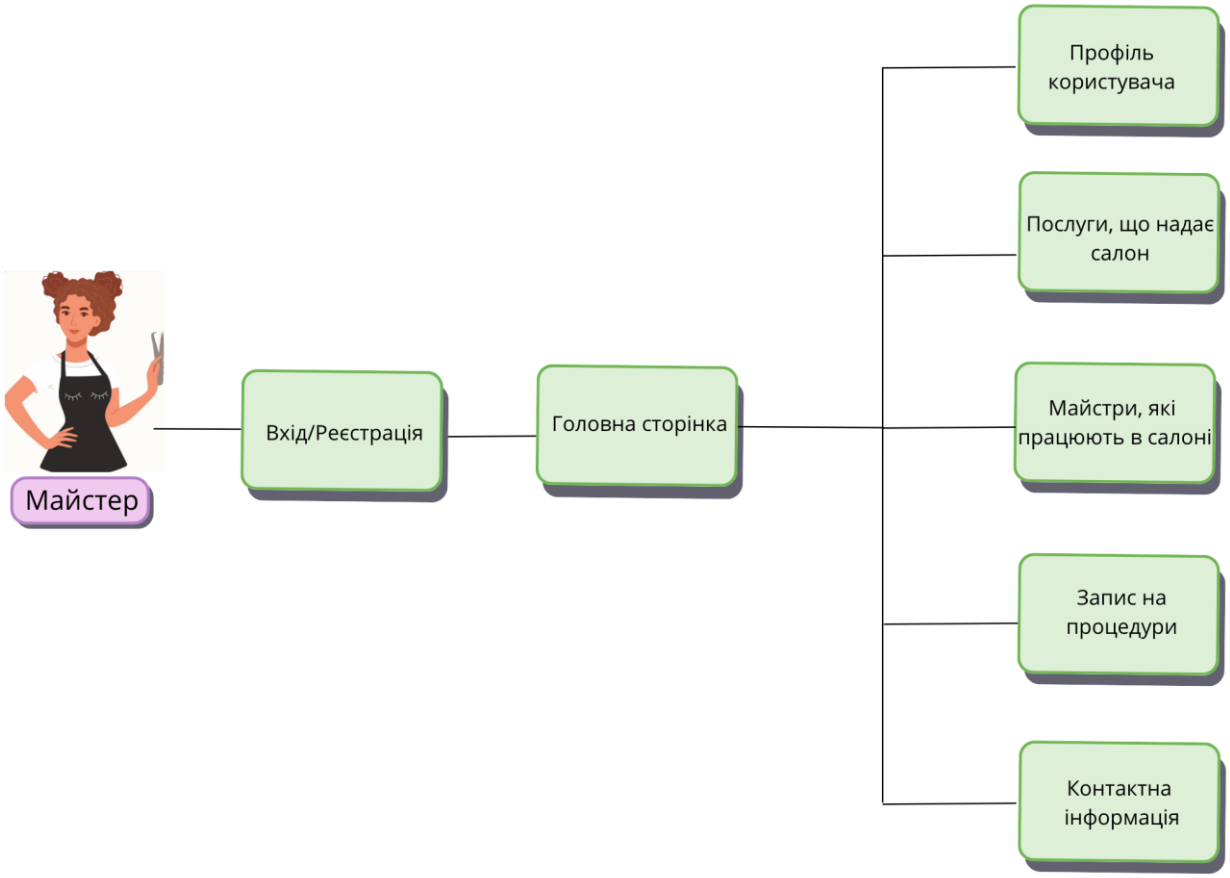


Рисунок 3.4 – Ієрархія форм підсистеми користувача Майстер

Вхід в систему та всі основні сторінки також відносяться і до адміністратора, але його можливості значно більші за інших користувачів. Для початку сторінка звітів поділяється на:

– доходи, де можна відслідкувати прибуток за весь час або в період між обраними датами;

– витрати, де можна відслідкувати розходи за весь час або в період між обраними датами;

– рейтинг майстрів за прибутком, що вони принесли салону останнім місяцем;

– рейтинг популярності послуг;

– відмінені клієнтами записи;

– календар записів усіх майстрів.

Ієрархія екранних форм для користувача Адміністратор зображена на рис. 3.5.

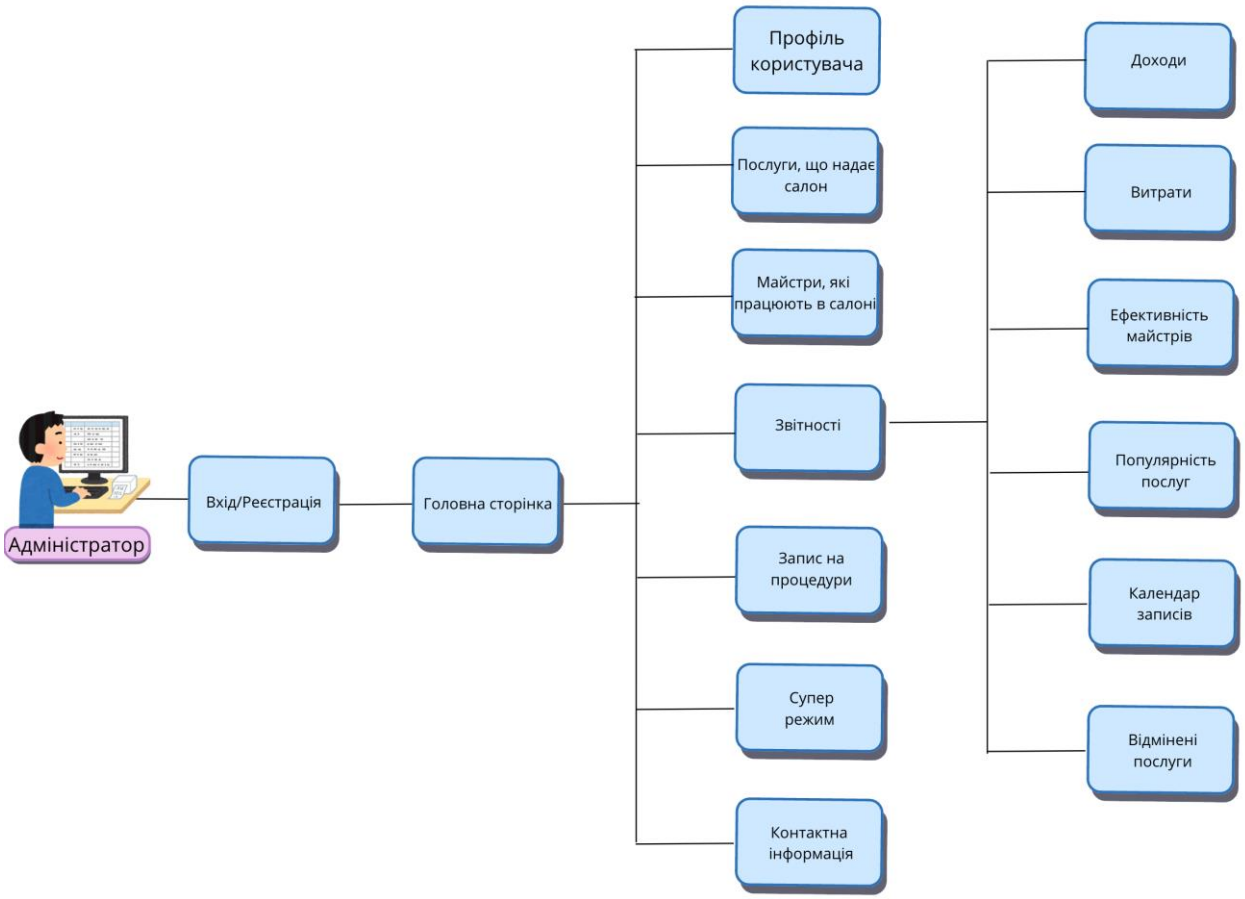


Рисунок 3.5 – Ієрархія форм підсистеми користувача Адміністратор

Також користувач Адміністратор може управляти всією інформацією про майстрів, клієнтів, послуги, філіали салонів, записами та всіма доступними можливостями зайшовши на сторінку супер режим.

3.4 Вибір технологій для реалізації системи

Для створення мережевої системи для організації діяльності філіалів салону краси «Call of Beauty» обрані наступні інструменти.

1) PostgreSQL як система управління базами даних. Цю СУБД обрано через її надійність, масштабованість, що дозволяє ефективно керувати даними

та легко зберігати їх. PostgreSQL добре підходить для розподілення задач між користувачами, а також через безкоштовність використання, що дозволяє зробити розробку системи менш затратною [6].

2) Мова програмування PHP, що широко використовується для розробки веб-застосунків завдяки своїй простоті, гнучкості, забезпеченню безпеки та багатому набору функцій для роботи з веб-серверами. Простота використання дозволяє швидко навчитися і почати розробку, що є важливим фактором при обмежених ресурсах та часу. Легке інтегрування з різними базами даних, в тому числі PostgreSQL є основним фактором вибору [7].

3) Фреймворк Laravel обрано для підвищення ефективності розробки та полегшення створення веб-додатків. Однією з переваг Laravel є те, що він дозволяє будувати логічні архітектури для проектів будь-якої складності та типу. Цей фреймворк характеризується високою продуктивністю та можливістю інтеграції з іншими платформами та бібліотеками [8].

4) Мова програмування Dart пропонує синтаксичну простоту та високу продуктивність завдяки можливості компіляції у машинний код. Що забезпечує швидке виконання програм та плавність роботи, що є критично важливим для систем, які вимагають високої швидкості обробки даних та швидкої реакції на дії користувачів [9].

5) Фреймворк Flatter дозволяє створювати високопродуктивні кросплатформенні додатки з одним спільним кодом для Android, iOS, web та desktop платформ [10].

4 ПРОЕКТУВАННЯ МЕРЕЖІ ФІЛІАЛІВ

4.1 Побудова структурно-логічної схеми комп’ютерної мережі

Після аналізу фізичної топології мережі для відображення планів приміщень і з врахування вимог підприємства, обрана топологія «Дерево», як найбільш оптимальний варіант.

Топологія дерева (Tree Topology) є структурою мережі, яка поєднує в собі особливості топології «Зірка» і «Шина». У такій мережі вузли організовані у вигляді ієрархічної структури, де кожен вузол може мати під собою декілька підключених вузлів. Ця топологія дозволяє покращити ефективність та масштабованість мережі.

У топології дерева існує центральний вузол або комутатор, який виконує роль кореня дерева. Від цього кореня виходять гілки, які представляють собою підключені підмережі або піддерева. Кожен внутрішній вузол може включати в себе піддерева або окремі пристрої, а кінцеві пристрої (комп'ютери, сервери) зазвичай підключені безпосередньо до внутрішніх вузлів. У топології дерева кожен рівень ієрархії представляє собою окремий рівень комутації. Це означає, що дана структура мережі надає гнучкість і можливість ефективного управління трафіком. Крім того, топологія дерева є дуже масштабованою, оскільки можна легко додавати нові гілки чи розширювати існуючі рівні для врахування зростаючого обсягу трафіку та підключених пристроїв [11].

Структурно логічна схема вказує на візуальне представлення структури та логіки взаємодії компонентів мережі. Така топологія має низький ступінь деталізації та відображає концепцію мережі як єдиної інформаційної системи. Структурно-логічні схеми використовуються під час проектування систем та програм для того, щоб зрозуміти та визначити архітектуру системи, розподіл функцій між компонентами та послідовність їхньої роботи.

Для побудови структурно-логічної схеми, застосований графічний редактор Microsoft Visio, а сама схема наведена у додатку Б.

4.2 Вибір активного та пасивного обладнання для реалізації мережі

Перелік обраного пасивного обладнання для мережі представлено далі.

а) Кабель [12] зображено на рис. 4.1:

1) тип кабелю – UTP Cat.5E;
2) виробник – Одесакабель;
3) кількість струменевих жил – 4 пари;
4) матеріал жил – мідь.

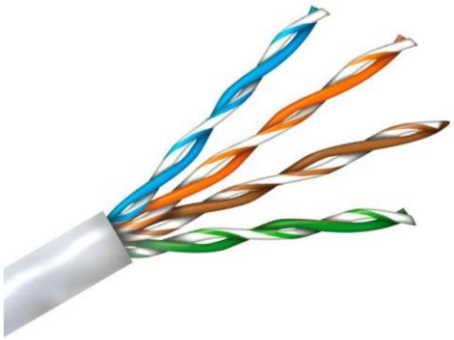


Рисунок 4.1 – Кабель UTP CAT 5E

б) Кабельний канал [13] зображено на рисунку 4.2:

1) довжина короба 2000мм;
2) розмір – 16 х 16 мм;
3) матеріал – важко займистий полівінілхлорид;
4) тип замку – подвійний.

Рисунок 4.2 – Кабельний канал

в) Розетки [14] зображено на рисунку 4.3:

1) тип – RJ45 5 категорія UTP;
2) вид монтажу – Прихований;
3) клас захисту IP20;
4) вага/довжина/ширина – 75 г/85 мм/85 мм.

