МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: «Інформаційна система організації замовлення косметичних послуг»

за спеціальністю 122 «Комп'ютерні науки»,

освітньо-професійна програма «Інформаційні технології проектування»

Виконавець роботи: студент групи ІТ.м-01 Киричок Анастасія Вадимівна

Кваліфікаційну роботу
захищено на засіданні ЕК
з оцінкою _____ «___» лютого 2022 р.

Науковий керівник _____ Чибіряк Я.І.

(підпис)

Голова комісії _____ Шифрін Д.М

(підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент _____

(підпис)

Суми-2022

Сумськии˘ державнии˘ університет

Факультет електроніки та інформаціӥних технологіӥ

Кафедра комп'ютерних наук

Секція інформаціӥних технологіӥ проектування

Спеціальність 122 «Комп'ютерні науки»

Освітньо-професіи˘на програма «Інформаціӥні технології проектування»

ЗАТВЕРДЖУЮ

Зав. секцією ІТП

_____

В.В. Шендрик

«__» _____ 2022р.

ЗАВДАННЯ

НА КВАЛІФІКАЦИИ˘НУ РОБОТУ МАГІСТРА СТУДЕНТУ

Киричок Анастасія Вадимівна

1. Тема роботи *Інформаціӥна система організації̈ замовлення косметичних послуг*

керівник роботи *Чибіряк Яна Іванівна*,

затверджені наказом по університету від « 29 » жовтня 2021 р. № 0787-IV

2. Строк подання студентом роботи «14» лютого 2022 р.

3. Вхідні дані до роботи технічне завдання на розробку ІС, контент

4. Зміст розрахунково-пояснювальноі¨ записки (перелік питань, які потрібно розробити) аналіз предметноі¨ області, постановка задачі та мети дослідження, проектування ІС, розробка ІС

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) актуальність, мета і задачі, аналіз та порівняння аналогів, функціональні вимоги, контекстна діаграма та ії̈ декомпозиція, діаграма

варіантів використання, er-діаграма, вибір інструментарію, архітектура web- додатку, приклади реалізації, висновки

6. Консультанти розділів роботи:

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7.Дата видачі завдання 01.10.2022

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітки
1.	Визначення мети проекту	03.09.2021 -10.09.2021	
2.	Визначення вимог до проекту	11.09.2021- 19.09.2021	
3.	Аналіз предметної області	26.09.2021 - 03.10.2021	
4.	Вибір існуючих інструментів для реалізації поставленої мети	04.10.2021 – 14.10.2021	
5.	Структурно-функціональне моделювання	15.10.2021 – 25.10.2021	
6.	Розробка інтерфейсу ІС	26.10.2021 – 10.11.2021	
7.	Програмна реалізація ІС	11.11.2021 – 01.01.2022	
8.	Встановлення та налаштування ІС	02.01.2022 – 05.02.2022	
9.	Оформлення пояснювальної записки	06.02.2022 – 10.02.2022	

Студент

Киричок А.В. _____

(підпис)

Керівник роботи _____ Чибіряк Я.І.

(підпис)

РЕФЕРАТ

Тема кваліфікаційної роботи магістра «Інформаційна система організації замовлення косметичних послуг».

Пояснювальна записка складається зі вступу, 4 розділів, висновків, списку використаних джерел із 30 найменувань, додатків. Загальний обсяг роботи − 54 сторінок, у тому числі 30 сторінок основного тексту, 2 сторінки списку використаних джерел, 12 сторінок додатків.

Кваліфікаційну роботу магістра присвячено розробці мобільного додатку для замовлення косметичних послуг. В роботі проведено аналіз та дослідження наявних аналогів, виявлено вимоги до реалізації, також наявний аналіз переваг та недоліків останніх.

У роботі виконано:

- дослідження предметної області;
- формування ТЗ;
- зроблено вибір методів розробки;
- змодельовано роботу системи;
- розроблено інформаційну систему замовлення косметичних послуг;
- протестовано додаток на коректність роботи.

Результатом проведеної роботи є розробка мобільного додатка для замовлення косметичних послуг, що реалізовано за допомогою мови програмування C# та СУБД Firebase. Практичне значення роботи полягає у розробці мобільного додатка, що є досить актуальним в наш час пандемії. Так мінімізуються черги, а користувач матиме

змогу зробити це в будь-який час доби. Також електронна клієнтська база не тільки є надійною, а ще може принести масу переваг не тільки для споживачів, але й для власників. Це дозволило впровадити рейтинг майстрів для кращого вибору клієнта.

Ключові слова: мобільний додаток, косметичні послуги, замовлення послуг, СУБД.

ЗМІСТ

ВСТУП

Мобільний додаток, також званий мобільним додатком або просто програмою, — це комп’ютерна програма або програмне забезпечення, призначене для роботи на мобільному пристрої, такому як телефон, планшет або годинник. Мобільні програми часто відрізняються від настільних програм, які призначені для запуску на настільних комп’ютерах, і веб-програм, які працюють у мобільних веб-браузерах, а не безпосередньо на мобільному пристрої.

З поширенням Інтернет технологій з’явилося безліч мобільних додатків, без яких важко уявити сучасне життя. Кожного дня ми використовуємо програми на смартфонах, комп’ютерах задля забезпечення звичного ритму буття.

Інтернет простір надає змогу не тільки навчатися, спілкуватися, а ще є гарним варіантом для ведення бізнесу в умовах карантину. В домашніх умовах є змога не тільки користуватися перевагами та послугами, а ще й можна слідкувати за ходом дій бізнесу та направляти його в потрібному напрямку. Все що потрібно – підключення до мережі Інтернет.

Проте є один нюанс, це виключення стосується замовлення певних послуг, а саме - косметичних. В цьому випадку на допомогу стає в нагоду мобільний додаток.

Нерідко можна зустріти запис та ведення статистистики через дзвінки та помітки в звичайних журналах. Ефективності та зворотнього зв’язку із такої реалізації майже немає.

Розглянемо інший реалізації – мобільний додаток в інтернеті. Відображення даних та функцій може набувати різноманітних форм. Від запису із нагадуванням до онлайн-консультацій з відгуками та фото майстрів. Прочитане або побачене не залишить байдужим користувача будь-якої вікової категорії, що є головною метою власника бізнесу з замовлення послуг.

Метою даної дипломної роботи є розробка інформаційної системи «Замовлення косметичних послуг». Інформаційна система буде реалізована у вигляді мобільного засосунку. Тому послуги швидко знайдуть свого потенціального клієнта.

Для споживача буде доступний наступний функціонал:

1. Запис на косметичну процедуру;
2. Перегляд послуг;
3. Перегляд майстрів;
4. Перегляд рейтингу майстра;
5. Перегляд рейтингу процедури;
6. Залишення відгуку;
7. Налаштування зовнішніх факторів відображення;
8. Вибір персональних налаштувань для відображення послуг.

Це значно покращить роботу не тільки для кінцевого споживача, а ще й для майданчику замовлення косметичних послуг. Такий застосунок буде гарна нагодою, щоб покращити та підвищити ефективність бізнесу, з урахуванням рейтингу та оціненням усіх переваг та недоліків.

Такий вид замовлення є досить актуальним в наш час пандемії. Так мінімізуються черги, а користувач матиме змогу зробити це в будь-який час доби. Також електронна клієнтська база не тільки є надійною, а ще може принести масу переваг не тільки для споживачів, але й для власників. Це дозволить впровадити бонусну систему, систему знижок, програму лояльності, рейтинг майстрів та послуг. Що неодмінно матиме перевагу над будь-якими іншими додатками.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд предметної області

Мобільний додаток — один з видів програми, який використовується, як в робочіх так і неробочих цілях на сучасних смартфонах або планшетах. Додатки зазвичай являють собою компактні програмні блоки з обмеженим фунуціоналом, в яких є можливість надавати кінцевому користувачу якісну сферу обслуговування.

Пандемія внесла свої корективи не тільки в життя людей, но і погляд на проведення робочого процесу. Задля забезпечення безпеки здоров'ю більшість офісів тепер знаходяться вдома, а мобільні додатки давно вийшлі за рамки ігрових. Тепер це платформи для замовлення послуг, онлайн магазини та ін.

Головною відмінністю між десктопними та мобільними додатками є часткова відмова від ітегрованих програмних систем. Так кожен застосунок надає ізольований та обмежений функціонал. Прикладами можуть слугувати такі додатки, як калькулятор, помітки, мобільна гра або ж мобільний браузер.

Обмеженість програмних ресурсів попередніх мобільних засобів створювала для застосунків обмеження в багатофункціональності. Хоча пристрої, що набули загального вжитку на момент сьогодення є більш складнішими в технічній реалізації, проте мобільні додатки залишаються вузькофункціональними. Саме в такий спосіб

власники застосунків дозволяють споживачам обирати вручну функції, які повинні мати їхні пристрої. [1]

Більш детально розглянемо види мобільних застосунків задля кращого розуміння спектру функціоналу останнього та обрання типу для кращої реалізації застосунку сфери косметичних послуг.

Нативний додаток.

Нативні програми створені для однієї мобільної операційної системи. Тому їх називають рідними – вони є рідними для певної платформи чи пристрою. Більшість мобільних додатків сьогодні створено для таких систем, як Android або iOS. Характеристики для завантаження такого застосунку опісують те, що неможливо встановити та використовувати додаток ОС Android на iPhone, і навпаки.

Основною перевагою нативних програм є їх висока продуктивність і чудовий досвід користувача. Зрештою, розробники, які створюють їх, використовують рідний інтерфейс пристрою. Доступ до широкого спектру API також допомагає прискорити роботу з розробки та розширити межі використання програми. Нативні програми можна завантажувати лише з магазинів додатків і встановлювати безпосередньо на пристрої. Тому спочатку вони повинні пройти суворий процес видавництва.

Найважливішим недоліком нативних програм є їх вартість. Щоб створити, підтримувати та підтримувати програму для Android та iOS, в основному необхідні дві команди розробників, що неодмінно піднімає вартість розробки програмного продукту.

Веб-додаток.

Веб-програми – це програми, які мають подібний функціонал до рідних засосунків і працюють на мобільних пристроях. Однак між нативними та останніми є значні відмінності. По-перше, веб-додатки використовують браузери для запуску, і зазвичай вони реалізовані на CSS, HTML5 або JavaScript. Такі програми перенаправляють користувача на URL-адресу, пропонуючи встановити їх. Створюючи закладку на веб-сторінці, такі застосунки використовують мінімум пам’яті пристрою.

Оскільки всі персональні бази даних будуть збережені на сервері, користувачі можуть використовувати програму лише за наявності підключення до Інтернету. Це основний недолік веб-програм – вони завжди потребують хорошого підключення до Інтернету. Інакше є ризик надати неякісний користувацький досвід.

Гібридні програми.

Гібридні програми створені за допомогою таких веб-технологій, як JavaScript, CSS та HTML 5. Таку назву вони отримали оскільки, додатки в основному працюють як веб-програми, замасковані під рідну оболонку.

Гібридні програми легко та швидко розробляються, що є безсумнівною перевагою. Є можливість отримання єдиної кодової бази для всіх платформ, що знижує витрати на обслуговування та спрощує процес оновлення. Розробники також можуть скористатися перевагами багатьох API для таких функцій, як гіроскоп або геолокація. [1]

З іншого боку, гібридним додаткам може не вистачати швидкості та продуктивності. Крім того, у вас можуть виникнути деякі проблеми з дизайном, оскільки додаток може виглядати не однаково на двох або більше платформах.

Розглянувши, додатки в програмній реалізації, представимо види по використанню. Мобільні програми бувають різних форм і розмірів. Нижче наведено найпопулярніші типи мобільних додатків, які є поточними тенденціями в мобільному ландшафті.

- Ігрові застосунки – одна з найактуальниших категорій мобільного додатку. Наразі компанії, орієнтуючись на сучасний ринок, охоче фінансують створення та впровадження мобільних аналогів десктопних додатків.
- Додатки для ведення бізнесу або ж обліку продуктивності – додатки, що займають значний сегмент IT ринку. Таке визнання данні застосунки отримали, через швидкий та мобільний темп життя. Наприклад, додатки можуть допомогти користувачам забронювати квитки, надсилати електронні листи або відстежувати хід роботи. Бізнес-додатки спрямовані на підвищення продуктивності та мінімізацію витрат, оскільки вони дозволяють користувачам

виконувати широкий спектр завдань, від покупки нових картриджів для офісних принтерів до найму нового офіс-менеджера.

- Освітні програми – програми для саморозвитку з метою покращення знань та вмінь. Наприклад, програми для вивчення мови, такі як Duolingo, стали неймовірно популярними, оскільки вони дають користувачам гнучкість, яку вони шукають у навчанні.
- Розвиваючі ігрові програми – чудовий інструмент для зовсім юних користувачів. Не тільки діти, проте й вчителі з батьками використовують такі додатки, щоб покращити процес засвоєння нобутих знань.
- Застосунки життєвого стилю – категорія, що включає в себе додатки для підтримання особистого життя. Наприклад застосункі для замовлення одежі або інших речей, віртуальні знайомства, онлайн тренування. Данні засосунки фокусують увагу на ризноманітних галузях приватного життя.
- Додатки М-комерції – додатки, що дублюють функціонал для здійснення онлайн шопінгу, такі як Aliexpress або OLX. Останні надають можливість здійснення шопінгу у будь-яку годину дня і ночі.
- Розважальні застосунки – програми, що надають можливість здійснення спілкування та передачу медіа контенту, як в особистому листуванні так і в вільному доступі.
- Допоміжні програми – застосунки, що являють собою сканери бач та штрих-кодів, статистичні або програми медичного змісту. [2]

Тому для обраної теми магістерської роботи, а саме організації проведення косметичних послуг було обрано реалізувати додаток підвиду для бізнесу або продуктивності, що спрямовані на підвищення продуктивності та мінімізацію витрат, оскільки вони дозволяють користувачам виконувати широкий спектр завдань, від запису на обрану послугу до перегляду рейтингу майстрів та перегляд відгуків, що значно полегшує вибір кінцевого споживача.

1.2 Розгляд наявних аналогів

Розглянемо також вже готові майданчики для запису на косметичні послуги. Задля кращого розуміння та аналізу сфери, мінімізації багів та ефективної реалізації.

EasyWeek.

Система для онлайн запису 24/7. Сервіс являє собою просту CRM, віджет для онлайн-бронювання клієнтам.

Зовнішній вигляд віджету для онлайн-запису можна налаштувати під фірмовий стиль компанії. Крім того, у разі використання сервісу буде сторінка з унікальним посиланням, де клієнти зможуть записатися на прийом.

Є база з клієнтами, де доступна інформація про візити та покупки, а також статистика майстрів та ТОП послуг. Доступні нагадування клієнту про візит по SMS та Email, план на день/тиждень для співробітників.

