

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
Національний університет «Запорізька політехніка»

Інститут інформатики та радіоелектроніки  
Факультет комп'ютерних наук та технологій  
(повне найменування інституту, факультету)

Кафедра комп'ютерних систем та мереж  
(повне найменування кафедри)

**Пояснювальна записка**

до дипломного проекту (роботи)

бакалавра

(ступінь вищої освіти (освітній ступінь))

на тему РОЗРОБКА КОМП'ЮТЕРНОЇ СИСТЕМИ КОСМЕТОЛОГІЧНОЇ  
КЛІНІКИ

Виконав: студент 4 курсу, групи КНТ-527  
спеціальності \_\_\_\_\_

123 «Комп'ютерна інженерія»

(код і найменування спеціальності)

Освітня програма (спеціалізація) \_\_\_\_\_

«Комп'ютерна інженерія»

Бухер Ольга Русланівна

(прізвище та ініціали)

Керівник Кудерметов Р.К.

(прізвище та ініціали)

Рецензент Куляба-Харитоновна Т.І.

(прізвище та ініціали)

м. Запоріжжя  
2021 рік

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
**Національний університет «Запорізька політехніка»**  
 (повне найменування вищого навчального закладу)

Інститут, факультет інформатики та радіоелектроніки, комп'ютерних наук і технологій  
 Кафедра «Комп'ютерні системи та мережі»  
 Ступінь вищої освіти (освітній ступінь) бакалаврський  
 Спеціальність 123 Комп'ютерна інженерія  
 (код і найменування)  
 Освітня програма (спеціалізація) Комп'ютерна інженерія  
 (назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри Кудерметов Р.К.

*Р.К. Кудерметов*  
 “ ” 2021 року

**ЗАВДАННЯ**  
**НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) СТУДЕНТА**

Бухер Ольги Русланівни

(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) Розробка комп'ютерної системи косметологічної клініки

керівник проекту (роботи) Кудерметов Равіль Камілович, к. т. н., доцент

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

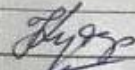
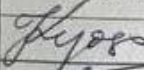
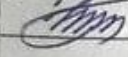
затверджені наказом вищого навчального закладу від “17” березня 2021 року № 81

2. Строк подання студентом проекту (роботи) 15 травня 2021 року3. Вихідні дані до проекту (роботи) опис предметної області (бриф-анкета, бізнес-процеси, інше)4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1) Аналіз технічного завдання;2) Моделювання програмного забезпечення;3) Проектування програмного забезпечення;4) Розробка програмного забезпечення.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

Плакат 1 - Діаграма прецедентів;Плакат 2 - ER-модель бази даних;Плакат 3 - Макети головної сторінки та форм, архітектура системи;Плакат 4 - Головні форми системи.

# 6. Консультанти розділів проекту (роботи)

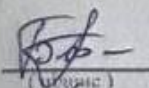
| Розділ        | Прізвище, ініціали та посада консультанта | Підпис, дата                                                                        |                                                                                     |
|---------------|-------------------------------------------|-------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|
|               |                                           | завдання видав                                                                      | прийняв виконавець завдання                                                         |
| 1-4           | Кудерметов Р.К., к.т.н., доцент           |  |  |
| нормоконтроль | Щербак Н.В., ст. викл.                    |  |  |
|               |                                           |                                                                                     |                                                                                     |
|               |                                           |                                                                                     |                                                                                     |
|               |                                           |                                                                                     |                                                                                     |
|               |                                           |                                                                                     |                                                                                     |
|               |                                           |                                                                                     |                                                                                     |

7. Дата видачі завдання 01.02.2021 р.

## КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів дипломного проекту (роботи)                              | Строк виконання етапів проекту (роботи) | Примітка |
|-------|-----------------------------------------------------------------------|-----------------------------------------|----------|
| 1     | Аналіз предметної області                                             | 05.02.2021 р.                           |          |
| 2     | Аналіз предметної області, розробка вимог до програмного забезпечення | 10.02.2021 р.                           |          |
| 3     | Проектування бази даних                                               | 01.03.2021 р.                           |          |
| 4     | Проектування інтерфейсу користувача                                   | 15.03.2021 р.                           |          |
| 5     | Аналіз та вибір програмних засобів                                    | 25.03.2021 р.                           |          |
| 6     | Розробка бази даних                                                   | 31.03.2021 р.                           |          |
| 7     | Розробка інтерфейсу користувача                                       | 30.04.2021 р.                           |          |
| 8     | Оформлення пояснювальної записки                                      | 5.05.2021 р.                            |          |
| 9     | Оформлення допоміжного та графічного матеріалу                        | 15.05.2021 р.                           |          |
|       |                                                                       |                                         |          |
|       |                                                                       |                                         |          |
|       |                                                                       |                                         |          |
|       |                                                                       |                                         |          |
|       |                                                                       |                                         |          |
|       |                                                                       |                                         |          |
|       |                                                                       |                                         |          |

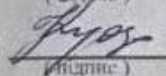
Студент

  
(ініціали)

О.Р. Бухар

(ініціали та прізвище)

Керівник проекту (роботи)

  
(ініціали)

Р.К. Кудерметов

(ініціали та прізвище)

## РЕФЕРАТ

ПЗ: 79 с., 14 рис., 16 табл., 2 додатки, 23 джерел.

BOOTSTRAP, MYSQL, PHP, БЕЗПЕКА, ІНТЕРФЕЙС КОРИСТУВАЧА,  
КЛІЄНТ-СЕРВЕР, КОМП'ЮТЕРНА СИСТЕМА, МОДЕЛЮВАННЯ,  
ОНЛАЙН ЗАПИС НА ПРИЙОМ, РОЗРОБКА, ХЕШ-ФУНКЦІЯ

Об'єкт розробки – комп'ютерна система косметологічної клініки.

Метою дипломної роботи є розширення можливостей діяльності косметологічної клініки, покращення іміджу клініки серед конкурентів, підвищення популярності комп'ютерної системи та залучення більшої кількості потенційних клієнтів за допомогою модуля для запису на прийом до спеціаліста.

Дипломна робота складається з чотирьох розділів.

Перший розділ присвячено предметному опису косметологічної клініки, проводиться порівняльний аналіз аналогів, визначаються вимоги до системи.

Другий розділ присвячено моделюванню програмного забезпечення, виділенню основних акторів, сценаріїв та діаграм послідовностей.

Третій розділ присвячено проєктуванню програмного забезпечення. Обрано тип архітектури для системи, проведено вибір системи керування базами даних, отримано ER-модель бази даних та розроблені макети сторінок інтерфейсу користувача.

У четвертому розділі проведено розробку, тестування бази даних та інтерфейсу користувача.

В результаті виконання дипломної роботи розроблено комп'ютерну систему косметологічної клініки з можливістю онлайн запису на прийом.

## ЗМІСТ

|                                                                            |    |
|----------------------------------------------------------------------------|----|
| Вступ.....                                                                 | 6  |
| 1 Аналіз технічного завдання.....                                          | 8  |
| 1.1 Опис предметної області .....                                          | 8  |
| 1.2 Розробка вимог до програмного забезпечення .....                       | 10 |
| 2 Моделювання програмного забезпечення .....                               | 15 |
| 3 Проєктування програмного забезпечення .....                              | 27 |
| 3.1 Архітектура програмного забезпечення .....                             | 27 |
| 3.2 Проєктування бази даних .....                                          | 29 |
| 3.2.1 Концептуальне проєктування бази даних.....                           | 29 |
| 3.2.2 Обґрунтування і вибір СКБД .....                                     | 35 |
| 3.3 Проєктування інтерфейсу користувача.....                               | 39 |
| 3.3.1 Проєктування інтерфейсу користувача .....                            | 40 |
| 3.3.2 Аналіз та вибір технологій для розробки інтерфейсу користувача ..... | 44 |
| 4 Розробка програмного забезпечення.....                                   | 52 |
| 4.1 Розробка бази даних.....                                               | 52 |
| 4.2 Розробка інтерфейсу користувача .....                                  | 60 |
| 4.3 Тестування системи .....                                               | 67 |
| Висновки .....                                                             | 69 |
| Перелік джерел посилання .....                                             | 71 |
| Додаток А .....                                                            | 74 |
| Додаток Б.....                                                             | 75 |

## ВСТУП

З появою приватних клінік в Україні суттєво розширився спектр надання послуг та покращилось обслуговування пацієнтів. Це пов'язано з більш досконалою діагностикою і ефективним лікуванням за рахунок сучасного оснащення клінік та удосконалених технологій. Така система обслуговування здатна задовольнити вимоги різних верств населення і забезпечити здорове та якісне життя клієнтам.

Кожного року люди все більше звертаються до приватних клінік. Успіх компаній залежить від вдалої реклами, відгуків клієнтів, цінової політики та правильного подання інформації про сфери послуг. Для сучасних клінік в зв'язку з розвитком технологій найвигідніше використовувати комп'ютерні системи.

Комп'ютерна система - це взаємопов'язана сукупність засобів, методів і інформації, які використовуються для зберігання, обробки та виведення інформації в інтересах досягнення поставленої мети, в залежності від предметної області [1].

За сферою застосування розрізняють комп'ютерні системи в промисловості, в економіці, в торгівлі, на транспорті, в медицині, в правовій сфері, в навчальних закладах і т.д.

Оскільки комп'ютерні системи створюються для задоволення інформаційних потреб в рамках конкретної предметної області, то кожній предметній області (сфері застосування) відповідає свій тип комп'ютерної системи.

Медична комп'ютерна система - комп'ютерна система, призначена для використання в лікувальному або лікувально-профілактичному закладі [1].

Комп'ютерні системи значно спрощують та прискорюють процеси роботи, дозволяють швидко обробляти інформацію з метою забезпечення інформаційних потреб користувача, підтримувати цілісність та збереження інформації та розвантажувати діяльності людини.

Отже, для вирішення цієї проблеми розроблюється медична комп'ютерна система косметологічної клініки. Також система забезпечить автоматизацію реєстрації клієнта на прийом до спеціаліста.

## 1 АНАЛІЗ ТЕХНІЧНОГО ЗАВДАННЯ

### 1.1 Опис предметної області

Кожна приватна клініка надає різноманітний спектр послуг для клієнтів, але ці послуги можуть надаватися в декількох клініках (конкурентах). Клініка мусить виділятися на тлі конкурентів, щоб залучити якомога більше потенційних клієнтів. Це призводить до того, що ми бачимо багато різноманітної реклами. Потенційний клієнт в більшості випадків чітко знає, що саме йому потрібно і що очікує від клініки та наданих послуг. Отже, клієнт бажає швидко отримати необхідну для нього інформацію про клініку:

- де знаходиться потрібна клініка;
- які послуги надаються клінікою;
- якого лікаря обрати;
- на яку дату та час можна записатися;
- які відгуки про клініку;
- останні новини про клініку.

Для сучасних клінік в зв'язку з розвитком технологій та технологій мобільного зв'язку для зберігання, обробки та видачі інформації в інтересах досягнення поставленої мети створюються комп'ютерні системи. Саме такі системи створюють перше враження про приватну клініку, коли клієнт переходить за посиланням.

По суті ж комп'ютерна система — це віддзеркалення успішності фірми, її обличчя. Комп'ютерна система виконує такі основні завдання [2]:

- збір і реєстрація інформаційних ресурсів;
- зберігання інформаційних ресурсів;
- актуалізація інформаційних ресурсів, для того щоб ця модель була практично корисною, необхідно своєчасно і адекватно відображати в ній зміни стану предметної області;
- обробка інформаційних ресурсів;

- надання інформаційних ресурсів користувачам.

Для косметологічної клініки основною метою системи є надання швидкого доступу на посилання клініки, надання інформації про клініку, послуги, спеціалістів, останні новини та оновлення клініки. Також система повинна мати розділ коментарів, де кожен може залишити свій коментар, подяку та, навіть, негативний відгук про обслуговування або персонал.

Основними користувачами системи являються: звичайні відвідувачі системи, клієнти клініки, спеціалісти клініки та адміністратор клініки.

Для виділення системи косметологічної клініки з поміж конкурентів було вирішено розширити можливості системи відповідно до користувачів системи. Для клієнтів додати можливість запису на прийом та можливість реєстрації до особистого кабінету, де клієнт зможе переглядати особисту інформацію. Для спеціалістів та адміністратора клініки розширення можливостей за рахунок особистого кабінету та особливих функцій для кожного з них.

Структура сайту повинна бути в першу чергу зручна. Тести проведені психологами стверджують, що людина, коли вперше потрапляє на головну сторінку сайту, приймає рішення близько 8 секунд.

Всі розділи системи та інформація повинні бути легко доступні, а навігація ергономічна. Розміщення корисної інформації на сторінках системи дозволяє збільшувати частоти відвідувань та створювати позитивний імідж приватних клінік.

При розробці сторінки комп'ютерної системи повинні бути доступні (і правильно відображатися) для максимально можливого числа користувачів та дисплеїв їх гаджетів. Всі сторінки повинні бути оформлені в одному стилі: використовувати не більше трьох основних шрифтів, включаючи шрифти, які застосовуються при створенні графічних об'єктів. Використовувати три основних кольори, можна використовувати відтінки. Текст не повинен напружувати зір, очі не повинні втомлюватися.

Отже, система повинна мати зрозумілий та приємний інтерфейс, зручну систему навігації. Забезпечувати потенційних клієнтів необхідною інформацією

про клініку, послуги та спеціалістів. Вся інформація повинна відображатися коректно на будь-якому дисплеї користувача.

Також система повинна мати зручну форму для реєстрація на прийом до спеціаліста. Дозволяти клієнтам переглядати власну інформацію з особистого кабінету та надавати доступ до особистого кабінету для спеціалістів та адміністраторів клініки.

## **1.2 Розробка вимог до програмного забезпечення**

Комп'ютерна система повинна спростити управління, доступ та перевірку інформації косметологічної клініки. Основні призначення для системи:

- надання інформації про клініку;
- автоматизація запису на прийом до спеціаліста клініки лазерної медицини та косметології;
- спрощення введення даних;
- підвищення якості (повноти, точності, достовірності, своєчасності, узгодженості) інформації;
- забезпечити збереження даних та інформації.

Простота управління і гнучкість системи повинна дозволяти використовувати її з самого моменту впровадження. Перспективи впровадження системи: дана система буде корисна для клініки, що піклується про своїх клієнтів, та економить затрати часу за рахунок автоматизації.

Для вирішення бізнес-задач з отриманням більшого прибутку та для задоволення потреб клієнта проведемо порівняльну характеристику систем клінік, які займають передові місця у сфері косметологічної медицини - «ПрофіМед» та «МОНЕ». Результати порівнянь надані в таблиці 1.1.

Таблиця 1.1 – Порівняльна характеристика сайтів клініки «ПрофіМед» та «МОНЕ»

| Критерії порівняння                     | «ПрофіМед»                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | «МОНЕ»                                                                                                                                                                                                                                                                                                                 |
|-----------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1                                       | 2                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    | 3                                                                                                                                                                                                                                                                                                                      |
| Алгоритм реєстрації особистого кабінету | <p><b>Крок 1.</b> Ввести email (який є логіном), пароль, підтвердження паролю.</p> <p><b>Крок 2.</b> Погодження на безстрокове зберігання медичних документів.</p> <p><b>Крок 3.</b> Підтвердження реєстрації.</p> <p><b>Переваги:</b> не потрібно вводити багато персональних даних.</p>                                                                                                                                                                                                                                                                            | <p><b>Крок 1.</b> Ввести прізвище, ім'я, по батькові, email, пароль.</p> <p><b>Крок 2.</b> Перевірка захисту «я не робот».</p> <p><b>Крок 3.</b> Підтвердження реєстрації.</p> <p><b>Недолік:</b> потрібно ввести багато персональних даних.</p>                                                                       |
| Алгоритм запису на прийом               | <p><b>Крок 1.</b> Вибрати спеціальність з переліку. Вибрати місто та філії.</p> <p><b>Крок 2.</b> В залежності від спеціальності обрати спеціаліста зі списку.</p> <p><b>Крок 3.</b> Обрати дату и час.</p> <p><b>Крок 4.</b> Заповнити анкету пацієнта, всі поля обов'язкові (прізвище, ім'я, по батькові, дата народження).</p> <p><b>Крок 5.</b> Заповнити контактні дані: мобільний телефон (обов'язково) та email (необов'язково).</p> <p><b>Крок 6.</b> Підтвердження введення.</p> <p><b>Недолік:</b> нема зворотного зв'язку про вдалий запис на прийом.</p> | <p><b>Крок 1.</b> Заповнити поля контактної форми. Вести ім'я, номер телефона, email, послуга, дата та час прийому (усі поля обов'язкові).</p> <p><b>Крок 2.</b> Погодження з політикою конфіденційності.</p> <p><b>Крок 3.</b> Відправити запит.</p> <p><b>Переваги:</b> повідомлення про вдалий запис на прийом.</p> |
| Вхід до особистого кабінету             | <p>Два варіанти входу.</p> <p><b>Варіант 1.</b> Введення логіну та паролю за умови попередньої реєстрації.</p> <p><b>Варіант 2.</b> Вхід через зареєстровані профіль користувача соціальних мереж.</p> <p><b>Переваги:</b> можливість швидкого доступу через профіль користувача соціальних мереж.</p>                                                                                                                                                                                                                                                               | <p>Введення логіну та паролю за умови попередньої реєстрації.</p> <p><b>Недоліки:</b> один варіант входу до особистого кабінету.</p>                                                                                                                                                                                   |
| Алгоритм відновлення паролю             | <p><b>Крок 1.</b> Ввести email.</p> <p><b>Крок 2.</b> Ввести код підтвердження.</p> <p><b>Крок 3.</b> Придумати новий пароль. Підтвердити зміну.</p>                                                                                                                                                                                                                                                                                                                                                                                                                 | <p><b>Крок 1.</b> Ввести логін.</p> <p><b>Крок 2.</b> Ввести новий пароль.</p> <p><b>Крок 3.</b> Підтвердити зміни.</p>                                                                                                                                                                                                |

Продовження таблиці 1.1

| 1                 | 2                                                                                                                                                                                                                                               | 3                                                                                                                                                |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| Особистий кабінет | Відображається профіль пацієнта, який можна редагувати, а також історія лікування в клініці. Присутня можливість завантаження файлів (листки призначень та рецепти, виписки, протоколи операцій, лабораторні дослідження, фінансові документи). | Відображається профіль пацієнта та послуга на яку він записаний.<br>Недолік: немає можливості скасувати запис на прийом через особистий кабінет. |
| Структура сайту   | Деревоподібна.                                                                                                                                                                                                                                  | Лінійна структура з відгалуженням.                                                                                                               |
| Навігація         | Глобальна (посилання, які видні з будь-якої сторінки сайту).                                                                                                                                                                                    | Глобальна (посилання, які видні з будь-якої сторінки сайту).                                                                                     |

Проаналізувавши сайти клініки «ПрофіМед» та «МОНЕ» було виявлено переваги та недоліки. В розроблювальній системі бажано відтворити переваги та не допустити виявлені недоліки.

З метою реалізації поставлених задач визначено наступні функціональні вимоги системи:

- відображення інформації про клініку та її послуги;
- відображення цін на послуги;
- відображення відгуків про клініку та спеціалістів;
- інтернет-сторінки повинні надавати повну можливість навігації і вибір послуг за допомогою миші, та використання миші і клавіатури при введенні даних.
- можливість запису на прийом;
- мінімізація можливих помилок клієнта при реєстрації на прийом до спеціаліста;
- при записі на прийом швидкий та коректний вибір даних клініки;
- неможливо зберігати декілька записів на прийом з повним співпаданням відповідностей послуга, спеціаліст, дата та час;
- реєстрація особистого кабінету для клієнтів;
- перевіряти відповідність логіну та паролю клієнта для входу до особистого кабінету;

- кроссплатформеність, система повинна правильно працювати на обладнанні незалежно від встановленої операційної системи сімейства Windows, GNU / Linux;
- адаптивність, система повинна коректно відображатися незалежно від дисплея користувача;
- кросбраузерність, повинна працювати з наступними інтернет-браузерами: google chrome, mozilla та іншими популярними браузерами;
- дані системи повинні зберігатися в базі даних.

До кваліфікації персоналу та клієнтів, що будуть використовувати системи особливих технічних знань не виявлено.

