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

**Кафедра цифрової економіки та бізнес- аналітики**

# КУРСОВА РОБОТА з навчальної дисципліни “Проектування та адміністрування БД і СД”

**Тема:**

**«Інформаційна система для нотаріальної контори»**

**Науковий керівник: Виконавець:**

к.ф.-м.н., доц. Депутат Б.Я.Вовк О.М.

(прізвище, ім’я, по-батькові) (прізвище, ім’я, по-батькові)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (підпис) УФЕ-31с група

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (підпис)

**“\_\_\_” \_\_\_\_\_\_\_\_\_\_\_\_\_ 2023 р. “\_\_\_” \_\_\_\_\_\_\_\_\_\_\_\_ 2023 р.**

**Загальна кількість балів \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(підписи, ПІП членів комісії)

**Львів 2023**

# ЗМІСТ

# ПЛАН

**ВСТУП………………………………………………………………………………3**

**РОЗДІЛ 1. АНАЛІЗ ВИМОГ………….…………………………………………5**

1.1 Постановка завдання………………………………………………………..5

1.2 Розробка моделі варіантів використання веб-сайту…………………...…6

1.3 Логічна структура сайту…………………………………………………..11

1.4 Аналіз засобів реалізації (техніко-економічне обґрунтування вибору)…………………………………………………………………………12

**РОЗДІЛ 2. РОЗРОБКА БАЗИ ДАНИХ……………………………………..14**

2.1 Опис моделі даних………………………………………………………...14

2.2 Нормалізація відношень…………………………………………………..16

2.3 Визначення типів даних…………………………………………………..17

2.4 Обмеження цілісності даних……………………………………………...19

2.5 Реалізація SQL-скрипту…………………………………………………..21

**Розділ 3. РОЗРОБКА ВЕБ-ДОДАТКУ…………………………………….27**

3.1. Структура веб-сайту…………………………………………………..….27

3.2. Макети сторінок веб-сайту………………………………………………29

3.3. Програмування серверної частини………………………………………31

3.4 Програмування клієнтської частини……………………………………..33

3.4. Розміщення веб-сайту на локальному віртуальному середовищі або в Інтернеті……………………………………………………………………..…36

**ВИСНОВКИ…………………………………………………………………….38  
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ……………………………………39**

**ВСТУП**

**Актуальність теми дослідження**. Тема дослідження "Нотаріуси в Україні" є вкрай актуальною, оскільки нотаріальна діяльність є невід'ємною частиною правової системи України та має важливе значення для захисту прав та інтересів громадян.

На сьогоднішній день в Україні діють більше 3 тис. нотаріусів, які здійснюють широкий спектр нотаріальних дій, таких як складання договорів, засвідчення підписів, проведення спадкових справ, оформлення прав на нерухомість та багато іншого. Зростаюча кількість нотаріусів та їх діяльність потребують постійного аналізу та вдосконалення.

Враховуючи що ми живемо в 21 столітті, власний веб-сайт для нотаріуса є хорошим, а в майбутньому й не від’ємним атрибутом.

**Мета і завдання дослідження**. **Метою курсової** робити є дослідження та аналіз актуальності інформаційної системи для нотаріальних контор в Україні.

Для досягнення мети в роботі поставлено й вирішено такі теоретичні та практичні **завдання**:

* визначити особливості створення веб сайту для нотаріальної контори в Україні;
* розробити інформаційну систему(веб-сайт), що дає змогу переглянути послуги даного нотаріуса та онлайн записатись;
* перевірити програмний продукт на наявність дефектів та знайти шляхи покращення.

**Об’єктом дослідження** виступає конкретна нотаріальна контора, для якого створюється інформаційна система та процес розробки відповідного програмного забезпечення.

**Предметом** є теоретичні, методичні та практичні аспекти розробки програмного забезпечення мовою розмітки: html5, css3; мовою програмування: Php, Js, мовою запитів MySql для створення офіційного веб- сайту салону.

**Практичне значення отриманих результатів**. Програма може використовуватись юридичною особою, яка хоче розмістити інформацію про своє підприємство в мережі інтернет з метою залучення клієнтів.

**Використане програмне забезпечення**. Для створення програмного продукту використовувалось середовище розробки Microsoft Visual Studio, SublimeText3, локальний сервер – XAMPP та певні мови програмування.

**Структура роботи**. Курсова робота складається з трьох розділів («Аналіз вимог», «Розробка бази даних» та «Розробка веб-додатку»), висновків, списку використаних джерел та додатків.

Загальний обсяг роботи − **39** сторінок.

# РОЗДІЛ 1. Аналіз вимог

* 1. **Постановка завдання**

Окремим аспектом розвитку сучасного бізнесу є впровадження інформаційних систем та веб-сайтів для поліпшення управлінської діяльності. В рамках нотаріальної контори це також має велике значення. Завданням даної роботи є проведення аналізу актуальності створення інформаційної системи для нотаріуса в Україні. В разі позитивного результату такий вебсайт повинен бути розроблений спеціально для даної нотаріальної контори.

Одним з ключових аспектів управлінської інформації є економічна інформація, яка визначає науково-технічну і економічну доцільність використання комп'ютерної техніки. Інформаційні системи виконують важливу функцію збору, зберігання, обробки, пошуку та передачі економічної інформації, необхідної для прийняття рішень керівництвом. Оскільки економічна інформація має специфічні особливості, такі як великі обсяги, повторюваність циклів отримання та обробки, різноманітні джерела та споживачі, а також використання логічних операцій при обробці, важливо мати належну інформаційну систему.

Інформаційна система (ІС) для нотаріальної контори є системою, що забезпечує інформаційне обслуговування керівництва. Вона формується та функціонує відповідно до методів та структури управлінської діяльності, що специфічні для даної нотаріальної контори, та виконує поставлені перед нею цілі та завдання. Метою ІС є опис економічного об'єкта, його стану та взаємодії через економічні показники. ІС має за завдання надавати керівництву своєчасну та достатню інформацію для прийняття ефективних рішень.

Функціональна структура ІС повинна відповідати інформаційним потребам користувачів, які змінюються залежно від їх ролі та відповідальності в нотаріальній конторі. У склад ІС можуть входити різні компоненти, такі як системи збору, зберігання та обробки даних, системи забезпечення безпеки інформації, системи звітності та аналізу, системи комунікації та інтерфейсу користувача. Однак, при розробці інформаційної системи для нотаріальної контори необхідно враховувати особливості нотаріальної діяльності, такі як обробка документів, аутентифікація сторін, зберігання електронних записів тощо. Всі ці аспекти мають бути уважно враховані під час розробки інформаційної системи, щоб забезпечити її оптимальну функціональність та відповідність потребам нотаріальної контори.

* 1. **Розробка моделі варіантів використання веб-сайту**

При розробці веб-сайту одним з головних завдань є створення привабливого та естетичного дизайну, який здатен привернути увагу користувачів і зробити перебування на сайті приємним і захоплюючим. Вибір правильного шрифту та його оформлення грає важливу роль у створенні візуальної привабливості і забезпеченні зручного читання контенту. Кожен шрифт має свою особливість і стиль, і важливо обрати той, який буде відповідати тематиці сайту та передавати його повідомлення інтуїтивно.