Рисунок 4.3 – Комп’ютерна розетка

г) Антивандальна шафа В-150 [15] зображено на рисунку 4.4:

1) розмір – Висота 320/ширина 520/глибина 400 мм;
2) кріплення – настінне;
3) вага – 8,47 кг;
4) максимально допустиме навантаження – 60 кг.

Рисунок 4.4 – Антивандальна шафа

Перелік обраного активного обладнання для мережі представлено далі.

а) Маршрутизатор [16] зображено на рисунку 4.5:

1) марка – FoxGate S9524-GS12M2;
2) кількість портів – 24;
3) продуктивність 128Гбіт/с;
4) підтримка стандартного та розширеного ACL;
5) підтримка Voice Vlan.

Рисунок 4.5 – Маршрутизатор

б) Комутатор [17] зображено на рисунку 4.6:

1) марка – FoxGate S6224-S4;
2) 24 порти 10/100BASE-T Ethernet RJ-45;
3) підтримка IEEE802.1Q VLAN, QoS, ACL.

Рисунок 4.6 – Комутатор

в) WIFI роутер [18] зображено на рисунку 4.7:

1) марка – TP-LINK Archer C50;
2) частота роботи WIFI – 5 ГГц + 2.4 ГГц;

3) інтерфейси – 4 порти LAN 10/100 Мбіт/с
4) функції безпеки – захист від DoS атак.

Рисунок 4.7 – WIFI роутер

4.3 Моделювання мережі в середовищі Packet Tracer

У відповідності з побудованою структурно-логічною схемою мережі, в середовищі Cisco Packet Tracer вибрано усе необхідне обладнання, враховуючи кінцеві вузли (ПК користувачів). Крім того вибрано обладнання, аналогічне тому, що описане в розділі 4.2.

Усі кінцеві вузли отримують адреси статично, окрім Wi-Fi пристроїв.

У якості DHCP-серверу використовується роутер. Для коректного налаштування DHCP, статичних адрес та забезпечення безпеки, з локальної мережі було виділено 2 підмережі. Вони приведені в таблиці 4.1.

Таблиця 4. 1 – Адресація мережі

Ім’я підмережі	Адреса	Маска	Діапазон доступних адрес
Salon1	10.8.0.0	255.255.240.0	10.8.0.1-10.8.15.255
SalonN	10.8.16.0	255.255.240.0	10.8.16.1-10.8.31.255

Схема побудована в Cisco Packet Tracer для мережі філіалів салону «Call of Beauty» представлена у додатку В.

Політика безпеки мережі:

– налаштування NAT;

– створення DMZ на серверах страхової компанії;

– налаштування VPN для безпечного мережного зв’язку між офісами;

– створення ACL.

Визначено наступні правила для ACL.

а) DMZ на веб-сервері:

1) дозволено доступ з інтернету в DMZ;

2) дозволено доступ з локальної мережі в DMZ;

3) дозволено доступ з локальної мережі в Інтернет.

б) Salon1:

1) дозволено вихід в інтернет для всіх пристроїв;

2) дозволено доступ до DMZ для всіх пристроїв.

в) SalonN:

1) дозволено вихід в інтернет для всіх пристроїв;

2) дозволено доступ до DMZ для всіх пристроїв.

NAT (Network Address Translation) – технологія трансляції мережевих адрес, тобто підміни адрес в заголовку IP-пакета (іноді може ще і порт міняти в заголовках TCP / UDP).

Перед тим, як налаштувати NAT, налаштуємо зовнішній порт роутера та підмережу інтернет-провайдера (лістинг 4.1).

```
interface FastEthernet0/1
ip address 210.210.1.2 255.255.255.252
ip nat outside
ip route 0.0.0.0 0.0.0.0 210.210.1.1
```

Лістинг 4.1 – Налаштування NAT на зовнішньому порту роутера

За допомогою листів доступу налаштуємо порти для транслювання (лістинг 4.2).

```
permit ip 10.8.0.0 0.0.15.255 any
permit ip 10.8.16.0 0.0.15.255 any
```

Лістинг 4.2 – ACL для доступу комп'ютерів локальної мережі в інтернет

Підключимо сервіс NAT:

```
R1(config)#ip nat inside source list FOR-NAT int fa0/1 overload
```

В зоні DMZ є веб-сервер доступ до якого мають користувачі локальної та зовнішньої мереж. Налаштування DMZ на маршрутизаторі головного філіалу салону краси за допомогою ACL (лістинги 4.3, 4.4).

```
R1(config)#ip access-list extended FROM-OUTSIDE
R1(config-ext-nacl)# permit tcp any any established
R1(config-ext-nacl)# permit icmp any any echo-reply
R1(config-ext-nacl)#permit icmp any any unreachable
R1(config)#int fa0/1
R1(config-if)#ip access-list FROM-OUTSIDE in
```

Лістинг 4.3 – Налаштування DMZ для зовнішньої мережі

```
ip access-list extended FROM-DMZ
permit tcp host 210.210.10.2 any established
permit icmp host 210.210.10.2 any echo-reply
permit icmp host 210.210.10.2 any unreachable
int fa1/0
ip access-group FROM_DMZ in
```

Лістинг 4.4 – Налаштування DMZ для внутрішньої мережі

VPN – узагальнена назва технологій, які дозволяють створювати віртуальні захищені мережі поверх інших мереж із меншим рівнем довіри. VPN-тунель, який створюється між двома вузлами, дозволяє приєднаному клієнту бути повноцінним учасником віддаленої мережі і користуватись її сервісами – внутрішніми сайтами, базами, принтерами, політиками виходу в Інтернет.

Для безпечного обміну пакетами між філіалами салону налаштовано VPN. На прикладі маршрутизатора першого філіалу виконано налаштування VPN (лістинг 4.5).

```
crypto isakmp policy 1
 encr 3des
 hash md5
 authentication pre-share
group 2
crypto isakmp key cisco address 210.210.5.2
crypto ipsec transform-set TS esp-3des esp-md5-hmac
crypto map CMAP 10 ipsec-isakmp
 set peer 210.210.5.2
 set transform-set TS
 match address FOR-VPN
ip access-list extended FOR-VPN
permit ip 10.8.0.0 0.0.15.255 10.8.16.0 0.0.15.255
```

Лістинг 4.5 – Налаштування VPN між філіалами салону

Найбільше, що можна зробити для безпеки в бездротових мережах, це аутентифікацію за паролем. Використовуємо протокол WPA2-PSK з алгоритмом шифрування AES, що є одним з найбільш безпечних та широко використовуваних стандартів захисту бездротових мереж. Він забезпечує високий рівень безпеки завдяки використанню протоколу CCMP (Counter Mode with Cipher Block Chaining Message Authentication Code Protocol).

Приклад налаштування WiFi роутера зображено на рисунку 4.9.

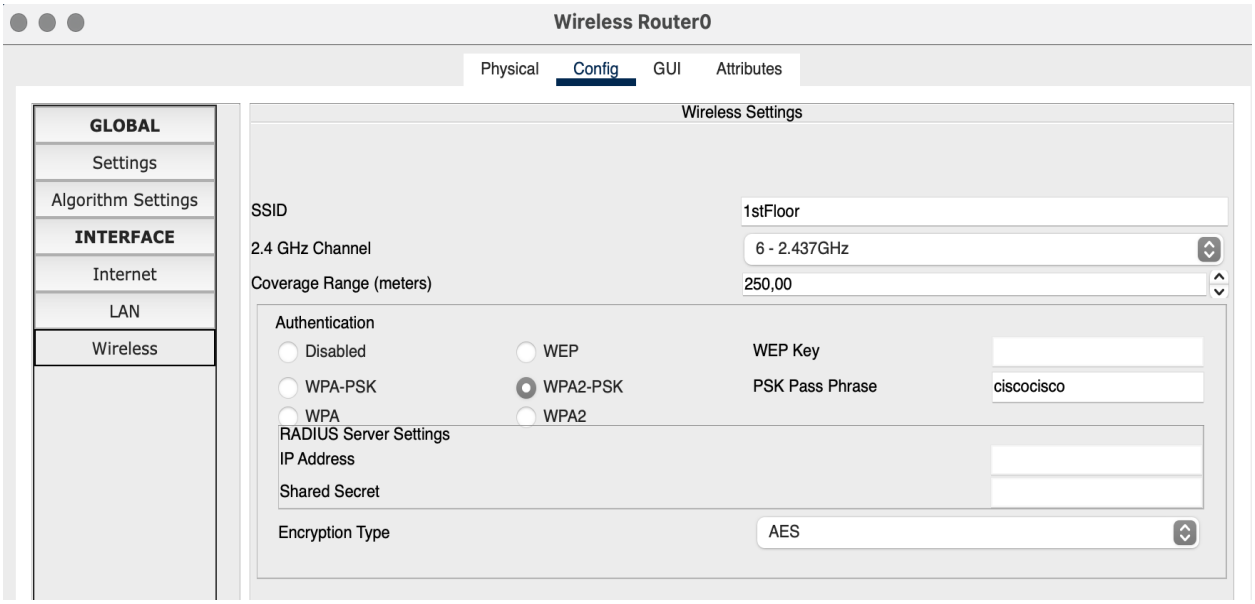


Рисунок 4.9 – Налаштування WiFi роутера

В якості операційної системи серверів обрано Ubuntu Server, якає однією з найпопулярніших операційних систем для серверів у сімействі Linux. Вона базується на Ubuntu та надає стабільну та надійну платформу для розгортання різноманітних серверних застосунків.

Для забезпечення безпеки пристроїв використовується антивірусне забезпечення SentinelOne Antivirus. SentinelOne – це сучасне антивірусне рішення, спрямоване на захист комп'ютерних систем від різних загроз, таких як віруси, шкідливі програми та інші види зловмисного програмного забезпечення.

Для виявлення вторгнень обрано систему Snort, а саме його безкоштовна версія Snort Community Ruleset – це набір правил для системи виявлення і запобігання вторгненням (IDS/IPS).

5 ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ САЛОНІВ КРАСИ

5.1 Створення бази даних

В цьому розділі описані основні об’єкти бази даних та права доступу користувачів до неї. Повний код створення бази даних наведено у додатку Г.

Створення таблиці відбувається з використанням команди CREATE TABLE. Ця команда дозволяє визначити структуру таблиці, включаючи імена стовпців, їх типи даних та обмеження. Загальний синтаксис створення таблиці в PostgreSQL має такий вигляд:

```
CREATE TABLE table_name (
    column1 data_type1 constraints,
    column2 data_type2 constraints,
    ...
);
```

де table_name – ім'я таблиці,

column1, column2, ... – імена стовпців таблиці.

data_type1, data_type2, ... – типи даних для кожного стовпця.

constraints – обмеження для стовпців, такі як NOT NULL, PRIMARY KEY, FOREIGN KEY та інші.

Детально розглянемо запит на створення таблиці Appointments. Для визначення одного з її атрибутів знадобилося створити перелічувальний тип даних:

```
create type Status as enum (
  ('pending',
  'done',
  'cancelled');
```

Створення типу відбувається за допомогою команди CREATE TYPE, далі вказується ім'я типу. За допомогою ENUM (...) перераховані допустимі значення для атрибутів даного типу. Тобто значення статусу запису може бути готовий, відмінений та в очікувані.

Для створення таблиці Appointments використовується наступний запит:

```
CREATE TABLE appointments (
    id SERIAL PRIMARY KEY,
    customer_id integer NOT NULL,
    employee_id integer NOT NULL,
    service_id integer NOT NULL,
    appointment_datetime timestamp(0) without time zone NOT NULL,
    status status,
    FOREIGN KEY (customer_id) REFERENCES customers(id) ON DELETE
CASCADE,
    FOREIGN KEY (employee_id) REFERENCES employees(id) ON DELETE
CASCADE,
    FOREIGN KEY (service_id) REFERENCES services(id) ON DELETE
CASCADE
     );
```

Id – стовпець, що є автоінкрементним та не може бути рівним NULL. Також цей стовпець є первинним ключем для таблиці.

Customer_id – стовпець звичайного числового типу INTEGER, який також не може бути порожнім і є зовнішнім ключем, що посилається на таблицю customers стовпець id.

Employee_id – стовпець звичайного числового типу INTEGER, який також не може бути порожнім і є зовнішнім ключем, що посилається на таблицю employees стовпець id.

Service_id– стовпець звичайного числового типу INTEGER, який також не може бути порожнім і є зовнішнім ключем, що посилається на таблицю services стовпець id.

Appointment_datetime є полям для дати і часу з точністю до секунди. Status є полем типу Status, яке може бути NUL та значення має бути одним з 'pending', 'done' або 'cancelled'.

Зовнішні ключі (FOREIGN KEY) вказують на те, що значення customer_id, employee_id і service_id повинні існувати в відповідних таблицях і будуть видалені каскадно (ON DELETE CASCADE), якщо відповідні записи в цих таблицях видаляються.

5.2 Запити до бази даних для вирішення задач

Далі представлені запити до бази даних, які використовуються у застосунку для отримання, вставки, оновлення або видалення даних з таблиць. Для початку розглянуто задачі, що виконує клієнт.