Для системи виділено наступні припущення і залежності:

- залежність від обладнання. Система працює правильно та без збоїв тільки за умови, що все необхідне для неї обладнання знаходиться в робочому стані;
- система правильно функціонує тільки за умови, що між її вузлами існує стабільний (не перерваний протягом сесії використання системи) зв'язок.

В результаті аналізу предметної області було виділено такі класи користувачів:

- відвідувач. Відвідувач - це користувач мережі інтернет. Всі клієнти мають доступ до Інтернету вдома або на робочих місцях, а також володіють стаціонарним або переносними пристроями доступу до інтернету. Відвідувач заходить на сайт на сайт з метою перегляду інформації про клініку, основну інформацію про спеціалістів та має можливість без спеціальних технічних знань записатися на прийом до спеціаліста і реєструватися до особистого кабінету;
- клієнт. Клієнт – це зареєстрований користувач системи. Він повинен мати доступ до особистого кабінету. Та може виконувати ті самі дії, що й звичайний відвідувач;
- спеціаліст клініки. Спеціаліст – це працівник клініки. Він має особливі права доступу при вході до особистого кабінету;

- адміністратор клініки. Адміністратор – це працівник клініки. Він має особливі права доступу при вході до особистого кабінету, але функції та права відрізняються від прав спеціалістів клініки.

Для кожного користувача системи виділено можливість авторизації до особистого кабінету, але з різними правами доступу (спеціаліст, адміністратор клініки мають особливі права доступу, як працівники клініки).

Функціональні вимоги системи для користувача (клієнт, зареєстрований користувач):

- відновлення паролю;
- відображення персональних записів на прийом;
- перегляд цін на послуги;
- можливість зворотного зв'язку;
- запис на прийом;
- скасування запису на прийом;
- можливість вибору дати та часу запису на прийом;
- можливість вибору спеціаліста при записі на прийом.

Функціональні вимоги системи для спеціалістів клініки:

- створення електронної довідки для клієнта;
- відображення графіку роботи.

Функціональні вимоги для адміністратора клініки:

- відображення інформації про записи клієнтів;
- можливість фільтрації даних за датою;
- можливість фільтрації даних за спеціалістом;
- можливість фільтрації даних за послугою.

Система повинна відповідати наступним вимогам до дизайну:

- мати зручний та інтуїтивний інтерфейс;
- сторінки повинні бути оформлені в одному стилі;
- зручна система навігації.

Отже, проаналізувавши предметну область, провівши порівняльну характеристику аналогів систем та після визначення основних класів користувачів

було виділено функціональні та не функціональні вимоги для комп'ютерної системи косметологічної клініки.

## **2 МОДЕЛЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ**

Діаграма прецедентів — діаграма, на якій відображення відношення між акторами та прецедентами в системі [3]. Іншими словами перекладається як діаграма варіантів використання.

Діаграма прецедентів представляє собою граф, що складається з множини акторів та прецедентів (варіантів використання) обмежених границею системи (представляє собою прямокутник), асоціацій між акторами та прецедентами, відношень серед прецедентів, та відношень узагальнення між акторами. Діаграми прецедентів відображають елементи моделі варіантів використання.

Суть даної діаграми полягає в наступному: проектована система представляється у вигляді безлічі сутностей чи акторів, що взаємодіють із системою за допомогою так званих варіантів використання. Варіант використання (англ. use case) використовують для описання послуг, які система надає актору. Іншими словами, кожен варіант використання визначає деякий набір дій, який виконує система при діалозі з актором. При цьому нічого не говориться про те, яким чином буде реалізована взаємодія акторів із системою.

У мові UML є декілька стандартних видів відношень між акторами та варіантами використання:

- асоціації (англ. association relationship);
- включення (англ. include relationship);
- розширення (англ. extend relationship);
- узагальнення (англ. generalization relationship).

Відношення асоціації — одне з фундаментальних понять у мові UML і в тій чи іншій мірі використовується при побудові всіх графічних моделей систем у формі канонічних діаграм.

Включення у мові UML — це різновид відношення залежності між базовим варіантом використання і його спеціальним випадком. При цьому відношенням залежності (англ. *dependency*) є таке відношення між двома елементами моделі, при якому зміна одного елемента (незалежного) приводить до зміни іншого елемента (залежного).

Відношення розширення (англ. *extend*) визначає взаємозв'язок базового варіанта використання з іншим варіантом використання, функціональна поведінка якого задіюється базовим не завжди, а тільки при виконанні додаткових умов [4].

Діаграма прецедентів, яка відображає процеси, пов'язані з роботою косметологічної клініки зображено на рисунку 2.1.

Розглянемо основні прецеденти для інформаційної системи косметологічної клініки «Estelife».

Прецедент П1. Авторизація до особистого кабінету.

Основний виконавець: користувач, а саме клієнт, спеціаліст клініки, адміністратор клініки.

Зацікавлені особи та їхні вимоги:

- користувач. Хоче авторизуватися в системі;
- система. Хоча ідентифікувати користувача і надати йому права доступу.

Результати (після умови): користувач успішно авторизований з відповідними правами доступу.

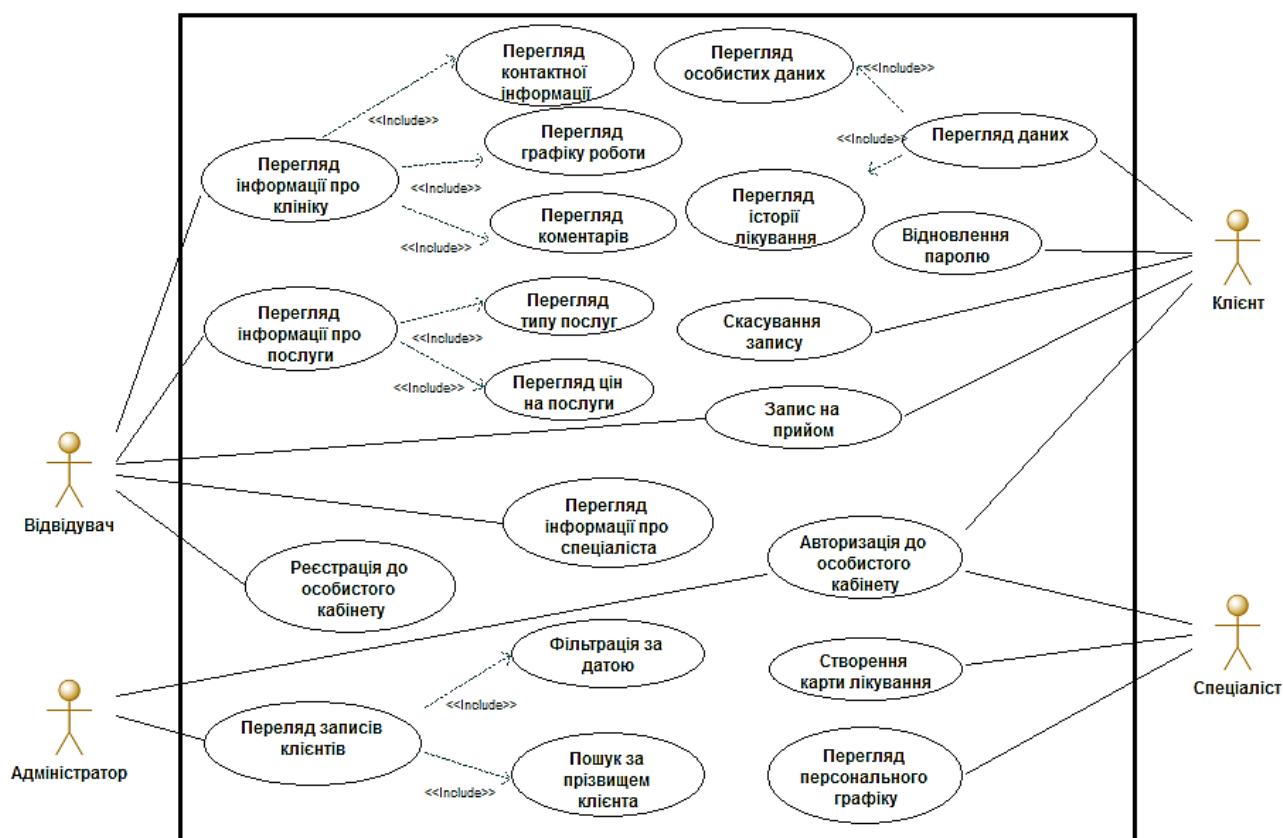


Рисунок 2.1 – Діаграма прецедентів, яка відображає процеси, пов'язані з клінікою

Основний успішний сценарій (або основний процес):

- користувач запускає форму авторизації шляхом натискання на кнопку «Авторизація» на головній сторінці системи.
- система відкриває форму для введення логіну і паролю;
- користувач вводить логін (обов'язкове поле);
- користувач вводить пароль (обов'язкове поле);
- користувач натискає кнопку підтвердження введення;
- форма відправляє дані до бази даних;
- база даних перевіряє дійсність логіну;
- якщо логін дійсний, то база даних порівнює вказаний клієнтом пароль з тим, що зберігається в базі даних.
- база даних визначає тип користувача (клієнт, спеціаліст, адміністратор);

- в разі успіху база даних передає повідомлення в систему про вірність даних;
- в разі успіху система повертає користувачеві повідомлення з приводу успішної авторизації (посилання на повідомлення);
- система відкриває сторінку особистого кабінету користувача;
- користувач потрапляє до власного кабінету.

Розширення (альтернативні потоки) наведено в таблиці 2.1.

Таблиця 2.1 – Альтернативні потоки для авторизації до особистого кабінету

|    |                                                                                                                                                                                               |
|----|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| *а | Немає доступу до БД. Система видає повідомлення (посилання на повідомлення). Результат: користувач не може увійти.                                                                            |
| *б | Помилка користувача при роботі з системою.                                                                                                                                                    |
| 1а | В налаштуваннях безпеки для даного ІР адреси існує заборона на вхід в систему. Результат: форма авторизації не надається, система видає повідомлення користувачу (посилання на повідомлення). |
| 3а | Користувач з введеними логіном і паролем не знайдено. Результат: відмова в авторизації. Система видає повідомлення (посилання на повідомлення). Перехід на крок 2.                            |
| 4а | Користувач з введеними логіном і паролем не знайдено. Результат: відмова в авторизації. Система видає повідомлення (посилання на повідомлення). Перехід на крок 2.                            |
| 4б | Користувач тричі ввів неправильно пароль. Результат: відмова в авторизації. Система відкриває форму для відновлення паролю.                                                                   |
| 5в | Збій при натисканні кнопки. Результат: дані не збережені в БД.                                                                                                                                |

Діаграма послідовності, яка відображає авторизацію до особистого кабінету наведено на рисунку 2.2.

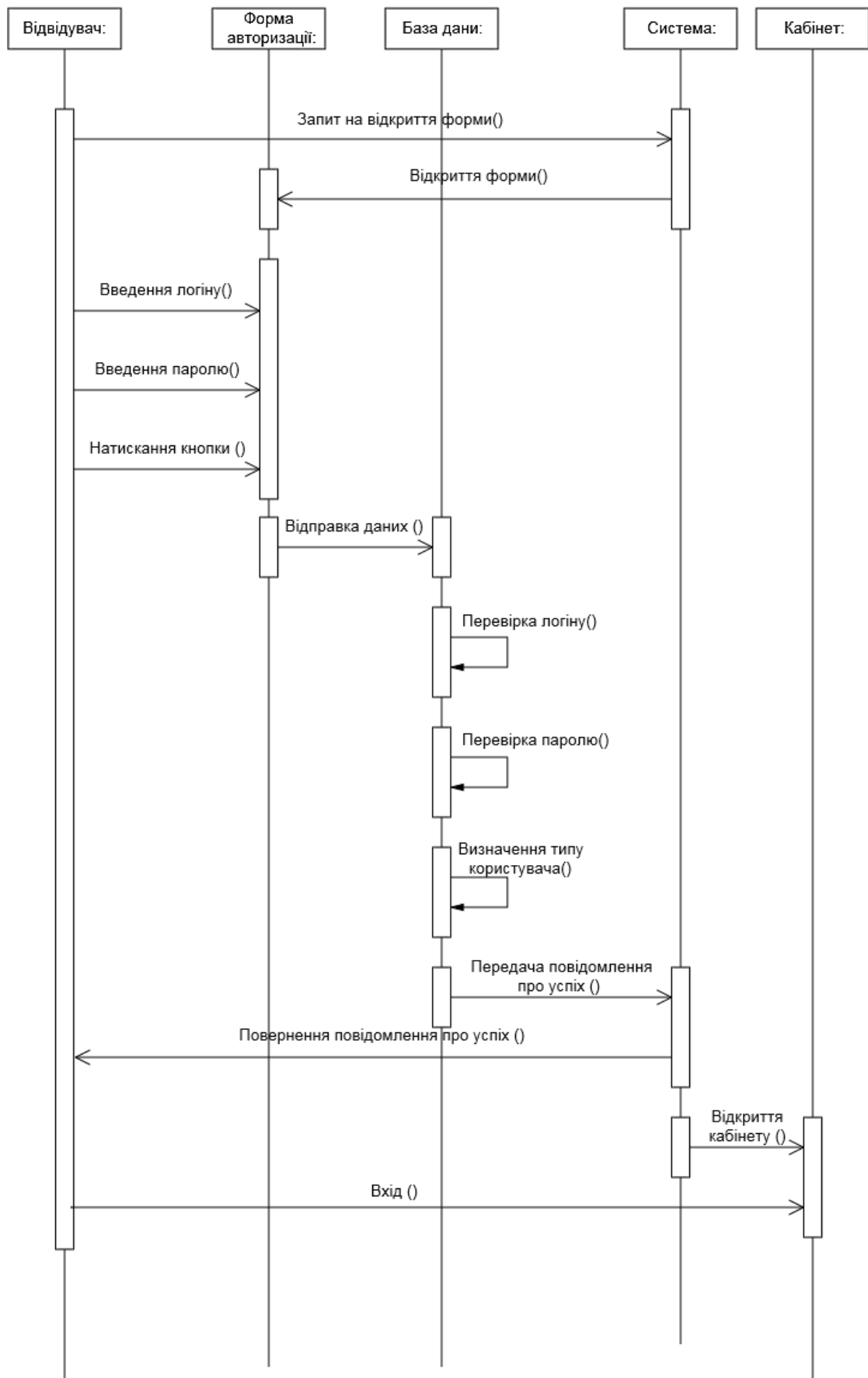


Рисунок 2.2 - Діаграма послідовності, яка відображає авторизацію до особистого кабінету

Прецедент П2. Реєстрація до особистого кабінету.

Основний виконавець: відвідувач (гість).

Зацікавлені особи та їхні вимоги:

- відвідувач: реєстрація в системі (створення власного кабінету);
- система: зареєструвати користувача і призначити права.

Результати (після умови): користувач успішно зареєстрований і може працювати з системою.

Основний успішний сценарій (або основний процес) для реєстрації до особистого кабінету:

- користувач відкриває форму для реєстрації шляхом натискання кнопки «Реєстрація»;
- система відкриває форму для реєстрації до особистого кабінету;
- користувач вводить логін (який є email, поле обов'язкове);
- користувач вводить пароль (поле обов'язкове);
- користувач вводить підтвердження пароля (поле обов'язкове);
- користувач натискає кнопку «Зареєструватися»;
- форма відправляє дані до бази даних;
- база даних перевіряє коректність логіну;
- база даних перевіряє коректність паролю;
- запис ведених логіну та паролю;
- якщо дані записано, то база даних перевіряє на збіг ведені дані в полях пароль та підтвердження паролю;
- в разі успіху база даних передає повідомлення в систему про вірність даних;
- в разі успіху система повертає користувачеві повідомлення з приводу успішної реєстрації (посилання на повідомлення);
- система відкриває сторінку особистого кабінету користувача;
- користувач потрапляє до власного кабінету.

Розширення (альтернативні потоки) наведено в таблиці 2.2.

Таблиця 2.2 – Альтернативні потоки для реєстрації до особистого кабінету

|    |                                                                                                                    |
|----|--------------------------------------------------------------------------------------------------------------------|
| *а | Немає доступу до БД. Система видає повідомлення (посилання на повідомлення). Результат: користувач не може увійти. |
| *б | Помила користувача при роботі з системою.                                                                          |
| 3б | Введений логін вже існує. Результат: відмова в реєстрації.                                                         |
| 5б | Поля пароллю не співпадають. Результат: буде світитися неправильне поле. Реєстрацію не буде виконано;              |
| 3в | Клієнт не заповнив поле логіну. Результат: запит не буде відправлено, буде світитися незаповнене поле.             |
| 4в | Клієнт не заповнив поле пароллю. Результат: запит не буде відправлено, буде світитися незаповнене поле.            |

Діаграма послідовності, яка відображає реєстрацію до особистого кабінету наведено на рисунку 2.3.

Прецедент ПЗ. Запис на прийом.

Основний виконавець: клієнт, відвідувач.

Зацікавлені особи та їхні вимоги:

- адміністратор. Хоче переглядати інформацію про записаних на прийом клієнтів;
- відвідувач. Хоче ознайомитися з послугами компанії і оформити запис на прийом;
- клієнт. Хоче оформити запис на прийом;
- компанія. Хоче записувати дані про клієнтів і задовольняти інтереси клієнтів. Хоче бути упевнена, що служба реєстрації на прийом зафіксувала необхідні дані клієнта. Зацікавлена в забезпеченні стійкості до збоїв.

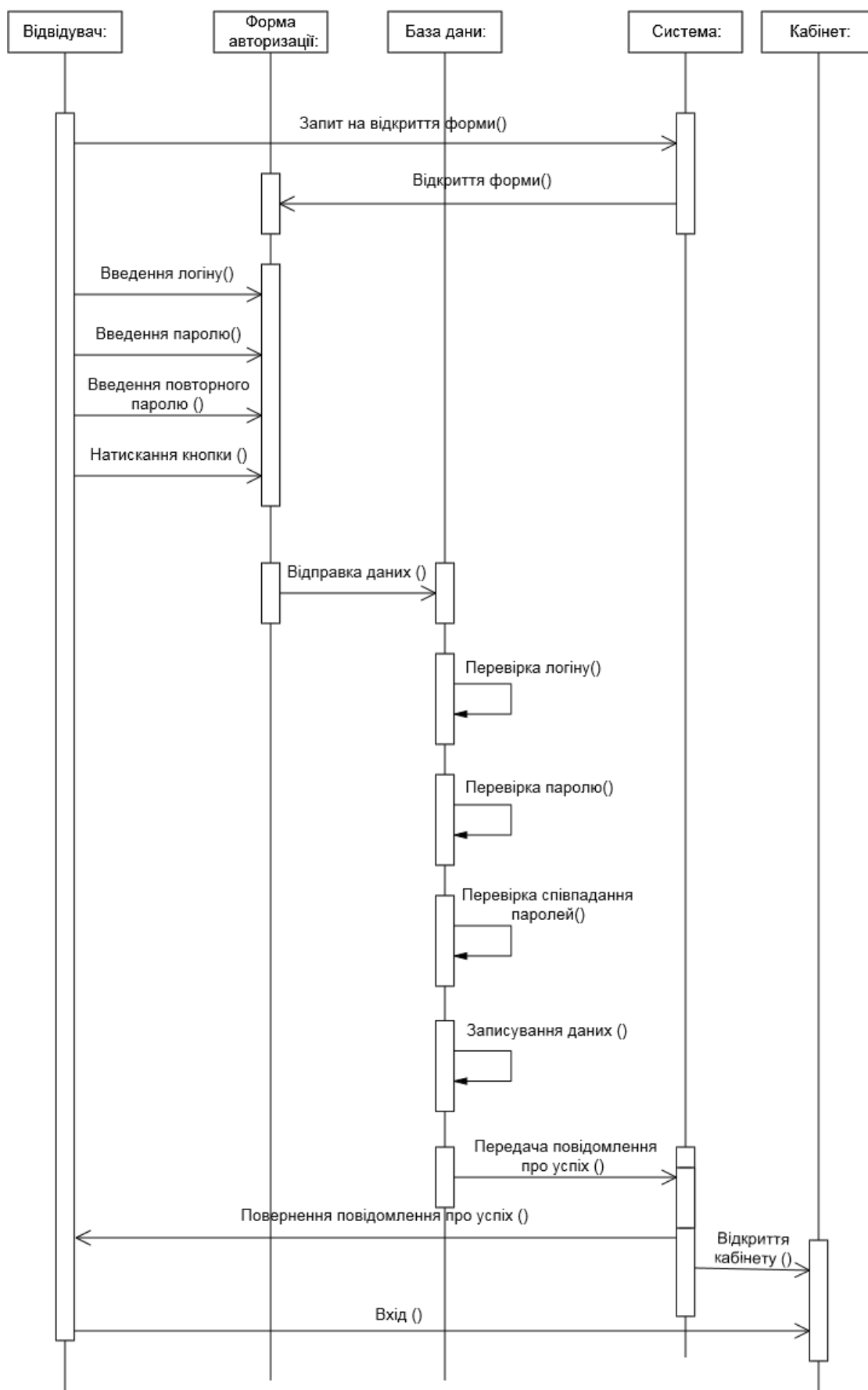


Рисунок 2.3 - Діаграма послідовності, яка відображає реєстрацію до особистого кабінету

Передумови: клієнт ідентифікований та автентифікований, а так само і не ідентифікований, автентифікований.