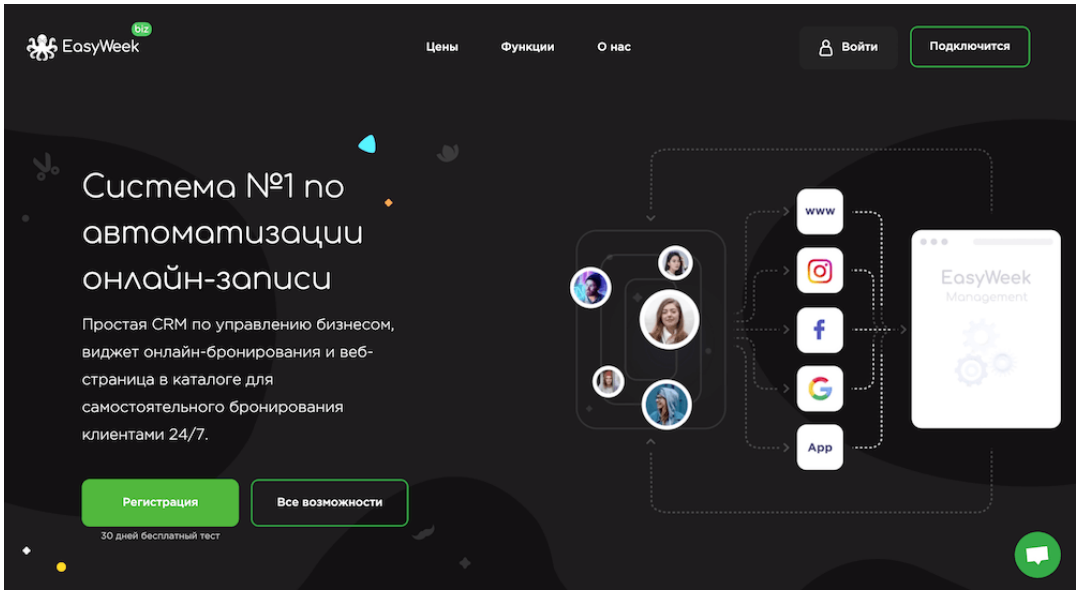

Рис. 1.1 – Приклад роботи EasyWeek.

Арніка.

Це хмарна система управління бізнесом у сфері послуг. За допомогою сервісу є можливість налаштувати онлайн-розклад, завдяки якому наочно видно завантаженість майстрів, клієнт може самостійно записатися онлайн 24/7. В Арніку можна вести клієнтську базу — налаштувати нагадування та сповіщення за допомогою WhatsApp або SMS.

Для управління фінансами є можливість відстежувати онлайн такі показники, як виручка, прибуток, витрати, доходи, всі операції з каси фіксуються у програмі. Є змога налаштувати автоматизований розрахунок зарплати працівникам.

Програма дозволяє вести облік товарів та матеріалів, аналіз їх використання у розрізі майстрів, послуг, клієнтів. Автоматичне списання матеріалів під час надання послуг.

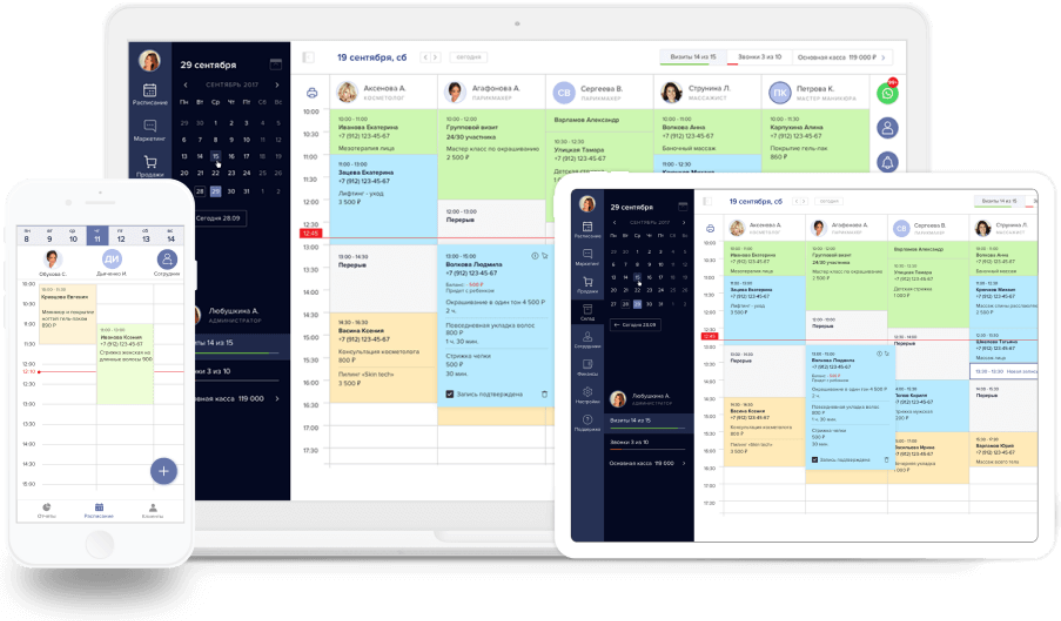

Рис. 1.2 – Приклад роботи Арніка.

UNIVERSE-CRM.

Це одна із нових web-програма для салонів краси із сервісом онлайн запису. Система є потрібною власникам салонів, барбершопів, перукарень, нейл-студій – будь-яким підприємствам, які надають послуги у сфері краси.

UNIVERSE-CRM створена для автоматизації основних бізнес-процесів: ведення та сегментація клієнтської бази, облік персональних знижок, контроль роботи майстрів, управління товарними запасами та фінансами. І, звісно, для онлайн-запису клієнтів.

CRM система дає можливість створювати брендовані віджети, які можна розмістити на сайті, соціальних мережах, в Instagram. Гнучкі індивідуальні налаштування сервісу онлайн-запису дозволяють відображати певних співробітників

у записі, приховувати конкретні послуги, додавати опис та фотографії до послуг, створювати мережеві віджети для кількох філій.

Для косметологічних салонів є можливість організації запису із бронюванням апаратів. Співробітники мають можливість заходити до свого особистого кабінету для перегляду історії відвідувань та майбутніх записів. Клієнти можуть залишати відгуки на сайті та також користуватися особистим кабінетом для скасування або перенесення записів.

CRM автоматично фіксує, звідки було зроблено онлайн-запис і видає інформацію про ефективність рекламних джерел у звітах-дашбордах. Програма відправляє e-mail повідомлення про підтвердження досконалих записів, а також нагадування про майбутні візити – клієнти, адміністратори салону та майстри.

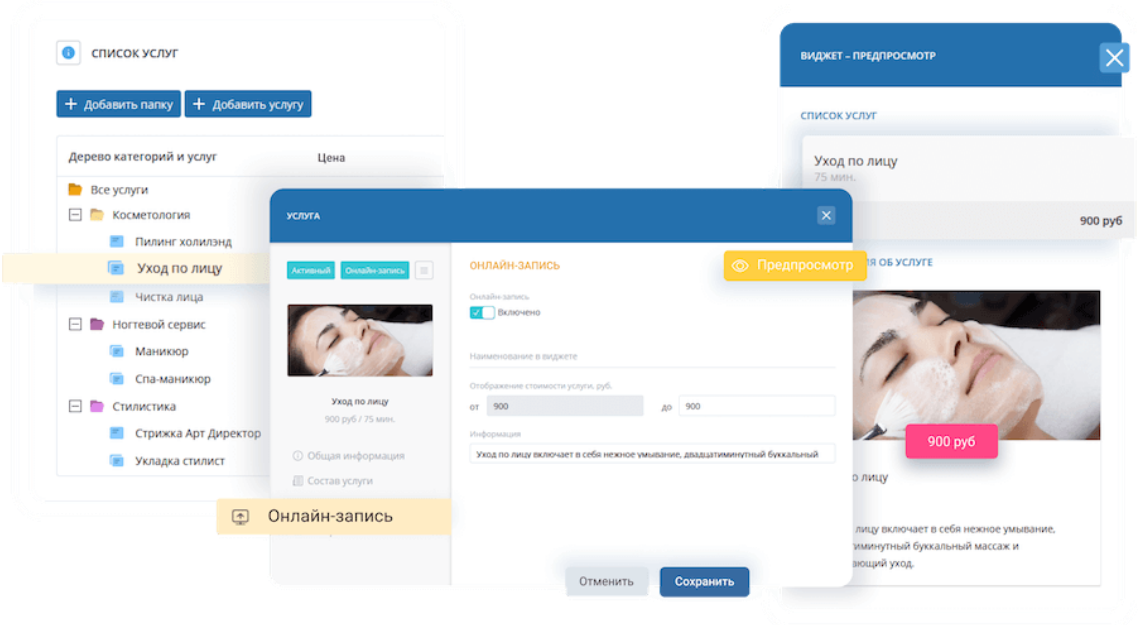

Рис. 1.3 – Приклад роботи UNIVERSE-CRM.

Torrow.Net.

Це багатофункціональна хмарна платформа з мобільними додатками iOS/Android та Web сайтом. Розробники також додали програму до магазину додатків AppGallery.

Torrow включає інструменти для управління заявками і роботою співробітників офлайн і онлайн бізнесу будь-якого масштабу. Платформа універсальна для бізнесу

будь-якої сфери послуг та дозволяє налаштувати роботу як самозайнятого спеціаліста, так і великого бізнесу із сотнями співробітників.

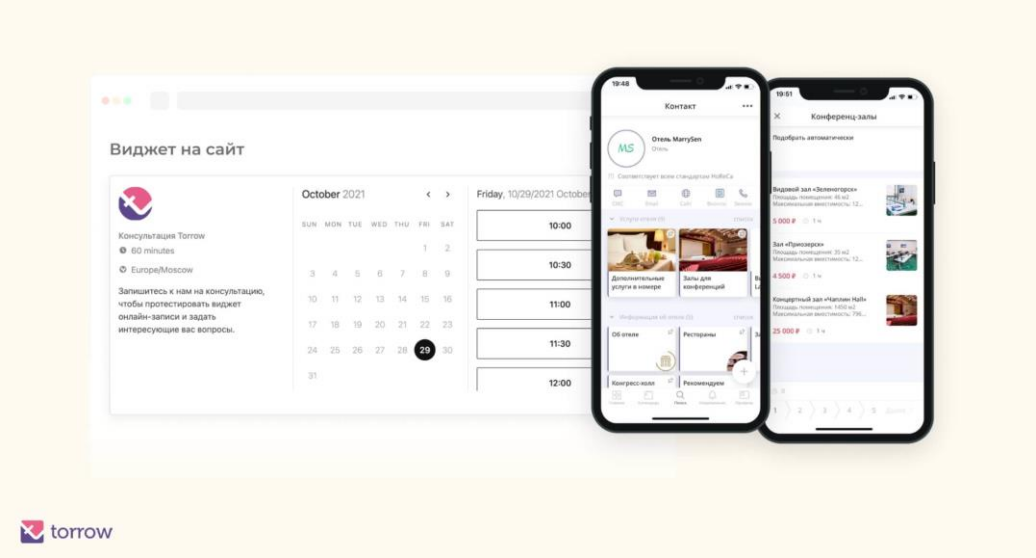

Рис. 1.4 – Приклад роботи Torrow.Net.

Плюси:

- можна налаштувати всі види послуг на одній платформі (індивідуальний запис на час, групові сеанси занять, інтернет замовлення, заявки, оренда по днях або годинах);
- підтримуються особливості роботи як офлайн, так і онлайн бізнесу;
- декілька алгоритмів призначення виконавців на заявки (ручний, автоматичний, черга);
- клієнт може записатися на послугу без встановлення програми або автентифікації у системі;
- облік часових поясів для розсилки повідомлень чи взаємодії з клієнтами;
- можливість інтеграції з Google та Microsoft Outlook календарями для розрахунку зайнятого часу працівників під час онлайн-запису;
- є можливість швидкого налаштувати сайт-візитку бізнесу із послугами;
- інтеграція з безліччю систем через webhook (AmoCrm, 1С, розсилки, облікові системи тощо).

Мінуси:

- складне настроювання програми для нетипових послуг. [3]

Попередньо було продемонстровано широкий спектр видів мобільних додатків. Проаналізувавши аналоги розроблювального застосунку, можна зробити наступні висновки з функціоналу останніх.

Таблиця Б.1 – Порівняння аналогів розроблювальної ІС

Характеристика	**EasyWeek**	**Арніка**	**Torrow.Net.**	**UNIVERSE-CRM**
Зручний інтерфейс	-	-	+	+
Реєстрація користувачів	-	+	+	+
Безкоштовне користування	-	-	-	-
Дійсність посилань	+	+	+	+
Швидкий зворотній зв'язок	+	-	-	+
Навігація	+	+	+	+
Сучасний дизайн	+	-	-	+

З таблиці видно, що не всі мають реєстрацію чи сучасний дизайн, що впливає на можливість зручного використання. Також важливим фактором є оплата за додаток, абсолютно усі представлені аналоги беруть плату за користування

застосунком. Розроблювальна ІС буде цілком безкоштовною. Проте слід зазначити, що мобільний додаток повинен бути розміщений в онлайн магазині додатків.

Найбільшими магазинами додатків є Google Play для Android, App Store для iOS і Microsoft Store для Windows 10, Windows 10 Mobile і Xbox One. Кожен із яких має свої особливості для розміщення мобільних додатків.

Google Play.

Google Play (раніше відомий як Android Market) — це міжнародний онлайн-магазин програмного забезпечення, розроблений Google для пристроїв Android. Він відкрився в жовтні 2008 року. У липні 2013 року кількість програм, завантажених через Google Play Store, перевищила 50 мільярдів із понад 1 мільйона доступних програм. Станом на вересень 2016 року, за даними Statista, кількість доступних додатків перевищила 2,4 мільйона. Більше 80% програм у магазині Google Play можна безкоштовно завантажити. [4]

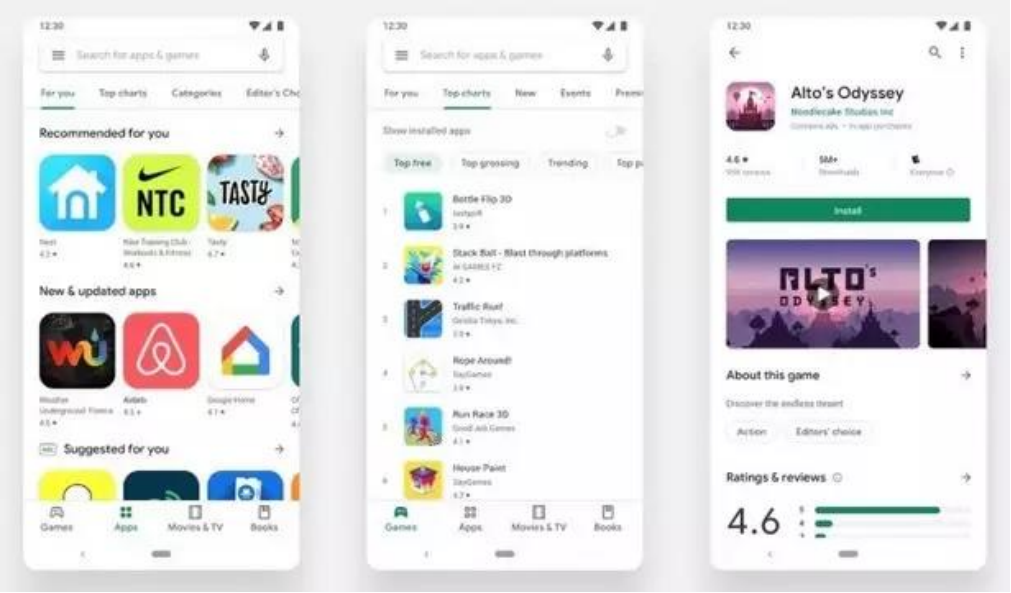

Рис. 1.5 – Приклад роботи GooglePlay

App Store.