– Задача (К1) для реєстрації клієнтів. Код для вирішення задачі наведено у лістингу 5.1.

```
INSERT INTO customers (
    user_id,
    customer_type_id,
    created_at,
    updated_at
)
VALUES (
    1,  -- ID користувача
    2,  -- Тип клієнта
    NOW(),  -- Поточний час для created_at
    NOW()   -- Поточний час для updated_at
);
```

Лістинг 5.1 – Код для вирішення задачі К1

– Переглядати інформацію про майстрів (К2). Код для вирішення задачі наведено у лістингу 5.2.

```
SELECT
    e.id AS employee_id, u.first_name, u.last_name,
e.working_hours, e.salary, e.created_at, e.updated_at
FROM
    employees e
INNER JOIN
    user_profiles u ON e.user_id = u.user_id;
```

Лістинг 5.2 – Код для вирішення задачі К2

– Задача (К3) перегляд відгуків. Код для вирішення задачі наведено у лістингу 5.3.

```
SELECT
    id AS review_id, customer_id, rating, comment, created_at,
updated_at
FROM customer_reviews;
```

Лістинг 5.3 – Код для вирішення задачі К3

– Задача (К4) залишити відгук про салон. Код для вирішення задачі наведено у лістингу 5.4.

```
INSERT INTO customer_reviews (
    customer_id,
    rating,
    comment,
    created_at,
    updated_at
)VALUES (
    1,  -- ID клієнта, який залишив відгук
    5,  -- Оцінка (ціле число)
    'Дуже задоволений послугою!',  -- Коментар
    NOW(),  -- Поточний час для created_at
    NOW()   -- Поточний час для updated_at);
```

Лістинг 5.4 – Код для вирішення задачі К4

– Переглядати інформацію про послуги (К5). Код для вирішення задачі наведено у лістингу 5.5.

```
SELECT
    s.id AS service_id, st.name AS service_type, s.name AS
service_name, s.description, s.price, s.expenses, s.duration,
s.created_at, s.updated_at
FROM services s
INNER JOIN
    service_types st ON s.service_type_id = st.id;
```

Лістинг 5.5 – Код для вирішення задачі К5

– Переглядати інформацію про можливі дати запису на послугу (К6). Вирішення цієї задачі відбувається за допомоги функції check_working_hours_and_appointments(), яка знаходиться в додатку Д.

– Редагувати свій обліковий запис (К7). Код для вирішення задачі наведено у лістингу 5.6.

```
UPDATE user_profiles
SET first_name = 'Нове ім'я', last_name = 'Нове прізвище'
WHERE user_id = 1;
```

Лістинг 5.6 – Код для вирішення задачі К7

– Задача (К8) записатись на процедуру. Код для вирішення задачі наведено у лістингу 5.7.

```
INSERT INTO appointments
(customer_id, employee_id, service_id, appointment_datetime,
status)
VALUES (1, 2, 3, '2024-06-01 10:00:00', 'in pending');
```

Лістинг 5.7 – Код для вирішення задачі К8

Далі розглянуто задачі, що виконує робітник, майстер салону та запити для їх вирішення.

– Переглядати свій календар записів (М1). Код для вирішення задачі наведено у лістингу 5.8.

```
SELECT * FROM appointments
WHERE employee_id = 1
AND status = 'pending';
```

Лістинг 5.8 – Код для вирішення задачі М1

– Змінити статус запису (М2). Код для вирішення задачі наведено у лістингу 5.9.

```
UPDATE appointments
SET status = 'cancelled'
WHERE id = 1;
```

Лістинг 5.9 – Код для вирішення задачі М2

– Задача (М3) додавання нових записів на процедуру. Код для вирішення задачі наведено у лістингу 5.10.

```
INSERT INTO appointments (customer_id, employee_id, service_id,
appointment_datetime, status)
VALUES (
1, 2, 3, '2024-06-01 10:00:00', 'in pending');
```

Лістинг 5.10 – Код для вирішення задачі М3

– Переглянути кількість наданих послуг за певний проміжок часу (М4) можна за допомогою функції, що зазначена в додатку Д.

Наступним кроком розглянуто задачі, що виконує адміністратор та запити для їх вирішення.

– Додати майстра (А1). Код запиту для вирішення задачі наведено у лістингу 5.11.

```
INSERT INTO employees (user_id, working_hours, salary, created_at,
updated_at)
VALUES(2,
'[{"day":"Monday","shiftBeginsAt":"08:00","shiftEndsAt":"12:00"}
,{"day":"Tuesday","shiftBeginsAt":"09:00","shiftEndsAt":"13:00"}
,{"day":"Wednesday","shiftBeginsAt":"08:00","shiftEndsAt":"12:00
"},{"day":"Thursday","shiftBeginsAt":"08:00","shiftEndsAt":"12:0
0"},{"day":"Friday","shiftBeginsAt":"12:00","shiftEndsAt":"16:00
"}]',  -- Робочі години у форматі JSON
    30000,  -- Заробітна плата
    NOW(),  -- Дата створення
    NOW()   -- Дата оновлення
);
```

Лістинг 5.11 – Код для вирішення задачі А1

– Видалити майстра (А2). Код запиту для вирішення задачі наведено у лістингу 5.12.

```
DELETE FROM employees
WHERE user_id = 2;
```

Лістинг 5.12 – Код для вирішення задачі А2

– Редагувати майстра (A3). Код для запиту для вирішення задачі наведено у лістингу 5.13.

```
UPDATE employees
SET salary = 55000,
    updated_at = NOW()
WHERE user_id = 2;
```

Лістинг 5.13 – Код для вирішення задачі A3

– Додавання нового клієнта (A4). Код запиту для вирішення задачі наведено у лістингу 5.14.

```
INSERT INTO customers (user_id, customer_type_id, created_at,
updated_at)
VALUES (3,  -- ID користувача в таблиці users дорівнює 3
    1,  -- Припустимо, що ID типу клієнта в таблиці
customer_types дорівнює 1
    NOW(),  -- Дата створення
    NOW()   -- Дата оновлення);
```

Лістинг 5.14 – Код для вирішення задачі A4

– Видалити клієнта (A5). Код запиту для вирішення задачі наведено у лістингу 5.15.

```
DELETE FROM customers
WHERE user_id = 3;
```

Лістинг 5.15 – Код для вирішення задачі A5

– Редагувати клієнта (A6). Код запиту для вирішення задачі наведено у лістингу 5.16.

```
UPDATE customers
SET customer_type_id = 2  -- Нове значення customer_type_id
WHERE user_id = 3;
```

Лістинг 5.16 – Код для вирішення задачі A6

– Задача (А7) додати нову послугу. Код запиту для вирішення задачі наведено у лістингу 5.17.

```
INSERT INTO services (
    service_type_id, name, description, price, expenses,
duration, created_at, updated_at )
VALUES (
    1,  -- Припустимо, що ID типу послуги в таблиці
service_types дорівнює 1
    'Haircut',  -- Назва послуги
    'Basic haircut service',  -- Опис послуги
    30.00,  -- Ціна послуги
    10.00,  -- Витрати на послугу
    '00:30:00',  -- Тривалість послуги
    NOW(),  -- Дата створення
    NOW()   -- Дата оновлення);
```

Лістинг 5.17 – Код для вирішення задачі А7

– Видалити послугу (А8). Код запиту для вирішення задачі наведено у лістингу 5.18.

```
DELETE FROM services
WHERE name = 'Haircut';
```

Лістинг 5.18 – Код для вирішення задачі А8

– Редагувати послугу (А9). Код запиту для вирішення задачі наведено у лістингу 5.19.

```
UPDATE services
SET
    name = 'Luxury Haircut',  -- Нова назва
    price = 50.00,  -- Нова ціна
    expenses = 15.00,  -- Нові витрати
    duration = '01:00:00',  -- Нова тривалість
    description = 'Luxury haircut service with additional
benefits',  -- Новий опис
    updated_at = NOW()  -- Оновлення часу
WHERE id = 1;
```

Лістинг 5.19 – Код для вирішення задачі А9

– Додати новий тип послуги (A10). Код запиту для вирішення задачі наведено у лістингу 5.20.

```
INSERT INTO service_types (
    name,
    description)
VALUES (
    'Hair Services',  -- Назва типу послуги
    'All types of hair-related services including haircuts,
styling, and treatments.'  -- Опис типу послуги
);
```

Лістинг 5.20 – Код для вирішення задачі A10

– Видалити тип послуги (A11). Код запиту для вирішення задачі наведено у лістингу 5.21.

```
DELETE FROM service_types
WHERE name = 'Hair Services';
```

Лістинг 5.21 – Код для вирішення задачі A11

– Додати новий філіал салону (A12). Код запиту для вирішення задачі наведено у лістингу 5.22.

```
INSERT INTO salons (
    city,
    address,
    created_at,
    updated_at
)
VALUES (
    'Kyiv',  -- Місто, де розташований салон
    '123 Main Street',  -- Адреса салону
    NOW(),  -- Поточний час для created_at
    NOW()   -- Поточний час для updated_at);
```

Лістинг 5.22 – Код для вирішення задачі A12

– Видалити філіал салону (A13). Код запиту для вирішення задачі наведено у лістингу 5.23.

```
DELETE FROM salons
WHERE city = 'Kyiv' AND address = '123 Main Street';
```

Лістинг 5.23 – Код для вирішення задачі А13

– Видаляти записи клієнтів (А14). Код запиту для вирішення задачі наведено у лістингу 5.24.

```
DELETE FROM appointments
WHERE customer_id = 1;
```

Лістинг 5.24 – Код для вирішення задачі А14

– Переглядати звітності про доходи та розходи (А15) можна за допомогою функції, що зазначена в додатку Д.

– Переглядати звітності про ефективність майстрів за прибутком (А16). Код запиту для вирішення задачі наведено у лістингу 5.25.

```
CREATE OR REPLACE FUNCTION
calculate_master_efficiency(month_input date)
RETURNS TABLE (
    first_name character varying(255),
    last_name character varying(255),
    total_revenue double precision
) AS $$
BEGIN
    RETURN QUERY
    SELECT
        u.first_name,
        u.last_name,
        SUM(s.price - s.expenses) AS total_revenue
    FROM
        appointments a
        INNER JOIN employees e ON a.employee_id = e.id
        INNER JOIN services s ON a.service_id = s.id
        INNER JOIN user_profiles u ON e.user_id = u.user_id
    WHERE
        DATE_TRUNC('month', a.appointment_datetime) =
DATE_TRUNC('month', month_input)
        AND a.status = 'done'
```

Лістинг 5.25 – Код для вирішення задачі А16

```
GROUP BY
        u.first_name,
        u.last_name
    ORDER BY
        total_revenue DESC;
END;
$$ LANGUAGE plpgsql;
```

Лістинг 5.25, аркуш 2

– Переглядати звітності про відмінені клієнтом записи (A17). Код запиту для вирішення задачі наведено у лістингу 5.26.

```
CREATE OR REPLACE FUNCTION cancelled_services_list()
RETURNS TABLE
(
    service_name character varying(255),
    appointment_datetime timestamp without time zone,
    status character varying(255)
)
AS $$
BEGIN
    RETURN QUERY
    SELECT
        s.name AS service_name,
        s. description AS service_description,
        a.appointment_datetime,
        a.status
    FROM
        appointments a
    INNER JOIN
        services s ON a.service_id = s.id
    WHERE
        a.status = 'cancelled'
    ORDER BY
        a.appointment_datetime DESC;
END;
$$ LANGUAGE plpgsql;
```

Лістинг 5.26 – Код для вирішення задачі A17

– Переглядати інформацію про популярність послуг (A18). Код запиту для вирішення задачі наведено у лістингу 5.27.

```
CREATE OR REPLACE FUNCTION service_popularity(month_input date)
RETURNS TABLE (
    service_name character varying(255),
    total_revenue double precision,
    requests_count bigint
)
AS $$
BEGIN
    RETURN QUERY
    SELECT
        ser.name AS service_name,
        SUM(s.price - s.expenses) AS total_revenue,
        COUNT(*) AS requests_count
    FROM
        appointments a
        INNER JOIN
        services s ON a.service_id = s.id
        INNER JOIN
        service_types ser ON s.service_type_id = ser.id
    WHERE
        DATE_TRUNC('month', a.appointment_datetime) =
DATE_TRUNC('month', month_input)
        AND a.status = 'done'
    GROUP BY
        ser.name
    ORDER BY
        requests_count DESC;
END;
$$ LANGUAGE plpgsql;
```

Лістинг 5.27 – Код для вирішення задачі A18

– Перегляд графіку роботи майстрів (A19). Код запиту для вирішення задачі наведено у лістингу 5.28.

```
SELECT *
FROM
appointments
WHERE
employee_id = 1
AND
status = 'pending';
```

Лістинг 5.28 – Код для вирішення задачі A19

5.3 Безпека інформаційної системи

Безпека на рівні БД забезпечується шляхом створення ролей користувачів і надання їм відповідних привілеїв. В інформаційній системі Call of beauty реалізовано 3 ролі: Адміністратор, Робітник (майстер салону), Клієнт. У таблиці 9.1 наведені привілеї ролей на таблиці та елементи бази даних. При цьому використані наступні позначення:

– C (Create) – створення (додавання) даних;
– R (Read) – читання даних;
– U (Update) – оновлення даних;
– D (Delete) – видалення даних.