Результати (після умови):

- дані про записи на прийом збережені;
- запис внесено в базу;
- дані оновлені.

Основний успішний сценарій (або основний процес):

- користувач переходить на форму запису на прийом шляхом натискання кнопки «Записатися на прийом»;
- система відкриває форму для запису на прийом;
- вводить прізвище (поле обов'язкове);
- вводить ім'я (поле обов'язкове);
- вводить по батькові (поле обов'язкове);
- вводить номер телефону (поле обов'язкове, записується в заданому форматі);
- форма показує список доступних відділі клініки отриманих з бази даних;
- користувач з поля зі списком вибирає відділ;
- форма відправляє запит до бази даних для відображення (назви) послуг в залежності від обраного відділу;
- база даних повертає результат запиту у вигляді списку доступних послуг в залежності від обраного відділу;
- в залежності від обраного відділу з поля зі списком користувач обирає послугу;
- форма відправляє запит до бази даних для відображення прізвищ спеціалістів, які займаються вибраною послугою;
- база даних повертає результат запиту у вигляді списку доступних спеціалістів в залежності від обраної послуги;
- в залежності від обраної послуги з поля зі списком користувач обирає прізвище спеціаліста;

- форма відправляє запит до бази даних для відображення можливих дат запису на прийом до обраного спеціаліста, на обрану послугу;
- база даних повертає результат запиту у вигляді списку доступних дат запису в залежності від попередньо обраних варіантів послуги та спеціаліста;
- в залежності від обраного спеціаліста з поля зі користувач обирає дату запису на прийом;
- форма відправляє запит до бази даних для відображення часу запису на прийом до обраного спеціаліста, на обрану дату;
- база даних повертає результат запиту у вигляді списку доступного часу для запису в залежності від попередньо обраних варіантів послуги, спеціаліста та дати;
- користувач обирає вільний час для запису на прийом до обраного спеціаліста на обрану дату;
- натискає кнопку «відправити»;
- форма записує введені дані;
- в разі успіху база даних передає повідомлення в систему про успішність запису на прийом;
- в разі успіху система видає користувачеві повідомлення з приводу успішного запису на прийом (посилання на повідомлення);
- клієнт залишає форму запису.

Розширення (альтернативні потоки) наведено в таблиці 2.3.

Таблиця 2.3 – Альтернативні потоки для запису клієнта на прийом

|     |                                                                                                                                        |
|-----|----------------------------------------------------------------------------------------------------------------------------------------|
| *а  | Немає доступу до БД. Система видає повідомлення (посилання на повідомлення). Результат: користувач не може увійти.                     |
| *б  | Помила користувача при роботі з формою.                                                                                                |
| 2б  | Клієнт не ввів прізвище. Результат: запит не буде відправлено, буде світитися незаповнене поле.                                        |
| 3б  | Клієнт не ввів ім'я. Результат: запит не буде відправлено, буде світитися незаповнене поле.                                            |
| 4б  | Клієнт не ввів по батькові. Результат: запит не буде відправлено, буде світитися незаповнене поле.                                     |
| 5б  | Телефон введено не в правильному форматі або поле не заповнено. Результат: запит не буде відправлено. Буде світитися незаповнене поле. |
| 8в  | Варіанти типів послуг не відображаються. Запит не було відправлено до БД.                                                              |
| 11в | Варіанти послуг не відображаються. Запит не було відправлено до БД.                                                                    |
| 14в | Варіанти прізвищ спеціалістів не відображаються. Запит не було відправлено до БД.                                                      |
| 17в | Варіанти дати запису на прийом не відображаються. Запит не було відправлено до БД.                                                     |
| 20в | Варіанти часу запису на прийом не відображаються. Запит не було відправлено до БД.                                                     |
| 11б | Варіанти послуг не відображаються. Користувач не обрав попередньо тип послуги.                                                         |
| 14б | Варіанти прізвищ спеціалістів не відображаються. Користувач не обрав попередньо послугу.                                               |
| 17б | Варіанти дати запису на прийом не відображаються. Користувач не обрав попередньо прізвище спеціаліста.                                 |
| 20б | Варіанти часу запису на прийом не відображаються. Користувач не обрав попередньо дату запису на прийом.                                |

Діаграма послідовності, яка відображає запис клієнта на прийом до спеціаліста наведено на рисунку 2.4.

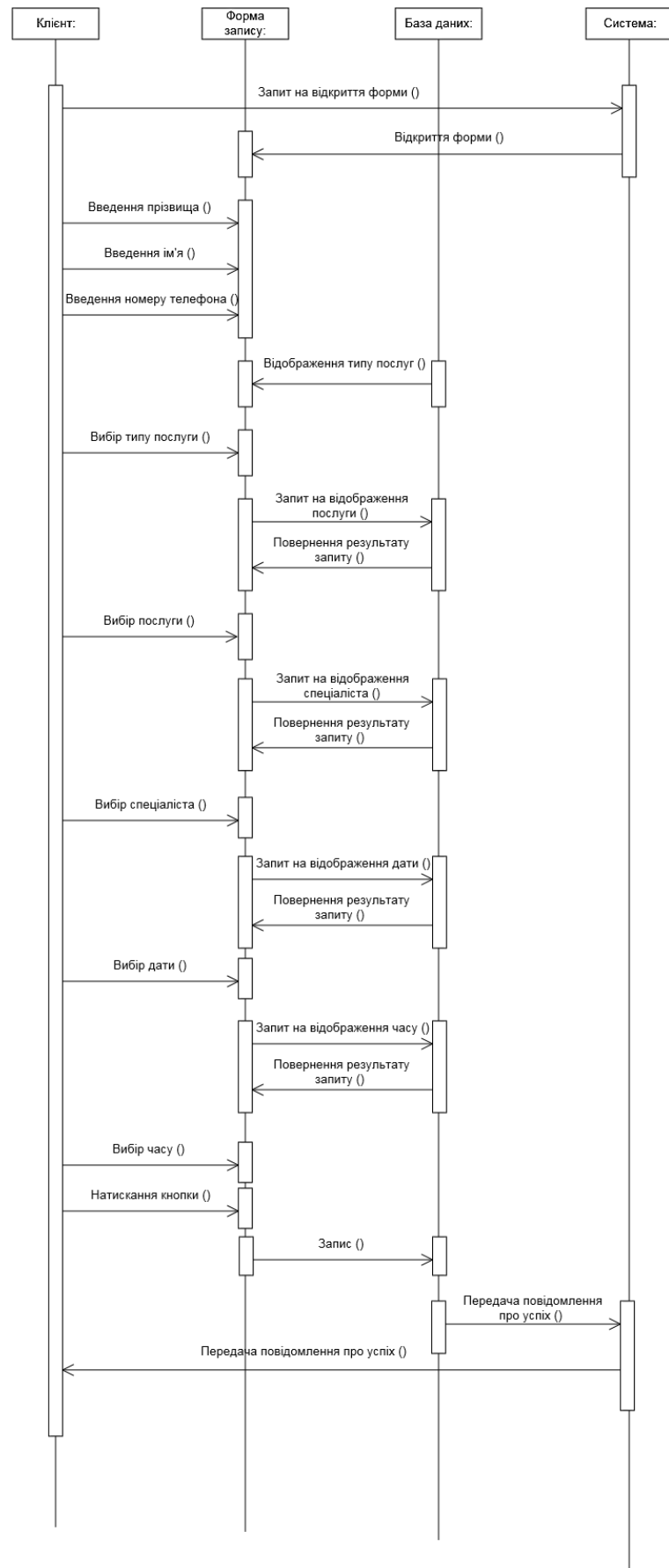


Рисунок 2.4 - Діаграма послідовності, яка відображає запис клієнта на прийом до спеціаліста

## 3 ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

### 3.1 Архітектура програмного забезпечення

Архітектура програмного забезпечення — це структура програми або обчислювальної системи, яка включає програмні компоненти, видимі зовні властивості цих компонентів, а також відносини між ними [5].

Архітектура клієнт-сервер є одним із архітектурних шаблонів програмного забезпечення та є домінуючою концепцією у створенні розподілених мережних застосувань, передбачає взаємодію та обмін даними між ними.

До технологій та протоколів побудованих навколо моделі «клієнт-сервер» відносяться:

- протокол передачі гіпертексту (HTTP);
- система доменних імен (DNS);
- простий протокол передачі пошти (SMTP);
- Telnet.

Структурна схема архітектури клієнт-сервер наведено на рисунку 3.1.

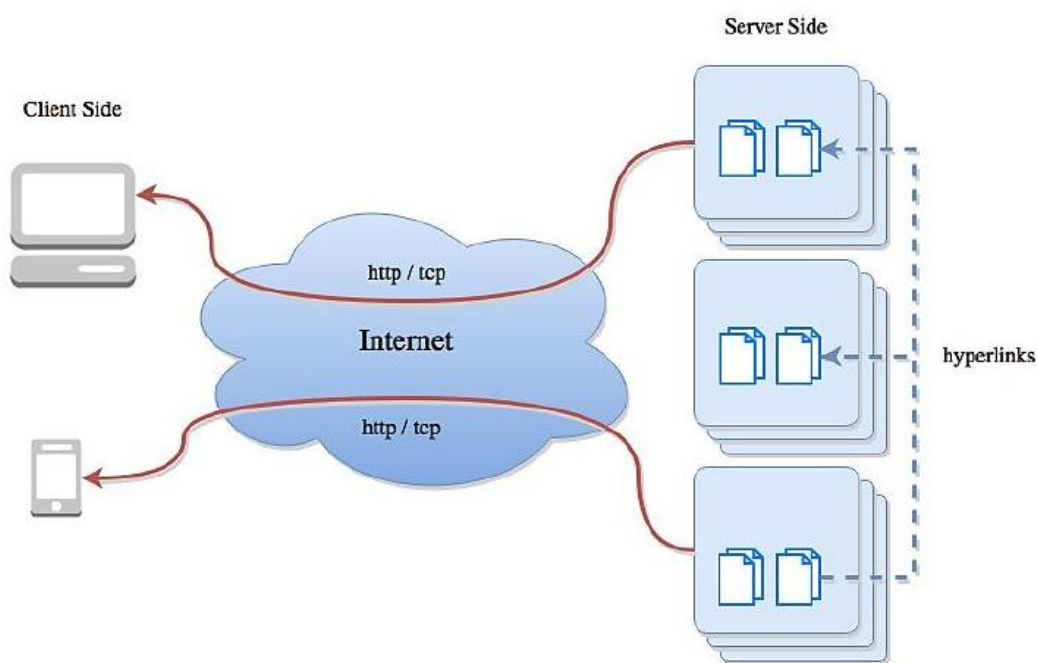


Рисунок 3.1 – Архітектура «Клієнт-сервер»

Архітектура «Клієнт-сервер» передбачає такі основні компоненти:

- набір серверів, які надають інформацію або інші послуги програмам, які звертаються до них (в найпростішому випадку — сервер один);
- набір клієнтів, які використовують сервіси, що надаються серверами;
- мережа, яка забезпечує взаємодію між клієнтами та серверами.

Сервери не залежать один від одного. Клієнти функціонують паралельно і незалежно один від одного. Немає жорсткої прив'язки клієнтів до серверів. Найбільш типовою є ситуація, коли один сервер одночасно обробляє запити від різних клієнтів; з іншого боку, клієнт може звертатися то до одного сервера, то до іншого. Клієнти мають знати про доступні сервери, але можуть не мати жодного уявлення про існування інших клієнтів. Загальноприйнятим є положення, що клієнти та сервери – це перш за все програмні модулі. Найчастіше вони знаходяться на різних комп'ютерах, але бувають ситуації, коли обидві програми – і клієнтська, і серверна, фізично розміщуються на одній машині; в такій ситуації сервер часто називається локальним сервером [6].

Переваги та недоліки архітектури клієнт-сервер. Переваги:

- дозволяють організовувати мережі з великою кількістю робочих станцій;
- забезпечують централізоване управління обліковими записами користувачів, безпекою та доступом, що спрощує мережне адміністрування;
- ефективний доступ до мережеских ресурсів;
- користувачеві потрібен один пароль для входу в мережу і для отримання доступу до всіх ресурсів, на які поширюються права користувача.

Поряд з перевагами клієнт – серверна архітектура має і ряд недоліків [7]:

- несправність сервера може зробити мережу непрацездатною, як мінімум втрату мережеских ресурсів;
- вимагають кваліфікованого персоналу для адміністрування;
- мають вищу вартість мереж і мережевого обладнання.

Отже, виходячи з визначених переваг, функцій та властивостей було вирішено використовувати для розроблювальної комп'ютерної системи архітектуру – клієнт-сервер.

## 3.2 Проєктування бази даних

### 3.2.1 Концептуальне проєктування бази даних

На основі аналізованої предметної області можна виділити наступні сутності:

- «Запис\_на\_прийом» (appointment), з атрибутами: «код\_призначення» (id\_appointment), «код\_залежності» (id\_relation), «код\_клієнта» (id\_client), «дата» (date), «час» (time);
- «Відділ» (branch), з атрибутами: «код\_відділу» (id\_branch), «відділ» (branch);
- «Картка\_клієнта» (card\_client), з атрибутами: «код\_картки» (id\_card\_client), «кож\_клієнта» (id\_client);
- «Клієнти» (clients), з атрибутами: «код\_клієнта» (id\_client), «прізвище\_клієнта» (surname\_client), «імя\_клієнта» (first\_name\_client), «по\_батькові\_клієнта» (last\_name\_client), «стать\_клієнта» (gender\_client), «телефон\_клієнта» (phone\_client), «пошта\_клієнта» (email\_client);
- «Коментарі» (comments), з атрибутами: «код\_коментаря» (id\_comment), «імя» (name), «пошта» (email), «текст\_коментаря» (text\_comments), «дата» (date);
- «Спеціалісти» (doctors), з атрибутами: «код\_спеціаліста» (id\_doctor), «прізвище\_спеціаліста» (surname\_doctor), «імя\_спеціаліста» (first\_name\_doctor), «по\_батькові\_спеціаліста» (last\_name\_doctor), «телефон\_спеціаліста» (phone\_doctor), «пошта\_спеціаліста» (email\_doctor), «спеціалізація\_спеціаліста» (specialty\_doctor), «логін\_спеціаліста» (login\_doctor), «пароль\_спеціаліста» (password\_doctor);
- «Деталі\_реєстрації» (login\_details), з атрибутами: «код\_деталей» (id\_details), «код\_клієнта» (id\_client), «логін\_клієнта» (login\_client), «пошта\_клієнта» (email\_client), «пароль\_клієнта» (password\_client);

- «Записи\_клієнта» (post\_client), з атрибутами: «код\_посту» (id\_post), «код\_картки» (id\_card\_client), «код\_спеціаліста» (id\_doctor), «код\_послуги» (id\_service), «дата\_візиту» (date\_visit), «коментар\_спеціаліста» (post\_comments);
- «Залежність» (relation), з атрибутами: «код\_залежності» (id\_relation), «код\_відділу» (id\_branch), «код\_послуги» (id\_service), «код\_спеціаліста» (id\_doctor);
- «Послуга» (services), з атрибутами: «код\_послуги» (id\_service), «послуга» (service);
- «Час» (time), з атрибутами «код\_часу» (id\_time), «час» (time\_all).

Між даними набором сутностей існують взаємозв'язки. Взаємозв'язок сутностей «Клієнти» (appointment) та «Запис\_на\_прийом» (appointment) (1:M). «Запис\_на\_прийом» (appointment) - таблиця, яка зберігає данні при реєстрації на прийом до спеціаліста. Таблиця «Клієнти» зберігає персональні данні про клієнтів.

Один клієнт може реєструватися на прийом декілька разів, та багато реєстрацій доступно для одного клієнта. Схема зв'язку між сутностями «Запис\_на\_прийом» (appointment) та «Клієнти» (clients) наведено на рисунку 3.2.

Властивості сутності «Запис\_на\_прийом» (appointment) наведено в таблиці 3.1. Властивості сутності «Клієнти» (clients) наведено в таблиці 3.2.

Таблиця 3.1 - Структура сутності «Запис\_на\_прийом» (appointment)

| Атрибут        | Тип даних | Допустиме значення   | Опис даних             | Обов'язковість |
|----------------|-----------|----------------------|------------------------|----------------|
| id_appointment | Числовий  | Натуральні числа     | Номер запису на прийом | Так            |
| id_relation    | Числовий  | Натуральні числа     | Номер даних запису     | Так            |
| id_client      | Числовий  | Натуральні числа     | Номер клієнту          | Так            |
| date           | Дата/Час  | Короткий формат дати | Дата запису на прийом  | Так            |
| time           | Текстовий | 5 символів           | Короткий формат дати   | Так            |

Таблиця 3.2 - Структура сутності «Клієнти» (clients)

| Атрибут           | Тип даних | Допустиме значення | Опис даних           | Обов'язковість |
|-------------------|-----------|--------------------|----------------------|----------------|
| id_client         | Числовий  | Натуральні числа   | Номер клієнта в базі | Так            |
| surname_client    | Текстовий | 40 символів        | Прізвище клієнта     | Так            |
| first_name_client | Текстовий | 40 символів        | Ім'я клієнта         | Так            |
| last_name_client  | Текстовий | 40 символів        | По батькові клієнта  | Ні             |
| gender_client     | Текстовий | 10 символів        | Стать клієнта        | Ні             |
| phone_client      | Текстовий | 15 символів        | Телефон клієнта      | Так            |
| email_client      | Текстовий | 40 символів        | Пошта клієнта        | Ні             |

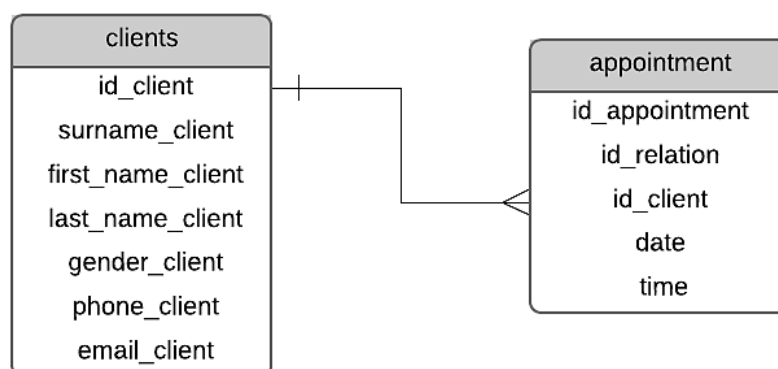


Рисунок 3.2 – Схема зв'язку сутностей «Запис\_на\_прийом»(appointment) та «Клієнти»(clients)

Взаємозв'язок «Деталі\_реєстрації» (login\_details) та «Клієнти»(clients)(1:1). Таблиця «Деталі\_реєстрації» (login\_details) зберігає та записує логін та пароль клієнтів для входу до особистого кабінету. Таблиця «Клієнти»(clients)зберігає персональні данні про клієнтів.

Кожний клієнт має лише один логін та пароль, відповідно один логін та пароль відповідають одному клієнту.

Властивості сутності «Деталі реєстрації» (login\_details) наведено в таблиці 3.3. Властивості сутності «Клієнти»(clients) наведено в таблиці 3.4.

Схема зв'язку «Деталі реєстрації» (login\_details) та «Клієнти» (clients) наведено на рисунку 3.3.