Apple App Store для iOS та iPadOS не був першим сервісом розповсюдження додатків, але він викликав революцію мобільних пристроїв і був відкритий 10 липня 2008 року, а станом на вересень 2016 року повідомляв про понад 140 мільярдів завантажень. Станом на 6 червня 2011 року було доступно 425 000 додатків, які завантажили 200 мільйонів користувачів iOS. Під час Всесвітньої конференції

розробників Apple у 2012 році генеральний директор Тім Кук оголосив, що App Store має 650 000 доступних програм для завантаження, а також 30 мільярдів додатків, завантажених із магазину до цієї дати. [5]

Рис. 1.6 – Приклад роботи AppStore

Microsoft Store

Microsoft Store (раніше відомий як Windows Store) був представлений Microsoft у 2012 році для своїх платформ Windows 8 і Windows RT. Хоча він також може містити списки традиційних настільних програм, сертифікованих на сумісність з Windows 8, він в основному використовується для розповсюдження «програм Windows Store», які в основному створені для використання на планшетах та інших сенсорних пристроях. [6]

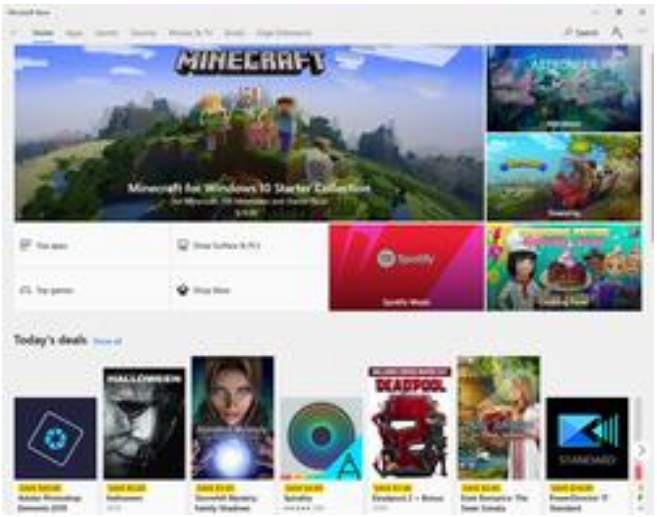

Рис. 1.7 – Приклад роботи Microsoft Store

З огляду на вище розглянуту інформацію можна зробити висновок, що мобільні застосунки набули широкого вжитку на різних програмних платформах.

2. ПОСТАНОВКА ЗАДАЧІ ТА МЕТОДИ ДОСЛІДЖЕННЯ

2.1 Мета та задачі дослідження

Об’єктом магістерської роботі є процес інформатизації замовлення косметичних послуг. Предметом виступає інформаційне забезпечення даних про замовлення косметичних послуг. Метою магістерською роботи є розробка мобільного додатку для організації роботи та замовлення косметичних послуг. Розробка мобільних додатків – це процес який ґрунтується на традиційній розробці програмного забезпечення. Однак він зосереджений на створенні ПЗ, яке використовує переваги унікальних функцій обладнання мобільних пристроїв.

Задачами дослідження будуть наступні кроки, такі як:

- дослідження предметної області;
- формування ТЗ;
- вибір методів розробки;
- моделювання роботи системи;
- розроблення інформаційної системи;
- тестування та перенесення на хостинг.

2.2. Методи дослідження

Методами дослідження будуть наступні кроки.

Оскільки, кращий підхід передбачає розробку спеціально для мобільного середовища. Це техніка, яка використовує всі переваги, що пропонують мобільні пристрої. Процес враховує їх обмеження та допомагає власникам бізнесу збалансувати вартість та функціональність.

Розробка мобільного додатка – це акт або процес, за допомогою якого розробляється мобільний додаток для мобільних пристроїв, таких як персональні

цифрові помічники, корпоративні цифрові помічники або мобільні телефони. Ці програмні програми розроблені для роботи на мобільних пристроях, таких як смартфон або планшет. Ці програми можуть бути попередньо встановлені на телефонах під час виробничих платформ або доставлені як веб-додатки, використовуючи обробку на стороні сервера або клієнта (наприклад, JavaScript), щоб забезпечити «подібний до програми» досвід у веб-браузері. Розробники програмного забезпечення також повинні враховувати великий набір розмірів екранів, апаратних характеристик та конфігурацій через жорстку конкуренцію мобільного програмного забезпечення та зміни на кожній із платформ. Розробка мобільних додатків неухильно зростає, прибутки та створені робочі місця. [7]

Оскільки, розробляється інформаційна система «Замовлення косметичних послуг», то обраний вид реалізації буде - додатки для бізнесу чи продуктивності. Тому що, ці програми займають значну частину ринку сьогодні, оскільки люди все частіше використовують свої смартфони та планшети для виконання багатьох складних завдань на ходу, що видно з рисунку 2.1.

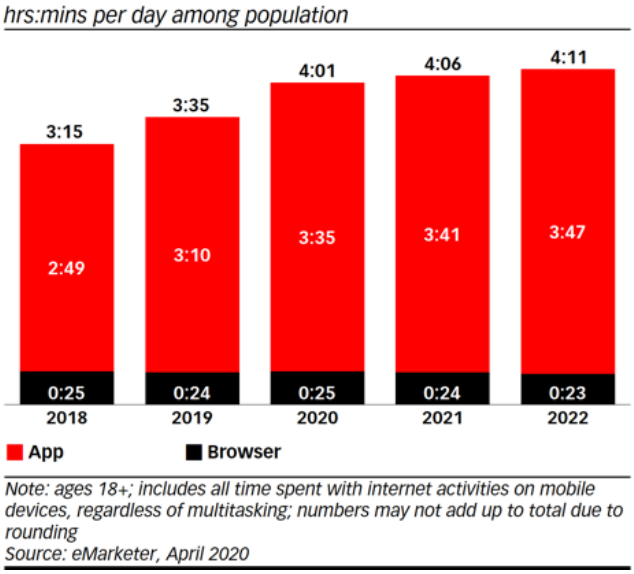

Рис. 2.1 – Діаграмма проведення в мобильних застосунках населенням

Згідно з дослідженнями сервісу eMarketer, 83% часу, проведеного в веб-застосунках в цому році, буде проводитися в мобільних додатках, що становить більший відсоток, ніж кілька років тому.

Користувачі мобільних пристроїв явно віддають перевагу внутрішньому додатку, оскільки це є більш зручнішим в користуванні, ніж постійний пошук в

браузері, що може працюапти некоректно. Пандемія також вплинула на поєднання додатків і мобільного Інтернету. Тепер більший відсоток засосунків базується на постійному використанні Інтрнету. [8]

По-друге, після визначення ідеї та функціоналу йде написання Технічного завдання з урахуванням побажань власника бізнес платформи та можливостей розробника на основі проведеного аналізу.

По-третє, на основі затвердженого ТЗ йде розробка дизайну. Візуально оформлюється функціонал майбутнього застосунку у вигляді макету в програмі «Photoshop».

По-четверте, не менш важливим є вибір мови програмування та розробка бази данних. Оскільки, наявна велика клієнтська база споживачів та замовлення послуг. Тому потрібно забезпечити якісне технічне виконання даного пункту.

По-п’яте, після програмної реалізації мобільного додатку необхідно виконати розміщення на веб-хостингу та провести тестування якості робити. Застосунок матиме різну вікову аудиторію користувачів, тому його робота повинна бути інтуїтивно зрозумілою.

Таким чином, з виконанням вище перелічених пунктів буде розроблений мобільний застосунок «Замовлення косметичних послуг».

2.3. Постановка задачі

Постановка задачі має наступний вигляд. Необхідно розробити мобільний додаток для замовлення послуг краси. Слід реалізувати візуальну та серверну частину (Front-end та Back-end) веб-плагіну, із таким функціоналом.

Клієнт:

1. Запис на косметичну процедуру;
2. Перегляд послуг;
3. Перегляд майстрів;
4. Перегляд рейтингу майстра;
5. Перегляд рейтингу процедури;

6. Залишення відгуку;
7. Налаштування зовнішніх факторів відображення;
8. Вибір персональних налаштувань для відображення послуг.

Адміністратор:

1. Всі можливості клієнта;
2. Доступ до панелі адміністратора та всі її можливості.

Сервер:

1. Доступ до бази даних, можливість вносити зміни;
2. Взаємодія з шаблонами додатку.

Також наявні наступні функціональні вимоги:

1. Розмежування прав доступу;
2. Інтеграція сторонніх інформаційних модулів;
3. Постійне оновлення контенту;
4. Можливість оцінювання послуг.

3. ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1 Структурно-функціональне моделювання процесу розробки додатку

Моделювання додатку зображено у вигляді нотації IDEF0, де було розбито на основні процеси функціонування в мобільному додатку. Ціль: забезпечення якісних косметичних послуг. Точка зору: адміністратор сайту.

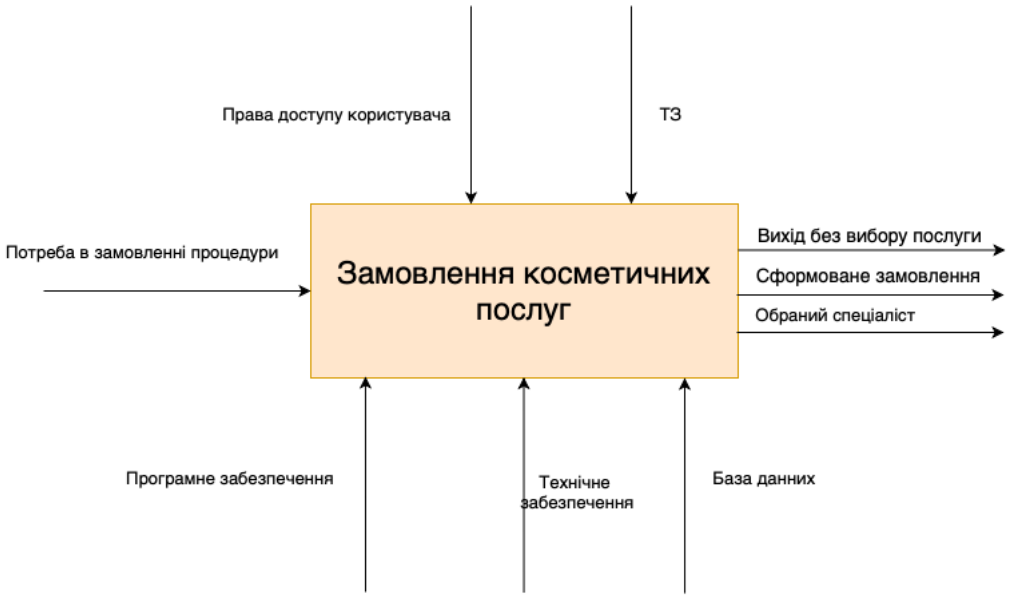

Рис. 3.1 – Контекстна діаграма

– Лівий бік, що зображає стрілки входу, тобто «Потреба в замовленні послуги» це вхідні данні, які є необхідними для роботи ІС.

– Верхня частина схеми, що зображає стрілки контролю - документи, регламенти, що визначають умови діяльності. В даному випадку це «Право доступу користування» та «ТЗ».

– Нижня частина схеми, що зображає технічні ресурси для забезпечення роботи функціоналу, тобто «Програмне забезпечення», «Технічне забезпечення» та «База даних».

– Правий бік, що зображає стрілки виходу – це результат виконання або закінчення процесу, в даному випадку це «Вихід без вибору послуги», «Сформоване замовлення», «Обраний спеціаліст».

На рисунку 3.2 подана декомпозиція моделі роботи мобільного додатку. Для декомпозиції також був обраний метод IDEF0.

При першому рівні декомпозиції було визначено наступні процеси:

– реєстрація/авторизація;

– опрацювання каталогу послуг;

– заповнення форми послуги/підтвердження вибору послуги;

– вибір спеціаліста за зведеним рейтингом;

– перегляд результату вибору.

Розміщення стрілок несе в собі ідентичне значення до контекстної діаграми. Зокрема йде додавання інших стрілок для пояснення функціоналу додатку.

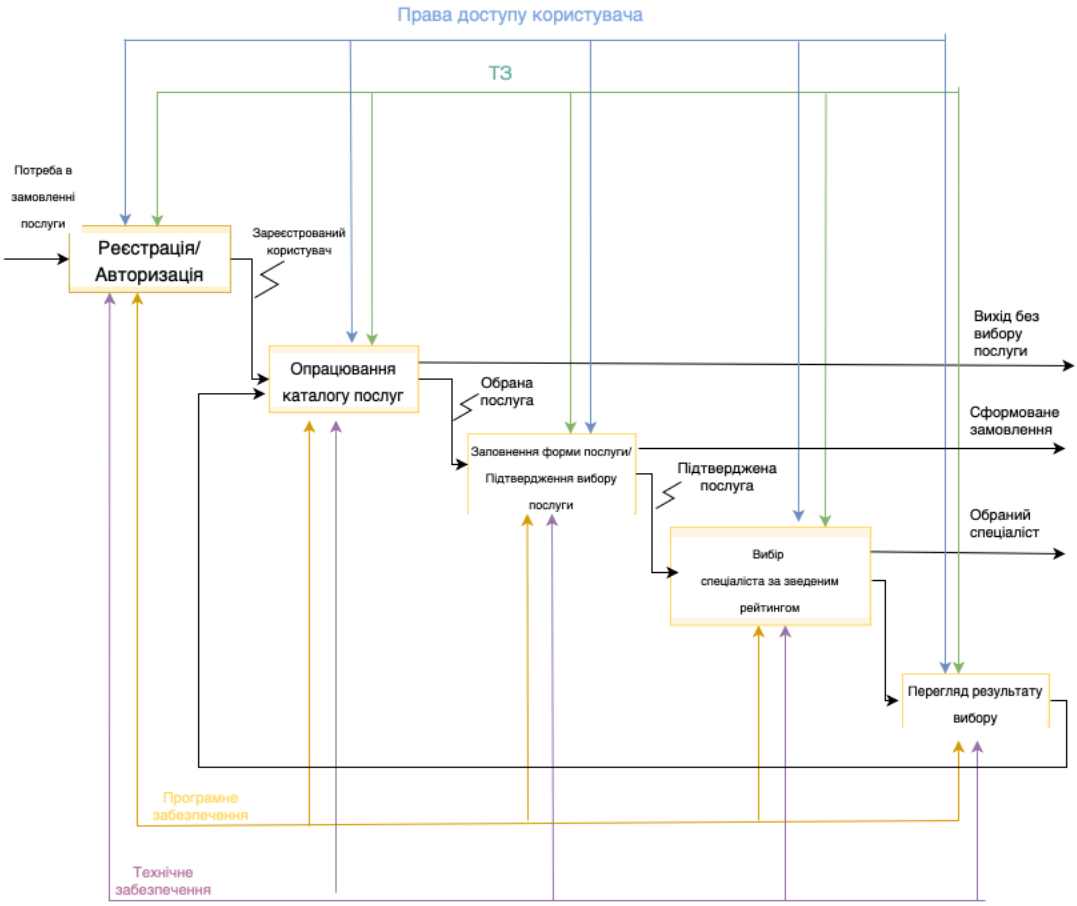

Рис. 3.2 – Декомпозиція моделі роботи мобільного додатку

За допомогою представленим вище моделям є змога наглядно продемонструвати взаємодію між компонентами ІС.

3.2. Проектування моделі баз даних

Інформаційна система (ІС) - інтегрований набір компонентів для збору, зберігання та обробки даних, а також для надання інформації, знань та цифрових продуктів. ІС повинна забезпечувати: отримання (введення або збір), зберігання, пошук, передачу та обробку даних, інформації. [9]

Розробка моделі інформаційної системи є систематизацією кожного компоненту в єдину ІС. Оскільки задачею магістерської роботи є реалізація мобільного додатку для замовлення косметичних послуг на мові C# з допомогою СУБД Firebase. [11]

Метою інформаційної моделі програмного додатку є модернізація та удосконалення бізнес процесів замовлення косметичних послуг.