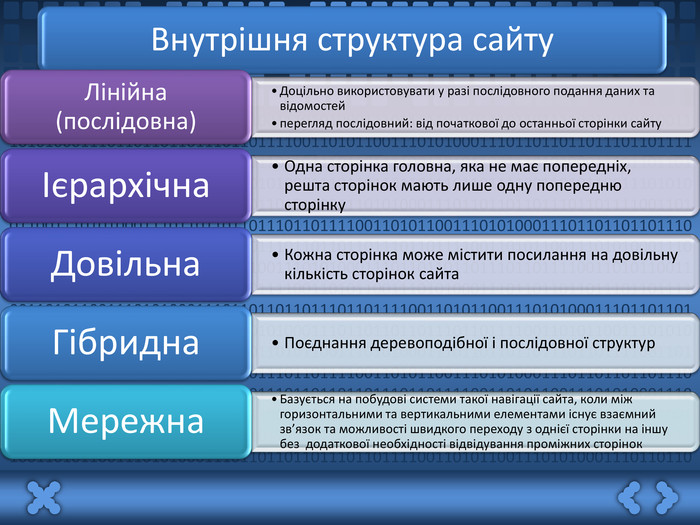
Колірна палітра також відіграє важливу роль у створенні естетичного враження і створенні належної атмосфери на веб-сайті. Гармонійне поєднання кольорів допомагає створити зручне та привабливе середовище для користувачів, сприяючи збереженню їх інтересу і залученню до подальшої взаємодії зі сторінками сайту. Кожен колір має своє значення і може викликати певні асоціації та емоції, тому важливо обирати кольори, що відповідають цілям та концепції сайту.

Однак, привабливий дизайн сам по собі не є достатнім. Важливо, щоб веб-сайт був легким у використанні та зрозумілим для користувачів. Це означає, що навігація повинна бути логічною, інтуїтивно зрозумілою та зручною для пересування по різних розділах та сторінках сайту. Елементи управління, такі як кнопки, посилання та меню, повинні бути візуально виокремлені та легко доступні. Зручна навігація дозволяє користувачам швидко знайти потрібну інформацію, покупки чи послуги, що сприяє поліпшенню користувацького досвіду та задоволенню від використання веб-сайту.

Таким чином, успішний веб-сайт поєднує привабливий дизайн з легкістю використання, створюючи гармонійне і привітне середовище для користувачів. Це допомагає залучити їх увагу, забезпечити комфортну навігацію та зберегти їх інтерес протягом відвідування сайту. Детальніша інформація наведена на рисунках 1.1 та 1.2.



**Рис.1.1 Структура сайту та її складові**



**Рис.1.2. Внутрішня структура сайту.**

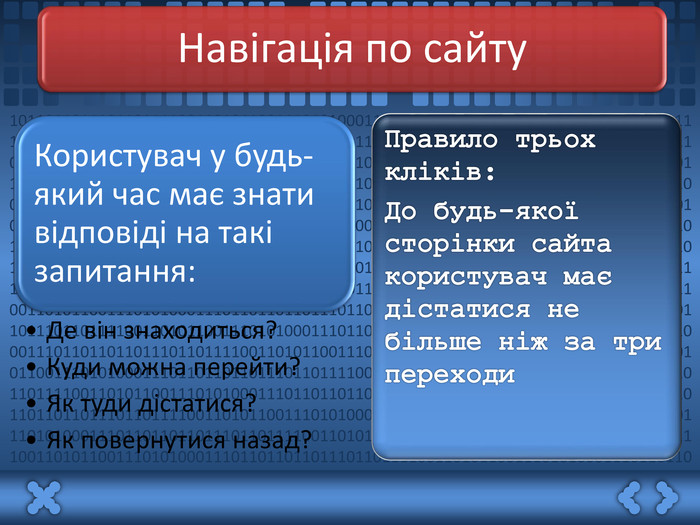
В кожній структурі та дизайні веб-сайту існує важлива складова, яка називається навігацією. Навігація є механізмом, що дозволяє користувачам орієнтуватися та пересуватися по веб-сайту, забезпечуючи зручність взаємодії та доступ до різних розділів і сторінок.

Навігація може мати різноманітні форми та елементи, залежно від конкретної моделі веб-сайту. Один з найпоширеніших варіантів - це горизонтальне меню, розташоване вгорі сторінки, де користувачі можуть знайти основні розділи сайту та переходити до них за допомогою кліку.

Крім горизонтального меню, існують інші типи навігації, такі як бокові панелі, випадаючі списки, хлібні крихти, кнопки з посиланнями та інші. Вони можуть бути розміщені на різних частинах сторінки, залежно від вимог та дизайну сайту.

Важливим аспектом навігації є її зрозумілість та інтуїтивність для користувачів. Елементи навігації повинні бути логічно впорядковані, зрозумілі та легкодоступні. Також важливо враховувати адаптивність навігації до різних пристроїв, забезпечуючи зручний доступ до сайту як на комп'ютерах, так і на мобільних пристроях.

Добре спроектована та ефективна навігація робить веб-сайт зручним у використанні, допомагає користувачам швидко знаходити потрібну інформацію та покращує загальний досвід взаємодії з сайтом. Тому розробка та впровадження ефективної навігації є важливим етапом при створенні будь-якого веб-сайту. (див. рис. 1.3).



**Рис 1.3. навігація сайту**

Існують різні моделі веб-сайтів, зокрема:

1. Модель "Контент":

Ця модель передбачає створення інтернет-ресурсу з цікавим та затребуваним контентом. Вона може орієнтуватись на широку аудиторію або на спеціалізованих користувачів, і контент може бути створений одним або кількома авторами. Такий сайт може містити графіку, тексти, відео, аудіо файли. Текстовий контент на цьому сайті зазвичай є унікальним та компактним, а сам сайт є легкодоступним для відвідувачів. Прикладами таких сайтів є інформаційні ресурси, сайти-візитівки, персональні сайти та інші.

2. Модель "Співтовариство":

Інтернет є не лише джерелом інформації, але й прекрасним середовищем для спілкування. Чати, блоги, коментарі, форуми та інші засоби спілкування вміють привернути увагу відвідувачів таких сайтів. Ця модель дозволяє створювати окремі спільноти або чати на сайті для людей з подібними інтересами, професійними ознаками, спільними хобі тощо. Людська потреба у взаємодії та спілкуванні забезпечує відвідуваність сайтів цієї моделі. Тематичні блоги і форуми часто є джерелом корисної інформації для відвідувачів.

3. Модель "Сервіс":

Інформаційна та комунікативна потреба людей призводить до створення різних інтернет-сервісів, таких як електронна пошта, пошукові системи, електронні платіжні сервіси тощо. Інтернет надає можливість для

фінансових операцій, онлайн-торгівлі та інших електронних послуг. Реалізація моделі "Сервіс" вимагає розробки нових та нетривіальних сервісів, але саме ця модель може привести до успішних та комерційно значущих інтернет-проектів.

Кожна з цих моделей може поєднуватись з іншими, і вибір головної моделі для веб-сайту залежить від господаря проекту. Це допоможе визначити характеристики ресурсу, виділитися серед конкурентів та краще розуміти його мету та напрямки розвитку.

При розробці інформаційної системи для нотаріальної контори, домінуючою моделлю буде "Модель сервіс". Оскільки цей сайт буде спрямований на надання різноманітних юридичних послуг та забезпечення зручного спілкування з клієнтами.

Застосовуючи модель "Сервіс", інформаційна система нотаріальної контори надаватиме користувачам такі сервіси як:

1. Онлайн запис на консультацію або нотаріальні послуги: Клієнти зможуть зручно запланувати свій візит до нотаріуса, обираючи зручний для них час та дату.

2. Електронна заявка на нотаріальні документи: Клієнти зможуть заповнити форму заявки онлайн та відправити необхідну інформацію для підготовки документів перед візитом до нотаріуса.