Таблиця 3.3- Структура сутності «Деталі реєстрації» (login\_details)

| Атрибут         | Тип даних        | Допустиме значення | Опис даних     | Обов'язковість |
|-----------------|------------------|--------------------|----------------|----------------|
| id_details      | Натуральні числа | Натуральні числа   | Номер деталей  | Так            |
| id_client       | Натуральні числа | Натуральні числа   | Номер клієнту  | Так            |
| login_client    | Текстовий        | 40 символів        | Логін клієнту  | Так            |
| email_client    | Текстовий        | 40 символів        | Пошта клієнту  | Так            |
| password_client | Текстовий        | 255 символів       | Пароль клієнта | Так            |

Таблиця 3.4 - Структура сутності «Клієнти» (clients)

| Атрибут           | Тип даних | Допустиме значення | Опис даних           | Обов'язковість |
|-------------------|-----------|--------------------|----------------------|----------------|
| id_client         | Числовий  | Натуральні числа   | Номер клієнта в базі | Так            |
| surname_client    | Текстовий | 40 символів        | Прізвище клієнта     | Так            |
| first_name_client | Текстовий | 40 символів        | Ім'я клієнта         | Так            |
| last_name_client  | Текстовий | 40 символів        | По батькові клієнта  | Ні             |
| gender_client     | Текстовий | 10 символів        | Стать клієнта        | Ні             |
| phone_client      | Текстовий | 15 символів        | Телефон клієнта      | Так            |
| email_client      | Текстовий | 40 символів        | Пошта клієнта        | Ні             |

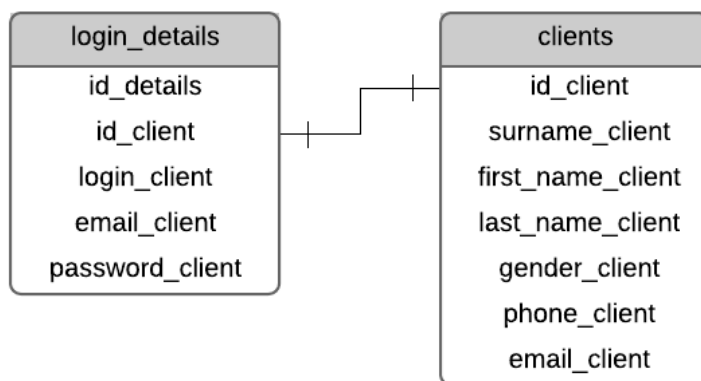


Рисунок 3.3 – Схема зв'язку сутностей «Деталі\_реєстрації»(login\_details) та «Клієнти»(clients)

Взаємозв'язок «Спеціалісти» (doctors) та «Запис\_на\_прийом»(appointment) (М:М). «Запис\_на\_прийом»(appointment) - таблиця, яка зберігає данні при реєстрації на прийом до спеціаліста. Таблиця «Спеціалісти» (doctors) зберігає данні про спеціалістів клініки.

Прізвище одного спеціаліста може знаходитися в багатьох записах клієнтів на прийом, та запис одного клієнта може включати декілька записів прізвищ спеціалістів.

Властивості сутності «Залежність» (relation) наведено в таблиці 3.5. Властивості сутності «Спеціалісти» (doctors) наведено в таблиці 3.6. Властивості сутності «Запис\_на\_прийом»(appointment) наведено в таблиці 3.7.

Таблиця 3.5 - Структура сутності «Залежність» (relation)

| Атрибут     | Тип даних | Допустиме значення | Опис даних        | Обов'язковість |
|-------------|-----------|--------------------|-------------------|----------------|
| id_relation | Числовий  | Натуральні числа   | Номер відношення  | Так            |
| id_branch   | Числовий  | Натуральні числа   | Номер відділу     | Так            |
| id_service  | Числовий  | Натуральні числа   | Номер послуги     | Так            |
| id_doctor   | Числовий  | Натуральні числа   | Номер спеціаліста | Так            |

Таблиця 3.6 - Структура сутності «Спеціалісти» (doctors)

| Атрибут          | Тип даних | Допустиме значення | Опис даних                                 | Обов'язковість |
|------------------|-----------|--------------------|--------------------------------------------|----------------|
| id_doctor        | Числовий  | Натуральні числа   | Номер спеціаліста                          | Так            |
| surname_doctor   | Текстовий | 40 символів        | Прізвище спеціаліста                       | Так            |
| firs_name_doctor | Текстовий | 40 символів        | Ім'я спеціаліста                           | Так            |
| last_name_doctor | Текстовий | 40 символів        | По батькові спеціаліста                    | Так            |
| phone_doctor     | Текстовий | 15 символів        | Телефон спеціаліста                        | Так            |
| email_doctor     | Текстовий | 40 символів        | Пошта спеціаліста                          | Так            |
| specialty_doctor | Текстовий | 40 символів        | Спеціалізація спеціаліста                  | Так            |
| login_doctor     | Текстовий | 20 символів        | Логін спеціаліста для особистого кабінету  | Так            |
| password_doctor  | Текстовий | 20 символів        | Пароль спеціаліста для особистого кабінету | Так            |

Таблиця 3.7 - Структура сутності «Запис\_на\_прийом» (appointment)

| Атрибут        | Тип даних | Допустиме значення   | Опис даних             | Обов'язковість |
|----------------|-----------|----------------------|------------------------|----------------|
| id_appointment | Числовий  | Натуральні числа     | Номер запису на прийом | Так            |
| id_relation    | Числовий  | Натуральні числа     | Номер даних запису     | Так            |
| id_client      | Числовий  | Натуральні числа     | Номер клієнту          | Так            |
| date           | Дата/Час  | Короткий формат дати | Дата запису на прийом  | Так            |
| time           | Текстовий | 5 символів           | Короткий формат дати   | Так            |

Схема зв'язку «Спеціалісти» (doctors) та «Запис\_на\_прийом» (appointment) наведено на рисунку 3.4.

На основі проведеного аналізу отримуємо загальний вигляд схеми бази даних, який знаходиться у додатку А, рисунок А.1.

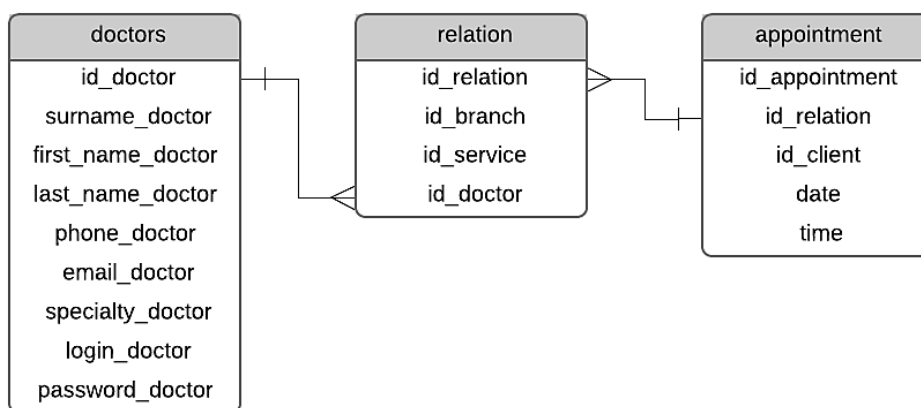


Рисунок 3.4 – Схема зв'язку сутностей «Спеціалісти» (doctors) та «Запис\_на\_прийом»(appointment)

### 3.2.2 Обґрунтування і вибір СКБД

Система керування базами даних - це універсальне програмний засіб, призначений для організації зберігання та обробки логічно взаємопов'язаних даних і забезпечення швидкого доступу до них [8].

До основних функцій СКБД відносяться [9, 10]:

- керування даними у зовнішній і оперативній пам'яті і забезпечення ефективного доступу до даних в процесі вирішення завдань;
- підтримка цілісності даних і управління транзакціями;
- ведення системного журналу змін до бази даних, що забезпечує відновлення бази даних після технічного або програмного збою;
- реалізація підтримки мови опису даних і мови запитів до даних;
- забезпечення безпеки даних;
- забезпечення паралельного доступу до даних декількох користувачів.

Однією з важливих задач комп'ютерної системи - забезпечення зручного інтерфейсу користувача БД з фізичними даними на зовнішніх носіях, тобто, зручність користувача в роботі з додатками. Комп'ютерна система повинна забезпечувати зручне введення і редагування даних.

Отже, для управління заданою комп'ютерною системою були розглянуті такі варіанти СКБД:

- MySQL;
- Oracle.

MySQL підтримує SQL (структурована мова запитів) і може застосовуватися в якості SQL-сервера [11]. Це означає, що спілкуватися з сервером можна на мові SQL: клієнт посилає серверу запит, той його обробляє і віддає клієнту тільки ті дані, які були отримані в результаті цього запиту. Тим самим клієнтові не потрібно викачувати дані і робити обчислення, як, наприклад, в Microsoft Access.

Крім того, MySQL - це програмне забезпечення з відкритим кодом, тобто його можна вільно вивчати та змінювати. Пакет розповсюджується на умовах GPL (GeneralPublicLicense), його можна безкоштовно завантажити з Інтернету для некомерційного використання. Перерахуємо переваги MySQL:

- швидкодія. Завдяки внутрішньому механізму багатопоточності швидкодія MySQL вельми висока;
- безпека. Досить високий рівень безпеки забезпечується завдяки базі даних MySQL, що створюється при установці пакету і містить п'ять таблиць. За допомогою цих таблиць можна описати, який користувач з будь-якого домену з якою таблицею може працювати і які команди він може застосовувати. Паролі, що зберігаються в базі даних, можна зашифрувати за допомогою вбудованої в MySQL функції password ();
- ліцензія. Зараз ця програма для некомерційних цілей поширюється безкоштовно;
- відкритість коду. Завдяки цьому ви зможете самі додавати в пакет потрібні функції, розширюючи його функціональність так, як вам потрібно;
- ресурси. Це може залежати від різних чинників, але в будь-якому випадку суперкомп'ютер вам не буде потрібно;
- кросплатформеність. В даний час існують версії програми для більшості поширених комп'ютерних платформ. Це говорить про те, що вам не нав'язують певну операційну систему. Ви самі можете вибрати, з чим працювати, наприклад з

Linux або Windows, але навіть в разі заміни ОС ви не втратите свої дані і вам навіть не знадобляться додаткові інструменти для їх перенесення.

Незважаючи на відсутність графічного інтерфейсу, в пакет MySQL входять досить потужні засоби адміністрування з інтерфейсом командного рядка.

Нижче наведено їх перелік зкороткими описами утиліт [12]:

- MySQLAdmin - головний інструмент адміністрування MySQL. З його допомогою ви можете створювати, знищувати, змінювати бази даних і повністю контролювати свій сервер;
- MySQLDump - утиліта резервування даних;
- MySQLAccess - дозволяє змінювати таблиці прав доступу і виводити їх зміст в зручному для читання вигляді;
- MySQLBug - в разі помилки в MySQL ця утиліта створює для розробників програми звіт про помилки, відсилаючи його також в поштовий список розсилки MySQL, щоб фахівці могли допомогти вирішити вашу проблему;
- MySQLImport - імпортує дані з файлу з роздільниками в базу даних;
- MySQLShow - показує структуру баз даних і таблиць, з яких вони складаються.

Oracle займає лідируючі позиції на ринку СКБД і, що особливо важливо, лідирує на платформах Unix і Windows [13]. Причина широку поширеність Oracle полягає насамперед у високих експлуатаційних характеристиках СКБД, великій кількості підготовлених вітчизняних фахівців з Oracle, наявності підтримуючої інфраструктури - навчальних центрів, широкої мережі партнерів Oracle, великому числу технічних курсів по Oracle в вузах і т.д.

З технічної точки зору важливим є те, що Oracle функціонує практично на всіх існуючих комп'ютерних платформах, в тому числі і на великих комп'ютерах (OS / 390) і на різних різновидах Unix, в тому числі Solaris, HP-UX, AIX, Linux, SCO Unix і т.д.

Іншою важливою характеристикою є підтримка Oracle всіх можливих варіантів архітектур, в тому числі симетричних багатопроцесорних систем, кластерів, систем з масовим паралелізмом і т.д.

Що стосується обсягів інформації, то максимальний обсяг бази даних Oracle Database XE становить 12 гігабайт (Гб). З них від 0.5 до 0.9 Гб використовуються словником даних, внутрішніми схемами і тимчасовим дисковим простором. Тому залишається 11.0 Гб даних.

Найважливішою перевагою перед конкурентами Oracle є ідентичність коду різних версій сервера баз даних Oracle для всіх платформ, гарантує ідентичність і передбачуваність роботи Oracle на всіх типах комп'ютерів, які не входили до її складу.

Всі варіанти сервера Oracle мають в своїй основі один і той же вихідний програмний код і функціонально ідентичні, за винятком деяких функцій, які, наприклад, можуть бути додані до Oracle Database Enterprise Edition і не можуть - в Oracle Database Standard Edition.

Таким чином, для всіх платформ існує єдина СКБД в різних версіях, яка поводить ся однаково і надає однакоvu функціональність незалежно від платформи, на якій вона встановлена.

Єдиним недоліком даної СКБД є складність адміністрування, однак усі витрати на її впровадження та освоєння надалі окупляться ефективною і надійною роботою. Серед основних властивостей СКБД Oracle слід зазначити такі, як [10]:

- висока надійність;
- можливість розбиття великих баз даних на розділи (великий розділ бази даних), що дає можливість ефективно управляти гігантськими гігабайтними базами;
- вільні таблиці (в інших СКБД все таблиці заповнюються відразу при створенні);
- розпаралелювання операцій в запиті;
- наявність широкого спектра засобів розробки, моніторингу та адміністрування;
- підтримка відомих платформ: Windows, AIX, Compaq Tru64 UNIX, HP 9000 серії HP-UX, Linux Intel, Sun Solaris;

- доступ по протоколу HTTP, що підтримує відправку SQL-запитів до БД із застосуванням URL-адрес;
- інструментальні засоби, що включають графічні засоби розробки схем, генерації запитів і коду.

В результаті аналізу СКБД для розроблювальної комп'ютерної системи було вибрано MySQL. Оскільки система, що розроблюється, буде відноситися до невеликих за розміром систем та інтерфейс користувача буде веб-додатком, потрібно обрати СКБД, з якою можуть співпрацювати мови програмування для написання веб-додатків.

### **3.3 Проєктування інтерфейсу користувача**

Оформлення або дизайн – це зовнішній вигляд системи, персональна візитка компанії. Перше враження від системи є дуже важливим, бо від нього залежить чи залишиться відвідувач або потенційний клієнт, чи знайомитиметься з інформацією, яку йому пропонують, або закрий вікно з цим сайтом і забуде про його існування.

Оформлення системи підказує відвідувачеві, куди він потрапив, допомагає орієнтуватися по системі, а може, і навпаки, збити відвідувача так, що навіть при добре розробленій структурі відвідувачеві важко буде зорієнтуватися [14].

Дизайн сайту повинен відповідати фірмовому стилю компанії, і добре асоціюватися з образом пропонованих товарів або послуг. Навіть в якій-небудь вузькій сфері діяльності образ компанії може бути переданий по-різному. Це залежить, від регіональних особливостей, від цільової аудиторії, на яку розрахований сайт.

Одні й ті ж послуги можуть бути адресовані різним соціальним та віковим групам, значить, і дизайн сайту краще виконувати в стилі, привабливому для цих цільових груп.

Основна увага при розробці дизайну приділяється зазвичай головній сторінці, оскільки саме їй відводиться роль «дверей» своєрідного віртуального офісу компанії, яким є сайт-представництво. У дизайні головної сторінки сайту присутні зазвичай логотип і слоган компанії, графічні зображення або ж фотоілюстрації продукції та послуг компанії.

Після того як розроблений дизайн головної сторінки сайту, приступають до створення дизайну всіх другорядних сторінок. Як правило, другорядні сторінки також містять логотип і слоган компанії або ж тільки логотип. Наявність різних інших елементів дизайну на другорядних сторінках сайту-представництва залежить від того, яким повинен бути контент цих сторінок [15].

### 3.3.1 Проектування інтерфейсу користувача

Макет зовнішньої структури головної сторінки представлено на рисунку 3.5. Сторінка має елементи, які розташовані в порядку відповідно до нумерації в макеті:

- іконки соціальних мереж;
- зображення – це логотип компанії, який повинен відображатися на кожній сторінці;
- кнопка для входу до особистого кабінету;
- кнопка для запису на прийом до спеціаліста;
- посилання, яке відображає електронну адресу клініки;
- навігаційне меню, повинно відображатися на кожній сторінці. Перше посилання – це головна сторінка. Інші пункти меню – це сторінки про спеціалістів, послуги, контактні інформації та новини клініки;
- слайдер із трьох картинок, який автоматично змінює картинку (фотографії) пов'язані з клінікою;
- блок інформації, який розташовано у три колонки;
- інформація, яка розташована у трьох блоках;
- блок інформації та маркований список, який відображає список послуг клініки;
- зображення;

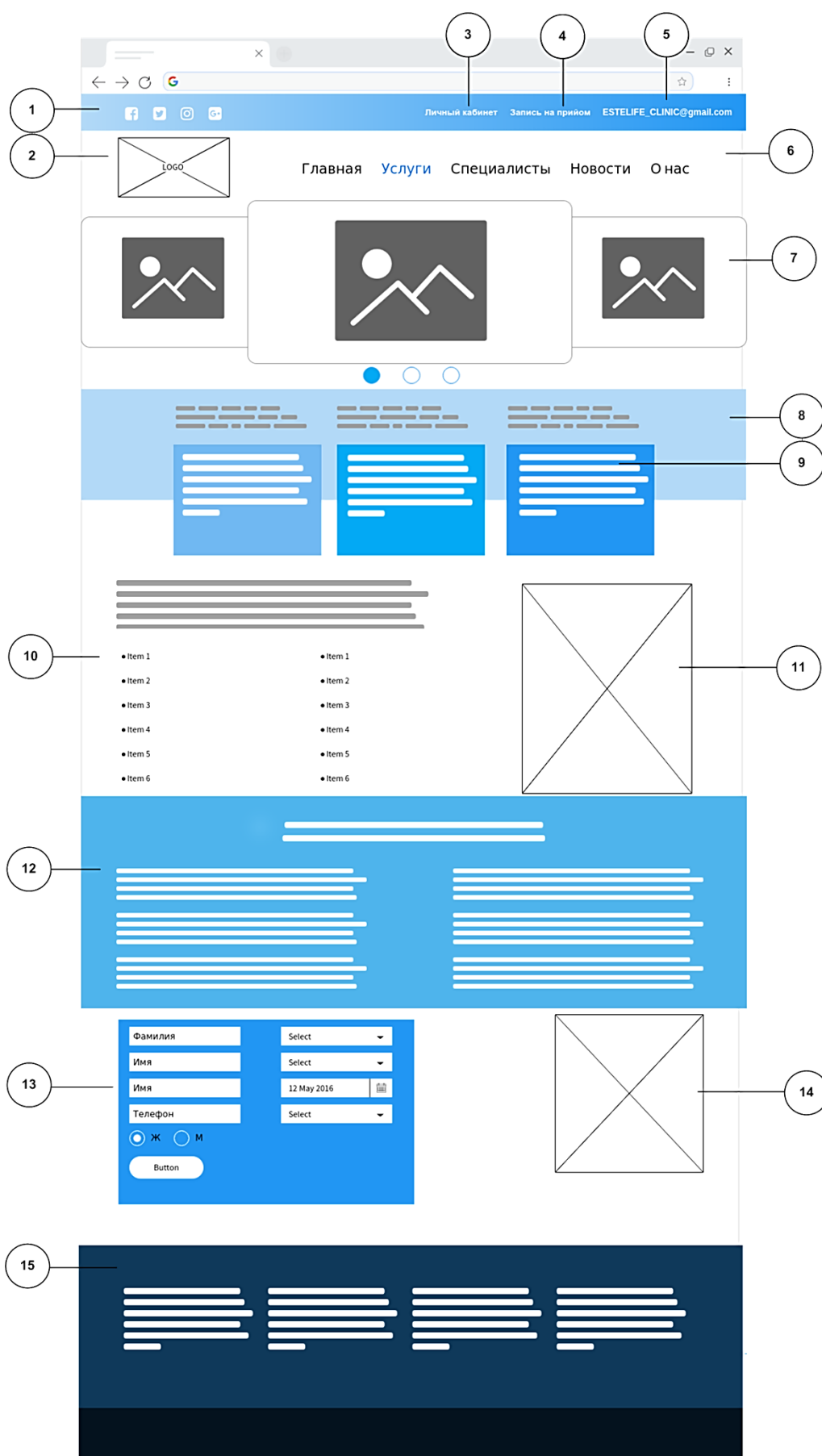


Рисунок 3.5 – Зовнішня структура сайту

- блок інформації про відділи клініки, які розташовані у дві колонки;
- форма запису на прийом, яка включає поля для заповнення: прізвище, ім'я, по батькові, телефон, послуга, спеціаліст, дата, час та кнопка відправки (див. рис. 3.6);
- зображення;
- футер, який розділено на чотири колонки. Включає логотип, останні три новини про клініку, швидкі посилання, контактну інформацію.