Передача та обробка даних здійснюється засобами СУБД Firebase. Firebase надає в режимі реального часу базу даних та бекенд як службу. Ця служба надає розробникам застосунків API, який дозволяє синхронізувати дані застосунків між клієнтами та зберігати їх у хмарі Firebase. [10]

Після обробки запиту користувач отримує інформацію о замовленні, таку як назва послуги, ім’я виконавця, час, вартість та місцезнаходження.

Отже, всі попередньо названі компоненти складають ІС, що має деякий функціонал з замовлення послуг, що при поступовій детальній реалізації із тестуванням складають добре злагоджений програмний продукт.

Табл. 3.1 – Деякі основні таблиці бази даних.

№	Назва	Призначення
1	procedure	Зберігає первинну інформацію про процедуру з рейтингом

2	Procedure_list	Зберігає більш детальну інформацію про процедуру, з переліком майстрів
3	Users_list	Зберігає більш детальну інформацію про користувачів
4	Workers_list	Зберігає більш детальну інформацію про працівників сфери краси
5	Cabinet	Зберігає інформацію про запис клієнтів на процедури

На рисунку 3.3 зображена ER-діаграма бази даних.

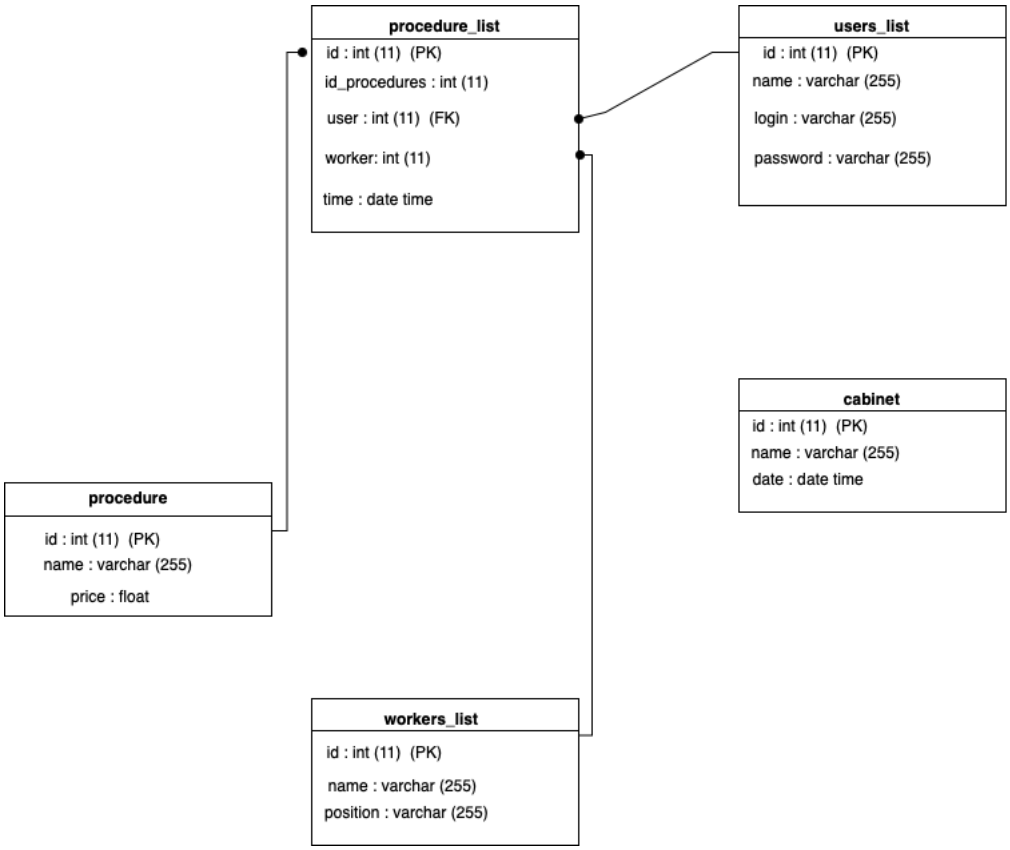

Рис. 3.3 – ER-діаграма бази даних

3.3. Моделювання варіантів використання

За допомогою діаграми варіантів використання можна зобраті функціонування системи в цілому, зокрема її складових, таких як актори, варіанти використання та відносини між ними.

На рис. 3.4 подана діаграма користувачів. Можна користувачів розділити на «Зареєстрований користувач», «Незареєстрований користувач» та «Адмін».

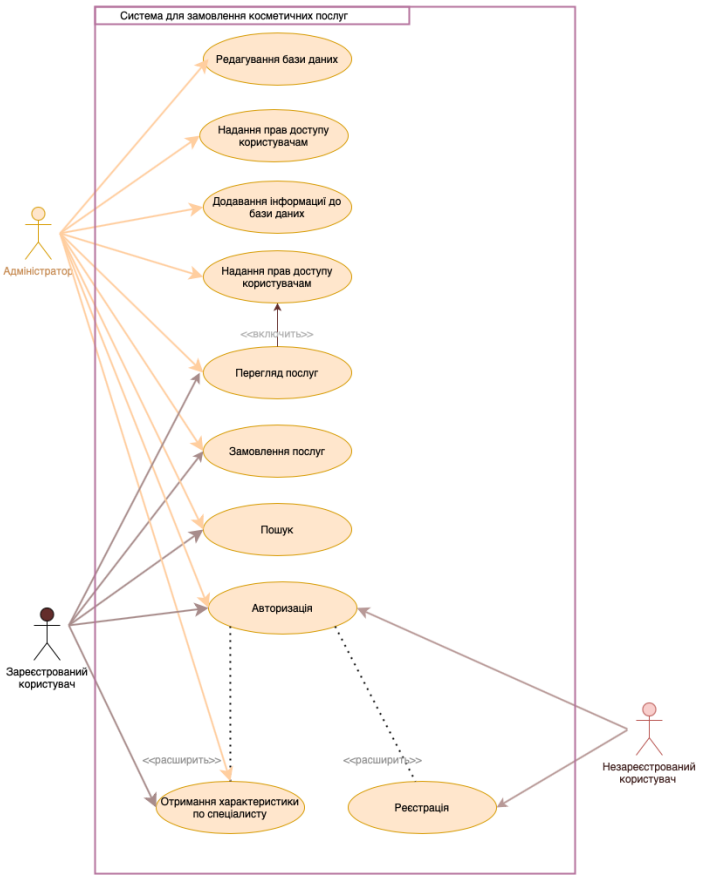

Рис. 3.4 – Діаграма використання

На рисунку зображена схема реалізації діграми користувачів, згідно якої, зареєстрований користувач має деякі свої права, незареєстрований має обмежені права, адміністратор має всі права клієнта та доступ до панелі адміністратора, а сервер працює с базою даних та оброблює запити згідно дій користувача.

4.РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ

4.1. Вибір методу реалізації

База даних, яку також називають електронною базою даних, будь-яка колекція даних або інформації, яка спеціально організована для швидкого пошуку та пошуку за допомогою комп’ютера. Бази даних структуровані так, щоб полегшити зберігання, пошук, модифікацію та видалення даних у поєднанні з різними операціями обробки даних. Система управління базами даних (СУБД) бере інформацію з бази даних у відповідь на запити. [11] [12]

СУБД дозволяє користувачам використовувати програмні додатки без оформлення. Мови запитів четвертого покоління, такі як SQL, використовуються разом із пакетом СУБД для взаємодії з базою даних.

Деякі інші приклади СУБД включають:

- MySQL;
- SQL Server;
- Oracle;
- dBASE;
- FoxPro. [13]

Створення віртуальної частини мобільного-додатку було реалізовано за допомогою мови програмування C#.

C# — це строго типізована об'єктно-орієнтована мова програмування, що є відкритим вихідним кодом, простим, сучасним, гнучким і універсальним. [14]

C# — це проста, сучасна й об'єктно-орієнтована мова програмування. Метою розробки данної мови програмування, було не тільки легкіть в виченні, а ще й підтримка сучасних функціональних можливостей для всіх видів розробки програмного забезпечення. [15]

C# розроблена, щоб враховувати бізнес концепцію або концепцію підприємств. Мова програмування створена для компаній, щоб створювати всі види програмного забезпечення за допомогою однієї мови програмування. Адже вона (мова програмування C#) надає функціональні можливості для підтримки сучасної розробки програмного забезпечення, підтримує потреби в розробці веб-, мобільних пристроїв і застосунків. Один з функціоналу, що містить сьогоденні мови програмування, які підтримує C#, — це генерики, типи змінних, автоматична ініціалізація типів і колекцій, лямбда-вирази, динамічне програмування, асинхронне програмування, кортежи, зіставлення шаблонів, розширене налагодження та обробка винятків тощо. [16]

Серверна частина реалізована засобами СУБД Firebase, так як вона є дінамичною та оброблює дані в реальному часі.

Firebase — це бекенд-як-сервіс (Baas). Він надає розробникам різноманітні інструменти та послуги, які допомагають їм розробляти якісні програми, розширювати базу користувачів та отримувати прибуток. Він побудований на інфраструктурі Google. [17]

Firebase класифікується як програма баз даних NoSQL, яка зберігає дані в JSON-подібних документах. [18]

Мобільна платформа для розробки від Google. Firebase, представлена в 2012 році, — це комплексна система для розробника, яка включає модулі для аутентифікації, розміщені в хмарі бази даних, зберігання файлів, обробку на стороні сервера, доставку вмісту, обмін повідомленнями, інтеграцію пошуку Google, обмін повідомленнями в додатку, віддалену конфігурацію та машинне навчання. Firebase також включає тестування та моніторинг продуктивності. [19]

Підключення до БД зображено на рисунку 4.1

```
class fireHelper
{
    IFirebaseConfig ifc = new FirebaseConfig()
    {
        AuthSecret = "aJjPE93WHoJDrMTvZYtXBHpiGwCaSvt2ac8FVLrs",
        BasePath = "https://cosmetics-b486c-default-rtdb.firebaseio.com/"
    };
    IFirebaseClient client;
    public fireHelper()
    {
```

Рис. 4.1. – Підключення до БД

4.2. Практична реалізація

Оскільки було реалізовано мобільний додаток, то використання з боку користувача виглядає наступним чином. Якщо юзер незареєстрований, то він повинен пройти процедуру реєстрації, де після вводу усіх необхідних данних отримає наступне повідомлення.

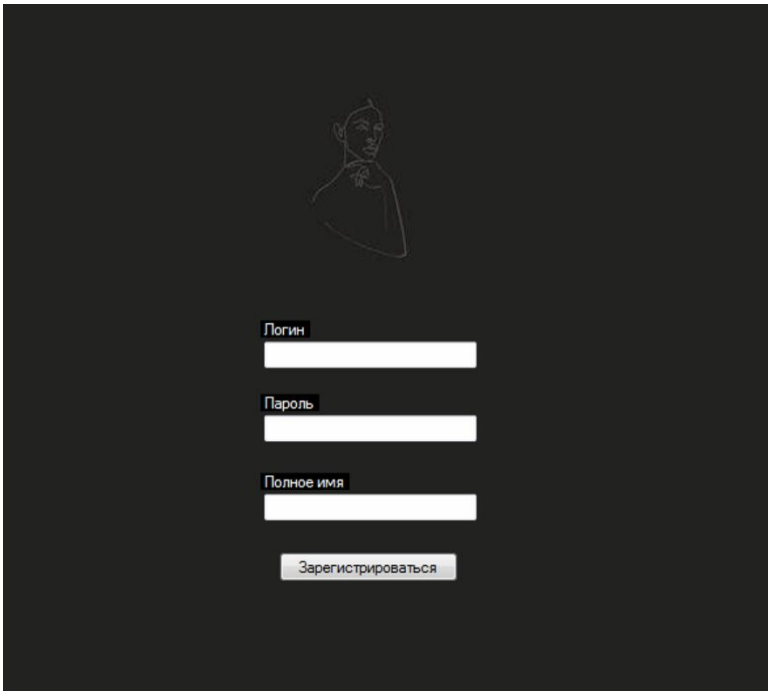

Рис. 4.2. – Реєстрація користувача

Після успішної реєстрації необхідно авторизуватись в системі.

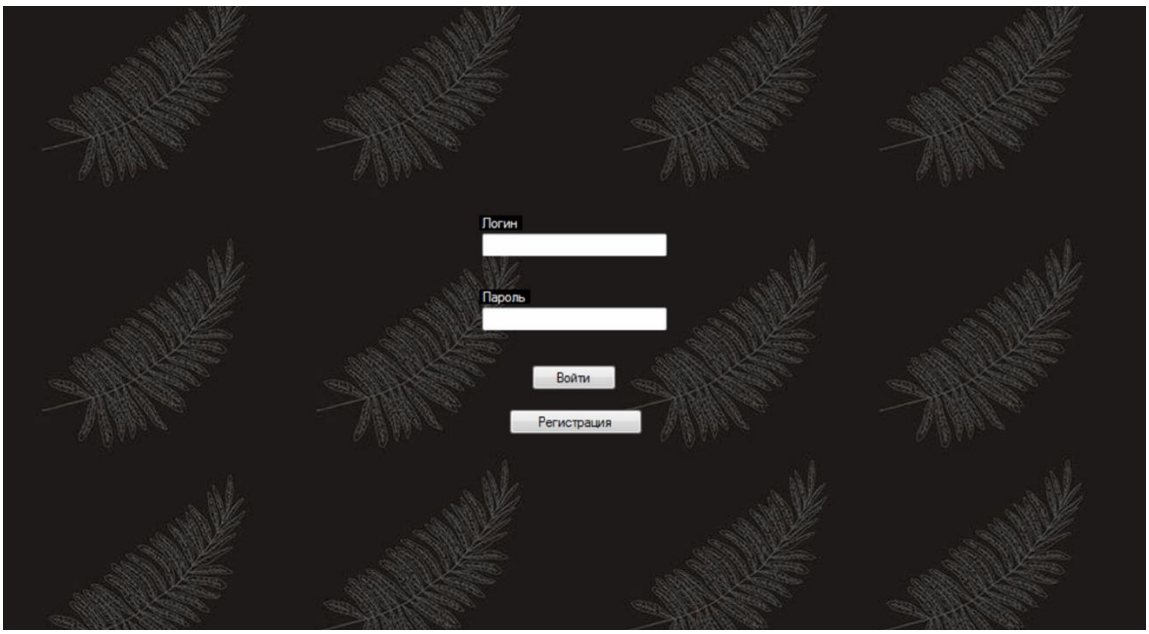

Рис. 4.3. – Авторизація користувача

Після успішної реєстрації необхідно авторизації в системі йде перенаправлення на сторінку з переліком послуг.

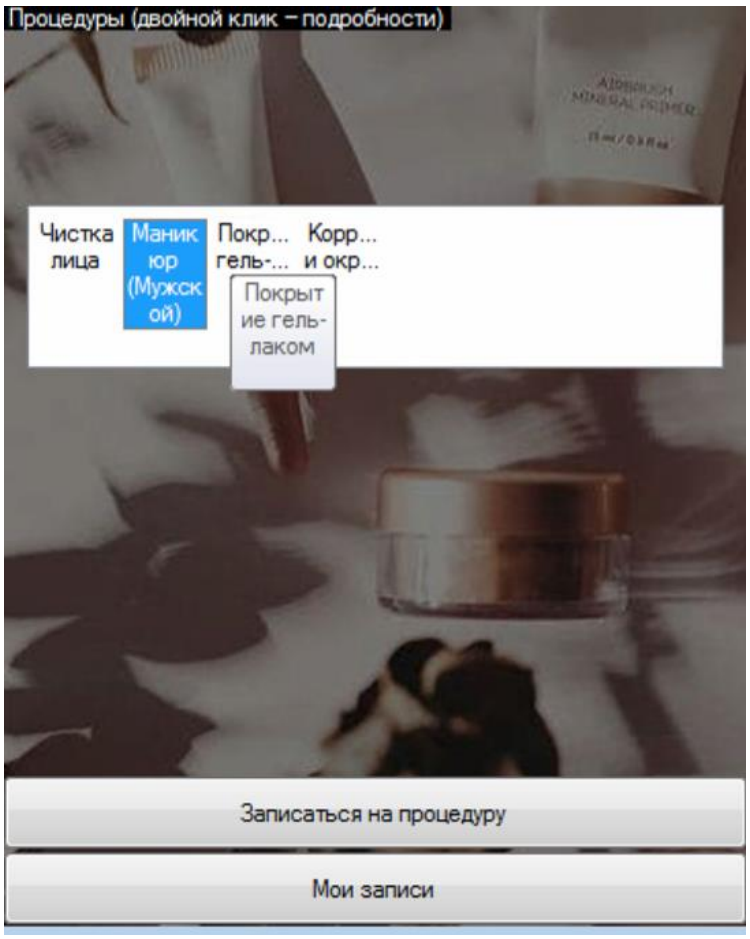

Рис. 4.4. – Перелік косметичних послуг

Натиснувши двічі, можно отримати інформацію про наявні косметичні послуги.