3. Онлайн-консультація з нотаріусом: Клієнти зможуть задавати питання та отримувати консультацію від нотаріуса в онлайн-режимі, що забезпечить швидку та зручну комунікацію.

Модель "Сервіс" дозволить нотаріальній конторі ефективно надавати свої послуги в онлайн-режимі, забезпечуючи зручність та швидкість взаємодії з клієнтами. Крім того, вона сприятиме автоматизації деяких процесів та полегшить отримання інформації для клієнтів, що збільшить задоволеність та лояльність клієнтів до нотаріальної контори.

## Логічна структура сайту

Структура сайту може мати різні моделі організації, такі як лінійна, "ґрати", "дерево" (ієрархія) і "павутина". Наприклад, лінійна модель дозволяє переходити від однієї сторінки до наступної в строго послідовному порядку. Однак ця модель рідко використовується, оскільки обмежує відвідувачів у виборі. Зазвичай лінійна структура сайту застосовується з деякими варіаціями, такими як пропуск певних сторінок або бічні відгалуження.

Модель "ґрати" зустрічається в інтернет-магазинах і має двонаправлену лінійну структуру з горизонтальними та вертикальними зв'язками між сторінками. Це дозволяє відвідувачам переглядати товари за різними категоріями, наприклад, за видом та ціною, забезпечуючи їм волю вибору.

Модель "дерево" є найпоширенішою ієрархічною структурою сайту. Вона дозволяє відвідувачам керувати глибиною перегляду сайту і обирати сторінки з різних рівнів ієрархії. Широта дерева визначає ступінь вибору, і занадто вузьке або занадто широке дерево може призвести до незручностей для відвідувачів.

Модель "павутина" використовується, коли структура сайту заплутана і складна. Це може бути результатом неправильного проектування або наміреного використання для виразного подання інформації. Такий підхід дозволяє відвідувачам отримати повну інформацію, але може стати заплутаним і важкодоступним.

Вибір правильної моделі структури сайту і врахування потреб відвідувачів є важливими для досягнення високої юзабіліті і ефективності сайту. Тому важливо ретельно підходити до розробки логічної структури сайту.  
Враховуючи попередні дані, можна зробити висновок, що найбільш оптимальним варіантом для створення веб-сайту нотаріальної контори буде застосування лінійної структури.

## 1.4 Аналіз засобів реалізації (техніко-економічне обґрунтування вибору)

Дана реалізації даної роботи було вибрано наступні технології:

-HTML5, СSS3, JS(зокрема і бібліотека Jquery), PhP, MySql(Для реалізації Back-end частини).

Для даної роботи були використані такі технології:

Для фронтенду (Front-end):

- HTML5, CSS3, та JavaScript (зокрема бібліотеку jQuery). Використання JavaScript дозволяє реалізувати складні рішення веб-додатків та зробити сторінки сайту більш інтерактивними, реагуючи на дії користувачів.

Для бекенду (Back-end):

- PHP та MySQL. Використання PHP дозволяє розробляти гнучкі веб-додатки, які можуть працювати на різних платформах та веб-серверах. MySQL використовується для зберігання та управління базами даних, забезпечуючи масштабованість, переносність та безпеку даних.

Ці технології були обрані через їхні переваги:

- JavaScript дозволяє реалізувати інтерактивність та динаміку на веб-сторінках, а його популярність зросла завдяки AJAX-технології, яка забезпечила новий рівень розробки сайтів.

- PHP є гнучкою мовою програмування, яка підтримується на багатьох платформах і веб-серверах. Вона має добре документовану документацію та широкий вибір систем керування сайтами, що написані саме на PHP.

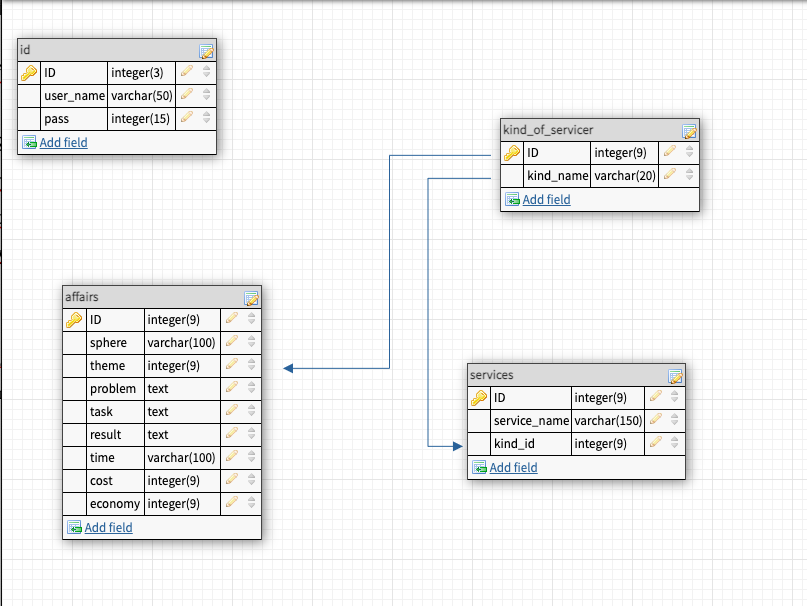
- MySQL має масштабованість для роботи з великими базами даних, переносність на різні платформи, зв'язаність для доступу до бази даних з різних додатків та високу швидкість виконання команд.

Використання цих технологій дозволило створити зручну та ефективну реалізацію роботи, забезпечивши функціональність та зовнішній вигляд веб-додатку.

# РОЗДІЛ 2. Розробка бази даних

## 2.1 Опис моделі даних

Модель даних - це фундаментальна структура, яка визначає, як дані будуть представлені, організовані та збережені у базі даних. Вона включає в себе набір понять і правил, які дозволяють описати структуру, стан і зміни даних в конкретній області, яку ви хочете відображати у базі даних. Ця модель даних визначає, які сутності (такі як таблиці, поля, відношення) існують, як вони пов'язані між собою і які обмеження встановлені на ці дані. Вона служить основою для створення та управління базою даних, забезпечуючи логічну структуру та орієнтацію для зберігання та обробки інформації. У ході практичного завдання була побудована наступна модель бази даних (Рис.2.1).



**Рис.2.1 Модель бази даних**

Також на рисунку(Рис.2.1) можна побачити, що дана модель складається з чотирьох таблиць: id, , kind\_of\_servicer, services та affairs.

Таблиця id у свою чергу складається з наступник стовпців:id, user\_name, pass. Таблиця kind\_of\_servicer має такі стовпці: id, kind\_name. Таблиця services вміщає в себе стовпці: id, services\_name, kind\_id. І таблиця affairs складається з id, sphere, theme, problem, task, result, time, cost та economy.

Первинний ключ на рисунку(рис.2.1) позначається жовтим кольором.

Існують такі види моделей даних:

Ієрархічна модель бази даних побудована у формі деревоподібної структури, де кожен об'єкт має головний елемент і може мати декілька підлеглих елементів. На верхньому рівні ієрархії знаходиться кореневий елемент. Прикладом ієрархічної структури даних може бути організація каталогів на комп'ютерному диску, класифікаційні системи, структура державної влади та інші подібні випадки.