### Запис на прийом

The form titled "Запис на прийом" (Appointment) contains the following elements:

- Input field for "Прізвище" (Last name)
- Input field for "Ім`я" (First name)
- Input field for "По батькові" (Patronymic)
- Input field for "Телефон" (Phone)
- Input field for "Дата" (Date) with a calendar icon
- Input field for "Час" (Time) with a dropdown arrow
- Input field for "Послуга" (Service) with a dropdown arrow
- Input field for "Спеціаліст" (Specialist) with a dropdown arrow
- Input field for "Відправка" (Send) with a dropdown arrow
- "Відправити" (Send) button

Рисунок 3.6 – Зовнішня структура форми запису на прийом до спеціаліста

Форма для авторизації до особистого кабінету представлено на рисунку 3.7. Форма має елементи, які розташовані в порядку відповідно до нумерації в макеті:

- поле для вводу email (логіну);
- поле для вводу паролю;
- кнопка запам'ятовування;
- кнопка «Вхід», якщо користувач вже був зареєстрований, то після натискання кнопки переходить до особистого кабінету.

The diagram shows a login form titled "Вхід до особистого кабінету". It contains four main elements, each indicated by a numbered circle and a line:

- 1: Login field (Логін) with placeholder text "name@domain.com".
- 2: Password field (Пароль) with placeholder text "Password".
- 3: Remember me checkbox (Запам'ятати мене) and a link "Збули пароль?".
- 4: Blue "Ok" button.

Рисунок 3.7 – Зовнішня структура модулю авторизації до особистого кабінету

Якщо користувач не був зареєстрований до особистого кабінету, то він переходить до форми реєстрація (див. рис. 3.8). Основні елементи, які розташовані в порядку відповідно до нумерації в макеті:

- поле для вводу прізвища;
- поле для вводу email (логіну);
- поле для вводу паролю;
- поле для повторного вводу паролю, для підтвердження дійсності;
- кнопка, для підтвердження введених даних, після чого користувач потрапляє до особистого кабінету;
- або можлива реєстрації за допомогою соціальних мереж.

The diagram shows a registration form titled "Реєстрація до особистого кабінету". It contains five numbered callouts pointing to specific elements: 1 points to the "Логін" (Login) input field, 2 points to the "Email" input field, 3 points to the "Пароль" (Password) input field, 4 points to the "Повторний пароль" (Repeat password) input field, and 5 points to the blue "Реєстрація" (Registration) button.

Рисунок 3.8 – Зовнішня структура модулю реєстрації до особистого кабінету

### 3.3.2 Аналіз та вибір технологій для розробки інтерфейсу користувача

У ході створення комп'ютерної системи буде використано декілька основних технологій [16]:

- HTML;
- CSS;
- JavaScript;
- PHP;
- Bootstrap фреймворк;
- Web-сервер Apache;
- Ajax.

HTML – це мова розмітки тексту. Всесвітня павутина складається з веб-сторінок, які створено у форматі HTML (HyperText Markup Language, «мова розмітки гіпертексту»). HTML - це фундаментальна, базова технологія Інтернету.

HTML використовує спеціальні оператори – теги (tag) чи інша назва дескриптори (descriptor) для розмітки текстового документа. Ці позначки

вказують в якому вигляді буде виведено текстовий чи інший елемент у вікні браузера.

HTML дозволяє формувати на сторінці сайту текстові блоки, додавати до них зображення, організовувати таблиці, керувати відтворенням кольору, додавати до дизайну сайту звуковий супровід, організовувати гіперпосилання з переходом до інших розділів сервера або звертатися до інших ресурсів Інтернету і компонувати всі ці елементи між собою. Документи, що створено лише засобами HTML мають розширення .htm або .html.

Однією з основних функціональних особливостей мови HTML, завдяки якій вона і отримала свою назву, є гіперпосилання.

Гіперпосилання (Hyperlink) — це базовий функціональний елемент HTML-документу, який реалізовує зв'язок певного об'єкту веб-сторінки з іншим об'єктом. Для гіперпосилання може використовуватися як фрагмент тексту, так і графічний об'єкт, а сам гіперзв'язок можна встановлювати як між об'єктами одного сайту, так і між об'єктами, що розміщені на різних сайтах Інтернету.

HTML є мовою, що лише інтерпретується, тому, для виконання коду, його не потрібно компілювати. Інтерпретатор мови втілено в браузер, і він «компілює» код безпосередньо під час відкривання документа. Якщо в коді сторінки виявлено помилку, інтерпретатор, зазвичай, не видає відповідного попередження, а просто ігнорує весь «помилковий» рядок, що може зіпсувати зовнішній вигляд завантаженої сторінки. Це є важливим для розробників, тому слід бути уважним під час складання HTML-коду і ретельно тестувати результати своєї роботи [17].

CSS (Cascading Style Sheets) — це технологія опису зовнішнього вигляду документа, що створено засобами HTML, XML і XHTML.

CSS використовується для завдання кольорів, шрифтів, розташування елементів сторінки тощо. До появи CSS оформлення веб-сторінок вказувалося безпосередньо в HTML-коді сторінки. Проте, з появою CSS стало можливим принципове розділення змісту і представлення документа. За рахунок такого нововведення стало можливим легке застосування єдиного стилю оформлення для кількох сторінок сайту, а також швидка зміна цього оформлення. Переваги:

- зменшення часу завантаження сторінок сайту за рахунок перенесення правил представлення даних до окремого CSS-файлу. В цьому випадку браузер завантажує лише структуру документа і дані, що містяться на сторінці. CSS-файл з правилами представлення цих даних завантажується браузером лише один раз і зберігається в кеші браузера.

- простота подальшої зміни дизайну. Не потрібно виправляти кожен сторінку, достатньо лише змінити кілька правил у CSS-файлі. додаткові можливості оформлення. Наприклад, за допомогою CSS-правил можна застосувати обтікання певного блоку текстом або зробити так, щоб меню фіксовано знаходилося в певному місці при перегортанні сторінки.

Основний недолік – це те що різні браузери можуть в різний спосіб інтерпретувати правила CSS, і відповідно, по різному відображати одні і ті ж фрагменти сторінки [17].

JavaScript – це мова, що призначена для написання сценаріїв для інтерактивних HTML-сторінок. Мова JavaScript не має жодного відношення до мови Java. Java розроблено фірмою SUN, а JavaScript – фірмою Netscape Communication Corporation. Первинною назвою було «LiveScript», але, з огляду на велику популярність мови Java, назву «LiveScript» з комерційних міркувань було змінено на «JavaScript».

JavaScript не призначено для створення автономних застосувань. Програмний код на JavaScript вписується безпосередньо в текст HTML-документа і інтерпретується браузером в міру завантаження цього документа. За допомогою JavaScript можна динамічно змінювати текст завантаженого HTML-документу і реагувати на події, які пов'язані з діями відвідувача або змінами стану документа чи вікна.

Важливою особливістю JavaScript є об'єктна орієнтованість. Програмісту є доступними численні об'єкти, такі, як документи, гіперпосилання, форми, фрейми тощо. Об'єкти характеризуються описовою інформацією (властивостями) і можливими діями (методами) [16].

PHP (Personal Home Page) - мова створення сценаріїв, яка давно переросла свою назву. PHP - це серверна мова. Конструкції PHP, що вставлено в HTML-текст, виконуються сервером при кожному відвідуванні сторінки. Результат обробки конструкцій разом із звичайним HTML-текстом передається браузеру [18].

Існують три основні області застосування PHP [18]:

- створення скриптів для виконання на стороні сервера. PHP найбільш широко використовується саме таким чином. Все, що вам знадобиться, це інтерпретатор PHP (у вигляді програми CGI або серверного модуля), веб-сервер і браузер. Для того щоб можна було переглядати результати виконання PHP-скриптів в браузері, потрібен працюючий веб-сервер і встановлений PHP. У випадку, якщо ви просто експериментуєте, ви цілком можете використовувати свій домашній комп'ютер замість сервера;

- створення скриптів для виконання в командному рядку. Ви можете створити PHP-скрипт, здатний запускатися незалежно від веб-сервера та броузера. Все, що вам буде потрібно – парсер PHP. Такий спосіб використання PHP ідеально підходить для скриптів, які повинні виконуватися регулярно, наприклад, за допомогою cron (на платформах \* nix або Linux) або за допомогою планувальника завдань (Task Scheduler) на платформах Windows. Ці скрипти також можуть бути використані в задачах простої обробки текстів;

- створення віконних додатків, що виконуються на стороні клієнта. Можливо, PHP є не найкращою мовою для створення подібних додатків, але, якщо ви дуже добре знаєте PHP і хотіли б використати деякі його можливості у своїх клієнт-додатках, ви можете використовувати PHP-GTK для створення таких додатків. Подібним чином ви можете створювати і крос-платформні додатки. PHP-GTK є розширенням PHP і не поставляється разом з дистрибутивом PHP.

PHP має ряд переваг, зокрема [18]:

- висока продуктивність;
- функціональність. Розробку PHP-програми можна відокремити від власне розробки веб-сторінки, що спрощує працю і для програміста і для дизайнера;

- ціна. Мова PHP є абсолютно безкоштовною;
- простота у використанні;
- переносимість. Один і той же PHP-код можна використовувати як в середовищі Windows, так і на платформах UNIX.

Фреймворк - це програмне забезпечення, що полегшує розробку і об'єднання різних компонентів великого програмного проекту. Ця платформа підходить для створення сайтів, бізнес-додатків і веб-сервісів.

Фреймворк відрізняється від бібліотеки тим, що бібліотека може бути використана в програмному продукті просто як набір підсистем близькою функціональності, не впливаючи на архітектуру основного програмного продукту і не накладаючи на неї ніяких обмежень. Фреймворк диктує правила побудови архітектури додатку, задаючи на початковому етапі розробки поведінка за умовчанням, формуючи каркас, який потрібно буде розширювати і змінювати відповідно до зазначених вимог. Фреймворк може включати допоміжні програми, бібліотеки коду, мова сценаріїв і інше програмне забезпечення, що полегшує розробку і об'єднання різних компонентів великого програмного проекту [19].

Bootstrap (також відомий як Twitter Bootstrap) - вільний набір інструментів для створення сайтів і веб-додатків. Включає в себе HTML- і CSS-шаблони оформлення для типографіки, веб-форм, кнопок, міток, блоків навігації та інших компонентів веб-інтерфейсу, включаючи JavaScript-розширення. Bootstrap використовує сучасні напрацювання в області CSS і HTML, тому необхідно бути уважним при підтримці старих браузерів.

Основні інструменти Bootstrap:

- сітки - заздалегідь задані розміри колонок, які можна відразу ж використовувати, наприклад ширина колонки 140 px відноситься до класу .span2 (.col-md-2 в третій версії фреймворка), який можна використовувати в CSS-описі документа;
- шаблони - фіксований або гумовий шаблон документа. Типографіка - опису шрифтів, визначення деяких класів для шрифтів, таких як код, цитати і т.д.;
- медіа - представляє деякий спосіб упорядкування зображень та відео;

- таблиці - кошти оформлення таблиць, аж до додавання функціональності сортування;
- форми - класи для оформлення форм і деяких подій, що відбуваються з ними;
- навігація - класи оформлення для табів, вкладок, сторінковий, меню і панелі інструментів;
- алерт - оформлення діалогових вікон, підказок і спливаючих вікон.

Переваги використання бібліотеки:

- висока швидкість розробки;
- адаптивний дизайн Bootstrap спрямований на створення макета під різні пристрої - ноутбуки, планшети, смартфони. При цьому код пишеться один, а масштабування в залежності від ширини пристрою бере на себе фреймворк.;
- відкрита програма Ви можете вільно вивчати вихідні коди бібліотеки, змінювати і розширювати її під власні потреби;
- сумісність з браузерами, компоненти бібліотеки написані і протестовані з урахуванням роботи різних браузерів. Це гарантує, що макет буде виглядати однаково незалежно від обраного браузера;
- низький поріг входження, щоб використовувати бібліотеку в своїй роботі, потрібно мати мінімальні знання по HTML, CSS і JavaScript. Це дозволяє створювати ефектні сайти навіть початківцям розробникам.

Недоліки, звичайно ж, теж є і вони впливають в основному з універсальності системи. По-перше, файли бібліотеки, навіть стислі, займають досить багато місця і збільшують навантаження на сервер. З підвищенням досвіду потрібно завантажувати і встановлювати тільки необхідні компоненти Bootstrap, скорочуючи тим самим обсяг файлів. По-друге, пропоновані в бібліотеці стилі можуть не підходити під дизайн розроблюваного сайту і доведеться багато переробляти. У таких випадках, як відомо, простіше написати все самому з нуля.

Загалом, Bootstrap годиться для типових сайтів, дизайн яких орієнтований на бібліотеку. А сайтів, які пропонують теми і шаблони, в тому числі безкоштовні, останнім часом народилося досить багато [19].

Apache HTTP-сервер - вільний веб-сервер. Apache є кросплатформним ПО, підтримує операційні системи Linux, BSD, Mac OS, Microsoft Windows, Novell NetWare, BeOS.

Основними достоїнствами Apache вважаються надійність і гнучкість конфігурації. Він дозволяє підключати зовнішні модулі для надання даних, використовувати СКБД для аутентифікації користувачів, модифікувати повідомлення про помилки і т.д. Входить в LAMP - поширений набір програмного забезпечення для створення веб-сайтів (Linux, Apache, MySQL, PHP).

Ядро Apache включає в себе основні функціональні можливості, такі як обробка конфігураційних файлів, протокол HTTP і система завантаження модулів. Ядро (на відміну від модулів) повністю розробляється Apache Software Foundation, без участі сторонніх програмістів.

Теоретично, ядро Apache може функціонувати в чистому вигляді, без використання модулів. Однак, функціональність такого рішення вкрай обмежена. Ядро Apache повністю написано на мові програмування C.

Система конфігурації Apache заснована на текстових конфігураційних файлах. Має три умовних рівня конфігурації:

- конфігурація сервера (httpd.conf);
- конфігурація віртуального хоста (httpd.conf с версії 2.2 extra / httpd-vhosts.conf);
- конфігурація рівня директорії (.htaccess).

Має власну мову конфігураційних файлів, заснований на блоках директив. Практично всі параметри ядра можуть бути змінені через конфігураційні файли, аж до управління MPM. Велика частина модулів має власні параметри.

Частина модулів використовує в своїй роботі конфігураційні файли операційної системи (наприклад / etc / passwd і / etc / hosts). Крім цього, параметри можуть бути задані через ключі командного рядка [20].

Ajax (Asynchronous Javascript And XML) це підхід до створення веб-застосувань за допомогою наступних технологій:

- стандартизоване представлення засобами XHTML і CSS;
- динамічне відображення і взаємодія з користувачем за допомогою DOM;
- обмін і обробка даних у вигляді XML и XSLT;
- широке застосування мови сценаріїв JavaScript;
- асинхронні запити за допомогою об'єкту XMLHttpRequest.

В стандартному веб-застосуванні обробкою всієї інформації займається сервер, браузер відповідає лише за взаємодію з користувачем, передачу запитів і виведення отриманих даних у форматі HTML.

В Ajax-застосуванні між користувачем і сервером з'являється ще один посередник – програмний механізм (рушій) Ajax. Він визначає, які запити можна обробити з боку клієнта, а які необхідно виконувати на сервері.

Поведінка сервера теж змінилася. Якщо раніше на кожен запит сервер видавав нову сторінку, то тепер він надсилає лише ті дані, які потрібні клієнту, а рушій Ajax в браузері формує з них HTML-код.

Асинхронність виявляється в тому, що далеко не кожен запит користувача скеровується до сервера, причому зворотне теж справедливо - далеко не кожна реакція сервера обумовлена запитом користувача. Велику частину запитів формує рушій Ajax, причому є можливим, щоб він передбачав запити користувача.

Зрозуміло, що за такою схемою роботи міняється якісне навантаження на сервер - якщо раніше запитів було мало, але кожен з них вимагав значних ресурсів (серверу потрібно витягнути інформацію з бази даних, сформувати з неї веб-сторінку і відправити до браузера), то тепер завдання сервера спрощується (формувати веб-сторінки не потрібно, та і об'єм даних, що передаються є значно меншим), але доводиться більше обробляти запитів.

Створювати застосування на Ajax є трудомістким і складним завданням. Потрібно написати і відлагодити на JavaScript рушій з десятих чи двадцятих тисяч рядків коду плюс реалізувати серверну частину.

На сьогодні спостерігається тенденція на стрімке поширення Ajax-застосувань. Масштабні Ajax-проекти представлено у Google - Google Suggest (сервіс, що підказує найбільш популярні запити), Gmail і Google Maps.

Для розробки комп'ютерної системи було обрано наступні технології: HTML, CSS, JavaScript, PHP, фреймворк Bootstarp, Web-сервер Apache, Ajax.

## **4 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ**

### **4.1 Розробка бази даних**

Логічного проектування бази даних полягає в перетворенні спроектованої концептуальної моделі даних у логічну модель даного підприємства з урахуванням обраного типу СКБД. Логічна модель даних є джерелом інформації для фази фізичного проектування. Вона надає розробнику фізичної моделі засоби проведення всебічного аналізу різних аспектів роботи з даними, що має винятково важливе значення для вибору дійсно ефективного проектного рішення, саме визначається оптимальний склад таблиць для зберігання вихідної інформації. Для кожної таблиці вказується її назва, перелік полів, первинний ключ, обов'язковість полів. Ідентифікуються зв'язки між таблицями [21-23].

На даному етапі виконуються наступні дії [22]:

- видалення складних зв'язків.
- видалення рекурсивних зв'язків.
- видалення зв'язків з атрибутами.
- видалення множинних атрибутів.
- перевірка моделі щодо транзакцій користувачів;
- перевірка зв'язків 1:1 «один до одного»;
- виділення атрибутів з фіксованим набором значень;
- перевірка моделі за допомогою правил нормалізації;
- видалення надлишкових зв'язків.

На основі технічного завдання були визначені ресурси, які необхідно зберігати в базі даних. Дана база даних складається з 11 таблиць реалізованих за допомогою MySQL:

- «Запис\_на\_прийом» (appointment);
- «Відділ» (branch);
- «Картка\_клієнта» (card\_client);
- «Клієнти» (clients);
- «Коментарі» (comments);
- «Спеціалісти» (doctors);
- «Деталі\_реєстрації» (login\_details);
- «Записи\_клієнта» (post\_client);
- «Залежність» (relation);
- «Послуга» (services);
- «Час» (time).

Дані про реєстрація на прийом зберігаються в таблиці «Запис\_на\_прийом» (appointment). Реалізація таблиці за допомогою MySQL наведено в лістингу 4.1, логічна структура таблиці наведена в таблиці 4.1.

#### Лістинг 4.1 – Реалізація таблиці «Запис\_на\_прийом» (appointment)

```
CREATE TABLE IF NOT EXISTS `appointment` (
  `id_appointment` int(11) NOT NULL AUTO_INCREMENT,
  `id_relation` int(11) NOT NULL,
  `id_client` int(11) NOT NULL,
  `surname` char(50) NOT NULL,
  `first_name` char(50) NOT NULL,
  `last_name` char(50) NOT NULL,
  `telephone` char(15) NOT NULL,
  `date` date NOT NULL,
  `time` char(5) NOT NULL,
  PRIMARY KEY (`id_appointment`),
  KEY `id_appointment` (`id_appointment`),
  KEY `id_relation` (`id_relation`),
  KEY `id_client` (`id_client`),
  CONSTRAINT `FK_appointment_relation` FOREIGN KEY
  (`id_relation`) REFERENCES `relation` (`id_relation`)
) ENGINE=InnoDB AUTO_INCREMENT=12 DEFAULT CHARSET=utf8
ROW_FORMAT=DYNAMIC;
```

Для даних ресурсів справедливі наступні правила:

- номер запису на прийом– ціле число (1,2,...), AUTO\_INCREMENT, тип даних INT, довжина значення 11, поле не може бути пустим;

- номер даних запису, номер клієнта – ціле число (1,2,...), тип даних INT, довжина значення 11, поле не може бути пустим;
- прізвище клієнта для запису на прийом телефон – тип даних VARCHAR, довжина значення 50, поле не може бути пустим;
- ім'я клієнта для запису на прийом телефон – тип даних VARCHAR, довжина значення 50, поле не може бути пустим;
- по батькові клієнта для запису на прийом телефон – тип даних VARCHAR, довжина значення 50, поле не може бути пустим;
- телефон – тип даних VARCHAR, довжина значення 15, поле не може бути пустим;
- дата запису на прийом – тип даних DATE, поле не може бути пустим;
- час запису на прийом – тип даних VARCHAR, довжина значення 5, поле не може бути пустим.