Таблиця 5.1. – Права доступу користувачів до бази даних

Об’єкти БД	Ролі		
	Адміністратор	Робітник	Клієнт
Таблиці			
Users	CRUD	-	-
User_profiles	CRUD	RU	CRU
Employee	CRUD	R	R
Salons	CRUD	R	R
Customers	CRUD	R	-
Customer Review	RD	R	CR
Customer Type	CRUD	R	-
Skills	CRUD	R	R
Service	CRUD	R	R
Service Type	CRUD	R	R
Appointments	CRUD	CRU	CRU

Продовження таблиці 5.1

Об’єкти БД	Ролі		
	Адміністратор	Робітник	Клієнт
Функції			
get_service_rank_by_employee	E	E	
calculate_expenses_by_date	E		
calculate_monthly_expenses	E		
calculate_income_by_month	E		
get_available_slots	E	E	E
cancelled_services_list	E		
ervice_popularity	E		
calculate_master_efficiency	E		

Створення ролей і наділення їх привілеями відповідно до наведеної вище таблиці здійснюється за допомогою наступних SQL-запитів:

```
    -- клієнт -----------------------------------
CREATE ROLE customer WITH LOGIN
GRANT INSERT, SELECT, UPDATE ON TABLE public."User_profiles" TO
customer;
GRANT SELECT ON TABLE public."Employee" TO customer;
GRANT SELECT ON TABLE public."Salons" TO customer;
GRANT SELECT ON TABLE public."Customer_Review" TO customer;
GRANT SELECT ON TABLE public."Skills" TO customer;
```

Лістинг 5.29 – Наділення привілеями відповідно до ролей

```
GRANT SELECT ON TABLE public."Service" TO customer;
GRANT SELECT ON TABLE public."Service_Type" TO customer;
GRANT INSERT, SELECT, UPDATE ON TABLE public."Appointments" TO
customer;
-- майстер ------------------------------------
CREATE ROLE employee WITH LOGIN;
GRANT SELECT, UPDATE ON TABLE public."User_profiles" TO
employee;
GRANT SELECT ON TABLE public."Employee" TO employee;
GRANT SELECT ON TABLE public."Salons" TO employee;
GRANT SELECT ON TABLE public."Customers" TO employee;
GRANT SELECT ON TABLE public."Customers" TO employee;
GRANT SELECT ON TABLE public."Customer_Review" TO employee;
GRANT SELECT ON TABLE public."Customer_Type" TO employee;
GRANT SELECT ON TABLE public."Skills" TO employee;
GRANT SELECT ON TABLE public."Service" TO employee;
GRANT SELECT ON TABLE public."Service_Type" TO employee;
GRANT SELECT, INSERT, UPDATE ON TABLE public."Appointments" TO
employee;
-- Адміністратор ----------------------------------------
CREATE ROLE admin  WITH LOGIN;
GRANT SELECT, DELETE ON ALL TABLES IN SCHEMA public TO admin;
GRANT INSERT, UPDATE ON TABLE public."Users" TO admin;
GRANT INSERT, UPDATE ON TABLE public."User_profiles" TO admin;
GRANT INSERT, UPDATE ON TABLE public."Employee" TO admin;
GRANT INSERT, UPDATE ON TABLE public."Salons" TO admin;
GRANT INSERT, UPDATE ON TABLE public."Customers" TO admin;
GRANT INSERT, UPDATE ON TABLE public."Customer_Type" TO admin;
GRANT INSERT, UPDATE ON TABLE public."Skills" TO admin;
GRANT INSERT, UPDATE ON TABLE public."Service" TO admin;
GRANT INSERT, UPDATE ON TABLE public."Service_Type" TO admin;
GRANT INSERT, UPDATE ON TABLE public."Appointments" TO admin;
```

Лістинг 5.29, аркуш 2

5.4 Інструкція користувача

Всі користувачі системи «Call of beauty», для початку мають пройти автентифікацію. Для цього на сторінці входу потрібно вести свої данні, так як зображено на рисунку 5.1. Якщо користувач ще не зареєстрований, він може це зробити натиснувши на реєстрація та заповнивши всі необхідні поля (рис. 5.2).

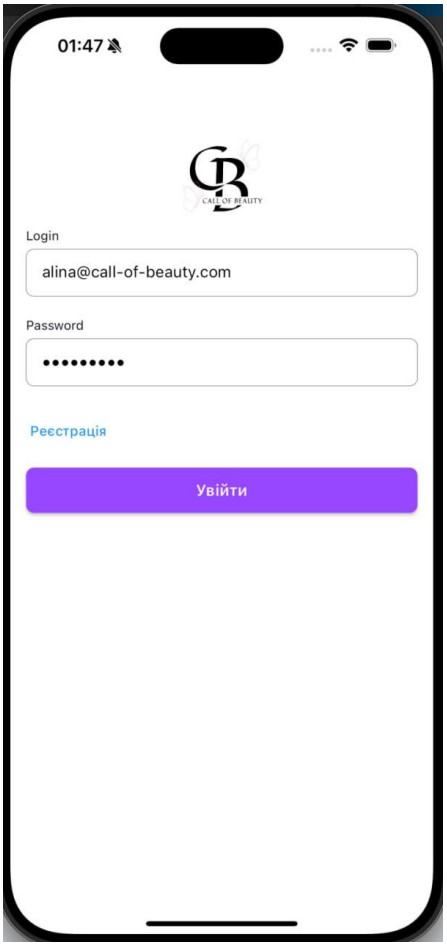


Рисунок 5.1 – Вхід в систему

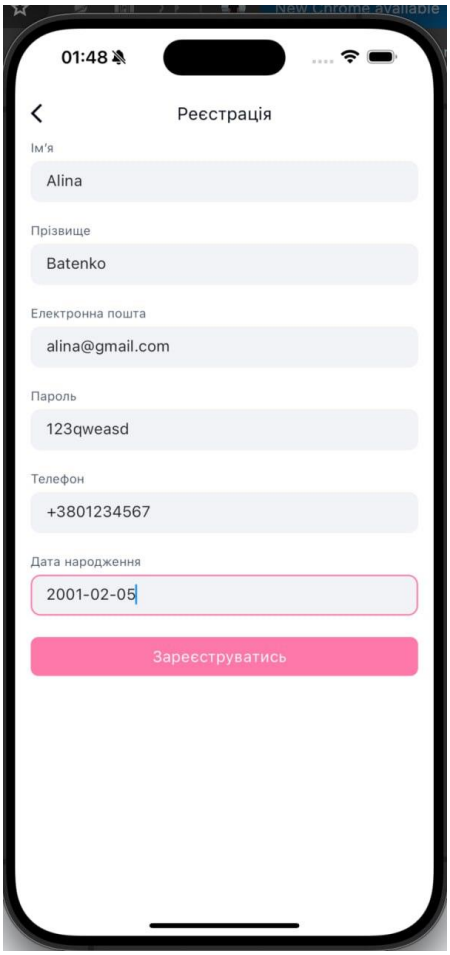


Рисунок 5.2 – Реєстрація в системі

Ввійшовши в систему користувач бачить головну сторінку з логотипом компанії та адреси філіалів салону краси, куди клієнт може завітати на процедури (рис. 5.3). Долі у лівому верхньому кутку знаходить бокове меню, натиснувши туди, можна побачити всю ієрархію сторінок для користувача адміністратор. Це головна сторінка, профіль, послуги, майстри, запис на

процедури, контактна інформація, звіти та супер режим. Меню зображено на рисунку 5.4.

Рисунок 5.3 – Головна сторінка

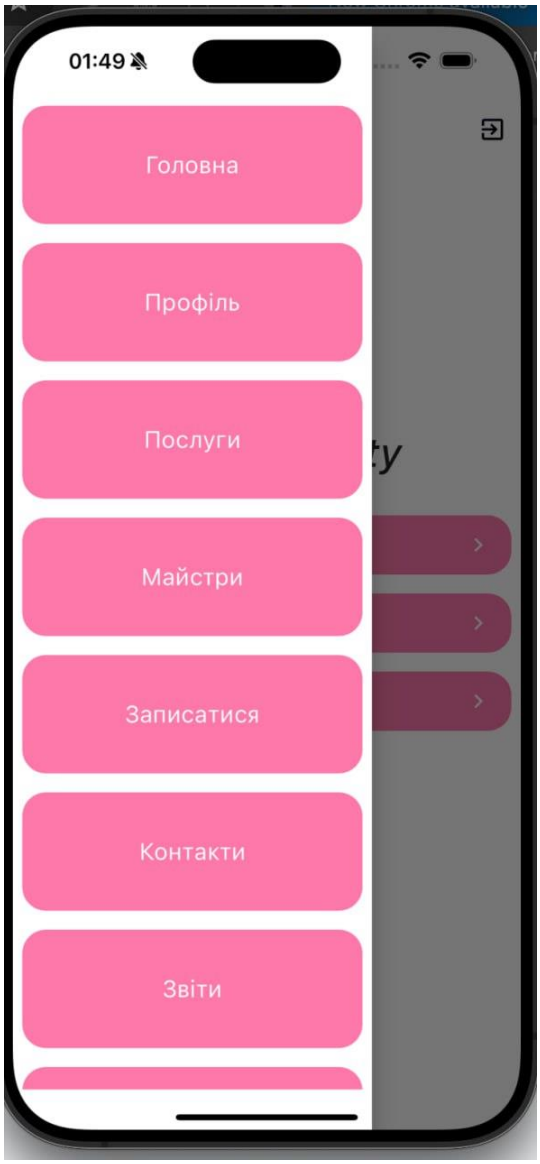


Рисунок 5.4 – Головне меню

Наступним кроком розглянуто профіль користувача, там знаходиться вся інформація про нього: ім’я, прізвище, електрона пошта, телефон та дата народження. На цій сторінці користувач також може змінити дані в наявних полях та зберегти їх. На рисунку 5.5 зображено профіль користувача. Далі можна переглянути послуги, що надає салон. На рисунку 5.6 продемонстровані типи послуг, натиснувши на які відкриваються конкретні процедури в своїй

категорії. Тобто в сферу догляду за волоссям входять ботокс, кератин. До категорії вії відноситься їх ламінування, фарбування та нарощення. У вкладці брови є ламінування, фарбування і корекція. Також в категорії макіяж салон пропонує денний, вечірній та весільний святковий макіяж. Список працівників салону можна переглянути в розділі майстри, що знаходиться в розділі головного меню. Майстри салону зображено на рисунку 5.7.

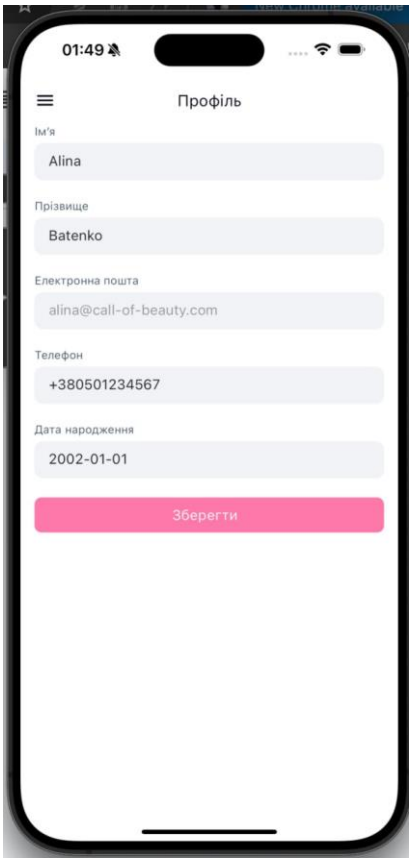
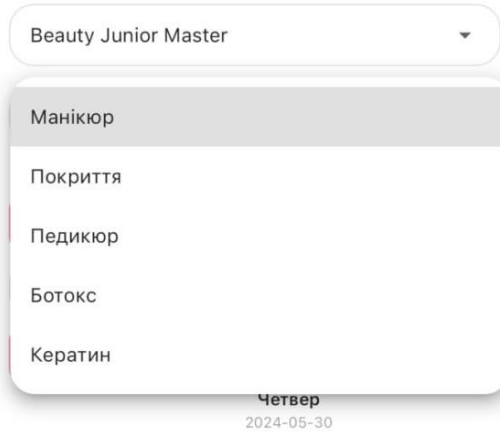


Рисунок 5.5 – Профіль Рисунок 5.6 – Послуги Рисунок 5.7 – Майстри

Визначившись з процедурою на яку клієнт хоче записатись та майстром, що буде її надавати, користувачу слід завітати до сторінки записів з головного меню. Вона зображена на рисунку 5.8. Там відкриється вікно, де можна обрати з списку наявних майстрів салону краси та послугу, що робить цей робітник (рис. 5.9). Також клієнт у зручному форматі календаря, обирає дату та час запису на процедуру.