Концептуальна схема ієрархічної моделі представляє собою набір типів записів, пов'язаних типами зв'язків у одному або декількох деревах. Всі зв'язки в цій моделі належать до типу "один до багатьох" і зображаються стрілками. Таким чином, взаємозв'язки між об'єктами схожі на взаємозв'язки у генеалогічному дереві, за винятком того, що кожен підлеглий об'єкт може мати лише один головний об'єкт. Іншими словами, ієрархічна модель даних дозволяє тільки два типи зв'язків: "один до одного" і "один до багатьох". Ієрархічні бази даних мають навігаційний характер, що означає, що доступ до даних можливий лише за допомогою попередньо визначених зв'язків. Однак, при моделюванні подій, часто потрібні зв'язки типу "багато до багатьох". Один з можливих способів подолати це обмеження - дублювати об'єкти. Однак, дублювання об'єктів може призводити до неузгодженості даних. Перевагою ієрархічної бази даних є швидкий доступ до даних вздовж попередньо визначених зв'язків.

Однак, негнучкість моделі даних і відсутність можливості мати кількох батьків для об'єкта, а також відсутність прямого доступу до даних роблять її непридатною для ситуацій, коли потрібно часто виконувати запити, які не були попередньо сплановані. Ще одним недоліком ієрархічної моделі даних є те, що не можна спрямовувати інформаційний пошук з нижніх рівнів ієрархії до вище розташованих вузлів.

Мережна модель бази даних дозволяє об'єкту виступати як головний і підлеглий елемент одночасно. Це означає, що об'єкт може мати довільну кількість зв'язків. Зв'язок може бути встановлений явно, коли значення деяких полів посилаються на дані, що містяться в іншому файлі. Прикладом мережної структури бази даних може бути система автобусних маршрутів, де існують зв'язки між різними місцями.

Реляційна модель бази даних представляє дані та їх зв'язки за допомогою прямокутних таблиць. Кожен рядок у таблиці даних називається записом, а стовпці - полями. Реляційна модель даних була розроблена в 70-х роках XX століття як більш зручний спосіб зберігання, вибірки та маніпулювання даними порівняно з ієрархічною та мережною моделями. Назва "реляційна" пов'язана з тим, що кожен запис у таблиці даних містить інформацію, пов'язану з конкретним об'єктом. Крім того, дані різних типів, пов'язані між собою, можуть бути розглянуті як єдине ціле.

## 2.2 Нормалізація відношень

Нормалізація відношень у базі даних - це процес розбиття початкових таблиць на менші, більш прості таблиці. Цей процес приводить схему бази даних до послідовних нормальних форм, кожна з яких має кращі властивості, ніж попередня. Кожна нормальна форма включає певний набір обмежень. Під час нормалізації структури таблиць у вищі форми, з таблиць вилучається зайвий описовий інформація. Цей процес базується на концепції функціональної залежності атрибутів.

Перша нормальна форма (1NF) вимагає, щоб кожне значення у таблиці було атомарним (неподільним), а групи, що повторюються, були видалені.

Друга нормальна форма (2NF) вимагає, щоб кожен описовий атрибут повністю залежав від первинного ключа, включаючи складений ключ.

Третя нормальна форма (3NF) вимагає, щоб всі описові атрибути були взаємно незалежними та повністю залежали від первинного ключа, тобто жоден описовий атрибут не залежав транзитивно від ключа.

Четверта нормальна форма (4NF) вимагає, щоб не було нетривіальних багатозначних залежностей між множинами атрибутів, які не витікають з ключів.

П'ята нормальна форма (5NF) вимагає, щоб не було нетривіальних залежностей об'єднання, які не витікають з обмежень ключів.

Отже, нормалізація відношень бази даних полягає у розбитті однієї великої таблиці на кілька менших таблиць, які мають чітко визначену змістовність.

Розглянемо нашу модель даних(Рис.2.1). Можна побачити що база даних „розбита” на декілька логічних змістом таблиць. Є окрема таблиця з адміном, окрема з видом сервісу, окрема з послугами нотаріуса. Також є таблиця із справами

### 2.3 Визначення типів даних



**Рис2.2(Визначення типів даних для таблиці „services”)**

Перед тим як створювати кожну таблицю у базі даних, потрібно визначити які дані будуть зберігатися у кожному стовпці. В мові SQL існує багато типів даних, які можна використовувати.

Нище в таблицях наведено всі типи даних в sql.

|  |  |
| --- | --- |
| *Тип даних* | *Опис* |
| *INT* | *Ціле число* |
| *BIGINT* | *Ціле число великої довжини* |
| *SMALLINT* | *Малий цілочисельний тип* |
| *TINYINT* | *Дуже малий цілочисельний тип* |
| *FLOAT* | *Число з плаваючою комою* |
| *DOUBLE* | *Подвійна точність числа* |
| *DECIMAL(p, s)* | *Число з фіксованою точністю* |
| *NUMERIC(p, s)* | *Число з фіксованою точністю* |
| *REAL* | *Число з плаваючою комою в одинарній точності* |
| *VARCHAR(n)* | *Рядок змінної довжини до n символів* |
| *CHAR(n)* | *Рядок фіксованої довжини n символів* |
| *TEXT* | *Текстове поле з великим обсягом даних* |
| *BOOLEAN* | *Логічне значення (TRUE або FALSE)* |
| *DATE* | *Дата* |
| *TIME* | *Час* |
| *DATETIME* | *Дата та час* |
| *TIMESTAMP* | *Дата та час з використанням часового поясу* |
| *BLOB* | *Бінарні дані* |
| *BINARY* | *Бінарний об'єкт фіксованої довжини* |
| *VARBINARY* | *Бінарний об'єкт змінної довжини* |
| *CLOB* | *Великі символьні дані* |
| *ARRAY* | *Масив значень* |
| *JSON* | *JSON-об'єкт* |
| *XML* | *XML-документ* |
| *INTERVAL* | *Інтервал часу або дати* |
| *UUID* | *Унікальний ідентифікатор* |
| *ENUM* | *Перелік значень* |
| *SET* | *Набір значень* |
| *GEOMETRY* | *Геометричний об'єкт* |
| *GEOGRAPHY* | *Географічний об'єкт* |
| *HIERARCHYID* | *Ієрархічний ідентифікатор* |
| *ROWVERSION* | *Версія рядка* |
| *SQL\_VARIANT* | *Змінний тип даних, що зберігає значення будь-якого типу* |
| *TABLE* | *Таблиця (використову ється для вкладених таблиць)* |
| *CURSOR* | *Вказівник на результат запиту* |
| *UNIQUEIDENTIFIER* | *Унікальний ідентифікатор* |
| *MONEY* | *Грошова сума* |
| *SMALLMONEY* | *Малий грошовий тип* |
| *NCHAR(n)* | *Рядок фіксованої довжини n символів (Unicode)* |
| *NVARCHAR(n)* | *Рядок змінної довжини до n символів (Unicode)* |
| *NTEXT* | *Текстове поле з великим обсягом даних (Unicode)* |
| *SYSNAME* | *Спеціальний тип, який використовується для збереження*  *назви об'єкта бази даних SQL Server* |

**Таблиця 2.1.Типи даних в sql**

В даному випадку використовуються наступні типи даних:

* int(9)-для стовпця id. Дане поле є первичним ключем, тобто повинно мати лище унікальні значання;
* varchar(150) -для поля service\_name;
* int(9)- для поля kind\_id;

Числа в дужка означають максимальну кількість символів яка може бути записана в даному полі. Тобто id і kind\_name не може перевищувати 9 символів, ім’я-150.