Таблиця 4.1 - Структура сутності «Запис\_на\_прийом» (appointment)

| Атрибут        | Тип даних    | Опис даних                               |
|----------------|--------------|------------------------------------------|
| id_appointment | INT(11)      | Номер запису на прийом                   |
| id_relation    | INT(11)      | Номер даних запису                       |
| id_client      | INT(11)      | Номер клієнту                            |
| surname        | VARCHAR (50) | Прізвище клієнта для запису на прийом    |
| first_name     | VARCHAR (50) | Ім'я клієнта для запису на прийом        |
| last_name      | VARCHAR (50) | По батькові клієнта для запису на прийом |
| telephone      | VARCHAR (15) | Телефон клієнта                          |
| date           | DATE         | Дата запису на прийом                    |
| time           | VARCHAR(5)   | Час запису на прийом                     |

Назва відділу зберігається в таблиці під назвою «Відділ» (branch). Реалізація таблиці за допомогою MySQL наведено в лістингу 4.2, логічна структура таблиці наведена в таблиці 4.2.

Інформація про клієнтів зберігається в таблиці під назвою «Клієнти» (clients). Реалізація таблиці за допомогою MySQL наведено в лістингу 4.2, логічна структура таблиці наведена в таблиці 4.2.

## Лістинг 4.2 – Реалізація таблиці «Клієнти» (clients)

```

CREATE TABLE IF NOT EXISTS `clients` (
  `id_client` int(11) NOT NULL AUTO_INCREMENT,
  `surname_client` varchar(40) NOT NULL,
  `first_name_client` varchar(40) NOT NULL,
  `last_name_client` varchar(40) NOT NULL,
  `gender_client` varchar(10) NOT NULL,
  `phone_client` varchar(15) NOT NULL,
  `email_client` varchar(40) NOT NULL,
  PRIMARY KEY (`id_client`),
  KEY `id_client` (`id_client`),
  CONSTRAINT `FK_clients_appointment` FOREIGN KEY (`id_client`)
REFERENCES `appointment` (`id_client`),
  CONSTRAINT `FK_clients_card_client` FOREIGN KEY (`id_client`)
REFERENCES `card_client` (`id_client`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 ROW_FORMAT=DYNAMIC;

```

Таблиця 4.2 - Структура сутності «Клієнти» (clients)

| Атрибут           | Тип даних   | Опис даних           |
|-------------------|-------------|----------------------|
| id_client         | INT(11)     | Номер клієнта в базі |
| surname_client    | VARCHAR(40) | Прізвище клієнта     |
| first_name_client | VARCHAR(40) | Ім'я клієнта         |
| last_name_client  | VARCHAR(40) | По батькові клієнта  |
| gender_client     | VARCHAR(10) | Стать клієнта        |
| phone_client      | VARCHAR(15) | Телефон клієнта      |
| email_client      | VARCHAR(40) | Пошта клієнта        |

Для даних ресурсів справедливі наступні правила:

- номер клієнта – ціле число (1,2,...), AUTO\_INCREMENT, тип даних INT, довжина значення 11, поле не може бути пустим;
- прізвище клієнта – тип даних VARCHAR, довжина значення 40, поле не може бути пустим;
- ім'я клієнта – тип даних VARCHAR, довжина значення 40, поле не може бути пустим;
- по батькові клієнта – тип даних VARCHAR, довжина значення 40, поле не може бути пустим;
- стать клієнта – тип даних VARCHAR, довжина значення 10, поле не може бути пустим, приймає значення: чоловік або жінка;

- телефон клієнта – тип даних VARCHAR, довжина значення 15, поле не може бути пустим;
- пошта клієнта – тип даних VARCHAR, довжина значення 40, поле не може бути пустим.

Інформація про спеціалістів знаходить в таблиці «Спеціалісти» (doctors). Реалізація таблиці за допомогою MySQL наведено в лістингу 4.3, логічна структура таблиці наведена в таблиці 4.3.

#### Лістинг 4.3 – Реалізація таблиці «Спеціалісти» (doctors)

```
CREATE TABLE IF NOT EXISTS `doctors` (
  `id_doctor` int(11) NOT NULL AUTO_INCREMENT,
  `surname_doctor` varchar(40) NOT NULL,
  `first_name_doctor` varchar(40) NOT NULL,
  `last_name_doctor` varchar(40) NOT NULL,
  `phone_doctor` varchar(15) NOT NULL,
  `email_doctor` varchar(40) NOT NULL,
  `specialty_doctor` varchar(40) NOT NULL,
  `login_doctor` varchar(20) NOT NULL,
  `password_doctor` varchar(20) NOT NULL,
  PRIMARY KEY (`id_doctor`),
  KEY `id_doctor` (`id_doctor`)
) ENGINE=InnoDB AUTO_INCREMENT=12 DEFAULT CHARSET=utf8
ROW_FORMAT=DYNAMIC;
```

Таблиця 4.3 - Структура сутності «Спеціалісти» (doctors)

| Атрибут          | Тип даних   | Опис даних                                 |
|------------------|-------------|--------------------------------------------|
| id_doctor        | INT(11)     | Номер спеціаліста                          |
| surname_doctor   | VARCHAR(40) | Прізвище спеціаліста                       |
| firs_name_doctor | VARCHAR(40) | Ім'я спеціаліста                           |
| last_name_doctor | VARCHAR(40) | По батькові спеціаліста                    |
| phone_doctor     | VARCHAR(15) | Телефон спеціаліста                        |
| email_doctor     | VARCHAR(40) | Пошта спеціаліста                          |
| specialty_doctor | VARCHAR(40) | Спеціалізація спеціаліста                  |
| login_doctor     | VARCHAR(20) | Логін спеціаліста для особистого кабінету  |
| password_doctor  | VARCHAR(20) | Пароль спеціаліста для особистого кабінету |

Для даних ресурсів справедливі наступні правила:

- номер спеціалістів – ціле число (1,2,...), AUTO\_INCREMENT, тип даних INT, довжина значення 11, поле не може бути пустим;
- прізвище спеціаліста – тип даних VARCHAR, довжина значення 40, поле не може бути пустим;
- ім'я спеціаліста – тип даних VARCHAR, довжина значення 40, поле не може бути пустим;
- по батькові спеціаліста – тип даних VARCHAR, довжина значення 40, поле не може бути пустим;
- для позначення прізвища, ім'я, по батькові спеціаліста використовуються наступні константи: Чешенко Дмитро В'ячеславович, Сірик Юрій Віталійович, Брезицький Євген Олегович, Джуман Степан Васильович, Тучинський Андрій Вікторович, Масько Ольга Віталіївна, Брезицький Анна Дмитрівна, Аносова Катерина Євгенівна, Галкіна Євгенія Василівна, Багатько Христина Трохимівна, Лагода Ірина Олександрівна;
- телефон спеціаліста – тип даних VARCHAR, довжина значення 15, поле не може бути пустим;
- пошта спеціаліста – тип даних VARCHAR, довжина значення 40, поле не може бути пустим;
- спеціалізація спеціаліста – тип даних VARCHAR, довжина значення 40, поле не може бути пустим;
- логін спеціаліста – тип даних VARCHAR, довжина значення 20, поле не може бути пустим;
- пароль спеціаліста – тип даних VARCHAR, довжина значення 20, поле не може бути пустим.

Інформація про співвідношення між відділом, послугою, та спеціалістом, який спеціалізується в даній послугі зберігається в таблиці під назвою «Залежність» (relation). Реалізація таблиці за допомогою MySQL наведено в лістингу 4.4, логічна структура таблиці наведена в таблиці 4.4.

## Лістинг 4.4 – Реалізація таблиці «Залежність» (relation)

```

CREATE TABLE IF NOT EXISTS `relation` (
  `id_relation` int(11) NOT NULL AUTO_INCREMENT,
  `id_branch` int(11) NOT NULL,
  `id_service` int(11) NOT NULL,
  `id_doctor` int(11) NOT NULL,
  PRIMARY KEY (`id_relation`),
  KEY `id_relation` (`id_relation`),
  KEY `id_branch` (`id_branch`),
  KEY `id_service` (`id_service`),
  KEY `id_doctor` (`id_doctor`),
  CONSTRAINT `FK_relation_branch` FOREIGN KEY (`id_branch`)
REFERENCES `branch` (`id_branch`),
  CONSTRAINT `FK_relation_doctors` FOREIGN KEY (`id_doctor`)
REFERENCES `doctors` (`id_doctor`),
  CONSTRAINT `FK_relation_services` FOREIGN KEY (`id_service`)
REFERENCES `services` (`id_service`)
) ENGINE=InnoDB AUTO_INCREMENT=38 DEFAULT CHARSET=utf8
ROW_FORMAT=DYNAMIC;

```

Таблиця 4.4 - Структура сутності «Залежність» (relation)

| Атрибут     | Тип даних | Опис даних        |
|-------------|-----------|-------------------|
| id_relation | NUMBER    | Номер відношення  |
| id_branch   | NUMBER    | Номер відділу     |
| id_service  | NUMBER    | Номер послуги     |
| id_doctor   | NUMBER    | Номер спеціаліста |

Для даних ресурсів справедливі наступні правила:

- номер відношення– ціле число (1,2,...), AUTO\_INCREMENT, тип даних INT, довжина значення 11, поле не може бути пустим;
- номер відділу, номер послуги, номер спеціаліста– ціле число (1,2,...), тип даних INT, довжина значення 11, поле не може бути пустим.

Інформація про логін та пароль клієнта зберігається в таблиці «Деталі\_реєстрації» (login\_details). Реалізація таблиці за допомогою MySQL наведено в лістингу 4.5, логічна структура таблиці наведена в таблиці 4.5.

## Лістинг 4.5 – Реалізація таблиці «Деталі\_реєстрації» (login\_details)

```

CREATE TABLE IF NOT EXISTS `login_details` (
  `id_details` int(11) unsigned NOT NULL AUTO_INCREMENT,

```

```

`id_client` int(11) unsigned NOT NULL DEFAULT '0',
`login_client` varchar(40) NOT NULL,
`email_client` varchar(40) NOT NULL,
`password_client` varchar(255) NOT NULL,
PRIMARY KEY (`id_details`)
) ENGINE=InnoDB AUTO_INCREMENT=18 DEFAULT CHARSET=utf8;

```

Таблиця 4.5- Структура сутності «Деталі\_реєстрації» (login\_details)

| Атрибут         | Тип даних    | Опис даних     |
|-----------------|--------------|----------------|
| id_details      | INT(11)      | Номер деталей  |
| id_client       | INT(11)      | Номер клієнту  |
| login_client    | VARCHAR(40)  | Логін клієнту  |
| email_client    | VARCHAR(40)  | Пошта клієнту  |
| password_client | VARCHAR(255) | Пароль клієнта |

Для даних ресурсів справедливі наступні правила:

- номер деталей – ціле число (1,2,...), AUTO\_INCREMENT, тип даних INT, довжина значення 11, поле не може бути пустим;
- номер клієнту – ціле число (1,2,...), тип даних INT, довжина значення 11, поле не може бути пустим;
- логін клієнту – тип даних VARCHAR, довжина значення 40, поле не може бути пустим;
- пошта клієнту – тип даних VARCHAR, довжина значення 40, поле не може бути пустим;
- пароль клієнта – тип даних VARCHAR, довжина значення 255, поле не може бути пустим.

Всі сутності знаходяться в третій нормальній формі (3НФ), так як всі атрибути кожної сутності знаходяться в повній функціональній залежності від ключа, в відношення немає груп, які повторюються, немає транзитивної і часткової функціональної залежності.

## 4.2 Розробка інтерфейсу користувача

Однією з особливостей комп'ютерної системи косметологічної клініки є можливість онлайн реєстрації клієнта на прийом до спеціаліста. Щоб задовольнити потреби користувача будемо виносити до форми реєстрації наступні пункти:

- прізвище клієнта;
- ім'я клієнта;
- по батькові клієнта;
- телефон клієнта;
- дату запису на прийом;
- відділ до якого хоче звернутися клієнт;
- обрана послуга;
- спеціаліст;
- час запису на прийом;

Для того, щоб задати єдину форму запису та перевіряти правильність заповнення форми клієнтом використовуємо атрибут `pattern`. Для прізвища, ім'я та по батькові атрибут містить тільки українські літери в будь-якому регістрі, для цього використовуємо `pattern=" [А-Яа-яІіїЄє]`. Для телефону використовуємо визначений формат запису та лише цифри, наприклад, `pattern="0[0-9]{2}-[0-9]{3}-[0-9]{2}-[0-9]{2}"`. Неможливий запис на минулу дату, тому мінімальним значенням дати буде - сьогоднішня дата.

Форма запису на прийом наведена в додатку Б, лістинг Б.1.

Після введення прізвища, ім'я, по батькові, телефону та обрання дати прийому клієнт мусить обрати відділ до якого він хоче записатися на прийом. Назву відділу завантажуюмо з бази даних, а в системі відображається за допомогою поля `select`. Код для вибору відділу наведено в лістингу 4.6.

## Лістинг 4.6 – Код для вибору відділу

```

<?php
$servername = "localhost";
$username = "root";
$password = "2106";
$link = mysqli_connect($servername, $username, $password);
if(!$link){
echo "Ошибка: Невозможно установить соединение с MySQL." . PHP_EOL;
echo "Код ошибки errno: " . mysqli_connect_errno() . PHP_EOL;
echo "Текст ошибки error: " . mysqli_connect_error() . PHP_EOL;
exit;
}
mysqli_select_db($link, 'estelife_new');
mysqli_set_charset($link, "utf8");
$sql = 'SELECT relation.id_branch, branch.branch FROM
relation INNER JOIN branch ON relation.id_branch = branch.id_branch
GROUP BY branch';
$result = mysqli_query($link, $sql);
echo "<select name='start'
onchange='showSecondSelect(this.value);'>";
echo "<option value=''>Відділ:</option>";
for($i = 0; $i < mysqli_num_rows($result); $i++){
    $f = mysqli_fetch_array($result, MYSQLI_ASSOC);
echo "<option
value='\".$f['id_branch'].\"'>\".$f['branch'].\"</option>";
}
echo "</select>";
mysqli_free_result($result);
mysqli_close($link);
?>

```

В залежності від обраного відділу необхідно обрати послугу на яку клієнт бажає зареєструватися. Назви послуг завантажуються з бази даних, а в системі відображається за допомогою поля select. Код для вибору послуги наведено в лістингу 4.7.

## Лістинг 4.7 – Код для вибору послуги

```

<?php
$servername = "localhost";
$username = "root";
$password = "2106";
$q = $_GET['value'];
$link = mysqli_connect($servername, $username, $password);
if(!$link){
    echo "Ошибка: Невозможно установить соединение с MySQL."
. PHP_EOL;

```

```

        echo "Код ошибки errno: " . mysqli_connect_errno() .
PHP_EOL;
        echo "Текст ошибки error: " . mysqli_connect_error() .
PHP_EOL;
        exit;
    }
    mysqli_select_db($link, 'estelife_new');
    mysqli_set_charset($link, "utf8");
    $sql = "SELECT relation.id_service, services.service FROM
relation INNER JOIN services ON
relation.id_service=services.id_service WHERE id_branch = ' ".$q."'
GROUP BY `id_service`";
    $result = mysqli_query($link, $sql);
    echo "<select name='medium'
onchange='showThird(this.value);'>";
    echo "<option value=''>Послуга:</option>";
    for($i = 0; $i < mysqli_num_rows($result); $i++){
        $f = mysqli_fetch_array($result, MYSQLI_ASSOC);
        echo "<option
value='".$f['id_service']."'>".$f['service']."'</option>";
    }
    echo "</select>";
    mysqli_free_result($result);
    mysqli_close($link);
?>

```

Після обрання послуги клієнт мусить обради спеціаліста. Прізвища спеціалістів завантажуються з бази даних, а в системі відображається за допомогою поля select. Код для вибору спеціаліста наведено в лістингу 4.8.

#### Лістинг 4.8 – Код для вибору спеціаліста

```

<?php
    $servername = "localhost";
    $username = "root";
    $password = "2106";
    $q = $_GET['service'];
    $q1 = $_GET['branch'];
    $link = mysqli_connect($servername, $username, $password);
    if(!$link){
        echo "Ошибка: Невозможно установить соединение с MySQL."
    . PHP_EOL;
        echo "Код ошибки errno: " . mysqli_connect_errno() .
PHP_EOL;
        echo "Текст ошибки error: " . mysqli_connect_error() .
PHP_EOL;
        exit;
    }
    mysqli_select_db($link, 'estelife_new');

```

```

        mysqli_set_charset($link, "utf8");
        $sql = "SELECT relation.id_relation,
doctors.surname_doctor, doctors.first_name_doctor,
doctors.last_name_doctor FROM relation, doctors WHERE
relation.id_doctor = doctors.id_doctor AND relation.id_branch = '"
.$q1. "' AND relation.id_service = '". $q. "'";
        $result = mysqli_query($link, $sql);
        echo "<select name='medium'
onchange='showFour(this.value);'>";
        echo "<option value=''>Доктор:</option>";
        for($i = 0; $i < mysqli_num_rows($result); $i++){
            $f = mysqli_fetch_array($result, MYSQLI_ASSOC);
            echo "<option
value='". $f['id_relation']. "'>". $f['surname_doctor']. "
". $f['first_name_doctor']. " ". $f['last_name_doctor']. "</option>";
        }
        echo "</select>";
        mysqli_free_result($result);
        mysqli_close($link);
    ?>

```

Після обрання спеціаліста необхідно обрати час запису на прийому. Неможливо записатися на один й той самий час на однакові вибрані попередні пункти. Код для вибору часу запису наведено в лістингу 4.9.

#### Лістинг 4.9 – Код для вибору часу

```

<?php
    $servername = "localhost";
    $username = "root";
    $password = "2106";
    $record= $_GET['record'];
    $date = $_GET['date'];
    $link = mysqli_connect($servername, $username, $password);
    if(!$link){
        echo "Ошибка: Невозможно установить соединение с MySQL." . PHP_EOL;
        echo "Код ошибки errno: " . mysqli_connect_errno() . PHP_EOL;
        echo "Текст ошибки error: " . mysqli_connect_error() . PHP_EOL;
        exit;
    }
    mysqli_select_db($link, 'estelife_new');
    mysqli_set_charset($link, "utf8");
    $sql = "SELECT time.timeall FROM time,appointment WHERE
time.timeall NOT IN (SELECT appointment.time FROM appointment WHERE
appointment.id_relation = " . $record . " AND appointment.date = '" .
$date . "' GROUP BY appointment.id_relation) GROUP BY time.timeall";
    $result = mysqli_query($link, $sql);
    echo "<select name='end'>";
    echo "<option value=''>Час:</option>";

```

```

for($i = 0; $i < mysqli_num_rows($result); $i++){
    $f = mysqli_fetch_array($result, MYSQLI_ASSOC);
echo "<option value='\".$f['timeall'].\"'>\".$f['timeall'].\"
</option>\";
    }
echo "</select>\";
    mysqli_free_result($result);
    mysqli_close($link);
?>

```

Після заповнення форми та натискання клієнтом кнопки «Відправити» дані записуються до бази даних.

Адміністратор клініки мусить переглядати та контролювати записи клієнтів та якщо необхідно, то зателефонувати. Запити для відображення інформації клієнта для запису на прийом наведено в лістингу 4.10.