Рис. 4.5. – Інформація про косметичну послугу

Після обрання необхідної косметисної послуги, де обираємо усі необхідні параметри. Також при наведенні на майстра є його рейтинг та відомості. Останні відсортовані за рейтингом, з кращим рейтингом знаходяться на перших позиціях відповідно.

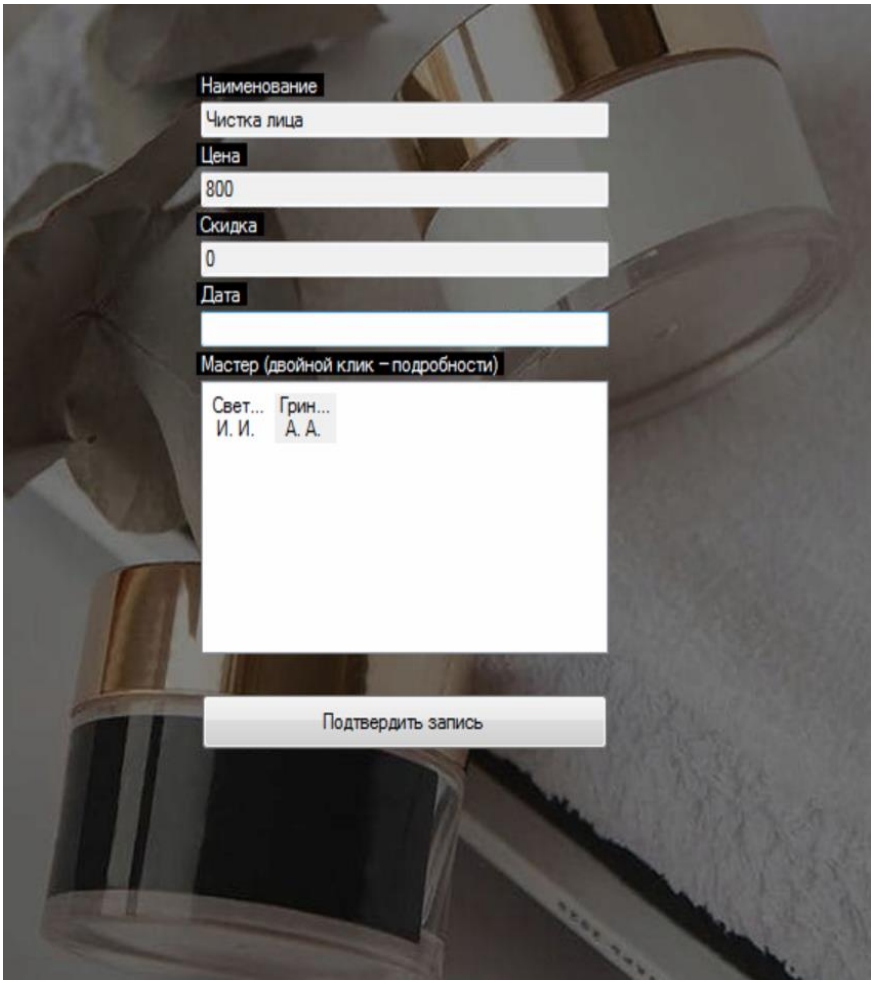

Рис. 4.6. – Замовлення косметичної послуги

Після здійснення замовлення, отримуємо наступне повідомлення.

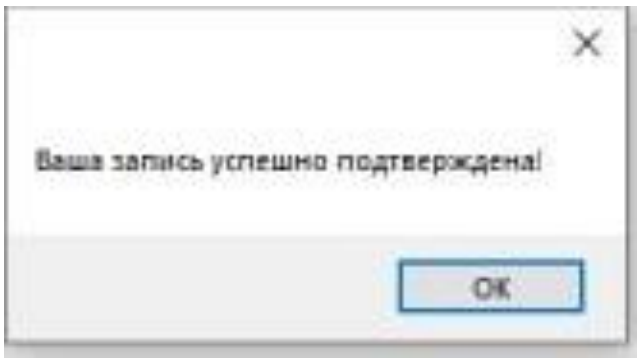

Рис. 4.7. – Повідомлення про підтвердження замовлення

Тепер переходимо до панелі адміністратора. Здійснивши вхід, додамо нову косметичну процедуру та нового майстра.

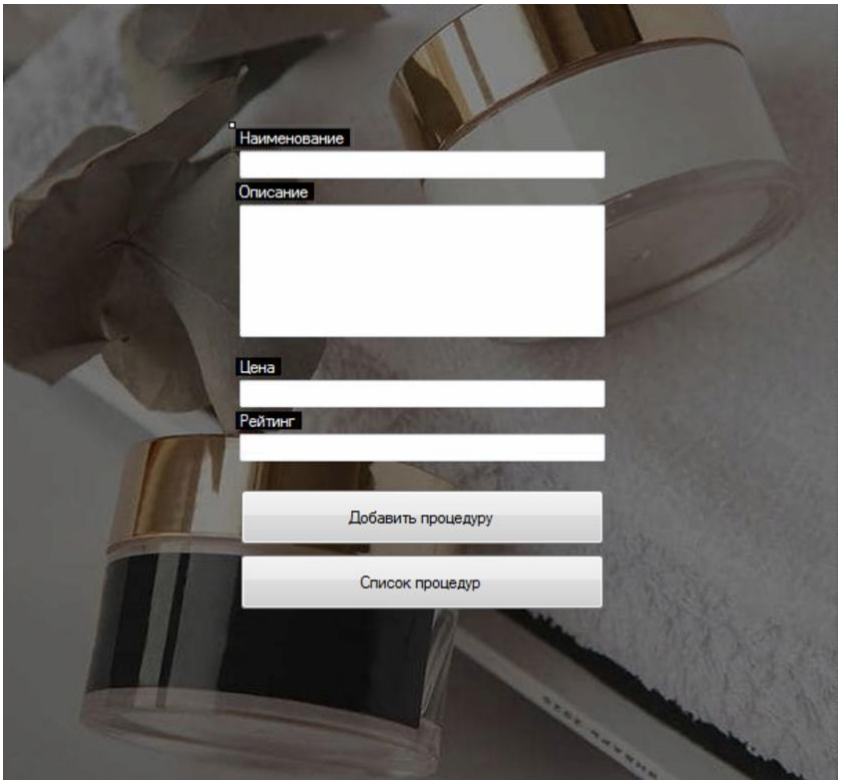

Рис. 4.8. – Додавання нової процедури

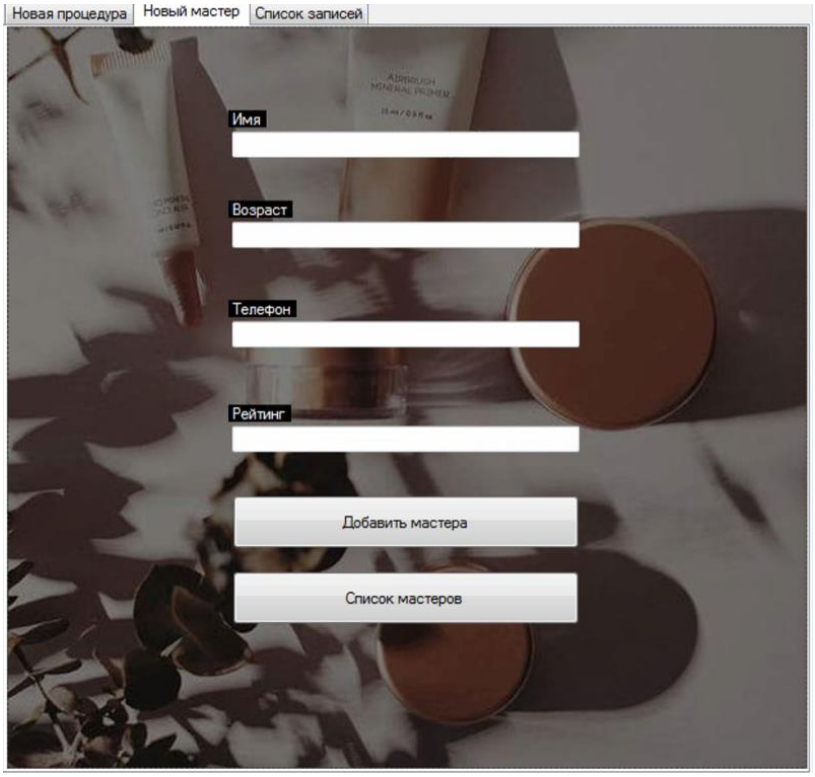

Рис. 4.9. – Додавання нового майстра

Вказавши необхідні данні, а саме назву, опис та ціну, отримуємо повидомлення. Також процедура з`являється в списку усіх процедур.

Також адмін має право перегляду замовлених процедур, так само як і користувач.

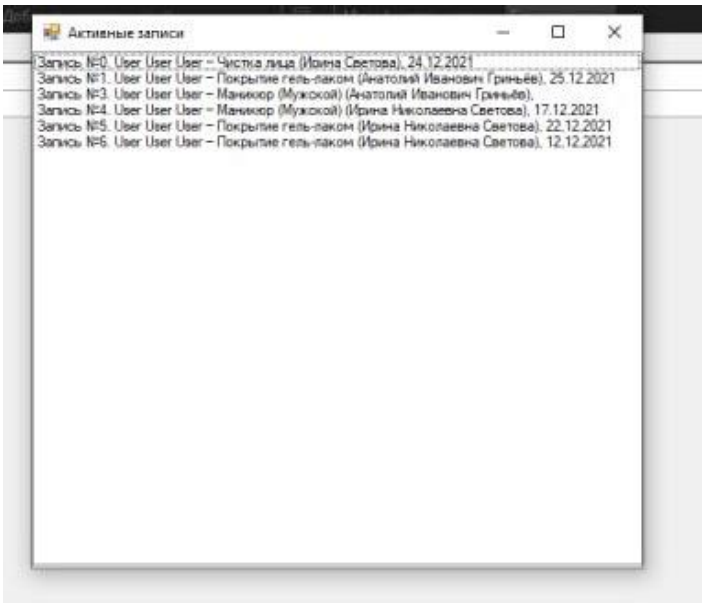

Рис. 4.10. – Список замовлених процедур

ВИСНОВОК

Таким чином, у ході дипломної роботи було запропоновано створення проекту для замовлення косметичних послуг. Програму реалізовано у формі мобільного додатку за допомогою мови програмування C# та СУБД Firebase з можливістю безкоштовного використання, що буде мати велику перевагу над іншими додатками із косметичної сфери. Тому послуги швидко знайдуть свого потенціального клієнта.

Такий вид замовлення є досить актуальним в наш час пандемії. Так мінімізуються черги, а користувач матиме змогу зробити це в будь-який час доби. Що неодмінно матиме перевагу над будь-якими іншими додатками.

Також було проведено аналіз та виявлено, що такий застосунок може покращити роботу майданчику, який надає дані послуги. Зараз ще нерідко можна зустріти запис та ведення статистистики через дзвінки та помітки в звичайних журналах. Ефективності та зворотнього зв’язку із такої реалізації майже немає. Тому

що клієнтська база може втратися, що призведе до збоїв в роботі. Натомість електронна клієнтська база не тільки є надійною, а ще може принести масу переваг не тільки для споживачів, але й для власників. Це дозволить впровадити рейтинг майстрів.

Отже, мобільний додаток із зручним та зрозумілим дизайн не залишить байдужим жодного клієнта будь-якої вікової категорії. Тобто така інформаційна система не тільки покращить роботу, а ще й більш точніше систематизує її.

СПИСОК ЛІТЕРАТУРИ

1. “App” voted 2010 word of the year by the American Dialect Society (UPDATED).
2. Види мобільних додатків [Електронний ресурс] – Режим доступу до ресурсу: https://www.thedroidsonroids.com/blog/what-is-a-mobile-app-app-development-basics-for-businesses.
3. Додатки для замовлення косметичних послуг [Електронний ресурс] – Режим доступу до ресурсу: https://postium.ru/programmy-i-servisy-onlajn-zapisi-klientov-10-luchshix/.
4. «The Future of Mobile Application». UAB. Retrieved 11 November 2015. «10 Billion App Countdown». Apple. 2011-01-14.
5. Miller, Michael (September 14, 2011). "Build: More Details On Building Windows 8 Metro Apps". PC Magazine. Archived from the original on February 17, 2012. Retrieved February 10, 2012.

6. Microsoff, Matt (February 9, 2012). "Here's Everything You Wanted To Know About Microsoft's Upcoming iPad Killers". Business Insider. Retrieved December 11, 2017.
7. Розробка мобільного додатка [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Mobile_app_development.
8. The Majority of Americans' Mobile Time Spent Takes Place in Apps [Електронний ресурс] – Режим доступу до ресурсу: https://www.emarketer.com/content/the-majority-of-americans-mobile-time-spent-takes-place-in-apps.
9. Інформаційна система [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Інформаційна_система
10. Firebase [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Firebase.
11. ГОСТ Р ІСО МЭК ТО 10032-2007: Эталонна модель управління даними (ідентифікований ISO/IEC TR 10032:2003 Інформаційні технології. Довідкова модель управління даними).
12. ISO/IEC TR 10032:2003 Information technology — Reference Model of Data Management [Електронний ресурс] – Режим доступу до ресурсу: https://www.iso.org/standard/38607.html
13. ГОСТ 33707-2016 (ISO/IEC 2382:2015) Інформаційні технології (IT). Словник
14. C# [Електронний ресурс] – Режим доступу до ресурсу: https://docs.microsoft.com/ru-ru/dotnet/csharp/tour-of-csharp/.
15. SO C# Language Specification.
16. C# [Електронний ресурс] – Режим доступу до ресурсу: https://docs.microsoft.com/ru-ru/dotnet/csharp/tour-of-csharp/tutorials/.
17. Firebase [Електронний ресурс] – Режим доступу до ресурсу: Firebase expands to become a unified app platform.
18. "Firebase is launching Cloud Firestore, a new document database featuring realtime sync, no-hassle scaling, and offline support". Android Police. 2017-10-03.
19. "Google's Cloud Firestore Lets You Focus On App Development | Androidheadlines.com". AndroidHeadlines.com. 2017-10-05.

20. WBS [Електронний ресурс] – Режим доступу до ресурсу:https://www.adeaca.com/blog/faq-items/what-is-a-work-breakdown-structure/.
21. OBS [Електронний ресурс] – Режим доступу до ресурсу: https://uplandsoftware.com/psa/resources/glossary/organization-breakdown-structure-obs/.