**2.4 Обмеження цілісності даних**

Правила цілісності бази даних визначають набір обмежень, які забезпечують коректність та зв'язність даних в базі даних. Ці правила допомагають уникнути введення некоректних або неповних даних, а також забезпечують зв'язок між різними таблицями. При створенні або модифікації таблиці можна встановлювати такі правила цілісності:

1. Правило CHECK: Контролює допустимі значення атрибутів. Встановлює обмеження на значення, які можуть бути введені в атрибут.

2. Правило NOT NULL/NULL: Забороняє або дозволяє використання значень NULL, тобто визначає, чи обов'язково має бути введене значення в атрибут.

3. Правило UNIQUE: Забезпечує унікальність значень атрибута в межах таблиці. Кожне значення в цьому атрибуті повинно бути унікальним.

4. Правило PRIMARY KEY: Використовується для ідентифікації унікальних рядків таблиці. Це основний ключ, і в таблиці може бути лише один атрибут, визначений як первинний ключ. Рядки з таким значенням не можуть мати невизначених або пустих значень.

5. Правило FOREIGN KEY: Використовується для встановлення зв'язку між двома таблицями. Вказує на поле в головній таблиці, на яке посилається зовнішній ключ. Поле зовнішнього ключа повинно бути визначеним як первинний ключ або мати обмеження UNIQUE. Поле зовнішнього ключа та поле головної таблиці, на яке воно посилається, повинні мати однаковий тип даних.

Правила FOREIGN KEY також дозволяють реалізувати каскадне оновлення та видалення даних, що означає, що зміни в головній таблиці автоматично відображатимуться в залежних таблицях.

При розробці бази даних для нотаріальної контори були використані такі правила цілісності: PRIMARY KEY (первинний ключ), FOREIGN KEY (зовнішній ключ), UNIQUE (унікальність значень) та було застосовано каскадне оновлення та видалення даних. Це допомогло забезпечити коректність та зв'язність даних у базі даних нотаріальної контори.

### 2.5 Реалізація SQL-скрипту

Sql-скрипт - це програмний код, який виконує запити до бази даних, здійснює оновлення, видалення та інші операції. Важливо зазначити, що виконання скрипту не завжди призводить до зміни структури бази даних.

Наприклад, якщо ми розглядаємо скрипт запиту, він просто отримує певну інформацію з бази даних, яка потім може бути оброблена і відображена для користувача.

Для виконання sql-скрипту необхідно спочатку встановити з'єднання з базою даних. Цей процес можна реалізувати за допомогою наступного методу.

<!DOCTYPE html>

<?php

$servername = '127.0.0.1:3306'; // Ім'я сервера бази даних

$username = 'root'; // Логін користувача бази даних

$password = ''; // Пароль користувача бази даних

$dbname = 'notalex'; // Назва бази даних, до якої ви хочете підключитися

// Підключення до бази даних

$conn = mysqli\_connect($servername, $username, $password, $dbname);

// Перевірка підключення

if (!$conn) {

die('Connection failed: ' . mysqli\_connect\_error());

}

?>

Перевіряєм, чи була відправлена форма з ім'ям користувача і паролем. Він виконує запит до бази даних, перевіряє правильність пароля і у разі співпадіння перенаправляє користувача на захищену сторінку, зберігаючи інформацію про користувача у сесії.

<?php

// перевірка, чи відправлена форма

if($\_SERVER["REQUEST\_METHOD"] == "POST") {

// отримання даних з форми

$username = $\_POST["username"];

$password = $\_POST["password"];

// формування запиту до бази даних по користувачу

$sql = "SELECT `id`, `user\_name`, `pass` FROM `user` WHERE `user\_name`='" .$username . "'";

$result = mysqli\_query($conn, $sql);

while($result\_array = mysqli\_fetch\_assoc($result)) {

// перевірка правильності паролю користувача

if ($result\_array["pass"] == $password){

session\_start();

// збереження інформації про користувача у сесії

$\_SESSION["id"] = $result\_array["id"];

// перенаправлення на захищену сторінку

header("Location: admin.php?id=" . $result\_array["id"]);

}

}

}

?>

Він виконує запит до бази даних для підрахунку кількості унікальних значень поля "id" в таблиці "kind\_of\_services". Потім він виводить блоки з інформацією про кожен тип послуги та список пов'язаних з ним послуг з бази даних. Після цього він виводить HTML-форму для додавання нової послуги. Код використовує з'єднання з базою даних, виконує запити та обробляє результати, використовуючи функції mysqli\_query() та mysqli\_fetch\_assoc(). Він також використовує глобальну змінну $\_SESSION["id"] для передачі ідентифікатора сесії.

function getServices($limit)

{ global $mysqli; connectDB(); $result = $mysqli-> query("SELECT \* FROM `services`

ORDER BY `id` ASC LIMIT $limit");

closeDB(); return resultToArray($result);

}

function getWorkers($limit)

{ global $mysqli; connectDB(); $result = $mysqli->

query("SELECT \* FROM `workers`

ORDER BY `id` ASC LIMIT $limit"); closeDB(); return resultToArray($result);

}

function resultToArray($result)

{

$array = array(); while (

($row = $result-> fetch\_assoc()

) != false)

$array[] = $row; return $array;

}

Цей код дозволяє взаємодіяти з базою даних, отримувати і відображати інформацію про послуги та типи послуг, а також додавати нові послуги за допомогою HTML-форми.

<?php

if($\_SERVER["REQUEST\_METHOD"] == "POST") {

// отримання даних з форми

$content\_id = $\_POST["content\_id"];

$new\_service\_name = $\_POST["service\_name"];

$sql = "UPDATE `services` SET `service\_name`='" . $new\_service\_name . "' WHERE `id`='" . $content\_id . "'"; // запит на оновлення послуги в таблиці

$result = mysqli\_query($conn, $sql);

if (mysqli\_query($conn, $sql)) {

header("Location: admin.php?id=" . $\_SESSION['id']);

} else {

echo "Помилка оновлення: " . mysqli\_error($conn);

}

}

$content\_id = $\_GET["content\_id"];

$sql = "SELECT \* FROM `services` WHERE `id`='" . $content\_id . "'";

$result = mysqli\_query($conn, $sql);

while($result\_array = mysqli\_fetch\_assoc($result)) {

$service\_name = $result\_array['service\_name'];

echo('<form method="post" action="edit.php?content\_id='. $result\_array['id'] .'&id=' . $\_SESSION['id'] . '">

<input type="text" class="form-input" id="service\_name" name="service\_name" value="' . $service\_name . '"><br>

<input type="hidden" name="content\_id" value="' . $result\_array['id'] . '">

<input type="submit" class="form-submit" value="Відправити">

</form>');

}

?>

</div>

</body>

</html>

<?php

}

?>

Код перевіряє, чи була відправлена форма методом POST. Якщо так, він отримує дані з форми, включаючи ідентифікатор контенту та нову назву послуги. Потім виконує запит до бази даних для оновлення назви послуги відповідно до заданого ідентифікатора. Якщо запит виконується успішно, користувач перенаправляється на сторінку адміністратора. У разі помилки виводиться повідомлення про помилку. Далі, код отримує ідентифікатор контенту через GET-параметр, виконує запит до бази даних для отримання інформації про послугу з відповідним ідентифікатором і виводить форму для редагування назви послуги.

**РОЗДІЛ 3. Розробка веб-додатку**

## 3.1 Структура веб-сайту

Зовнішня структура веб-сайту є першим враженням, яке користувач отримує при відвідуванні сайту. Вона складається з декількох основних елементів, кожен з яких виконує свою функцію і має важливе значення для коректної навігації та сприйняття контенту.