#### Лістинг 4.10 – Запити для відображення інформації

```

<?php
    $hostname = "localhost";
    $username = "root";
    $password = "2106";
    $databaseName = "estelife_new";
    $connect = mysqli_connect($hostname, $username, $password,
$databaseName);
    $query = "SELECT * FROM `appointment`";
    mysqli_set_charset($connect, "utf8");
    $result1 = mysqli_query($connect, $query);
    $sql = "SELECT appointment.id_relation, services.service FROM
appointment, relation, services, doctors WHERE
appointment.id_relation = relation.id_relation AND
relation.id_service = services.id_service GROUP BY
appointment.id_relation ";
    $result = mysqli_query($connect, $sql);
?>

```

Дані про записи для адміністратора будуть відображатися у вигляді таблиці. Реалізація таблиці наведено на лістингу 4.11.

#### Лістинг 4.11 – Таблиця для відображення інформації про записи на прийом

```

<tableid="table_admin" width="100%">
<thead>
<tr>
<th>№</th>

```

```

<th>Прізвище</th>
<th>І`мя</th>
<th>По батькові</th>
<th>Телефон</th>
<th>Дата прийому</th>
<th>Час прийому</th>
<th>Послуга</th>
</tr>
</thead>
<tbody>
<?php while($row1 = mysqli_fetch_array($result1) and $row2 =
mysqli_fetch_array($result)) :;>?>
<tr>
<td><?php echo $row1[0];?></td>
<td><?php echo $row1[2];?></td>
<td><?php echo $row1[3];?></td>
<td><?php echo $row1[4];?></td>
<td><?php echo $row1[5];?></td>
<td><?php echo $row1[6];?></td>
<td><?php echo $row1[7];?></td>
<td><?php echo $row2[1];?></td>
</tr>
<?php endwhile;?>
</tbody>
</table>

```

Наступною головною особливістю комп'ютерної системи є можливість реєстрації та входу до особистого кабінету.

В кодї для форми входу розроблено зв'язок с базою даних, форма, та перевірки, які будуть попереджати про неправильність введення логіну чи паролю. Код для форми входу до особистого кабінету наведено в додатку Б, Лістинг Б.2.

В кодї для форми реєстрації розроблено зв'язок з базою даних, форма, та перевірки, які будуть попереджати про неправильність заповнення даних, про неспівпадання паролів, та про вже наявність клієнта з таким логіном або email.

Пароль клієнта зберігається в базі в зашифрованому вигляді, за допомогою хеш-функції.

Функція хешування (хеш-функція) – це перетворення даних, яке переводить рядок бітів довільної довжини в рядок бітів деякої фіксованої довжини. Фіксована довжина може складати кілька десятків чи сотень біт.

Слово хешування походить від англійського дієслова “to hash” – кришити, рубати, плутати. Результат хешування називають хеш-образом, хеш-кодом або message digest.

Хеш-функція  $h(M)$  повинна задовольняти наступним умовам:

- хеш-функція  $h(M)$  повинна бути чутливою до будь-яких змін вхідної послідовності  $M$ ;
- для даного значення  $h(M)$  повинно бути неможливо знайти значення  $M$ .

Ситуація називається колізією, за якої для різних вхідних послідовностей  $M$ ,  $M'$  співпадають значення їхніх хеш-образів:

$$h(M) = h(M') \quad (4.1)$$

При побудові хеш-образу, вхідна послідовність  $M$  розбивається на блоки фіксованої довжини і оброблюється поблочно за формулою

$$H_{i+1} = f(H_i, M_i), \quad (4.2)$$

де  $H_1$  – довільний початковий вміст;

$f(u, v)$  – задана функція.

Хеш-значення, що обчислюється при введенні останнього блоку повідомлення, стає хеш-кодом  $h(M)$  всього повідомлення.

Код для форми реєстрації до особистого кабінету наведено в додатку Б, лістинг Б.3.

### 4.3 Тестування системи

В попередньому розділі було розроблено модуль для реєстрації на прийом до спеціаліста та розроблено форми для входу та реєстрації до особистого кабінету. Необхідно перевірити та протестувати працездатність модуля та форм.

Основний користувач модуля – це клієнт, отже, перевірка працездатності буде проводитися в основному зі сторони клієнта. Початковий вигляд форми запису на прийом до спеціаліста наведено на рисунку 4.1.

**Запис на прийом**

|              |                                                         |
|--------------|---------------------------------------------------------|
| Прізвище:    | <input type="text" value="Прізвище"/>                   |
| Ім'я:        | <input type="text" value="Ім'я"/>                       |
| По батькові: | <input type="text" value="По батькові"/>                |
| Телефон:     | <input type="text" value="Ваш телефон: 0xx-xxx-xx-xx"/> |
| Дата:        | <input type="text" value="дд.мм.гггг"/>                 |
| Відділ:      | <input type="text" value=""/>                           |

Рисунок 4.1 – Початковий вигляд форми запису на прийом до спеціаліста

Наступний етап – це перевірка правильності, коректності введення даних в форму, та обрання відділу клініки. Прізвища, ім'я, по батькові повинні бути введені на українській мові будь-якого регістру, телефон вводить в заданому форматі.

Наступні етапи - це обрання послуги, спеціаліста та часу запису на прийом.

Після натискання кнопки «Відправити» дані повинні записатися в таблицю «Прийом до лікаря» (appointment) та «Клієнти» (clients) відповідно. Перевірка запису даних до таблиці наведено на рисунку 4.2 та на рисунку 4.3.

estelife.appointment: 2 строк (приблизительно)

|  id_appointment |  id_relation |  id_client | date       | time  |
|--------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------|------------|-------|
| 1                                                                                                | 37                                                                                            | 1                                                                                           | 2019-04-21 | 09:00 |
| 2                                                                                                | 9                                                                                             | 2                                                                                           | 2019-04-21 | 11:30 |

Рисунок 4.2 – Запис даних до бази даних, таблиця «Прийом до лікаря» (appointment)

|  id_client | surname_client | first_name_client | last_name_client | gender_client | phone_client  | email_client |
|---------------------------------------------------------------------------------------------|----------------|-------------------|------------------|---------------|---------------|--------------|
| 1                                                                                           | Іванов         | Іван              | Іванович         | (NULL)        | 050-147-78-96 | (NULL)       |

Рисунок 4.3 – Запис даних до бази даних, таблиця «Клієнти» (clients)

Адміністратор клініки може переглядати усі записи на прийом, фільтрувати за датою, алфавітом та шукати за прізвищем.

Для наочної перевірки працездатності форм розроблено різноманітні перевірки на коректність введення даних. Для початку перевіримо форму реєстрації до особистого кабінету.

Неможливо зареєструвати, якщо логін або email вже використовується іншим користувачем. При правильному введенні даних реєстрація можлива.

## ВИСНОВКИ

В ході дипломної роботи були проведені всі етапи розробки комп'ютерної системи косметологічної клініки з можливістю онлайн запису на прийом.

При аналізі технічного завдання проведено опис косметологічної клініки, порівняльний аналіз аналогів, визначено вимоги до системи. Основні задачі системи:

- надання інформації про клініку;
- автоматизація запису на прийом до спеціаліста клініки лазерної медицини та косметології;
- спрощення введення даних;
- підвищення якості (повноти, точності, достовірності, своєчасності, узгодженості) інформації;
- забезпечити збереження даних та інформації.

Основні вимоги до задач системи:

- кросплатформеність;
- адаптивність;
- кросбраузерність;
- зручний та інтуїтивний інтерфейс;
- зручна система навігації;
- дані системи повинні зберігатися в базі даних.

При моделюванні програмного забезпечення було виділено основних акторів, сценаріїв та діаграм послідовностей.

Обрано для системи архітектуру «Клієнт-сервер», проведено проектування бази даних, при цьому були виділено сутності та отримано ER-модель бази даних. При проектуванні інтерфейсу користувача розроблено макети сторінок. Проведено обґрунтування вибору системи керування базами даних та технологій для реалізації інтерфейсу користувача.

При розробці комп'ютерної системи було використано наступні технології:

HTML, CSS, JavaScript, PHP, фреймворк Bootstarp, Web-сервер Apache, Ajax. База даних для системи реалізовано за допомогою MySQL.

Отже, розроблена комп'ютерна система може використовуватися підприємством та бути впровадженою до мережі Інтернет. За допомогою розробленої системи можливо покращити імідж компанії, збільшити обсяг продажів послуг та залучити більше потенційних клієнтів.

## ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Інформаційні системи та технології. URL: [https://stud.com.ua/94115/informatika/informatsiyni\\_sistemi\\_tehnologiyi](https://stud.com.ua/94115/informatika/informatsiyni_sistemi_tehnologiyi) (дата звернення 12.04.2021).
2. Функции информационных систем. URL: <https://neudov.net/4students/otvety-po-pive/funkcii-informacionnyx-sistem/> (дата звернення 12.04.2021).
3. «Діаграма прецедентів» [електронний ресурс] – Електронні дані. – Режим доступу: <http://fitm.nusta.edu.ua/mediawiki/> (дата звернення 12.04.2021) – Назва з екрану.
4. Диаграмма сценариев использования в процессе разработки ПО [електронний ресурс] – Електронні дані. – Режим доступу: <https://m.habr.com> (дата звернення 26.11.2021) – Назва з екрану.
5. «Архітектура програмного забезпечення» [електронний ресурс] – Електронні дані. – Режим доступу: <https://kpi.stu.cn.ua/arhitektura-programnogo-zabezpechenn/> (дата звернення 12.04.2021) – Назва з екрану.
6. «Архітектура клієнт-сервер. Сховища даних. Реляційні бази даних як основа сховищ даних» [електронний ресурс] – Електронні дані. – Режим доступу: <http://educational.mariroz.com/> (дата звернення 12.04.2021) – Назва з екрану.
7. «Архітектура клієнт – сервер» [електронний ресурс] – Електронні дані. – Режим доступу: <https://homework.net.ua/arkhitektura-klient-server/> (дата звернення 12.04.2021) – Назва з екрану.
8. «Понятие субд, основные функции субд.» [електронний ресурс] – Електронні дані. – Режим доступу: <https://studfiles.net/preview/6055065/page:2/> (дата звернення 12.04.2021) – Назва з екрану.

9. «Вибір СУБД для розробки проекту» [електронний ресурс] – Електронні дані. – Режим доступу: <https://mydocx.ru/11-104231.html> (дата звернення 12.04.2021) – Назва з екрану.

10. «Сравнительные характеристики СУБД» [електронний ресурс] – Електронні дані. – Режим доступу: [http://orastack.narod.ru/index\\_file/onebd.html](http://orastack.narod.ru/index_file/onebd.html) (дата звернення 12.04.2021) – Назва з екрану.

11. «Що таке MySQL?» [електронний ресурс] – Електронні дані. – Режим доступу: <https://uk.photo-555.com/5876083-what-is-mysql> (дата звернення 12.04.2021) – Назва з екрану.

12. «Стандартные утилиты в MySQL» [електронний ресурс] – Електронні дані. – Режим доступу: [https://kbss.ru/blog/bd\\_mysql/19.html](https://kbss.ru/blog/bd_mysql/19.html) (дата звернення 12.04.2021) – Назва з екрану.

13. «Автономная база данных Oracle: что она умеет делать для руководителей, аналитиков и DBA» [електронний ресурс] – Електронні дані. – Режим доступу: <https://channel4it.com/publications/Avtonomnaya-baza-dannyh-Oracle-chto-ona-umeet-delat-dlya-rukovoditeley-analitikov-i-DBA-30952.html> (дата звернення 12.04.2021) – Назва з екрану.

14. «Зовнішня структура» [електронний ресурс] – Електронні дані. – Режим доступу: <https://studfiles.net/preview/7037223/page:2/> (дата звернення 12.04.2021) – Назва з екрану.

15. «Створення дизайну сайту для інтернет-представництва» [електронний ресурс] – Електронні дані. – Режим доступу: <http://webstudio2u.net/ua/design-site/668-dizain-internet-predstavitelstva.html> (дата звернення 12.04.2021) – Назва з екрану.

16. «Сучасні веб-технології» [електронний ресурс] – Електронні дані. – Режим доступу: <https://sites.google.com/site/vivcaemowebdizajndistancijno/html/lekcia-2-sucasni-veb-tehnologiie> (дата звернення 15.04.2021) – Назва з екрану.

17. «Bootstrap - HTML- и CSS-шаблоны оформления и JavaScript-расширения. основы использования» [електронний ресурс] – Електронні дані. – Режим доступу: <https://intellect.icu/bootstrap-html-i-css-shablony-oformleniya-i->

javascript-rasshireniya-osnovy-ispolzovaniya-6826 (дата звернення 15.04.2021) – Назва з екрану.

18. «Що таке PHP» [електронний ресурс] – Електронні дані. – Режим доступу: <http://www.lib.mdpu.org.ua/e-book/web/lec1.htm>(дата звернення 15.04.2021) – Назва з екрану.

19. «Фреймворки в веб-розробці» [електронний ресурс] – Електронні дані. – Режим доступу: [https://web-creator.ru/articles/about\\_frameworks](https://web-creator.ru/articles/about_frameworks) (дата звернення 15.04.2021) – Назва з екрану.

20. «Apache-HTTP сервер» [електронний ресурс] – Електронні дані. – Режим доступу: [https://studbooks.net/2272039/informatika/apache\\_http\\_server](https://studbooks.net/2272039/informatika/apache_http_server) (дата звернення 15.04.2021) – Назва з екрану.

21. «Загальний огляд процедур проектування баз даних» [електронний ресурс] – Електронні дані. – Режим доступу: <http://shpora.me/Peter/12321> (дата звернення 15.04.2021) – Назва з екрану.

22. «Разработка концептуальной и логической схемы при создании базы данных» [електронний ресурс] – Електронні дані. – Режим доступу: <https://novainfo.ru/article/14280> (дата звернення 12.04.2021) – Назва з екрану.

23. «Модель сущность-атрибут-связь (er)» [електронний ресурс] – Електронні дані. – Режим доступу: <http://csaa.ru/model-sushhnost-atribut-svjaz-er/> (дата звернення 12.04.2021) – Назва з екрану.

## ДОДАТОК А

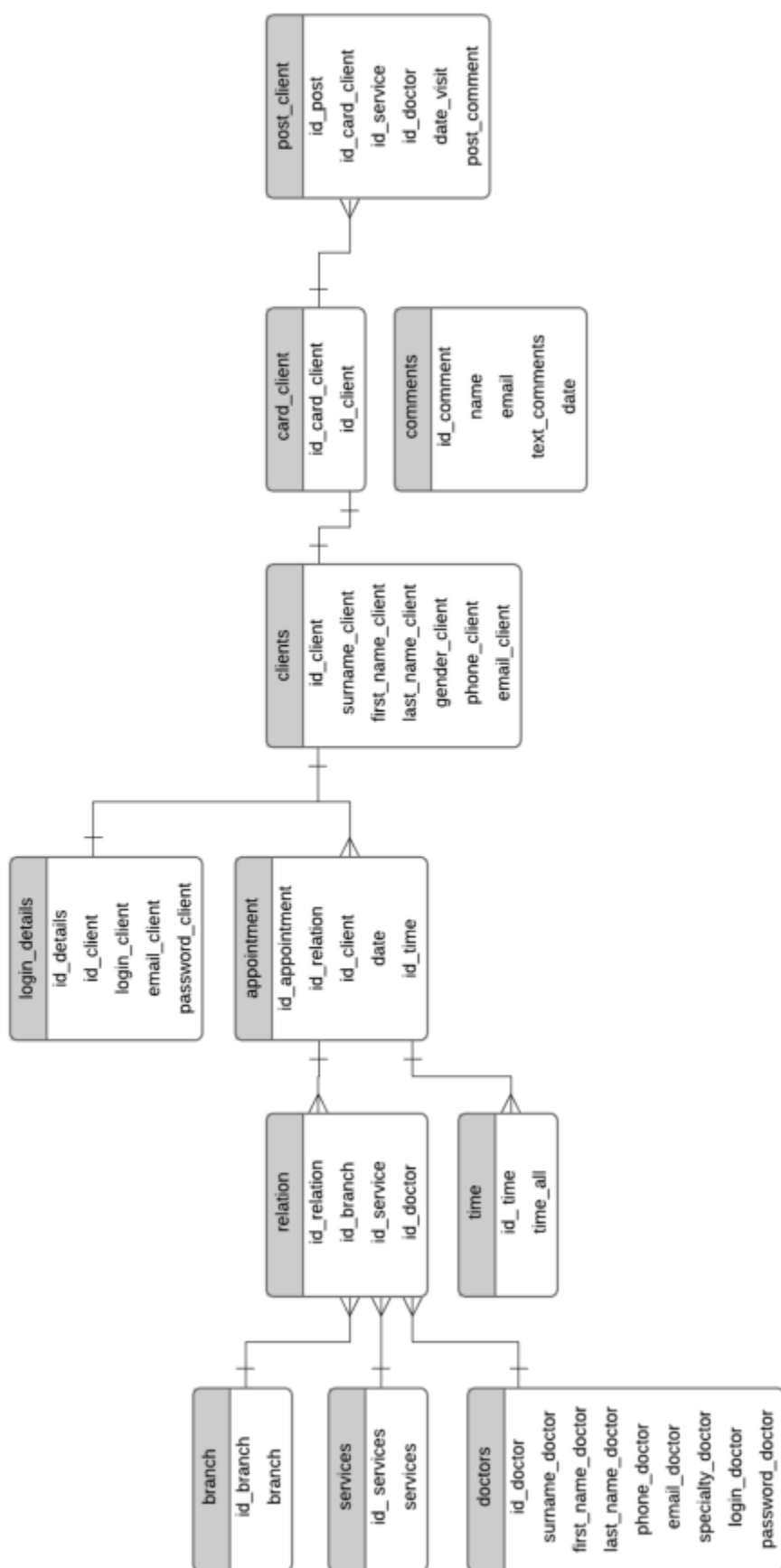


Рисунок А.1 - ER-модель бази даних

## ДОДАТОК Б

## Лістинг Б.1 – Форма запису на прийом

```

<section class="salon-book-an-record-area section_padding_100_0">
<div class="container">
<div class="row align-items-center">
<div class="col-12 col-lg-6">
<div class="record-form">
<h3>Запис на прийом</h3>
<form data-toggle="validator" name="forma" role="form">
<div class="form-group has-feedback">
<div class="input-group">
<span class="input-group-addon">Ім'я:</span>
<input type="text" class="form-control" id="record-name_client"
pattern="[А-Яа-яІіїіЄє ]+" required name="name_client"
placeholder="Ваше ім'я" />
</div>
<span class="glyphicon form-control-feedback" aria-
hidden="true"></span>
</div>
<div class="form-group has-feedback">
<div class="input-group">
<span class="input-group-addon">Телефон:</span>
<input type="tel" class="form-control" required name="phone_client"
id="record-phone_client" pattern="0[0-9]{2}-[0-9]{3}-[0-9]{2}-[0-
9]{2}" placeholder="Ваш телефон: 0xx-xxx-xx-xx"/>
</div>
<span class="glyphicon form-control-feedback" aria-
hidden="true"></span>
</div>
<div class="form-group has-feedback">
<div class="input-group">
<span class="input-group-addon">Дата:</span>
<input type="date" class="col-md-12 col-xs-12 record-date1"
id="record-date1" required name="date1" min="2019-04-20"
placeholder="Дата" >
</div>
<span class="glyphicon form-control-feedback" aria-
hidden="true"></span>
</div>
</form>
<div class="form-group has-feedback">
<div class="input-group">
<form id="InformationHint" name="secession" record-
secession"></form>
</div>
<span class="glyphicon form-control-feedback" aria-
hidden="true"></span>
</div>
<div class="form-group has-feedback">
<div class="input-group">