ДОДАТОК А. ПЛАНУВАННЯ РОБІТ

А.1 Ідентифікація мети ІТ-проекту

Деталізація мети проекту методом SMART. Продуктом дипломного проекту є мобільний додаток для замовлення косметичних послуг.

Результати деталізації методом SMART розміщені у табл. Б.1.

Таблиця Б.1 − Деталізація мети методом SMART

Specific (конкретна)	Розробити мобільний додаток для замовлення послуг краси.
Measurable (вимірювана)	Розробити мобільний додаток за три місяці.
Achievable (досяжна)	Поступова розробка додатку під смартфон

Relevant (реалістична)	Розробка класичного мобільного додатку під Android
Time-framed (обмежена у часі)	Розробка мобільного додатку впродож трьох місяців

1.2. Планування змісту структури робіт IT-проекту (WBS)

Структура розбивки робіт (WBS) — це ієрархічний план завдань, необхідних для завершення проекту. WBS «розбиває» структуру проекту на керовані результати. Кожному результату призначається завдання або серія завдань, які можна додатково розбити на підзадачі для задоволення потреб проекту.

Перевага використання WBS як частини управління життєвим циклом проекту полягає в тому, що він приймає великі, складні проекти і розбиває їх на менші, більш керовані завдання, які можна призначити окремим людям або командам для виконання.

Основною метою WBS є занадте планування графіка проекту. Тривалість кожного завдання планується разом із необхідними попередніми та наступними завданнями. Потім WBS надає загальний план, щоб керівник проекту міг бачити, як проект має розвиватися, і належним чином керувати робочим процесом.

Частини WBS включають:

- Завдання – номер, ідентифікатор, назва та опис кожного завдання.
- Власник завдання – відповідальний за виконання завдання.
- Залежність завдань і попередники – поєднання двох завдань разом, якщо одне залежить від виконання іншого.
- Дата початку та завершення завдання – оцінює час, який займе кожне завдання і, зрештою, весь проект.
- Тривалість – тривалість кожного завдання в календарі (зазвичай кількість днів або годин).
- Оцінка роботи – скільки годин/днів роботи потрібно для виконання завдання (об’єднання всіх ресурсних годин разом, якщо працюєте паралельно).

- Статус завдання – чи призначається кожне завдання власнику/ресурсу, розпочато, виконується, пізно, завершено тощо.
- Діаграма Ганта – візуалізація WBS із завданнями, представленими графічно з плином часу. [20]

Таблиця WBS була побудована на рис. Б.1.

Організаційна структура проекту (OBS) - це ієрархічна модель, що описує встановлену організаційну структуру для планування проекту, управління ресурсами, відстеження часу та витрат, розподілу витрат, звітності про доходи/прибутки та управління роботою.

Структура розбивки робіт (WBS) організовано фіксує всі елементи проектів. Розбиття великих, складних проектів на більш дрібні частини проекту забезпечує кращу основу для організації та управління поточними та майбутніми проектами. WBS полегшує розподіл ресурсів, призначення завдань, вимірювання та контроль вартості проекту та виставлення рахунків. WBS використовується на початку проекту для визначення обсягу, визначення центрів витрат і є відправною точкою для розробки планів проекту/діаграм Ганта.

Структура розбивки організації об’єднує подібні заходи проекту або «робочі пакети» та пов’язує їх із структурою організації. OBS (також відома як організаційна структура розбивки) використовується для визначення відповідальності за управління проектом, звітність про витрати, виставлення рахунків, складання бюджету та контроль проекту. OBS забезпечує організаційну перспективу проекту, а не задачу. Ієрархічна структура OBS дозволяє агрегувати (згортати) інформацію про проект на вищих рівнях. Коли обов’язки за проектом визначені та призначено роботу, OBS і WBS підключаються, надаючи потужну аналітику можливість вимірювати продуктивність проекту та робочої сили на дуже високому рівні (приклад продуктивності бізнес-підрозділу) або до деталей (приклад роботи користувача над завдання).

Таблиця OBS побудована на рис. Б.2. [21]

1.3. Побудова календарного графіка виконання ІТ-проекту. Календарний план для проекту: це корисний спосіб показати, яку роботу заплановано виконати в певні дні. Це допомагає менеджерам проектів і членам команд переглядати дати початку, дати завершення та віхи розкладу проекту на одній простій гістограмі з накопиченням.

Для кращої розробки календарного плану будемо використовувати діаграму Ганта. Діаграма Ганта, яка зазвичай використовується в управлінні проектами, є одним з найпопулярніших і найкорисніших способів відображення активності (завдань або подій), що відображається в залежності від часу. Ліворуч від діаграми наведено список заходів, а вгорі — відповідна шкала часу. Кожна діяльність представлена смугою; положення та довжина смуги відображає дату початку, тривалість та дату завершення діяльності. Це дозволяє з першого погляду побачити:

- Які різні види діяльності;
- Коли кожна діяльність починається і закінчується;
- Скільки триватиме кожна діяльність;
- Де види діяльності перетинаються з іншими видами діяльності та наскільки;
- Дата початку та закінчення всього проекту.

За допомогою програми Microsoft Project Professional побудовано діаграму Ганта (рис. Б.3).

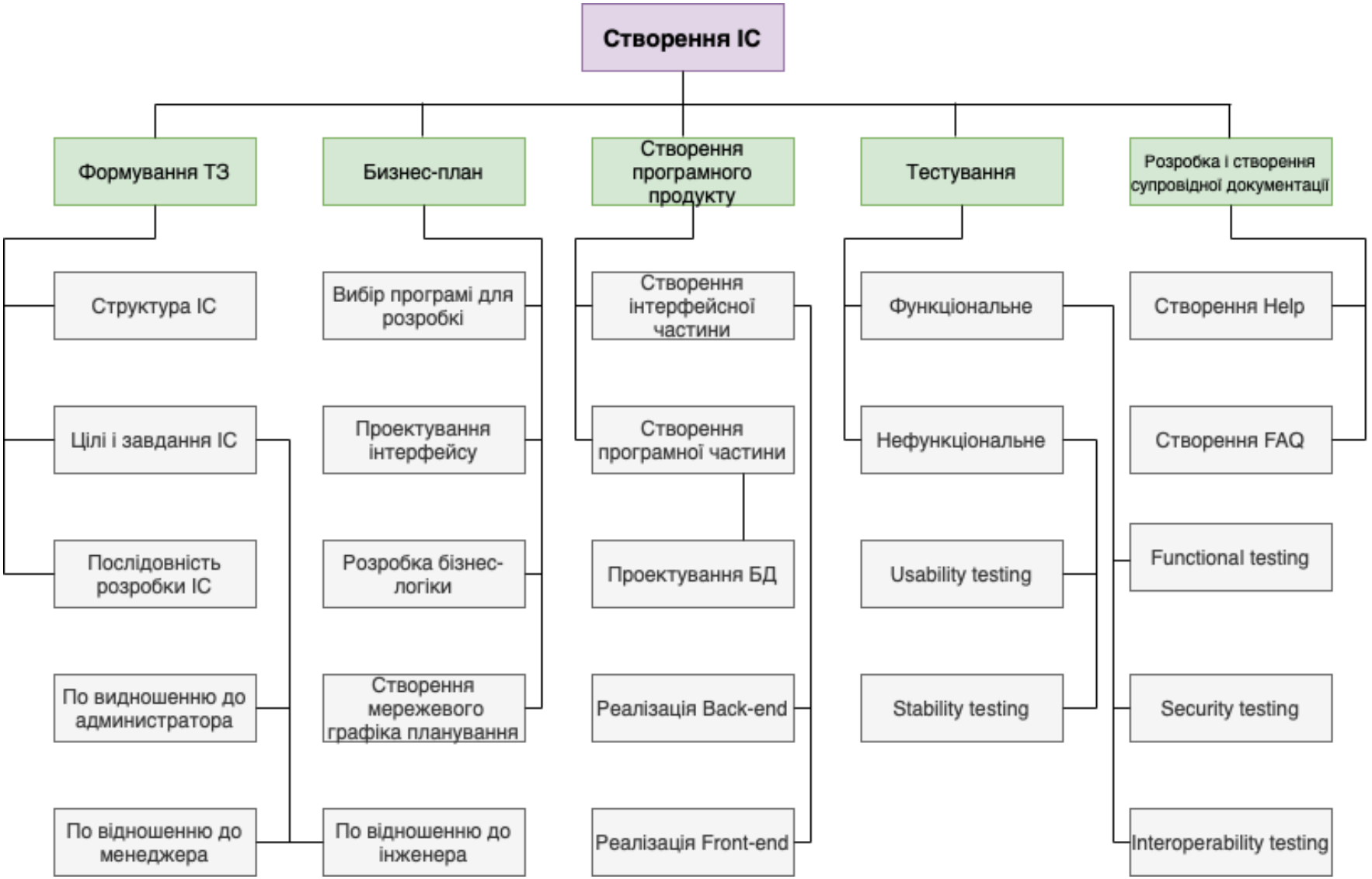

Рисунок Б.1 – таблиця WBS (work breakdown structure)

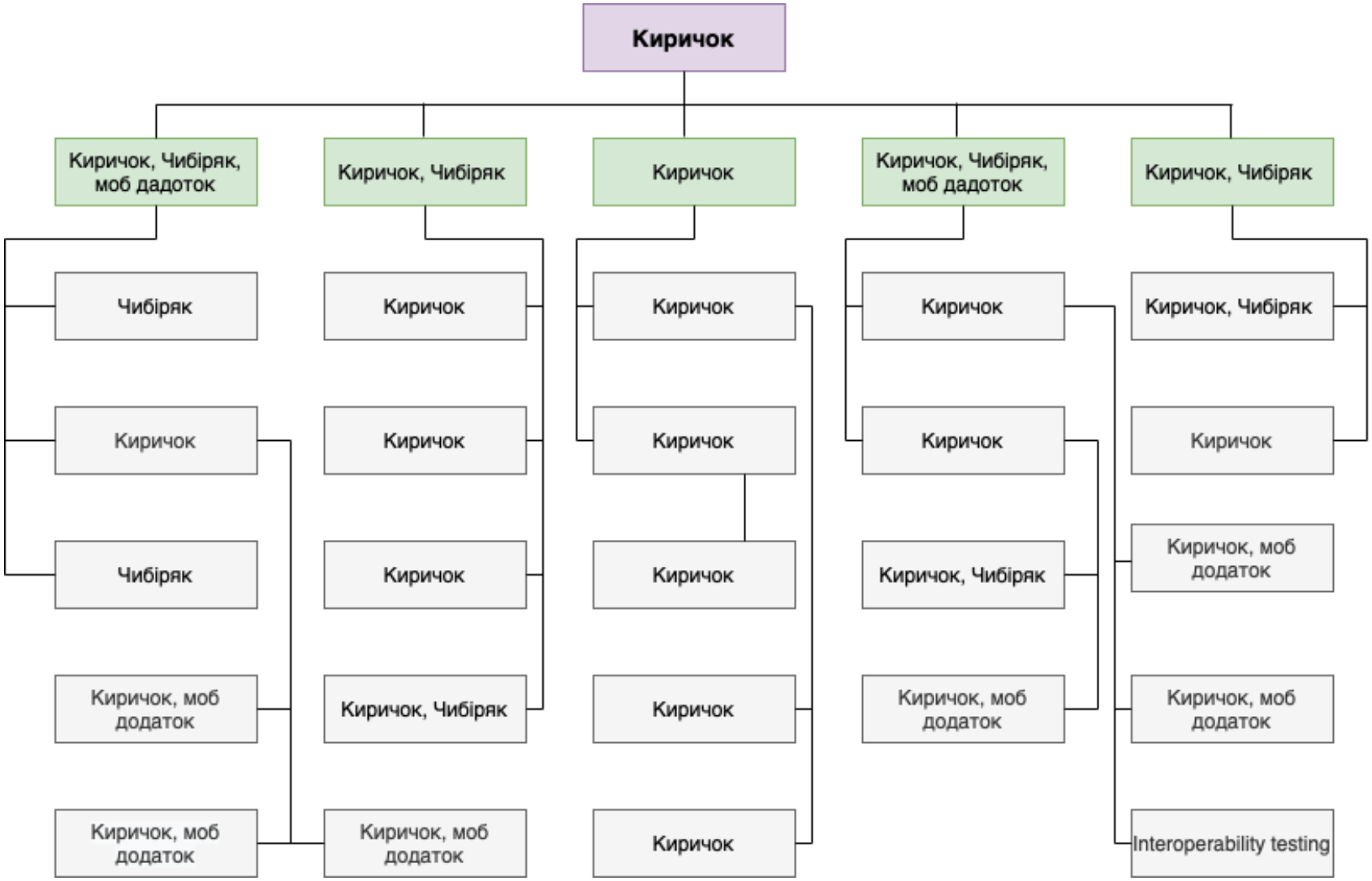

Рисунок Б.2 – таблиця OBS (Organization Breakdown Structure)

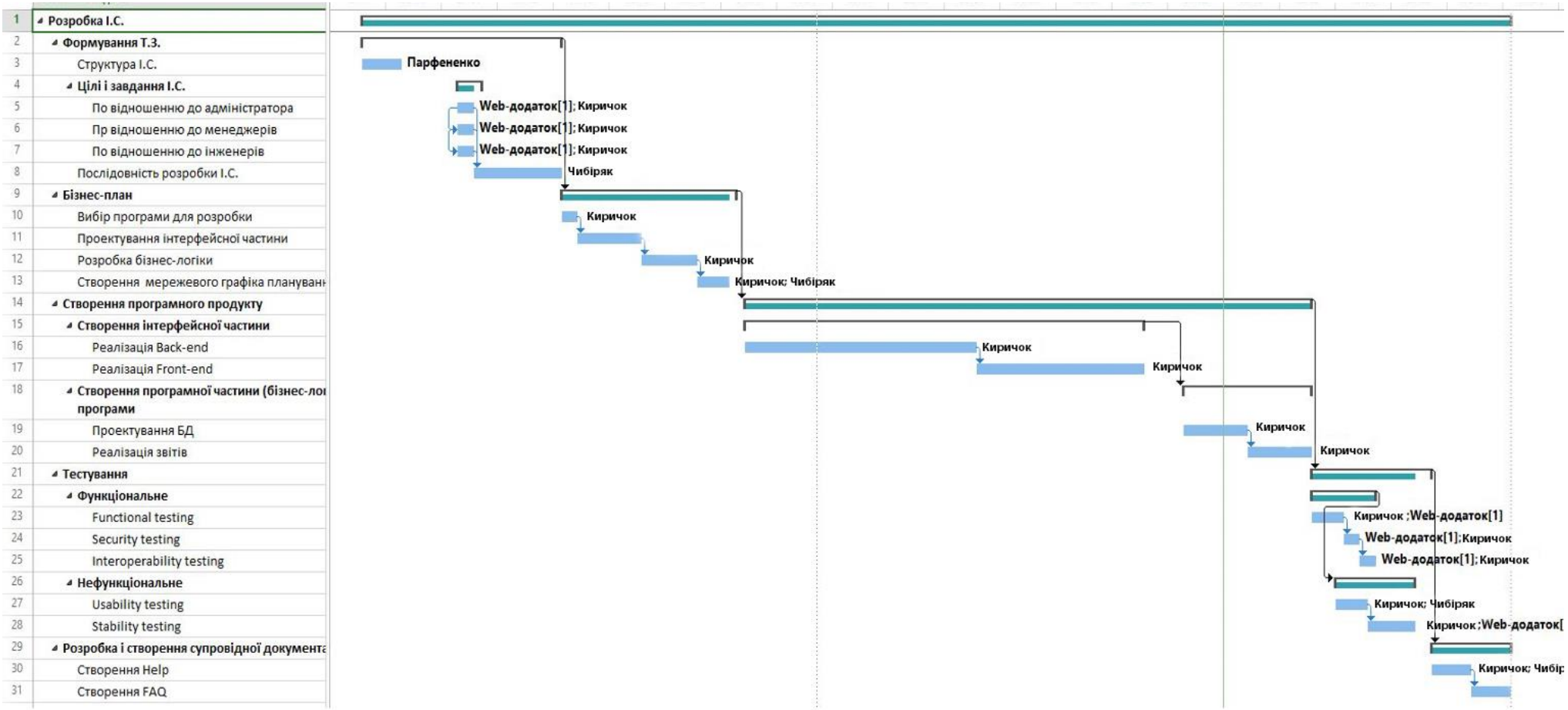

Рисунок Б.3 – Діаграма Ганта

1.4. Управління ризиками. Основними ризиками розробки «ІС для замовлення косметичних послуг» є:

- неглибоке вивчення предметної області неповне дослідження ;
- збільшення навантаження під час реалізації проекту;
- зміна цілей у ході реалізації проекту;
- людський фактор;
- відмова обладнання;
- відсутність кваліфікованого програміста;
- зміна строків виконання роботи;
- незнаходження спільної мови між керівником і виконавцем;
- зростання вимог до проекту.