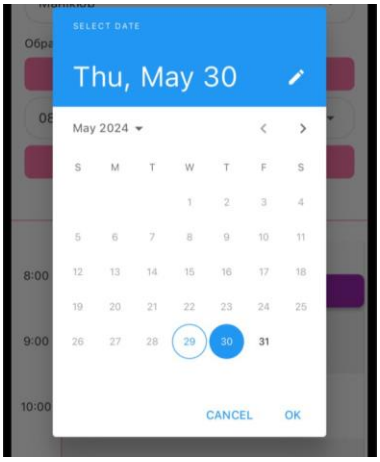


Рисунок 5.8 – Вибір процедури

Рисунок 5.9 – Вибір дати

Повне вікно запису на процедуру зображено на рисунку 5.10.

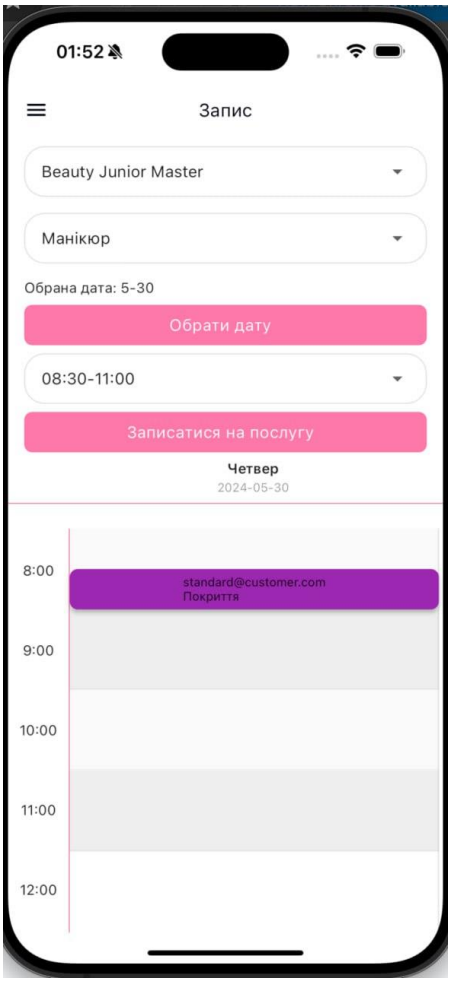


Рисунок 5.10 – Вікно запису на процедуру

Далі розглянуто найцікавіший розділ «Звіти», що доступний лише користувачу адміністратор. До цього розділу входять підрозділи зображені на рисунку 5.11. А також окремо продемонстровані на рисунках нижче.

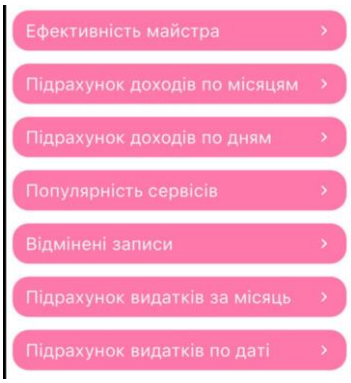
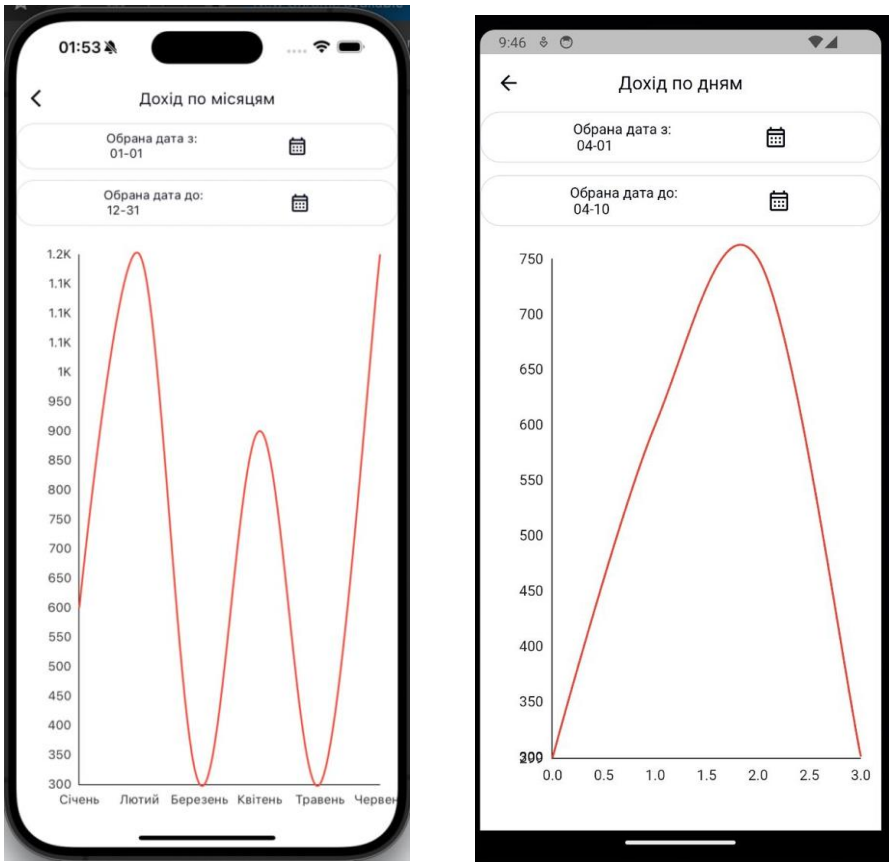


Рисунок 5.11 – Різновидність звітів

Графіки доходів за певний період в місяцях або днях (рис 5.12).

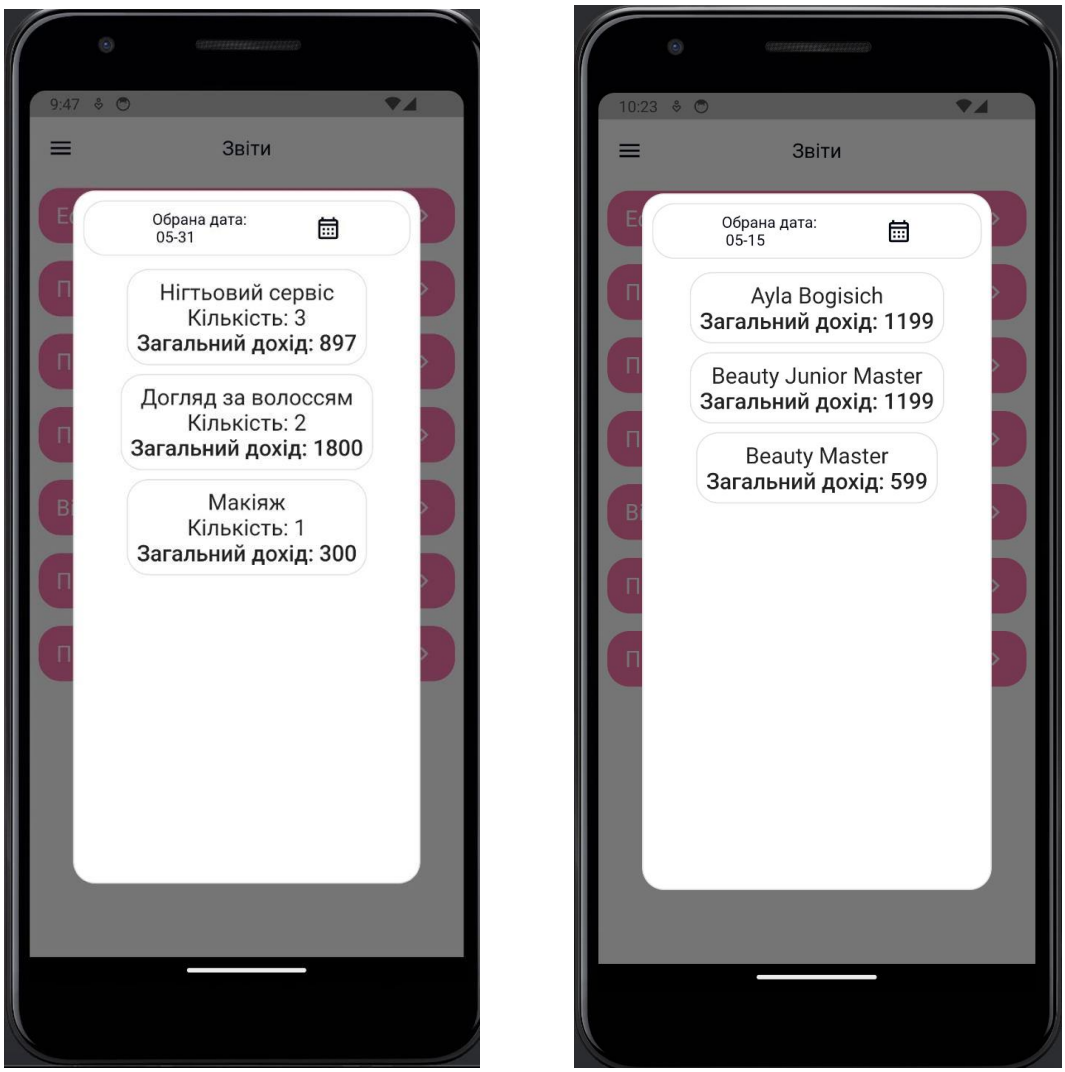


а) б)

Рисунок 5.12 – Статистика доходів за місяцями (а), днями (б)

Для перегляду доходів потрібно обрати дві дати, між якими будуть вираховуватись витрати з ціни на послугу та множитись на кількість послуг . Графік будується по місяцям та дням, між двома обраними датами, що зручно зробити за допомогою календаря.

На рисунку 5.13 зображені звіти, які надають рейтингову інформацію про популярність послуг та популярність майстрів, що їх надають. Додатково до рейтингу зазначається кількість наданих процедур за цей період та сумарна кількість доходів за ними.



а) б)

Рисунок 5.13 – Статистика: популярність послуг (а), майстрів (б)

До того ж, лише адміністратор може переглядати список відмінених послуг (рис 5.14).

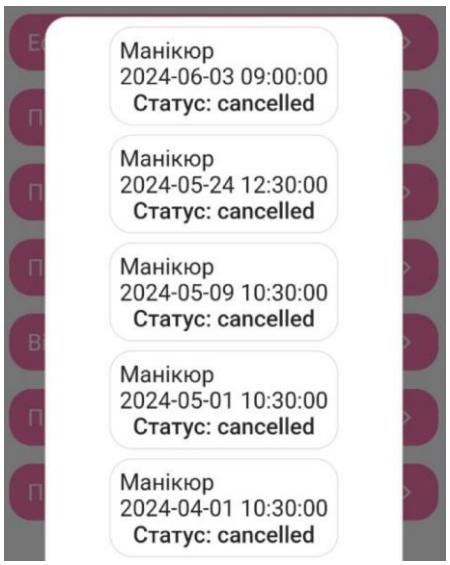
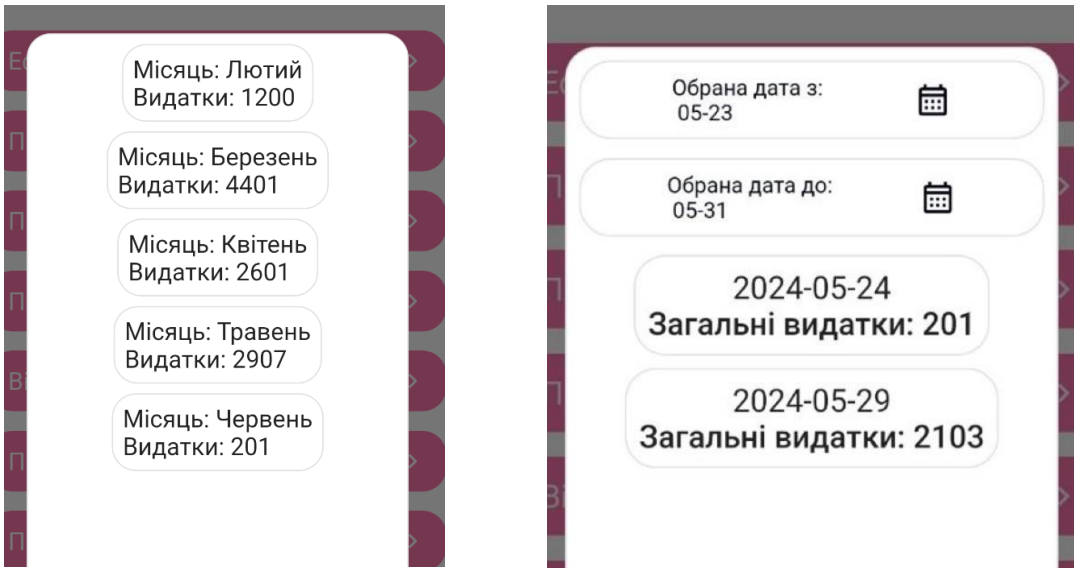


Рисунок 5.14 – Список відмінених послуг

Витрати можна перевірити за днями та місяцями, що зображено на рисунку 5.15.