```

```
<form id="SecondSelect" name="services" record-services"></form>
</div>
<span class="glyphicon form-control-feedback" aria-
hidden="true"></span>
</div>
<div class="form-group has-feedback">
<div class="input-group">
<form id="ThirdSelect" name="specialist" record-specialist"></form>
</div>
<span class="glyphicon form-control-feedback" aria-
hidden="true"></span>
</div>
<div class="form-group has-feedback">
<div class="input-group">
<form id="FourSelect" name="time" record-specialist ></form>
</div>
<span class="glyphicon form-control-feedback" aria-
hidden="true"></span>
</div>
<button id="btn-submit" class="col-md-3 col-xs-4 col-md-offset-9
col-xs-offset-8record-submit">Отправить</button>
<script src="js/submit_index.js"></script>
</div>
</div>
<div class="col-12 col-lg-6">
<div class="salon-record-thumb">

</div>
</div>
</div>
</div>
</section>
```

```

        echo '<div style="color: green "
id="errors">Ви увійшли до особистого кабінету. Вітаємо!</div>';
    }
    else {
        $errors[] = 'Помилка. Перевірте введені
дані.';
    }
}
else{
    $errors[] = 'Помилка. Перевірте введені дані.';
}
if(!empty($errors)){
    echo '<div style="color: red"
id="errors">'.array_shift($errors).'</div>';
}
}
?>
<form action="login.php" method="POST" class="form-panel
one">

    <div class="form-header">
        <h1>Від до особистого кабінету</h1>
    </div>
    <div class="form-content">
        <div class="form-group">
            <label for="username">Логін</label>
            <input type="text" name="login" value="<?php
echo @$data['login'] ?>">
        </div>
        <div class="form-group">
            <label for="password">Пароль</label>
            <input type="password" name="password"
value="<?php echo @$data['password']?>">
        </div>
        <div class="form-group">
            <label class="form-remember">
                <input type="checkbox"/>Запам'ятати
мене
            </label>
            <a class="form-recovery" href="#">Забули
пароль?</a>
        </div>
        <div class="form-group">
            <button type="submit"
name="do_login">Ok</button>
        </div>
    </div>
</form>
</div>
</body>
</html>

```

## Лістинг Б.3 – Форма для реєстрації до особистого кабінету

```

<!DOCTYPE html>
<html>
<head>
    <link rel="stylesheet" href="login.css">
</head>
<body>
    <div class="form">
        <?php require "db.php";
        $data = $_POST;
        if(isset($data['do_singup'])) {
            $errors = array();
            if ( trim($data['login']) == '' ) {
                $errors[] = 'Некоректний логін';
            }
            if ( trim($data['email']) == '' ) {
                $errors[] = 'Перевірте правильність введення
email!';
            }
            if ( ($data['password']) == '' ) {
                $errors[] = 'Некоректний пароль!';
            }
            if ( ($data['password_2']) != $data['password']) {
                $errors[] = 'Перевірте правильність введення
повторного паролю!';
            }
            if ( R::count('clientdetails', "login_client = ?",
array($data['login']) ) > 0 ) {
                $errors[] = 'Логін з таким імям вже існує.
Задайте інший логін.';
            }
            if ( R::count('clientdetails', "email_client = ?",
array($data['email'])) > 0 ) {
                $errors[] = 'Кабінет з такою поштою вже існує.
Задайте інший email.';
            }
            if(empty($errors)) {
                $details = R::dispense('clientdetails');
                $details->login_client = $data['login'];
                $details->email_client = $data['email'];
                $details->password_client =
password_hash($data['password'], PASSWORD_DEFAULT);
                R::store($details);
                echo '<div style="color: green " id="errors">Ви
zareestrovani do osobistogo kabinetu. Vitamo!!</div>';
            }
            else {
                echo '<div style="color: red"
id="errors">'.array_shift($errors).'</div>';
            }
        }
    ?>

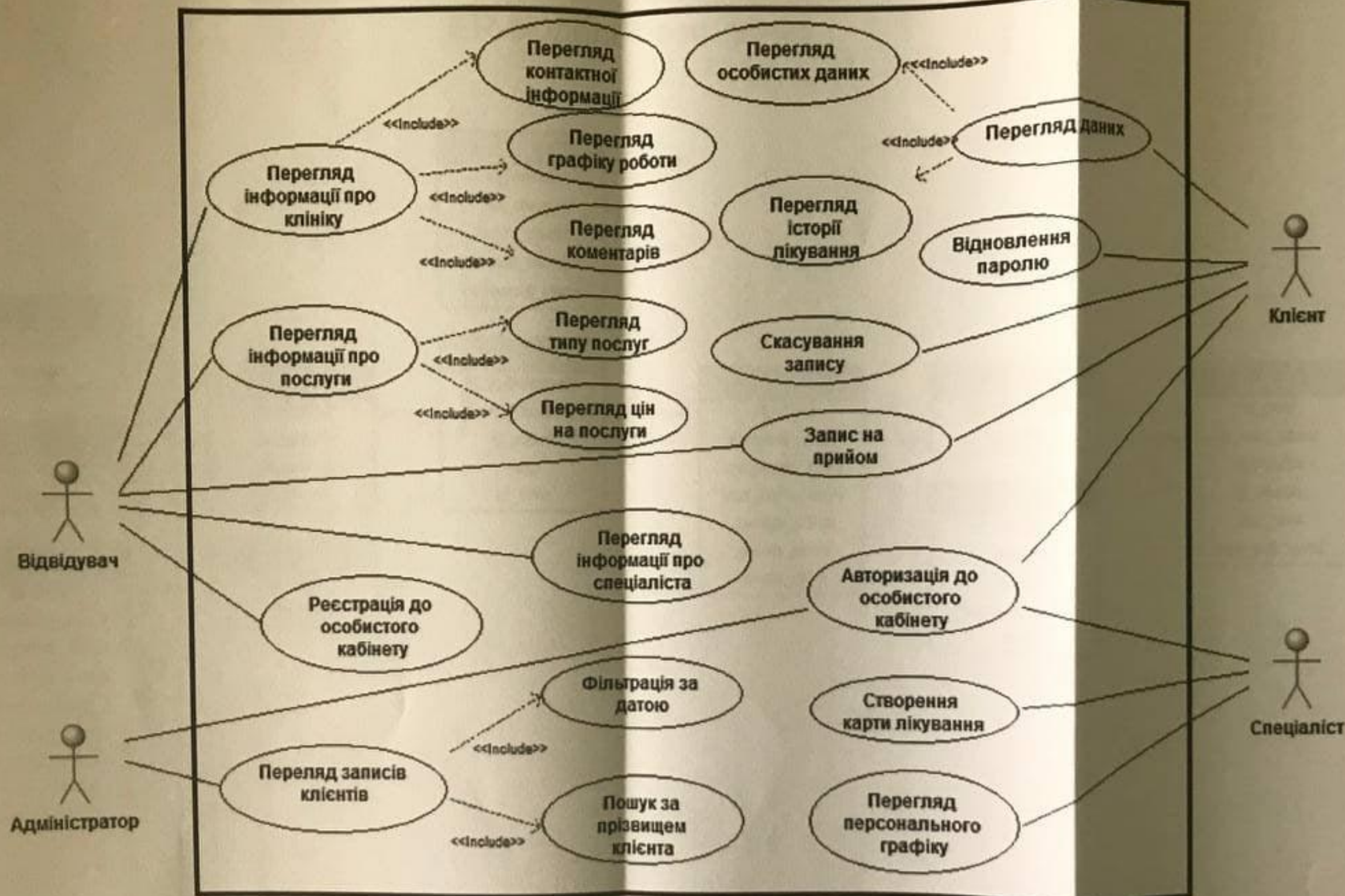
```

```

<form action="singup.php" method="POST" class="form-panel one">
  <div class="form-header">
    <h1>Реєстрація до особистого кабінету</h1>
  </div>
  <div class="form-content">
    <div class="form-group">
      <p><strong>Імя</strong></p>
      <input type="text" name="name_client" value="<?php
echo @$data['login'] ?>">
    </div>
    <div class="form-group">
      <p><strong>Телефон</strong></p>
      <input type="text" name="telefon" value="<?php echo
@$data['login'] ?>">
    </div>
    <div class="form-group">
      <p><strong>Login</strong></p>
      <input type="text" name="login" value="<?php echo
@$data['login'] ?>">
    </div>
    <div class="form-group">
      <p><strong>Email</strong></p>
      <input type="email" name="email" value="<?php echo
@$data['login']?>">
    </div>
    <div class="form-group">
      <p><strong>Password</strong></p>
      <input type="password" name="password" value="<?php
echo @$data['password']?>">
    </div>
    <div class="form-group">
      <p><strong>Password</strong></p>
      <input type="password" name="password_2" value="<?php
echo @$data['password_2']?>">
    </div>
    <div class="form-group">
      <button type="submit"
name="do_singup">Реєстрація</button>
    </div>
  </div>
</form>
</div>
</body>
</html>

```

# Діаграма прецедентів



| Зм.       | Лист | № документа     | Підпис             | Дата |
|-----------|------|-----------------|--------------------|------|
| Розробив  |      | Бухар О.Р.      | <i>[Signature]</i> |      |
| Перевірив |      | Кудерметов Р.К. | <i>[Signature]</i> |      |
| Н. контр. |      | Щербак Н.В.     | <i>[Signature]</i> |      |
| Затвердив |      | Кудерметов Р.К. | <i>[Signature]</i> |      |

13.02070849.00025 ПЛ1

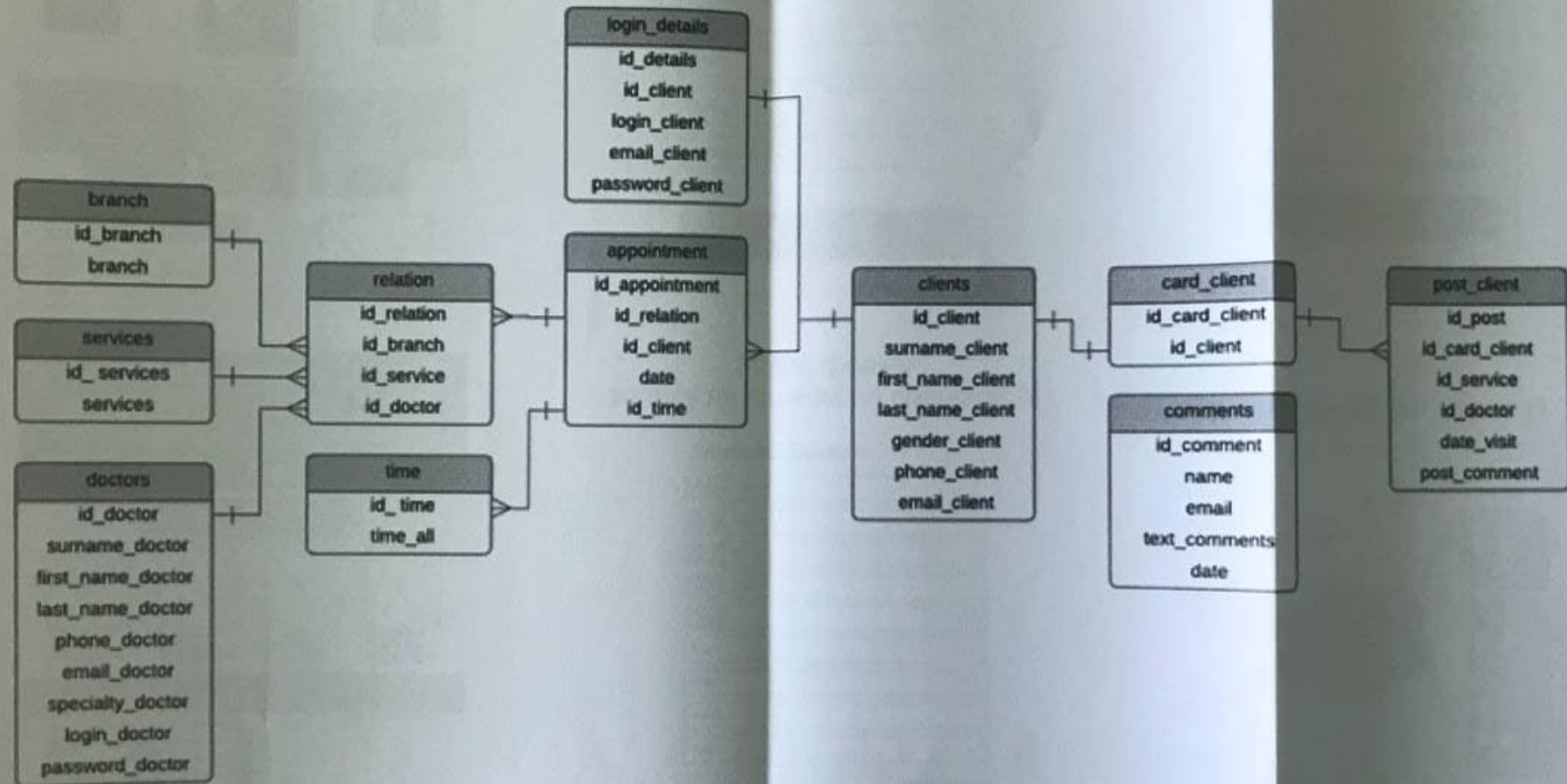
Розробка комп'ютерної системи косметологічної клініки

Діаграма прецедентів

| Літ. | Лист | Листів |
|------|------|--------|
|      | 1    | 1      |

НУ «Запорізька політехніка», група КНТ- 527

# ER-модель бази даних



| Зм.       | Лист            | № документа | Підпис | Дата |
|-----------|-----------------|-------------|--------|------|
| Розробив  | Бухар О.Р.      |             |        |      |
| Перевірив | Кудерметов Р.К. |             |        |      |
| Н. контр. | Цербах Н.В.     |             |        |      |
| Затвердив | Кудерметов Р.К. |             |        |      |

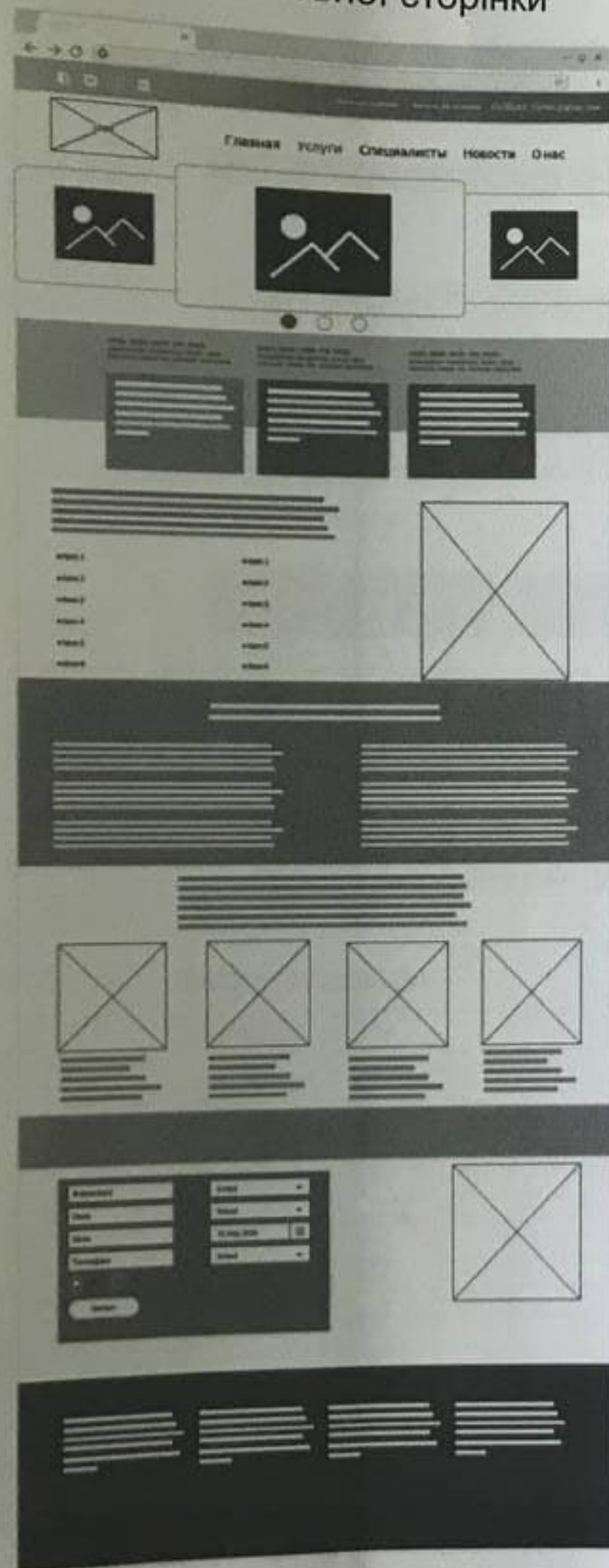
13.02070849.00025 ПЛ2

Розробка комп'ютерної системи косметологічної клініки

ER-модель бази даних

| Літ.                                        | Лист | Листів |
|---------------------------------------------|------|--------|
|                                             | 1    | 1      |
| НУ «Запорізька політехніка», група КНТ- 527 |      |        |

Макет головної сторінки



Макет форми реєстрації до особистого кабінету

**Реєстрація до особистого кабінету**

Логін

Email

Пароль

Повторний пароль

Макет форми входу до особистого кабінету

**Вхід до особистого кабінету**

логін

пароль

☒ Залим'ятати мене ☐ Забути пароль?

Макет форми запису на прийом до спеціаліста

**Запис на прийом**

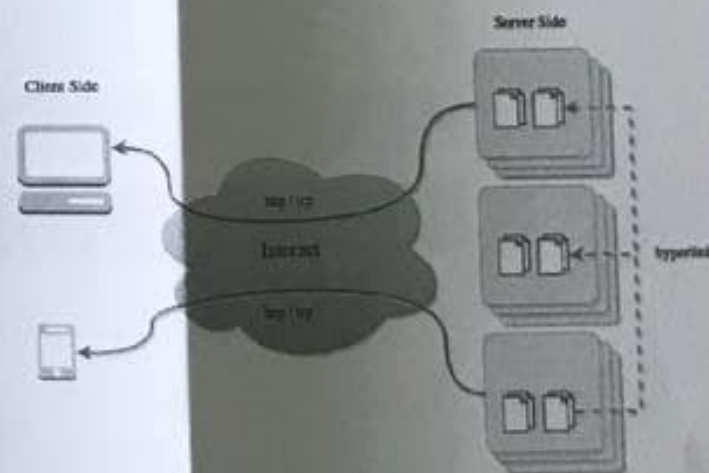
Прізвище

Ім'я

По батькові

Телефон

Архітектура «Клієнт-сервер»



| Зм.       | Лист | № документа     | Підпис             | Дата |
|-----------|------|-----------------|--------------------|------|
| Розробив  |      | Бухар О.Р.      | <i>[Signature]</i> |      |
| Перевірив |      | Кудерметов Р.К. | <i>[Signature]</i> |      |
| Н. контр. |      | Щербак Н.В.     | <i>[Signature]</i> |      |
| Затвердив |      | Кудерметов Р.К. | <i>[Signature]</i> |      |

13.02070849.00025 ПЛЗ

Розробка комп'ютерної системи космітологічної клініки  
Макети головної сторінки та форм, архітектура системи

| Літ.                                        | Лист | Листів |
|---------------------------------------------|------|--------|
|                                             | 1    | 1      |
| НУ «Запорізька політехніка», група КНТ- 527 |      |        |

Форма  
входу до особистого кабінету

ВІД ДО ОСОБИСТОГО КАБІНЕТУ

ЛОГІН

ПАРОЛЬ

☐ Запам'ятати мене

[Забули пароль?](#)

Форма  
запису на прийом до спеціаліста

Запис на прийом

|              |                                                         |
|--------------|---------------------------------------------------------|
| Прізвище:    | <input type="text"/>                                    |
| ІМЯ:         | <input type="text"/>                                    |
| По батькові: | <input type="text"/>                                    |
| Телефон:     | <input type="text" value="Ваш телефон: 000-xxx-xx-xx"/> |
| Дата:        | <input type="text" value="ДД.ММ.ГГГГ"/>                 |
| Відділ:      | <input type="text"/>                                    |

ВІДПРАВИТИ

Форма  
реєстрації до особистого кабінету

РЕЄСТРАЦІЯ ДО ОСОБИСТОГО  
КАБІНЕТУ

Login

Email

Password

Password



| Зм.       | Лист | № документа     | Підпис | Дата |
|-----------|------|-----------------|--------|------|
| Розробив  |      | Бухар О.Р.      |        |      |
| Перевіряв |      | Кудерметов Р.К. |        |      |
| Н. контр. |      | Щербак Н.В.     |        |      |
| Затвердив |      | Кудерметов Р.К. |        |      |

13.02070849.00025 ПЛ4

Розробка комп'ютерної  
системи космітологічної клініки

Головний фахівець системи

| Літ.                                           | Лист | Листів |
|------------------------------------------------|------|--------|
|                                                | 1    | 1      |
| НУ «Запорізька політехніка»,<br>група КНТ- 527 |      |        |