Таблиця Б.2. Ймовірність виникнення і величина ризику

№	Ризики	Виникнення	Втрати
1	Відсутність досвіду	3	2
2	Збільшення навантаження під час реалізації проекту	3	2
3	Зміна цілей у ході реалізації проекту	4	4
4	Людський фактор	2	3
5	Відмова обладнання	2	4
6	Відсутність кваліфікованого програміста	2	5
7	Зміна строків виконання роботи	2	4
8	Складнощі при впровадженні на місці	3	3
9	Зростання вимог до проекту	3	3

Матриця «Ймовірність – Втрати»

Ймовірність							
	-	5	5	10	15	20	25
	Зміна цілей у ході реалізації проекту	4	4	8	12	16	20
	Збільшення навантаження під час реалізації проекту Зростання вимог до проекту	3	3	6	9	12	15
	Відсутність досвіду Людський фактор Відмова обладнання Відсутність кваліфікованого програміста Зміна строків виконання роботи Складнощі при впровадженні на місці	2	2	4	6	8	10
	-	1	1	2	3	4	5
			1	2	3	4	5
			-	Збільшення навантаження під час реалізації проекту	Людський фактор Складнощі при впровадженні на місці Зростання вимог до проекту	Відсутність досвіду Зміна цілей у ході реалізації проекту Відмова обладнання Зміна строків виконання роботи	Відсутність кваліфікованого програміста
			Втрати				

Аналізуючи ризики за ймовірністю їх виникнення, можемо їх розділити на:

- Ігноровані
    - Відсутні
- Незначні
    - Збільшення навантаження під час реалізації проекту
    - Людський фактор
    - Відмова обладнання
    - Зміна строків виконання роботи
- Помірні
    - Відсутність кваліфікованого програміста
    - Зростання вимог до проекту
    - Відсутність досвіду
    - Складнощі при впровадженні на місці

- Істотні
    - Зміна цілей у ході реалізації проекту
- Критичні
    - Відсутні

Класифікація ризиків за рівнем впливу:

- Прийнятні
    - Відсутні
- Виправдані
    - Неглибоке вивчення предметної області
    - Збільшення навантаження під час реалізації проекту
    - Людський фактор
    - Відмова обладнання
    - Відсутність кваліфікованого програміста
    - Зміна строків виконання роботи
    - Не знаходження спільної мови між керівником і виконавцем
    - Зростання вимог до проекту
- Неприпустимі
    - Зміна цілей у ході реалізації проекту.

ДОДАТОК Б. Лістинг коду файлу admin_view.cs

```
namespace cosmetic_service
{
    partial class create_note
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.label3 = new System.Windows.Forms.Label();
            this.textBox3 = new System.Windows.Forms.TextBox();
            this.label1 = new System.Windows.Forms.Label();
            this.textBox1 = new System.Windows.Forms.TextBox();
            this.label2 = new System.Windows.Forms.Label();
            this.textBox2 = new System.Windows.Forms.TextBox();
            this.label4 = new System.Windows.Forms.Label();
            this.listView1 = new System.Windows.Forms.ListView();
            this.button1 = new System.Windows.Forms.Button();
            this.SuspendLayout();
            //
            // label3
            //
            this.label3.AutoSize = true;
            this.label3.Location = new System.Drawing.Point(52, 74);
            this.label3.Name = "label3";
            this.label3.Size = new System.Drawing.Size(33, 13);
```

```
            this.label3.TabIndex = 9;
            this.label3.Text = "Цена";
            //
            // textBox3
            //
            this.textBox3.Location = new System.Drawing.Point(55, 90);
            this.textBox3.Name = "textBox3";
            this.textBox3.ReadOnly = true;
            this.textBox3.Size = new System.Drawing.Size(265, 20);
            this.textBox3.TabIndex = 8;
            //
            // label1
            //
            this.label1.AutoSize = true;
            this.label1.Location = new System.Drawing.Point(52, 35);
            this.label1.Name = "label1";
            this.label1.Size = new System.Drawing.Size(83, 13);
            this.label1.TabIndex = 7;
            this.label1.Text = "Наименование";
            //
            // textBox1
            //
            this.textBox1.Location = new System.Drawing.Point(55, 51);
            this.textBox1.Name = "textBox1";
            this.textBox1.ReadOnly = true;
            this.textBox1.Size = new System.Drawing.Size(265, 20);
            this.textBox1.TabIndex = 6;
            //
            // label2
            //
            this.label2.AutoSize = true;
            this.label2.Location = new System.Drawing.Point(52, 113);
            this.label2.Name = "label2";
            this.label2.Size = new System.Drawing.Size(33, 13);
            this.label2.TabIndex = 11;
            this.label2.Text = "Дата";
            //
            // textBox2
            //
            this.textBox2.Location = new System.Drawing.Point(55, 129);
            this.textBox2.Name = "textBox2";
            this.textBox2.Size = new System.Drawing.Size(265, 20);
            this.textBox2.TabIndex = 10;
            //
            // label4
            //
            this.label4.AutoSize = true;
            this.label4.Location = new System.Drawing.Point(52, 152);
            this.label4.Name = "label4";
            this.label4.Size = new System.Drawing.Size(200, 13);
            this.label4.TabIndex = 12;
            this.label4.Text = "Мастер (двойной клик — подробности)";
            //
```

```
            // listView1
            //
            this.listView1.HideSelection = false;
            this.listView1.Location = new System.Drawing.Point(55, 168);
            this.listView1.Name = "listView1";
            this.listView1.Size = new System.Drawing.Size(265, 243);
            this.listView1.TabIndex = 13;
            this.listView1.UseCompatibleStateImageBehavior = false;
            this.listView1.MouseDoubleClick += new System.Windows.Forms.MouseEventHandler(this.listView1_MouseDoubleClick);
            //
            // button1
            //
            this.button1.Location = new System.Drawing.Point(55, 417);
            this.button1.Name = "button1";
            this.button1.Size = new System.Drawing.Size(265, 32);
            this.button1.TabIndex = 14;
            this.button1.Text = "Подтвердить запись";
            this.button1.UseVisualStyleBackColor = true;
            this.button1.Click += new System.EventHandler(this.button1_Click);
            //
            // create_note
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.ClientSize = new System.Drawing.Size(372, 461);
            this.Controls.Add(this.button1);
            this.Controls.Add(this.listView1);
            this.Controls.Add(this.label4);
            this.Controls.Add(this.label2);
            this.Controls.Add(this.textBox2);
            this.Controls.Add(this.label3);
            this.Controls.Add(this.textBox3);
            this.Controls.Add(this.label1);
            this.Controls.Add(this.textBox1);
            this.Name = "create_note";
            this.Text = "Новая запись";
            this.ResumeLayout(false);
            this.PerformLayout();

        }

        #endregion

        private System.Windows.Forms.Label label3;
        private System.Windows.Forms.TextBox textBox3;
        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.TextBox textBox1;
        private System.Windows.Forms.Label label2;
        private System.Windows.Forms.TextBox textBox2;
        private System.Windows.Forms.Label label4;
        private System.Windows.Forms.ListView listView1;
        private System.Windows.Forms.Button button1;
    }
```

ДОДАТОК В. Лістинг коду файлу adminview.resx

```
<xsd:schema id="root" xmlns="" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:msdata="urn:schemas-microsoft-com:xml-msdata">
  <xsd:import namespace="http://www.w3.org/XML/1998/namespace" />
  <xsd:element name="root" msdata:IsDataSet="true">
    <xsd:complexType>
      <xsd:choice maxOccurs="unbounded">
        <xsd:element name="metadata">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="value" type="xsd:string" minOccurs="0" />
            </xsd:sequence>
            <xsd:attribute name="name" use="required" type="xsd:string" />
            <xsd:attribute name="type" type="xsd:string" />
            <xsd:attribute name="mimetype" type="xsd:string" />
            <xsd:attribute ref="xml:space" />
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="assembly">
          <xsd:complexType>
            <xsd:attribute name="alias" type="xsd:string" />
            <xsd:attribute name="name" type="xsd:string" />
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="data">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="value" type="xsd:string" minOccurs="0" msdata:Ordinal="1" />
              <xsd:element name="comment" type="xsd:string" minOccurs="0" msdata:Ordinal="2" />
            </xsd:sequence>
            <xsd:attribute name="name" type="xsd:string" use="required" msdata:Ordinal="1" />
            <xsd:attribute name="type" type="xsd:string" msdata:Ordinal="3" />
            <xsd:attribute name="mimetype" type="xsd:string" msdata:Ordinal="4" />
            <xsd:attribute ref="xml:space" />
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="resheader">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="value" type="xsd:string" minOccurs="0" msdata:Ordinal="1" />
            </xsd:sequence>
            <xsd:attribute name="name" type="xsd:string" use="required" />
          </xsd:complexType>
        </xsd:element>
      </xsd:choice>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
<resheader name="resmimetype">
  <value>text/microsoft-resx</value>
</resheader>
<resheader name="version">
```

```
  <value>2.0</value>
 </resheader>
 <resheader name="reader">
  <value>System.Resources.ResXResourceReader, System.Windows.Forms, Version=4.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089</value>
 </resheader>
 <resheader name="writer">
  <value>System.Resources.ResXResourceWriter, System.Windows.Forms, Version=4.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089</value>
 </resheader>
</root>
```