а) б)

Рисунок 5.15 – Вікно витрат за місяцями (а), днями (б)

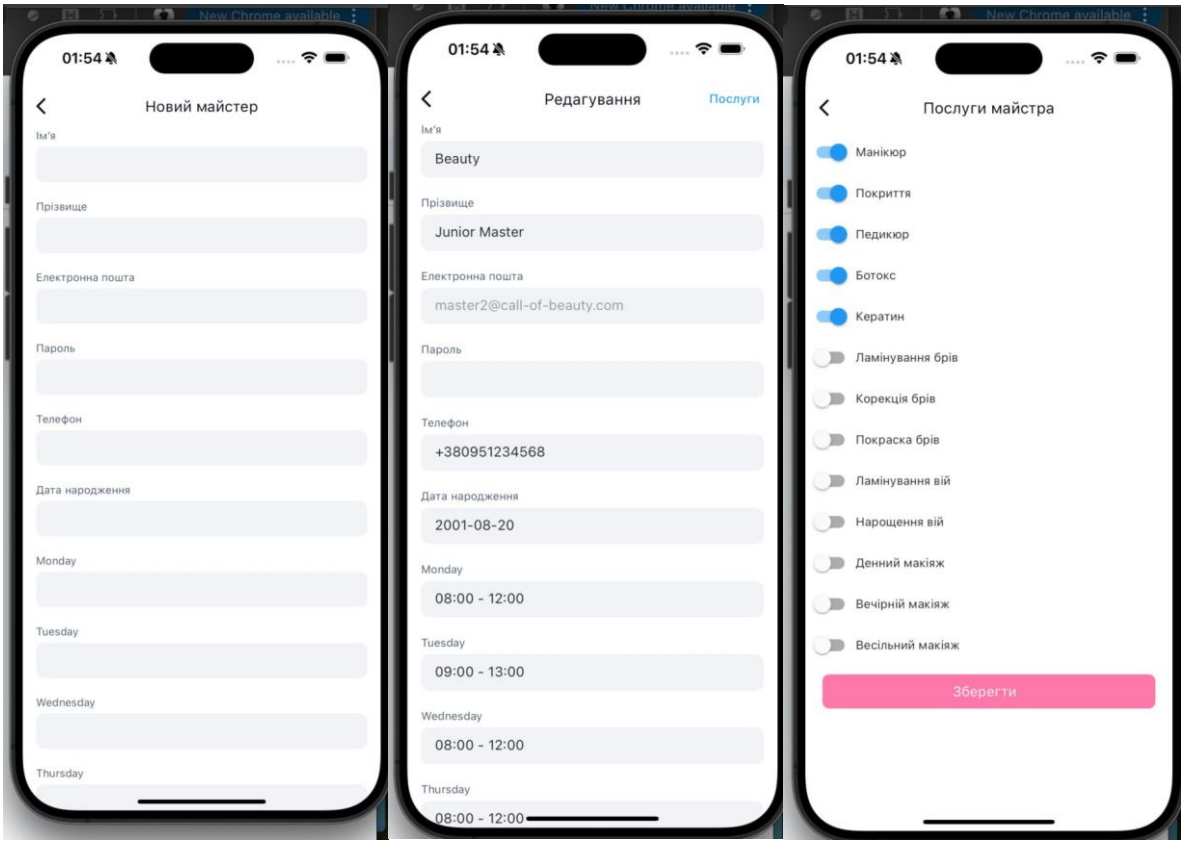
Також з головного меню можна обрати сторінку «Супер режим», що забезпечує управління всією інформацією про майстрів, клієнтів, послуги, філіали салонів, записами та всіма доступними можливостями, що відображено на рисунку 5.16.

а) б) в)

Рисунок 5.16 – Додавання: клієнтів (а), майстрів (б), послуг (в)

Наприклад, розглянуто процес додавання нового майстра до салону краси. Для цього необхідно вказати такі дані, як його ім'я, прізвище, електронну пошту, номер телефону, дату народження, а також дні та години роботи майстра.

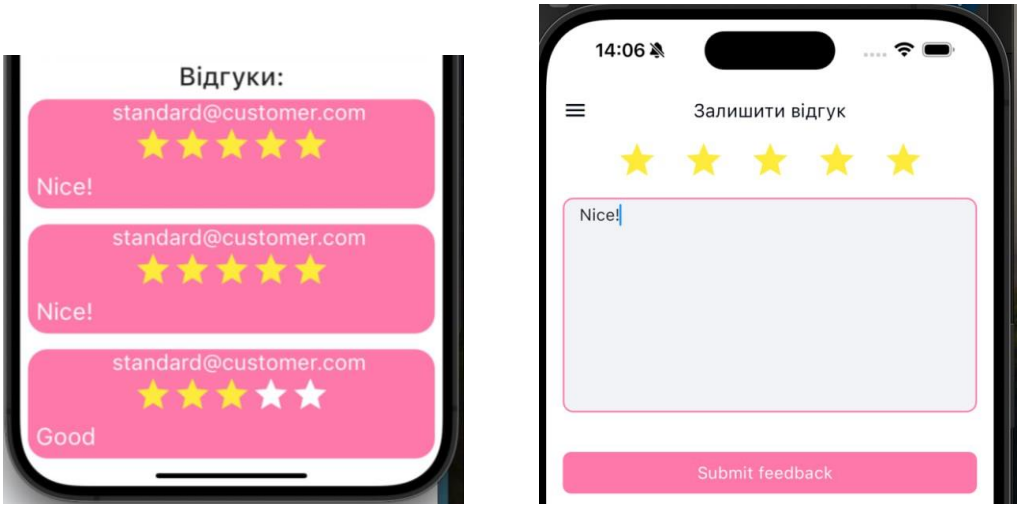
Крім того, система надає можливість редагування даних вже існуючих майстрів. Це включає зміну їхніх персональних даних, а також оновлення списку послуг, які вони надають. Ці ж самі поля можна редагувати та додавати нові навички, через відповідні послуги для майстра. Вікно для редагування стає доступним лише при виборі конкретного майстра. Процес додавання та редагування майстрів детально зображений на рисунку 5.17, де показано всі кроки та можливості, які надає система. Завдяки цій функціональності, керівники можуть ефективно управляти ресурсами салону та забезпечувати високий рівень обслуговування клієнтів.



а) б) в)

Рисунок 5.17 – Додавання та редагування: клієнтів (а), майстрів (б), послуг (в)

Також у клієнтів системи є можливість переглядати та залишати відгуки, що зображено на рисунку 5.18.



а) б)

Рисунок 5.18 – Перегляд відгуків (а), додавання відгуків (б)

Контактна інформація для зв’язку з салоном зображена на рисунку 5.19.

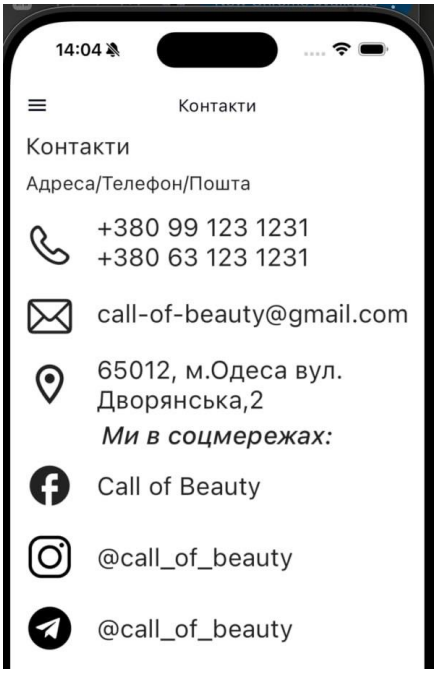


Рисунок 5.19 – Контактна інформація

Функціонал та вигляд web й desktop додатків нічим не відрізняється. Для прикладу на рисунку 5.20 зображена головна сторінка сайту.

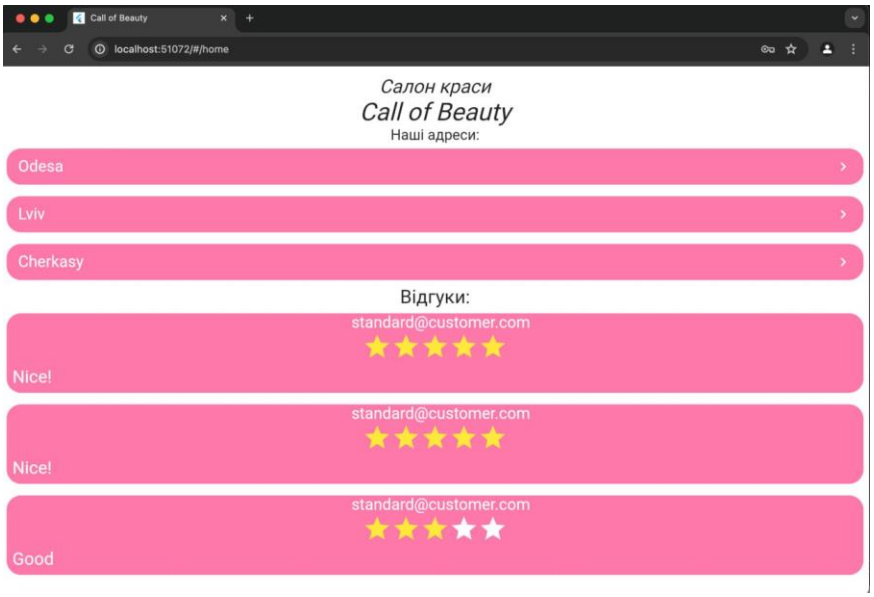


Рисунок 5.20 – Головна сторінка сайту

ВИСНОВКИ

Під час виконання кваліфікаційної роботи проаналізовано бізнес-процеси, що відбуваються у салонах краси, окреслено коло користувачів та виявлено їхні вимоги до створюваної системи, а також оглянуто можливості існуючих ІС для автоматизації ідентифікованих бізнес-процесів. В результаті аналізу конкурентних програмних продуктів, таких як EasyWeek, BloknotApp та Booksy, виявлено, що наявні інформаційні системи для салонів краси мають певні обмеження, такі як недостатня гнучкість налаштувань, обмежений функціонал для управління фінансами та висока вартість.

Створена система «Call of Beauty» враховує виявлені недоліки конкурентів та поєднує їхні переваги для використання конкретної мережі салонів. Система призначена для автоматизації бізнес-процесів салонів краси та забезпечення більшої ефективності управління. Вона має гнучку архітектуру, що дозволяє легко налаштовувати її під конкретні потреби кожного салону, додаючи нових майстрів та клієнтів, редагуючи послуги та керуючи філіалами. «Call of Beauty» включає розподілену автоматизовану систему управління з можливістю VPN-з'єднань між філіалами, що забезпечує безпеку та зручність обміну даними.

Архітектура створеної системи відповідає триланковій моделі клієнт-сервер, що дозволяє ефективно розділити функціональність між різними компонентами системи. Для зберігання інформації спроектована база даних, яка працює під управлінням СУБД PostgreSQL. Система розроблена за допомогою сучасних технологій для створення веб-застосунків, таких як мови програмування PHP та Dart, а також фреймворки Laravel та Flutter.

Розроблена мережева система допомагає полегшити та автоматизувати велику кількість бізнес-процесів у салонах краси, сприяє покращенню комунікації та співпраці між майстрами та клієнтами, значно спрощуючи їх взаємодію. Це забезпечує оперативне вирішення проблем, швидке формування документації та підвищення ефективності управління бізнесом.

Перспективи розвитку системи «Call of Beauty» полягають у розширенні її функціоналу для охоплення ширшого спектру бізнес-процесів, інтеграції з новими технологіями та адаптації до змінних потреб ринку. Це включає додавання нових інструментів для маркетингових кампаній, розширення можливостей аналітики та інтеграцію з іншими сервісами для покращення користувацького досвіду.