Шапка сайту є верхньою частиною, яка зазвичай містить логотип і назву сайту. Цей елемент відіграє роль ідентифікації та представлення сайту, створюючи впізнаваність і брендовий образ. Також у шапці може розташовуватися основне меню, яке надає можливість користувачам легко переходити до основних розділів або сторінок сайту.

Меню є важливим інструментом для навігації по сайту. Воно зазвичай містить список основних розділів або сторінок, які доступні користувачам. Це дозволяє швидко знаходити потрібну інформацію та переміщатися між різними частинами сайту.

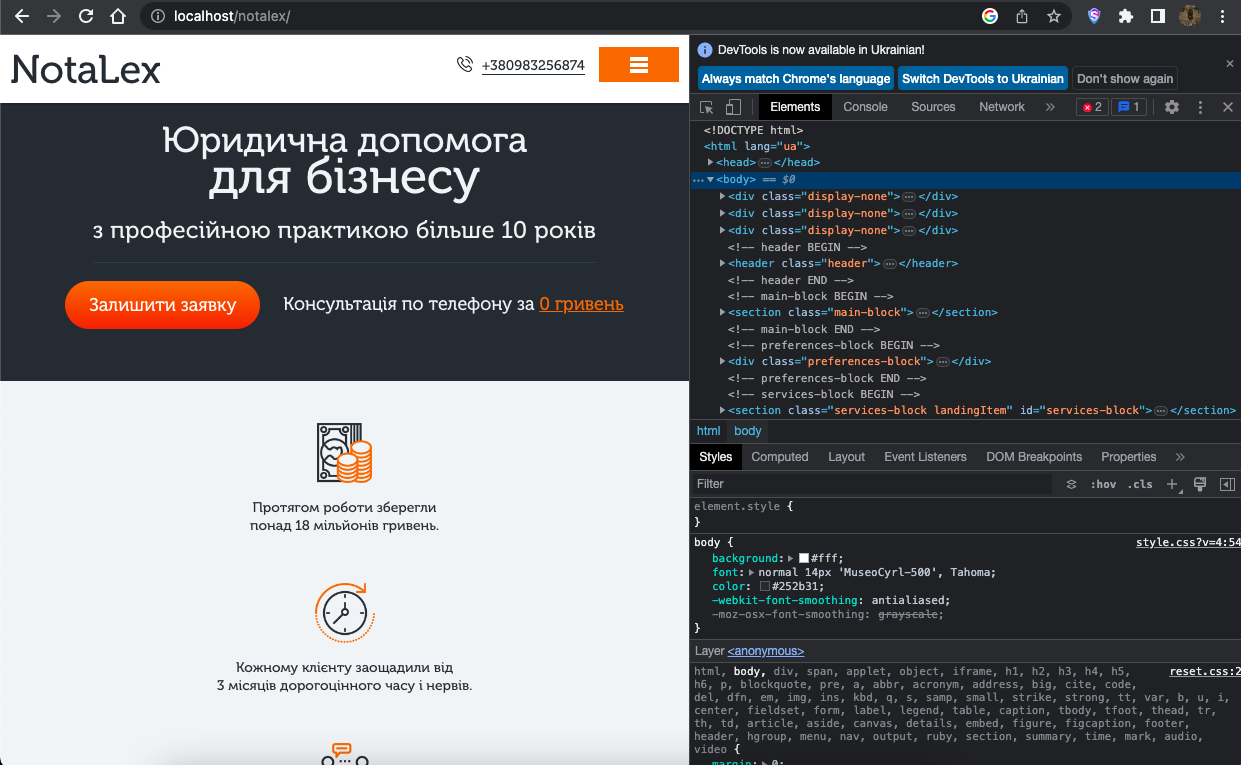
Контент є головною частиною сайту, оскільки саме він містить інформацію, яка цікава користувачам. Це можуть бути тексти, зображення, відео, звукові файли та інші елементи, які передають потрібну інформацію або створюють певний візуальний ефект. Наприклад, на головній сторінці шкільного сайту основним контентом можуть бути новини, які повідомляють про останні події та анонсують майбутні події.

Підвал або футер знаходиться у нижній частині сайту і містить додаткову інформацію, яка може бути менш важливою, але все ж важливою для користувачів. Наприклад, в підвалі можуть бути вказані контактні дані, посилання на політику конфіденційності або повідомлення про авторські права. Цей елемент зазвичай має стандартний дизайн на всіх сторінках сайту і надає додаткові ресурси або інформацію для користувачів.

Крім основних елементів, на сайті можуть бути розміщені посилання на інші сайти, різноманітні інформери або додаткові елементи, які можуть збагатити досвід користувача і надати йому додаткову інформацію або функціональність.

Внутрішня структура сайту представлена картою сайту, яка відображає ієрархію сторінок і зв'язки між ними. Це допомагає користувачам легко орієнтуватися на сайті та знаходити потрібну інформацію. Карта сайту може бути візуальною діаграмою або структурованим списком сторінок та розділів. Вона допомагає розробникам та адміністраторам сайту управляти та організовувати його контент.

Загалом, зовнішня і внутрішня структура сайту є важливими аспектами, які визначають як зовнішній вигляд, так і функціональність сайту. Вони сприяють зручній навігації, зрозумілому сприйняттю контенту та забезпечують позитивний досвід користувача при взаємодії з сайтом.



**Рис.3.1(Структура сайту нотаріальної контори)**

На малюнку (рис. 3.1) можна побачити чітку та належну організацію зовнішнього та внутрішнього вигляду веб-сайту, з врахуванням всіх вимог та рекомендацій.Важливо дотримуватися правильної структури при створенні сайту, оскільки у протилежному випадку можуть виникнути труднощі при просуванні та оптимізації SEO.

Use case діаграма(рис.3.2) використовується для моделювання функціональності нашої системи з точки зору її користувачів. На діаграмі зображені актори, які взаємодіють з системою, та use case'и, які представляють функціональні можливості системи.

Головні актори нашої системи включають Клієнта та Адміністратора. Клієнт взаємодіє з системою для отримання послуг нотаріуса. Адміністратор відповідає за управління системою, редагування інформації на сайті.

Усі ці актори використовують різні use case для взаємодії з системою.

Завдяки use case діаграмі ми можемо краще розуміти, які функції має виконувати наша система та як користувачі будуть взаємодіяти з нею. Це допомагає нам визначити потреби користувачів і розробити ефективну та зручну систему для нотаріальної контори.



**Рис.3.2(use case діаграма)**

**3.2 Макети сторінок веб-сайту**

Перед початком верстки сайту, передбаченої, необхідно спочатку створити макети сторінок. Це допоможе зекономити значну кількість часу та зусиль. Найпростішим способом відображення майбутнього сайту є блокова схема. Вона представляє основні блокові елементи сторінки на світлому фоні. Наступним кроком є розробка графічного макета сайту, який включає відтінки, відступи та інші параметри у статичному вигляді. Щодо інтерактивного представлення роботи майбутнього сайту, найбільш точним варіантом є прототип. З його допомогою можна оцінити не тільки зовнішній вигляд, але й функціонал різних елементів.

Макет сайту - це посереднє рішення між блоковою схемою та прототипом. Він є оптимальним варіантом для клієнта, оскільки не потребує додаткової оплати за інтерактивні ефекти прототипу. З іншого боку, макет не виглядає так просто, як блокова схема, і дозволяє верстальнику краще зрозуміти задуману концепцію сайту. Залежно від угоди між клієнтом та дизайнером, макет може виглядати як блок з вказівкою розмірів або майже готовий дизайн.

Переваги використання макета:

- Виявлення візуальних недоліків на ранніх етапах роботи над проектом.