ДОДАТОК Г. Лістинг коду файлу adminview.Designer.cs

```
namespace cosmetic_service
{
    partial class admin_view
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }
        #region Windows Form Designer generated code
        private void InitializeComponent()
        {
            this.tabControl1 = new System.Windows.Forms.TabControl();
            this.tabPage1 = new System.Windows.Forms.TabPage();
            this.button3 = new System.Windows.Forms.Button();
            this.button1 = new System.Windows.Forms.Button();
            this.label3 = new System.Windows.Forms.Label();
            this.textBox3 = new System.Windows.Forms.TextBox();
            this.label2 = new System.Windows.Forms.Label();
            this.textBox2 = new System.Windows.Forms.TextBox();
            this.label1 = new System.Windows.Forms.Label();
            this.textBox1 = new System.Windows.Forms.TextBox();
            this.tabPage2 = new System.Windows.Forms.TabPage();
            this.button4 = new System.Windows.Forms.Button();
            this.label7 = new System.Windows.Forms.Label();
            this.textBox7 = new System.Windows.Forms.TextBox();
            this.button2 = new System.Windows.Forms.Button();
            this.label4 = new System.Windows.Forms.Label();
            this.textBox4 = new System.Windows.Forms.TextBox();
            this.label5 = new System.Windows.Forms.Label();
            this.textBox5 = new System.Windows.Forms.TextBox();
            this.label6 = new System.Windows.Forms.Label();
            this.textBox6 = new System.Windows.Forms.TextBox();
            this.tabPage3 = new System.Windows.Forms.TabPage();
            this.listBox1 = new System.Windows.Forms.ListBox();
            this.tabControl1.SuspendLayout();
            this.tabPage1.SuspendLayout();
            this.tabPage2.SuspendLayout();
```

```
this.tabPage3.SuspendLayout();
this.SuspendLayout();
//
// tabControl1
//
this.tabControl1.Controls.Add(this.tabPage1);
this.tabControl1.Controls.Add(this.tabPage2);
this.tabControl1.Controls.Add(this.tabPage3);
this.tabControl1.Dock = System.Windows.Forms.DockStyle.Fill;
this.tabControl1.Location = new System.Drawing.Point(0, 0);
this.tabControl1.Name = "tabControl1";
this.tabControl1.SelectedIndex = 0;
this.tabControl1.Size = new System.Drawing.Size(499, 470);
this.tabControl1.TabIndex = 0;
//
// tabPage1
//
this.tabPage1.Controls.Add(this.button3);
this.tabPage1.Controls.Add(this.button1);
this.tabPage1.Controls.Add(this.label3);
this.tabPage1.Controls.Add(this.textBox3);
this.tabPage1.Controls.Add(this.label2);
this.tabPage1.Controls.Add(this.textBox2);
this.tabPage1.Controls.Add(this.label1);
this.tabPage1.Controls.Add(this.textBox1);
this.tabPage1.Location = new System.Drawing.Point(4, 22);
this.tabPage1.Name = "tabPage1";
this.tabPage1.Padding = new System.Windows.Forms.Padding(3);
this.tabPage1.Size = new System.Drawing.Size(491, 444);
this.tabPage1.TabIndex = 0;
this.tabPage1.Text = "Новая процедура";
this.tabPage1.UseVisualStyleBackColor = true;
//
// button3
//
this.button3.Location = new System.Drawing.Point(114, 376);
this.button3.Name = "button3";
this.button3.Size = new System.Drawing.Size(265, 41);
this.button3.TabIndex = 7;
this.button3.Text = "Список процедур";
this.button3.UseVisualStyleBackColor = true;
this.button3.Click += new System.EventHandler(this.button3_Click);
//
// button1
//
this.button1.Location = new System.Drawing.Point(114, 317);
this.button1.Name = "button1";
this.button1.Size = new System.Drawing.Size(265, 41);
this.button1.TabIndex = 6;
this.button1.Text = "Добавить процедуру";
this.button1.UseVisualStyleBackColor = true;
this.button1.Click += new System.EventHandler(this.button1_Click);
//
```

```
// label3
//
this.label3.AutoSize = true;
this.label3.Location = new System.Drawing.Point(111, 250);
this.label3.Name = "label3";
this.label3.Size = new System.Drawing.Size(33, 13);
this.label3.TabIndex = 5;
this.label3.Text = "Цена";
//
// textBox3
//
this.textBox3.Location = new System.Drawing.Point(114, 266);
this.textBox3.Name = "textBox3";
this.textBox3.Size = new System.Drawing.Size(265, 20);
this.textBox3.TabIndex = 2;
//
// label2
//
this.label2.AutoSize = true;
this.label2.Location = new System.Drawing.Point(111, 62);
this.label2.Name = "label2";
this.label2.Size = new System.Drawing.Size(57, 13);
this.label2.TabIndex = 3;
this.label2.Text = "Описание";
//
// textBox2
//
this.textBox2.Location = new System.Drawing.Point(114, 78);
this.textBox2.Multiline = true;
this.textBox2.Name = "textBox2";
this.textBox2.Size = new System.Drawing.Size(265, 169);
this.textBox2.TabIndex = 1;
//
// label1
//
this.label1.AutoSize = true;
this.label1.Location = new System.Drawing.Point(111, 23);
this.label1.Name = "label1";
this.label1.Size = new System.Drawing.Size(83, 13);
this.label1.TabIndex = 1;
this.label1.Text = "Наименование";
//
// textBox1
//
this.textBox1.Location = new System.Drawing.Point(114, 39);
this.textBox1.Name = "textBox1";
this.textBox1.Size = new System.Drawing.Size(265, 20);
this.textBox1.TabIndex = 0;
//
// tabPage2
//
this.tabPage2.Controls.Add(this.button4);
this.tabPage2.Controls.Add(this.label7);
```

```
this.tabPage2.Controls.Add(this.textBox7);
this.tabPage2.Controls.Add(this.button2);
this.tabPage2.Controls.Add(this.label4);
this.tabPage2.Controls.Add(this.textBox4);
this.tabPage2.Controls.Add(this.label5);
this.tabPage2.Controls.Add(this.textBox5);
this.tabPage2.Controls.Add(this.label6);
this.tabPage2.Controls.Add(this.textBox6);
this.tabPage2.Location = new System.Drawing.Point(4, 22);
this.tabPage2.Name = "tabPage2";
this.tabPage2.Padding = new System.Windows.Forms.Padding(3);
this.tabPage2.Size = new System.Drawing.Size(491, 444);
this.tabPage2.TabIndex = 1;
this.tabPage2.Text = "Новый мастер";
this.tabPage2.UseVisualStyleBackColor = true;
//
// button4
//
this.button4.Location = new System.Drawing.Point(114, 381);
this.button4.Name = "button4";
this.button4.Size = new System.Drawing.Size(265, 41);
this.button4.TabIndex = 16;
this.button4.Text = "Список мастеров";
this.button4.UseVisualStyleBackColor = true;
this.button4.Click += new System.EventHandler(this.button4_Click);
//
// label7
//
this.label7.AutoSize = true;
this.label7.Location = new System.Drawing.Point(111, 254);
this.label7.Name = "label7";
this.label7.Size = new System.Drawing.Size(48, 13);
this.label7.TabIndex = 15;
this.label7.Text = "Рейтинг";
//
// textBox7
//
this.textBox7.Location = new System.Drawing.Point(114, 270);
this.textBox7.Name = "textBox7";
this.textBox7.Size = new System.Drawing.Size(265, 20);
this.textBox7.TabIndex = 3;
//
// button2
//
this.button2.Location = new System.Drawing.Point(114, 323);
this.button2.Name = "button2";
this.button2.Size = new System.Drawing.Size(265, 41);
this.button2.TabIndex = 4;
this.button2.Text = "Добавить мастера";
this.button2.UseVisualStyleBackColor = true;
this.button2.Click += new System.EventHandler(this.button2_Click);
//
// label4
```

```
//
this.label4.AutoSize = true;
this.label4.Location = new System.Drawing.Point(111, 174);
this.label4.Name = "label4";
this.label4.Size = new System.Drawing.Size(52, 13);
this.label4.TabIndex = 12;
this.label4.Text = "Телефон";
//
// textBox4
//
this.textBox4.Location = new System.Drawing.Point(114, 45);
this.textBox4.Name = "textBox4";
this.textBox4.Size = new System.Drawing.Size(265, 20);
this.textBox4.TabIndex = 0;
this.label5.AutoSize = true;
this.label5.Location = new System.Drawing.Point(111, 98);
this.label5.Name = "label5";
this.label5.Size = new System.Drawing.Size(49, 13);
this.label5.TabIndex = 10;
this.label5.Text = "Возраст";
this.textBox5.Location = new System.Drawing.Point(114, 114);
this.textBox5.Name = "textBox5";
this.textBox5.Size = new System.Drawing.Size(265, 20);
this.textBox5.TabIndex = 1;
this.label6.AutoSize = true;
this.label6.Location = new System.Drawing.Point(111, 29);
this.label6.Name = "label6";
this.label6.Size = new System.Drawing.Size(29, 13);
this.label6.TabIndex = 8;
this.label6.Text = "Имя";
this.label6.Click += new System.EventHandler(this.label6_Click);
this.textBox6.Location = new System.Drawing.Point(114, 190);
this.textBox6.Name = "textBox6";
this.textBox6.Size = new System.Drawing.Size(265, 20);
this.textBox6.TabIndex = 2;
this.tabPage3.Controls.Add(this.listBox1);
this.tabPage3.Location = new System.Drawing.Point(4, 22);
this.tabPage3.Name = "tabPage3";
this.tabPage3.Padding = new System.Windows.Forms.Padding(3);
this.tabPage3.Size = new System.Drawing.Size(491, 444);
this.tabPage3.TabIndex = 2;
this.tabPage3.Text = "Список записей";
this.tabPage3.UseVisualStyleBackColor = true;
this.listBox1.Dock = System.Windows.Forms.DockStyle.Fill;
this.listBox1.FormattingEnabled = true;
this.listBox1.Location = new System.Drawing.Point(3, 3);
this.listBox1.Name = "listBox1";
this.listBox1.Size = new System.Drawing.Size(485, 438);
this.listBox1.TabIndex = 0;
this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
this.ClientSize = new System.Drawing.Size(499, 470);
this.Controls.Add(this.tabControl1);
```

```
        this.Name = "admin_view";
        this.Text = "Панель администратора";
        this.tabControl1.ResumeLayout(false);
        this.tabPage1.ResumeLayout(false);
        this.tabPage1.PerformLayout();
        this.tabPage2.ResumeLayout(false);
        this.tabPage2.PerformLayout();
        this.tabPage3.ResumeLayout(false);
        this.ResumeLayout(false);
    }
    #endregion
    private System.Windows.Forms.TabControl tabControl1;
    private System.Windows.Forms.TabPage tabPage1;
    private System.Windows.Forms.TabPage tabPage2;
    private System.Windows.Forms.TabPage tabPage3;
    private System.Windows.Forms.Button button1;
    private System.Windows.Forms.Label label3;
    private System.Windows.Forms.TextBox textBox3;
    private System.Windows.Forms.Label label2;
    private System.Windows.Forms.TextBox textBox2;
    private System.Windows.Forms.Label label1;
    private System.Windows.Forms.TextBox textBox1;
    private System.Windows.Forms.Button button2;
    private System.Windows.Forms.Label label4;
    private System.Windows.Forms.TextBox textBox4;
    private System.Windows.Forms.Label label5;
    private System.Windows.Forms.TextBox textBox5;
    private System.Windows.Forms.Label label6;
    private System.Windows.Forms.TextBox textBox6;
    private System.Windows.Forms.Label label7;
    private System.Windows.Forms.TextBox textBox7;
    private System.Windows.Forms.ListBox listBox1;
    private System.Windows.Forms.Button button3;
    private System.Windows.Forms.Button button4;
}
```

ДОДАТОК Ґ. Лістинг коду файлу cabinet.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace cosmetic_service
{
    public partial class cabinet : Form
    {
        fireHelper fire = new fireHelper();
        List<note> notes;
        user curr_user;
        public cabinet(user user)
        {
            InitializeComponent();
            curr_user = user;
            notes = fire.getAllNotes(curr_user);
            for (int i = 0; i < notes.Count; i++)
            {
                string itm = "Запись №" + notes[i].note_id + ". " +
                    notes[i].client.fullName + " — " + notes[i].procedure.name + " (" +
                    notes[i].master.name + "), " + notes[i].date;
                listBox1.Items.Add(itm);
            }
        }

        private void cabinet_Load(object sender, EventArgs e)
        {

        }
    }
}
```

ДОДАТОК Д. Лістинг коду файлу cabinet.resc

```
<xsd:schema id="root" xmlns="" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:msdata="urn:schemas-microsoft-com:xml-msdata">
  <xsd:import namespace="http://www.w3.org/XML/1998/namespace" />
  <xsd:element name="root" msdata:IsDataSet="true">
    <xsd:complexType>
      <xsd:choice maxOccurs="unbounded">
        <xsd:element name="metadata">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="value" type="xsd:string" minOccurs="0" />
            </xsd:sequence>
            <xsd:attribute name="name" use="required" type="xsd:string" />
            <xsd:attribute name="type" type="xsd:string" />
            <xsd:attribute name="mimetype" type="xsd:string" />
            <xsd:attribute ref="xml:space" />
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="assembly">
          <xsd:complexType>
            <xsd:attribute name="alias" type="xsd:string" />
            <xsd:attribute name="name" type="xsd:string" />
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="data">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="value" type="xsd:string" minOccurs="0" msdata:Ordinal="1" />
              <xsd:element name="comment" type="xsd:string" minOccurs="0" msdata:Ordinal="2" />
            </xsd:sequence>
            <xsd:attribute name="name" type="xsd:string" use="required" msdata:Ordinal="1" />
            <xsd:attribute name="type" type="xsd:string" msdata:Ordinal="3" />
            <xsd:attribute name="mimetype" type="xsd:string" msdata:Ordinal="4" />
            <xsd:attribute ref="xml:space" />
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="resheader">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="value" type="xsd:string" minOccurs="0" msdata:Ordinal="1" />
            </xsd:sequence>
            <xsd:attribute name="name" type="xsd:string" use="required" />
          </xsd:complexType>
        </xsd:element>
      </xsd:choice>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
<resheader name="resmimetype">
  <value>text/microsoft-resx</value>
</resheader>
<resheader name="version">
```

```
    <value>2.0</value>
  </resheader>
  <resheader name="reader">
    <value>System.Resources.ResXResourceReader, System.Windows.Forms, Version=4.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089</value>
  </resheader>
  <resheader name="writer">
    <value>System.Resources.ResXResourceWriter, System.Windows.Forms, Version=4.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089</value>
  </resheader>
</root>
```

ДОДАТОК Е. Лістинг коду файлу user_view.resc

```
<xsd:schema id="root" xmlns="" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:msdata="urn:schemas-microsoft-com:xml-msdata">
  <xsd:import namespace="http://www.w3.org/XML/1998/namespace" />
  <xsd:element name="root" msdata:IsDataSet="true">
    <xsd:complexType>
      <xsd:choice maxOccurs="unbounded">
        <xsd:element name="metadata">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="value" type="xsd:string" minOccurs="0" />
            </xsd:sequence>
            <xsd:attribute name="name" use="required" type="xsd:string" />
            <xsd:attribute name="type" type="xsd:string" />
            <xsd:attribute name="mimetype" type="xsd:string" />
            <xsd:attribute ref="xml:space" />
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="assembly">
          <xsd:complexType>
            <xsd:attribute name="alias" type="xsd:string" />
            <xsd:attribute name="name" type="xsd:string" />
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="data">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="value" type="xsd:string" minOccurs="0" msdata:Ordinal="1" />
              <xsd:element name="comment" type="xsd:string" minOccurs="0" msdata:Ordinal="2" />
            </xsd:sequence>
            <xsd:attribute name="name" type="xsd:string" use="required" msdata:Ordinal="1" />
            <xsd:attribute name="type" type="xsd:string" msdata:Ordinal="3" />
            <xsd:attribute name="mimetype" type="xsd:string" msdata:Ordinal="4" />
            <xsd:attribute ref="xml:space" />
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="resheader">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="value" type="xsd:string" minOccurs="0" msdata:Ordinal="1" />
            </xsd:sequence>
            <xsd:attribute name="name" type="xsd:string" use="required" />
          </xsd:complexType>
        </xsd:element>
      </xsd:choice>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
<resheader name="resmimetype">
  <value>text/microsoft-resx</value>
</resheader>
<resheader name="version">
```

```
  <value>2.0</value>
 </resheader>
 <resheader name="reader">
  <value>System.Resources.ResXResourceReader, System.Windows.Forms, Version=4.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089</value>
 </resheader>
 <resheader name="writer">
  <value>System.Resources.ResXResourceWriter, System.Windows.Forms, Version=4.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089</value>
 </resheader>
</root>
```

ДОДАТОК Є. Лістинг коду файлу user_view.Designer.cs

```
amespace cosmetic_service
{
   partial class user_view
   {
.IContainer components = null;
      protected override void Dispose(bool disposing)
      {
         if (disposing && (components != null))
         {
            components.Dispose();
         }
         base.Dispose(disposing);
      }

      #region Windows Form Designer generated code
      private void InitializeComponent()
      {
         this.listView1 = new System.Windows.Forms.ListView();
         this.button1 = new System.Windows.Forms.Button();
         this.button2 = new System.Windows.Forms.Button();
         this.label1 = new System.Windows.Forms.Label();
         this.SuspendLayout();
         //
         // listView1
         //
         this.listView1.HideSelection = false;
         this.listView1.Location = new System.Drawing.Point(0, 16);
         this.listView1.Name = "listView1";
         this.listView1.Size = new System.Drawing.Size(364, 434);
         this.listView1.TabIndex = 0;
         this.listView1.UseCompatibleStateImageBehavior = false;
         this.listView1.MouseDoubleClick += new System.Windows.Forms.MouseEventHandler(this.listView1_MouseDoubleClick);
         this.button1.Dock = System.Windows.Forms.DockStyle.Bottom;
         this.button1.Location = new System.Drawing.Point(0, 413);
         this.button1.Name = "button1";
         this.button1.Size = new System.Drawing.Size(364, 37);
         this.button1.TabIndex = 1;
         this.button1.Text = "Мои записи";
         this.button1.UseVisualStyleBackColor = true;
         this.button1.Click += new System.EventHandler(this.button2_Click);
         this.button2.Dock = System.Windows.Forms.DockStyle.Bottom;
         this.button2.Location = new System.Drawing.Point(0, 376);
         this.button2.Name = "button2";
         this.button2.Size = new System.Drawing.Size(364, 37);
         this.button2.TabIndex = 2;
         this.button2.Text = "Записаться на процедуру";
         this.button2.UseVisualStyleBackColor = true;
         this.button2.Click += new System.EventHandler(this.button1_Click);
         this.label1.AutoSize = true;
         this.label1.Dock = System.Windows.Forms.DockStyle.Top;
```

```
            this.label1.Location = new System.Drawing.Point(0, 0);
            this.label1.Name = "label1";
            this.label1.Size = new System.Drawing.Size(219, 13);
            this.label1.TabIndex = 3;
            this.label1.Text = "Процедуры (двойной клик — подробности)";
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.ClientSize = new System.Drawing.Size(364, 450);
            this.Controls.Add(this.label1);
            this.Controls.Add(this.button2);
            this.Controls.Add(this.button1);
            this.Controls.Add(this.listView1);
            this.Name = "user_view";
            this.Text = "Услуги";
            this.ResumeLayout(false);
            this.PerformLayout();
        }
        #endregion
        private System.Windows.Forms.ListView listView1;
        private System.Windows.Forms.Button button1;
        private System.Windows.Forms.Button button2;
        private System.Windows.Forms.Label label1;
    }
}
```

ДОДАТОК Ж. Лістинг коду файлу worker_list.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace cosmetic_service
{
    public partial class workers_list : Form
    {
        fireHelper fire = new fireHelper();
        List<worker> workers;
        public workers_list()
        {
            InitializeComponent();
            workers = fire.getAllWorkers();
            for (int i = 0; i < workers.Count; i++)
            {
                string[] splited_name = workers[i].name.Split(' ');
                string short_name = splited_name[2] + " " + splited_name[0].ToCharArray()[0] + ". " + splited_name[0].ToCharArray()[0] + ". ";
                listView1.Items.Add(short_name);
            }
        }

        private void listView1_MouseDoubleClick(object sender, MouseEventArgs e)
        {
            int selected_int = listView1.SelectedItems[0].Index;
            MessageBox.Show("Возраст: " + workers[selected_int].age
                + "\nРейтинг: " + workers[selected_int].rate + "/5",
                workers[selected_int].name);
        }
    }
}
```