За результатами даної роботи опубліковано тези на всеукраїнській конференції.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. CleverBox [Електронний ресурс] – Режим доступу: https://cleverbox-crm.com/
2. BloknotApp [Електронний ресурс] – Режим доступу: https://bloknotapp.com/
3. Booksy [Електронний ресурс] – Режим доступу: https://booksy.com/biz/en-us
4. Батенко А.І., Автоматизація діяльності підприємства малого бізнесу у промисловості краси / А.І. Батенко, І. М. Шпинарева // Інформатика, інформаційні системи та технології: тези доповідей двадцять першої всеукраїнської конференції студентів і молодих науковців. Одеса, 28 квітня 2024 р. – Одеса, 2024. – с. 141-143 [Електронний Ресурс] / Режим доступу:http://dspace.pdpu.edu.ua/bitstream/123456789/19142/1/2024%20Zbirka_tez_IIS%26T-2024.pdf
5. MVC [Електронний Ресурс] / Режим доступу: https://developer.mozilla.org/en-US/docs/Glossary/MVC
6. PostgreSQL: The World's Most Advanced Open Source Relational Database [Електронний ресурс] / Режим доступу: https://www.postgresql.org/
7. PHP [Електронний ресурс] / Режим доступу: https://www.php.net/
8. The PHP Framework for Web Artisans [Електронний ресурс] / Режим доступу: https://laravel.com
9. Flutter [Електронний ресурс] / Режим доступу: https://flutter.dev/
10. Dart [Електронний ресурс] / Режим доступу: https://dart.dev/
11. Топологія мереж [Електронний ресурс] / Режим доступу: https://kremenetskyy.blogspot.com/2017/10/blog-post.html
12. Кабель [Електронний ресурс] / Режим доступу: https://ohrana.ua
13. Кабельний канал [Електронний ресурс] / Режим доступу: https://nika-electro.com.ua/ru/kabel-kanal-16kh16-uec-96-m/
14. Антивандальна шафа [Електронний ресурс] / Режим доступу: https://ohrana.ua/peredacha-video-kabel/antivandalnie-yashchiki-i-seyfi

15. Розетка [Електронний ресурс] / Режим доступу: https://rozetka.com.ua/schneider_electric_asfora_eph4300121/p5180612/
16. Маршрутизатор [Електронний ресурс] / Режим доступу: https://www.mit-net.com.ua/товар/керований-ethernet%20комутатор-foxgate-s9524-gs12m2/
17. Комутатор [Електронний ресурс] / Режим доступу: https://klaster.ua/shop/peredacha-dannyh/setevoe-oborudovanie/setevye-kommutatory/upravlyaemyy-kommutator-l2-s6224-s4/
18. Роутер [Електронний ресурс] / Режим доступу: https://rozetka.com.ua/24747751/p24747751/

ДОДАТОК А

ER-діаграма для салонів краси «Call of Beauty»

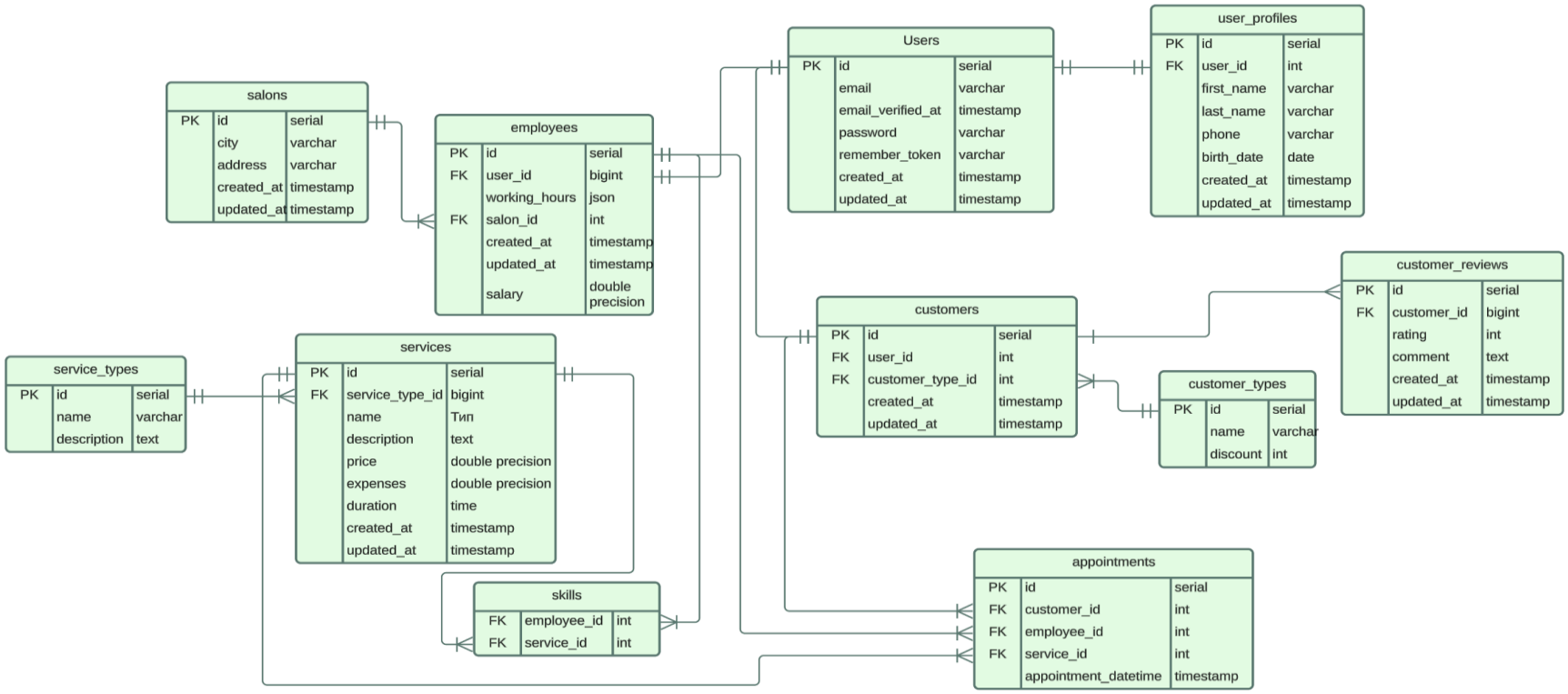


Рисунок А.1 – ER-діаграма для салонів краси “Call of Beauty”

ДОДАТОК Б

Структурно-логічна схема мережі салонів краси «Call of Beauty», побудована в графічному редакторі Microsoft Visio

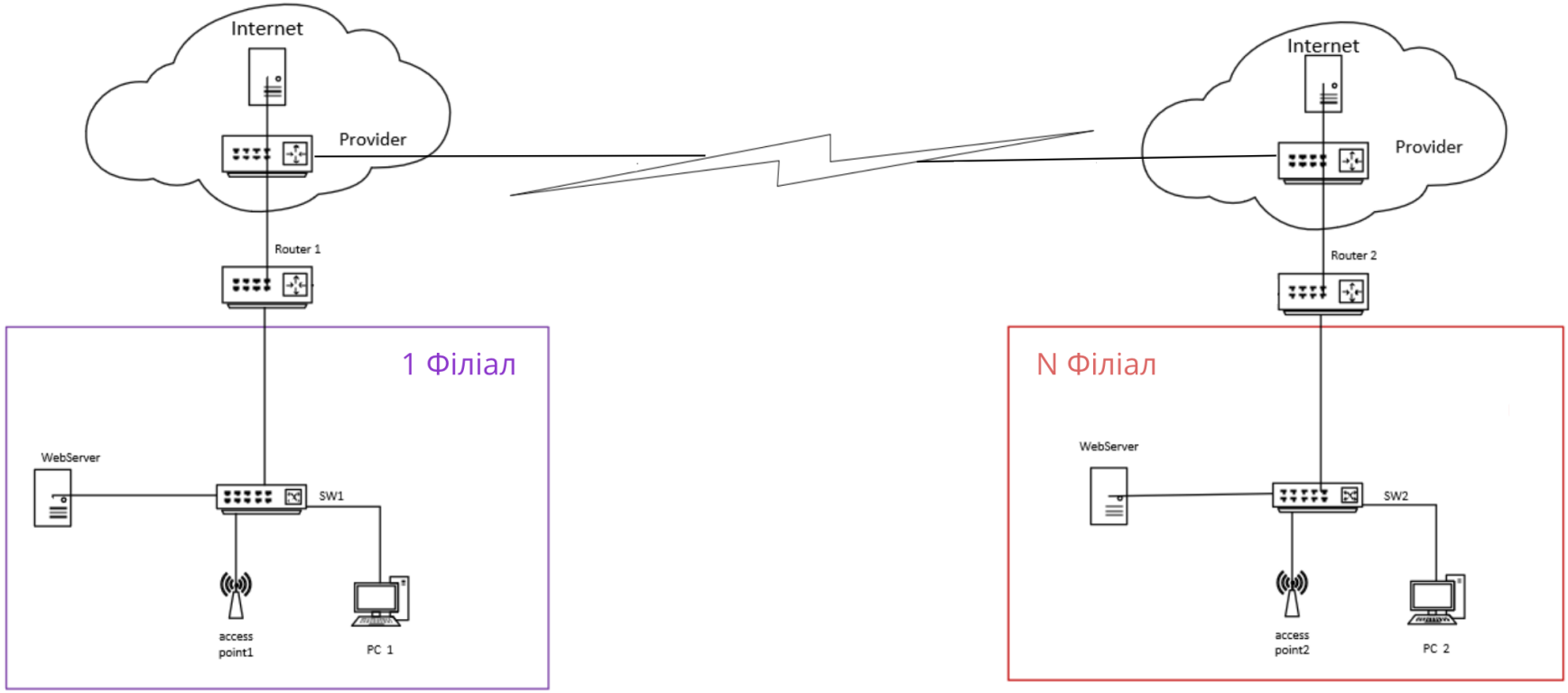


Рисунок Б.1 – Структурно-логічна схема

ДОДАТОК В

Змодельована мережа салонів краси «Call of Beauty»

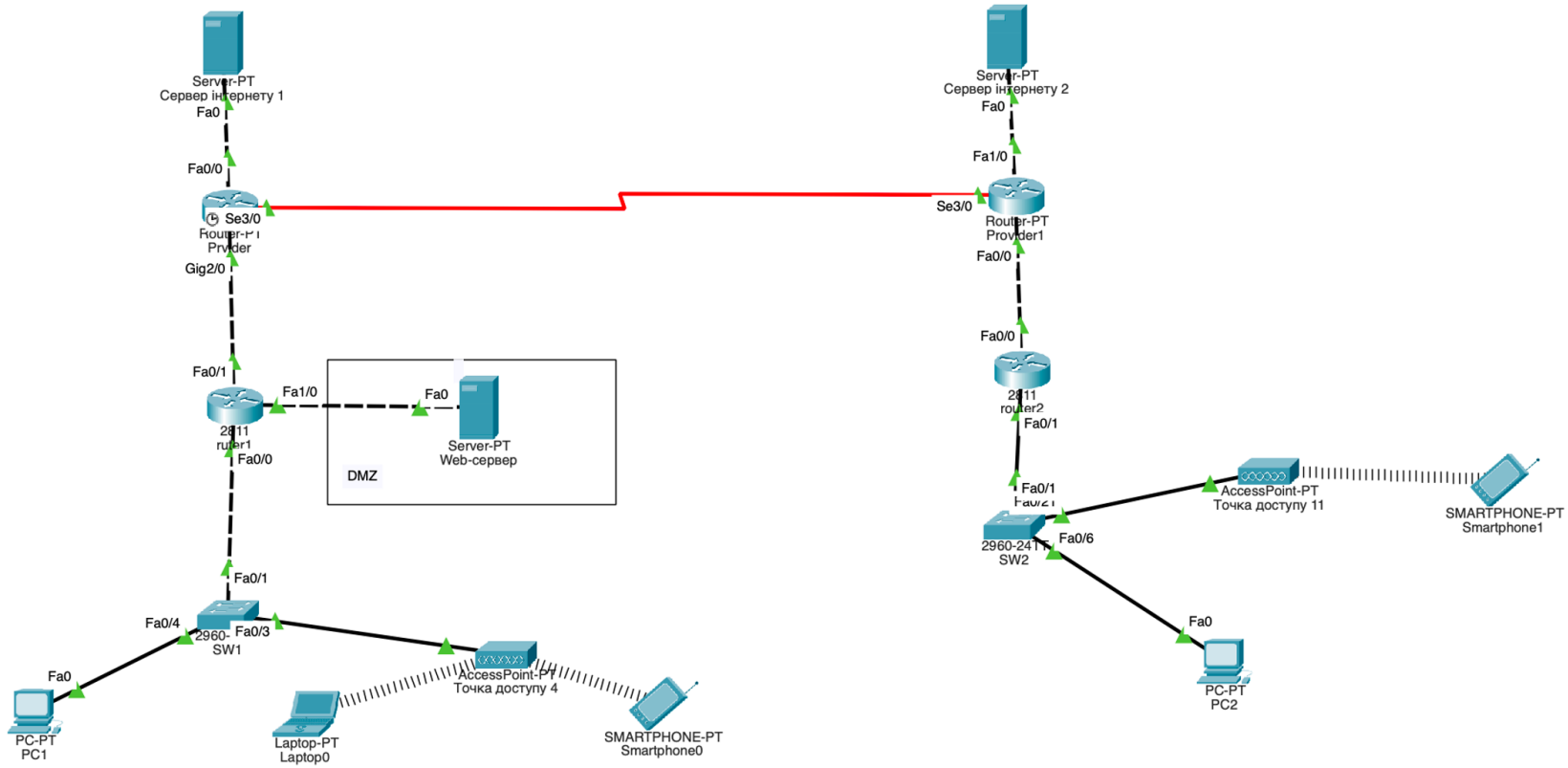


Рисунок В.1 – Змодельована мережа салонів краси «Call of Beauty»