- Можливість наочного демонстрування дизайнерських ідей клієнту.

- Спрощення роботи верстальника та програміста.

- Зрозуміння того, як готовий сайт буде виглядати.

Поширені помилки при створенні макетів:

- Зайве завантаження сторінки елементами.

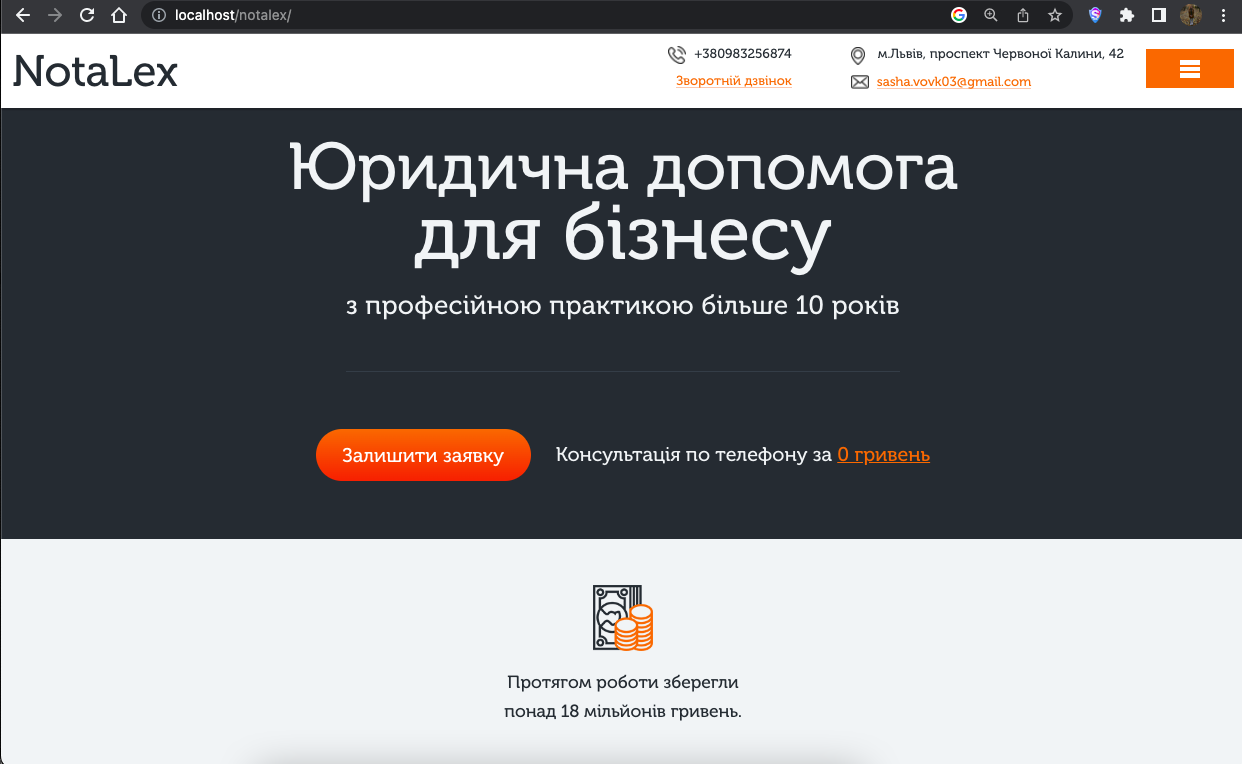
- Неврівноваженість кольорових відтінків.

- Непотрібні шари, які потрібно було вилучити.

- Невдалий фон та незручно читабельний шрифт.

- Відсутність адаптації під різні пристрої.

- Макет сайту з неправильними розмірами.



**Рис3.3(Приклад макету меню для мобільної версії із сайту нотаріальної контори)**

## 3.3 Програмування серверної частини

У розділі реалізації (практичній частині) інформаційної системи (вебсайту) для нотаріальної контори, серверна частина розроблена на мові програмування PHP.

Розглянемо фрагмент коду, який відповідає за створення форми зворотного зв'язку.

*$form['form-1'] = array(*

*'fields' => array(*

*'name' => array(*

*'title' => 'Імя: ',*

*),*

*'tell' => array(*

*'title' => 'Телефон: ',*

*'validate' => array(*

*'minlength' => '6',*

*),*

*),*

*),*

*'cfg' => array(*

*'charset' => 'utf-8',*

*'subject' => 'NotaLex – Зворотній дзвінок',*

*'title' => 'NotaLex – Зворотній дзвінок',*

*'ajax' => true,*

*'validate' => true,*

*'from\_email' => 'noreply@email.com',*

*'from\_name' => 'noreply',*

*'to\_email' => 'sasha.vovk03@gmail.com',*

*'to\_name' => 'noreply1, noreply2',*

*'geoip' => true,*

*'referer' => true,*

*'type' => 'html',*

*'tpl' => true,*

*'antispam' => 'email77',*

*'antispamjs' => 'address77',*

*'okay' => '*

*<div class="modal-item-ob">*

*<p class="modal-item-head">*

*Дякуємо ми вам<br> передзвонимо протягом 15хв.*

*</p>*

*</div>*

*',*

*'fuck' => 'Повідомлення відправлено - ERROR',*

*'spam' => 'Cпам робот',*

*'notify' => 'color-modal-textbox',*

*'usepresuf' => false*

*)*

*);*

Цей код створює форму зворотного зв'язку, яка складається з двох полів: ім'я та номер телефону. Поля мають заголовки і деякі правила валідації.

Конфігурація форми включає параметри, такі як кодування символів, тему та заголовок повідомлення, використання AJAX та валідації, електронну пошту відправника та одержувача, геолокацію та посилання, тип повідомлення та використання шаблону.

Крім того, є ряд повідомлень, які використовуються для різних ситуацій: успішної відправки форми, помилки відправки та спаму.

Цей код встановлює налаштування форми зворотного зв'язку, яка може бути використана для отримання повідомлень від користувачів.

## 3.4 Програмування клієнтської частини

Клієнтська частина програми, яка взаємодіє з користувачем, виконує наступні завдання: передає запити користувача серверу, отримує результат запиту від сервера і відображає його у зручному для користувача вигляді.

У цьому проекті було написано багато коду, що відповідає за оформлення сторінок і стилізацію, з метою поліпшення зовнішнього вигляду та інтерфейсу фронтенду.

Для прикладу розглянемо використання мови Java Script в navigarion: При завантаженні сторінки відбувається ініціалізація плагіна "liLanding" для елемента з класом "header". Зазначений плагін використовується для плавної прокрутки до розділів сторінки при кліку на посилання в навігації. Задано параметр "topMargin" зі значенням 96, який встановлює відступ від верхнього краю браузера при прокрутці до розділу.

/\* navigarion \*/

$(window).load(function(){

$('.header').liLanding({

topMargin: 96

});

})

logo: При кліку на елемент з класом "logo" (логотип) відбувається плавна прокрутка сторінки до верхньої частини за допомогою анімації "animate" впродовж 500 мілісекунд.

$('.logo').click(function() {

$('html, body').animate({scrollTop: 0},500);

return false;

})

slider SLICK: Ініціалізується плагін Slick для створення слайдерів. Визначені різні налаштування для слайдерів з класами "cases-slider" і "reviews-slider". Зокрема, встановлено параметри "dots" для відображення крапок-індикаторів, "arrows" для відображення стрілок навігації, "infinite" для безкінечної прокрутки слайдів, "speed" для швидкості прокрутки, "slidesToShow" для кількості видимих слайдів тощо. Для слайдера з класом "reviews-slider" також задано налаштування для адаптивного режиму під розміри екрану, де кількість видимих слайдів зменшується до 1 при досягненні розміру екрану менше 767 пікселів.