ДОДАТОК Г

Запити для створення бази даних

```
CREATE TABLE users (
    id SERIAL PRIMARY KEY,
    first_name character varying(255) NOT NULL,
    last_name character varying(255) NOT NULL,
    email character varying(255) NOT NULL,
    email_verified_at timestamp(0) without time zone,
    password character varying(255) NOT NULL,
    two_factor_secret text,
    two_factor_recovery_codes text,
    two_factor_confirmed_at timestamp(0) without time zone,
    remember_token character varying(100),
    current_team_id bigint,
    profile_photo_path character varying(2048),
    created_at timestamp(0) without time zone,
    updated_at timestamp(0) without time zone,
    UNIQUE (email)
);
```

Лістинг Г.1 – Створення таблиці користувачів

```
CREATE TABLE user_profiles (
    id SERIAL PRIMARY KEY,
    user_id integer NOT NULL,
    first_name character varying(255),
    last_name character varying(255),
    phone character varying(255),
    birth_date date NOT NULL,
    created_at timestamp(0) without time zone,
    updated_at timestamp(0) without time zone,
    FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE
CASCADE);
```

Лістинг Г.2 – Створення таблиці профілей користувачів

```
CREATE TABLE customer_types (
    id SERIAL PRIMARY KEY,
    name character varying(255) NOT NULL,
    discount integer DEFAULT 0 NOT NULL);
```

Лістинг Г.3 – Створення таблиці типів користувачів

```
CREATE TABLE customers (
    id SERIAL PRIMARY KEY,
    user_id integer NOT NULL,
    customer_type_id integer NOT NULL,
    created_at timestamp(0) without time zone,
    updated_at timestamp(0) without time zone,
    UNIQUE (user_id, customer_type_id),
    FOREIGN KEY (customer_type_id) REFERENCES customer_types(id)
ON DELETE CASCADE,
    FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE
CASCADE);
```

Лістинг Г.4 – Створення таблиці клієнтів

```
CREATE TABLE salons (
    id SERIAL PRIMARY KEY,
    city character varying(255) NOT NULL,
    address character varying(255) NOT NULL,
    created_at timestamp(0) without time zone,
    updated_at timestamp(0) without time zone);
```

Лістинг Г.5 – Створення таблиці філіалів салону

```
CREATE TABLE employees (
    id SERIAL PRIMARY KEY,
    user_id bigint NOT NULL,
    working_hours json NOT NULL,
    salary double precision NOT NULL,
    created_at timestamp(0) without time zone,
    updated_at timestamp(0) without time zone,
    FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE
CASCADE);
```

Лістинг Г.6 – Створення таблиці робітників

```
CREATE TABLE service_types (
    id SERIAL PRIMARY KEY,
    name character varying(255) NOT NULL,
    description text);
```

Лістинг Г.7 – Створення таблиці типів послуг

```
CREATE TABLE services (
    id SERIAL PRIMARY KEY,
```

Лістинг Г.8 – Створення таблиці послуг

```
    service_type_id bigint NOT NULL,
    name character varying(255) NOT NULL,
    description text NOT NULL,
    price double precision NOT NULL,
    expenses double precision NOT NULL,
    duration interval NOT NULL,
    created_at timestamp(0) without time zone,
    updated_at timestamp(0) without time zone,
    FOREIGN KEY (service_type_id) REFERENCES service_types(id)
ON DELETE CASCADE);
```

Лістинг Г.8, аркуш2

```
CREATE TABLE customer_reviews (id SERIAL PRIMARY KEY,
    customer_id bigint NOT NULL,
    rating integer NOT NULL,
    comment text,
    created_at timestamp(0) without time zone,
    updated_at timestamp(0) without time zone,
    FOREIGN KEY (customer_id) REFERENCES customers(id) ON DELETE
CASCADE);
```

Лістинг Г.9 – Створення таблиці клієнтів

```
CREATE TABLE skills (
    service_id integer NOT NULL,
    employee_id integer NOT NULL,
    FOREIGN KEY (employee_id) REFERENCES employees(id) ON DELETE
CASCADE,
    FOREIGN KEY (service_id) REFERENCES services(id) ON DELETE
CASCADE);
```

Лістинг Г.10 – Створення таблиці здібностей майстрів салону

```
CREATE TABLE appointments (id SERIAL PRIMARY KEY,
    customer_id integer NOT NULL,
    employee_id integer NOT NULL,
    service_id integer NOT NULL,
    appointment_datetime timestamp(0) without time zone NOT NULL,
    status status,FOREIGN KEY (customer_id) REFERENCES
customers(id) ON DELETE CASCADE,
    FOREIGN KEY (employee_id) REFERENCES employees(id) ON DELETE
CASCADE,
    FOREIGN KEY (service_id) REFERENCES services(id) ON DELETE
CASCADE);
```

Лістинг Г.11 – Створення таблиці записів на процедури

ДОДАТОК Д

Створення функцій

```
CREATE OR REPLACE FUNCTION
check_working_hours_and_appointments() RETURNS TRIGGER AS $$
DECLARE
    employee_working_hours json;
    day_working_hours json;
    appointment_start time;
    appointment_end time;
    shift_begins time;
    shift_ends time;
    appointment_day text;
    service_duration interval;
BEGIN
    SELECT
    e.working_hours INTO employee_working_hours
    FROM employees e
    WHERE e.id = NEW.employee_id;
    SELECT s.duration INTO service_duration
    FROM services s
    WHERE s.id = NEW.service_id;
    appointment_start := NEW.appointment_datetime::time;
    appointment_end := (NEW.appointment_datetime +
service_duration)::time;
    appointment_day := to_char(NEW.appointment_datetime,
'FMDay');
    -- Loop through the working hours to find the relevant day
    FOR day_working_hours IN SELECT * FROM
json_array_elements(employee_working_hours) LOOP
        IF day_working_hours->>'day' = appointment_day THEN
            -- Extract shift times
            shift_begins := (day_working_hours-
>>'shiftBeginsAt')::time;
            shift_ends := (day_working_hours-
>>'shiftEndsAt')::time;
            -- Check if the appointment time is within the shift
times
            IF appointment_start < shift_begins OR
appointment_end > shift_ends THEN
                RAISE EXCEPTION 'Appointment time is outside
working hours';
            END IF;
```

Лістинг Д.1 – Створення тригеру для перевірки робочих часів для записів

```
            -- Check for overlapping appointments for the same
employee
            IF EXISTS (
                SELECT 1
                FROM appointments a
                WHERE a.employee_id = NEW.employee_id
                  AND (
                    (NEW.appointment_datetime,
NEW.appointment_datetime + service_duration) OVERLAPS
                    (a.appointment_datetime,
a.appointment_datetime + (SELECT duration FROM services WHERE id
= a.service_id))
                  )
            ) THEN
                RAISE EXCEPTION 'Another appointment overlaps
with the same employee at the same time';
            END IF;

            RETURN NEW;
        END IF;
    END LOOP;

    -- If no matching day was found, raise an exception
    RAISE EXCEPTION 'No working hours found for the appointment
day';

    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER prevent_invalid_appointments
BEFORE INSERT ON appointments
FOR EACH ROW
EXECUTE FUNCTION check_working_hours_and_appointments();
```

Лістинг Д.1, аркуш2

```
CREATE OR REPLACE FUNCTION cancelled_services_list()
RETURNS TABLE (
    service_name character varying(255),
    appointment_datetime timestamp without time zone,
    status character varying(255)
) AS $$
```

Лістинг Д.2 – Створення функції для перегляду відмінених послуг

```
BEGIN
    RETURN QUERY
    SELECT
        s.name AS service_name,
        a.appointment_datetime,
        a.status
    FROM
        appointments a
        INNER JOIN services s ON a.service_id = s.id
    WHERE
        a.status = 'cancelled'
    ORDER BY
        a.appointment_datetime DESC;
END;
$$ LANGUAGE plpgsql;
```

Лістинг Д.2, аркуш 2

```
CREATE OR REPLACE FUNCTION service_popularity(month_input date)
RETURNS TABLE (
    service_name character varying(255),
    total_revenue double precision,
    requests_count bigint
) AS $$
BEGIN
    RETURN QUERY
    SELECT
        ser.name AS service_name,
        SUM(s.price - s.expenses)
        AS total_revenue,
        COUNT(*) AS requests_count
    FROM appointments a
         INNER JOIN
         services s ON a.service_id = s.id
         INNER JOIN
         service_types ser ON s.service_type_id = ser.id
    WHERE DATE_TRUNC('month', a.appointment_datetime) =
DATE_TRUNC('month', month_input)
        AND a.status = 'done'
    GROUP BY
        ser.name
    ORDER BY requests_count DESC;
END;
$$ LANGUAGE plpgsql;
```

Лістинг Д.3 – Створення функції для визначення популярності послуг

```
CREATE OR REPLACE FUNCTION
calculate_master_efficiency(month_input date)
RETURNS TABLE (
    first_name character varying(255),
    last_name character varying(255),
    total_revenue double precision
) AS $$
BEGIN
    RETURN QUERY
    SELECT
        u.first_name,
        u.last_name,
        SUM(s.price - s.expenses) AS total_revenue
    FROM
        appointments a
        INNER JOIN employees e ON a.employee_id = e.id
        INNER JOIN services s ON a.service_id = s.id
        INNER JOIN user_profiles u ON e.user_id = u.user_id
    WHERE
        DATE_TRUNC('month', a.appointment_datetime) =
DATE_TRUNC('month', month_input)
        AND a.status = 'done'
    GROUP BY
        u.first_name, u.last_name
    ORDER BY
        total_revenue DESC;
END;
$$ LANGUAGE plpgsql;
```

Лістинг Д.4 – Створення функції, що визначає ефективність майстрів

```
        CREATE OR REPLACE FUNCTION
calculate_monthly_expenses(start_date date, end_date date)
        RETURNS TABLE(month date, total_expenses numeric) AS $$
        BEGIN
            RETURN QUERY
            SELECT
                DATE_TRUNC('month',
a.appointment_datetime)::date AS month,
                COALESCE(SUM(s.expenses)::numeric, 0) AS
total_expenses
            FROM
                appointments a
```

Лістинг Д.5 – Створення функції розрахунку місячних витрат

```
                JOIN services s ON a.service_id = s.id
            WHERE
                a.appointment_datetime BETWEEN start_date AND
end_date
            GROUP BY
                DATE_TRUNC('month',
a.appointment_datetime)::date
            ORDER BY month;
        END;
        $$ LANGUAGE plpgsql;
```

Лістинг Д.5, аркуш 2

```
calculate_expenses_by_date
        CREATE OR REPLACE FUNCTION
calculate_expenses_by_date(start_date date, end_date date)
        RETURNS TABLE(appointment_date date, total_expenses
numeric) AS $$
        BEGIN
            RETURN QUERY
            SELECT
                DATE(a.appointment_datetime) AS
appointment_date,
                COALESCE(SUM(s.expenses)::numeric, 0) AS
total_expenses
            FROM appointments a
            JOIN services s ON a.service_id = s.id
            WHERE a.appointment_datetime BETWEEN start_date AND
end_date
            GROUP BY DATE(a.appointment_datetime)
            ORDER BY DATE(a.appointment_datetime);
        END;
        $$ LANGUAGE plpgsql;
```

Лістинг Д.6 – Створення функції розрахунку витрат по дням

```
get_service_rank_by_employee
        CREATE OR REPLACE FUNCTION
get_service_rank_by_employee(employee_id_param integer,
start_date_param date, end_date_param date)
        RETURNS TABLE(service_name character varying,
appointment_count integer, rank integer) AS $$
        BEGIN
```

Лістинг Д.7 – Створення функції популярності послуг

```
            RETURN QUERY
            SELECT
                s.name AS service_name,
                COUNT(*)::integer AS appointment_count,
                RANK() OVER (ORDER BY COUNT(*) DESC)::integer AS
rank
            FROM appointments a
            JOIN services s ON a.service_id = s.id
            WHERE
                a.employee_id = employee_id_param
                AND a.appointment_datetime BETWEEN
start_date_param AND end_date_param
                AND a.status = 'done'
            GROUP BY s.name
            ORDER BY
                appointment_count DESC;
        END;
        $$ LANGUAGE plpgsql;
```

Лістинг Д.7, аркуш 2

```
        CREATE OR REPLACE FUNCTION
calculate_income_by_month(start_date date, end_date date)
        RETURNS TABLE(month text, net_income numeric) AS $$
        BEGIN
            RETURN QUERY
            SELECT
                to_char(a.appointment_datetime, 'YYYY-MM') AS
month,
                COALESCE(SUM(s.price * (1 - ct.discount /
100.0)), 0) -
                COALESCE(SUM(s.expenses), 0) -
                COALESCE(SUM(e.salary), 0) AS net_income
            FROM appointments a
            JOIN services s ON a.service_id = s.id
            JOIN customers c ON a.customer_id = c.id
            JOIN customer_types ct ON c.customer_type_id = ct.id
            JOIN employees e ON a.employee_id = e.id
            WHERE a.appointment_datetime BETWEEN start_date AND
end_date
            GROUP BY to_char(a.appointment_datetime, 'YYYY-MM');
        END;
        $$ LANGUAGE plpgsql;
```

Лістинг Д.8 – Створення функції розрахунку доходу за місяць

Olga Roznovets
Digitally signed by Olga Roznovets
Date: 2024.06.13 21:28:02 +03'00'