/\* slider SLICK \*/

$('.cases-slider').slick({

dots: false,

arrows: true,

infinite: true,

speed: 100,

slidesToShow: 1,

slidesToScroll: 1

});

$('.reviews-slider').slick({

dots: true,

arrows: false,

infinite: true,

speed: 100,

slidesToShow: 3,

slidesToScroll: 1,

responsive: [

{

breakpoint: 767,

settings: {

slidesToShow: 1

}

}

]

});

Розробка клієнтської частини програми є не менш важливою, ніж серверної. Це пояснюється тим, що клієнту необхідно мати зручне та привабливе візуальне сприйняття інформації. Сайт не просто має працювати швидко та мати багато функціоналу, але й має мати добре оформлений та привабливий дизайн. Дизайн сайту є першим, що користувач бачить при відвідуванні певного веб-сайту.

## 3.5 Розміщення веб-сайту на локальному віртуальному середовищі або в Інтернеті

Для розміщення веб-сайту нотаріальної контори використовується локальне середовище XAMPP(рис.3.4).

XAMPP - це платформа, яка надає комплексне середовище для розробки та тестування веб-додатків безпосередньо на вашому комп'ютері. Він складається з п'яти основних компонентів: Apache, MySQL, PHP, Perl.

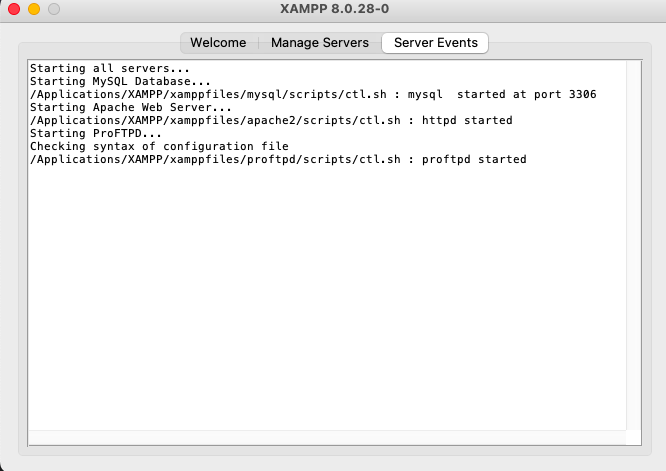
Однією з основних переваг XAMPP є простота встановлення. Він постачається з усіма необхідними компонентами, тому встановлення його на комп'ютер є швидким та простим процесом. Після встановлення XAMPP ви отримуєте повне середовище для розробки веб-додатків без необхідності окремого налаштування кожного компонента.

Комплексність XAMPP полягає в тому, що він включає всі основні компоненти, необхідні для розробки веб-додатків. Веб-сервер Apache відповідає за обробку запитів від клієнтів і доставку веб-сторінок. MySQL - система управління базами даних, яка забезпечує зберігання та управління даними веб-додатків. PHP - мова програмування, яка використовується для розробки динамічних веб-сторінок. Perl також є мовою програмування, яка використовується для розробки веб-додатків.

XAMPP також підтримує кросплатформеність, що означає, що він доступний для використання на різних операційних системах, таких як Windows, macOS та Linux. Це дає можливість розробникам використовувати XAMPP на будь-якому пристрої з уподобаною ОС.

Крім основних компонентів, XAMPP також має додаткові інструменти, такі як phpMyAdmin, який є інтерфейсом для управління базами даних MySQL, OpenSSL - для шифрування комунікації і багато інших.

Загалом, XAMPP є потужним інструментом для розробки веб-додатків, який забезпечує зручне середовище для розробки, тестування та налагодження веб-додатків локально на вашому комп'ютері, перед їхнім розгортанням на сервері. Це дозволяє розробникам ефективно працювати над проектом та забезпечує швидкий та зручний процес розробки веб-сайту для нотаріальної контори.



**Рис. 3.4(Приклад роботи XAMPP)**

**ВИСНОВКИ**

У процесі дослідження були досягнуті поставлені мета і завдання, а саме успішно реалізовано теоретичні та практичні аспекти побудови інформаційної системи - веб-сайту для нотаріальної контори. На основі отриманих результатів дослідження можна зробити наступні висновки:

1) Аналізуючи джерела літератури та проводячи власний аналіз інформаційної системи, стало очевидним, що веб-сайт для нотаріальної контори є актуальним і потрібним.

2) Функціональні можливості веб-сайту є важливими. Перед розробкою сайту необхідно детально проаналізувати його призначення і створити макет. Наприклад, у випадку нотаріальної контори, було вирішено включити форму зворотнього зв'язку, оскільки авторизація не є необхідною.

3) Правильний вибір технологій сприяє полегшенню розробки веб-сайту. У даному випадку, використання технологій PHP, MySQL і JavaScript дозволило уникнути будь-яких перешкод як з боку Front-end, так і з боку Back-end частини сайту.

Ураховуючи зазначене вище, можна стверджувати, що інформаційні системи та веб-сайти відіграють важливу роль у розвитку різних сфер діяльності, включаючи нотаріальні контори. Побудована інформаційна система позитивно впливає на розвиток нотаріального бізнесу, оскільки надає клієнтам можливість онлайн ознайомитись з послугами, процедурами, розкладом роботи та контактними даними нотаріуса. Це забезпечує зручність та ефективність у взаємодії з клієнтами, а також допомагає залучати нових клієнтів і зберігати існуючу клієнтську базу.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Мартин Грабер :SQL; Видавництво «Лори», 2016. -643 с.
2. [Моррисон М.](http://padabum.com/search.php?author=%D0%9C%D0%BE%D1%80%D1%80%D0%B8%D1%81%D0%BE%D0%BD%20%D0%9C.) : Изучаем JavaScript; Видавництво «O`REILLY», 2012. – 606 c.
3. Веб-дизайн, структура сайту [Електронний ресурс]. – Режим доступу: <https://webstudio2u.net/ua/design-web/403-structure-models.html>.
4. Переваги MySQl [Електронний ресурс]. – Режим доступу: <https://studfile.net/preview/5607354/page:3/>.
5. Переваги PhP [Електронний ресурс]. – Режим доступу:

[https://blog.ithillel.ua/ua/articles/chomu-vybyraiut-7-perevah-movyprohramuvannia-php.](https://blog.ithillel.ua/ua/articles/chomu-vybyraiut-7-perevah-movy-prohramuvannia-php)

1. Моделі даних [Електронний ресурс]. – Режим доступу: [https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D0%B5%D0%BB%](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D0%B5%D0%BB%D1%96_%D0%B1%D0%B0%D0%B7_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85)

[D1%96\_%D0%B1%D0%B0%D0%B7\_%D0%B4%D0%B0%D0%BD%D0%B8 %D1%85.](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D0%B5%D0%BB%D1%96_%D0%B1%D0%B0%D0%B7_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85)

1. Типи даних MySQL [Електронний ресурс]. – Режим доступу: [https://uk.wikibooks.org/wiki/SQL/%D0%A2%D0%B8%D0%BF%D0%B8\_%D0 %B4%D0%B0%D0%BD%D0%B8%D1%85\_MySQL.](https://uk.wikibooks.org/wiki/SQL/%D0%A2%D0%B8%D0%BF%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85_MySQL)