МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ НАУК ТА ТЕХНОЛОГІЙ

Кафедра Комп'ютерних інформаційних технологій

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач випускової кафедри
Аліна САВЧЕНКО.
«_____» _____ 2023р.

КВАЛІФІКАЦІЙНА РОБОТА

(ДИПЛОМНИЙ ПРОЄКТ, ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ **"БАКАЛАВР"**
ЗА ОСВІТНЬО-ПРОФЕСІЙНОЮ ПРОГРАМОЮ
"ІНФОРМАЦІЙНІ УПРАВЛЯЮЧІ СИСТЕМИ ТА ТЕХНОЛОГІЇ"

Тема: "Web-застосунок надання адвокатських послуг клієнтам"

Виконавець: студент групи УС-411Б Лось Назар Сергійович

Керівник: к.т.н., доцент Райчев Ігор Едуардович

Нормоконтролер: Олександр ШЕВЧЕНКО

Київ – 2023

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет *комп'ютерних наук та технологій*

Кафедра *Комп'ютерних інформаційних технологій*

Галузь знань, спеціальність, освітньо-професійна програма: 12 "Інформаційні технології", 122 "Комп'ютерні науки", "Інформаційні управляючі системи та технології"

ЗАТВЕРДЖУЮ

Завідувач випускової кафедри

_____ Аліна САВЧЕНКО

«____»_____2023р.

ЗАВДАННЯ

на виконання кваліфікаційної роботи студента

Лося Назара Сергійовича

(прізвище, ім'я, по батькові)

1. **Тема роботи:** «Web-застосунок надання адвокатських послуг клієнтам» затверджена наказом ректора від "01" травня 2023р. за № 623/ст.
2. **Термін виконання роботи:** 15.05.2023 – 25.06.2023
3. **Вихідні дані до роботи:** програмне забезпечення Visual Studio Code та OpenServer, завдання для розробки та вимоги до web-застосунку й web-сайту.
4. **Зміст пояснювальної записки:** дослідження предметної області, аналіз визначених програмних засобів та технологій для розробки програмного комплексу з web-застосунком, розробка на основі web-технологій, web-дизайн.
5. **Перелік обов'язкового графічного матеріалу:** Слайди презентації MS PowerPoint.

6. Календарний план-графік

№ з/п	*Завдання*	*Термін виконання*	*Підпис керівника*
1.	Дослідження та аналіз потреб адвокатської фірми	15.05.2023 – 16.05.2023	
2	Опрацювання інформації за тематикою дипломного проекту	16.05.2023 – 18.05.2023	
3	Розробка дизайну web-сайту та web-застосунку	18.05.2023 – 25.05.2023	
4	Розробка структури та бази даних комплексу	25.05.2023 – 30.05.2023	
5	Розробка web-сайту та web-застосунку	30.05.2023 – 05.06.2023	
6	Написання пояснювальної записки дипломного проекту	05.06.2023 – 15.06.2023	
7	Підготовка демонстраційного матеріалу та доповіді	15.06.2023 – 19.06.2023	

Дата видачі завдання: 15 травня 2023 р.

Керівник дипломної роботи _____ Ігор РАЙЧЕВ

(підпис керівника) (П.І.Б.)

Завдання прийняв до виконання _____ Назар Лось

(підпис випускника) (П.І.Б.)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи «Web-застосунок надання адвокатських послуг клієнтам» складається зі вступу, трьох розділів, висновку, списку бібліографічних посилань і містить 74 сторінки, 3 таблиці та 42 рисунка. Список бібліографічних посилань складається з 20 найменувань.

Мета кваліфікаційної роботи – застосування здобутих навичок та знань для розробки програмного комплексу на основі web-технологій.

Об’єкт дослідження – web-сайт та web-додаток на базі основних web-технологій.

Предмет дослідження – розробка web-сайту та web-застосунку адвокатських послуг.

Методи дослідження – аналіз електронних посібників і джерел інформації пов’язаних з web-технологіями.

Результатом кваліфікаційної роботи є розроблений повноцінний програмний комплекс з надання адвокатських послуг клієнтам. Web-застосунок слугує корисним інструментом при обслуговуванні клієнтів, з можливістю працювати з інформацією у базі даних, та календарем запланованих зустрічей фірми. А розроблений web-сайт, стає чудовою можливістю для розширення бази клієнтів шляхом реклами в мережі Інтернет. Практичне значення роботи полягає у створені зручного та функціонального інструменту для адвокатів, що дозволяє зберігати усю необхідну інформацію, про клієнтів, структурованою та цілісною, з можливостями редагувати її.

WEB-ТЕХНОЛОГІЇ, WEB-САЙТ, WEB-ЗАСТОСУНОК, КЛІЄНТ, КОРИСТУВАЧ, АДВОКАТ, HTML, CSS, JAVASCRIPT, PHP, LARAVEL, MYSQL, БАЗА ДАНИХ.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ, ТЕРМІНІВ 7
ВСТУП 8
РОЗДІЛ 1. ФУНКЦІОНУВАННЯ WEB-ЗАСТОСУНКУ 10
1.1. Веб браузер 10
1.2. Web-застосунок 12
1.3. Структура web-застосунків 14
1.4. Огляд сучасного стану адвокатських послуг 17
1.5. Огляд ринку компаній що надають адвокатські послуги 18
1.6. Розробка дизайну web-сайту 20
1.7. Обґрунтування створення web-сайту власної компанії 21
1.8. Функціональні та технологічні можливості web-сайту 23
1.9. Висновки до розділу 1 25
РОЗДІЛ 2. МЕТОДИ ТА ЗАСОБИ РЕАЛІЗАЦІЇ ПРОГРАМНОГО КОМПЛЕКСУ 27
2.1. Вибір архітектури програмного комплексу 27
2.2. Опис інструментів для розробки web-застосунку 30
2.3. Гіпертекстова мова розмітки HTML 32
2.4. Каскадна мова стилів CSS 35
2.5. Мова програмування JavaScript 39
2.6. Серверна мова програмування PHP 42
2.7. Система керування базами даних MySQL 45
2.8. Опис бази даних 46
2.9. Висновки до розділу 2 50
РОЗДІЛ 3. РЕАЛІЗАЦІЯ ПРОГРАМНОГО КОМПЛЕКСУ 51
3.1. Розробка та налагодження web-застосунку 51
3.2. Структура та функції клієнтської частини програмного комплексу 55

3.3. Результат розробки та тестування функцій при роботі з програмним комплексом 58
3.4. Висновки до розділу 3 71
ВИСНОВКИ 72
СПИСОК БІБЛІОГРАФІЧНИХ ПОСИЛАНЬ ВИКОРИСТАНИХ ДЖЕРЕЛ .. 73

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ, ТЕРМІНІВ

AJAX – Asynchronous JavaScript and XML

API – Application Programming Interface

CLI – Command-Line Interface

CSS – Cascading Style Sheets

HTML – HyperText Markup Language

HTTP – Hypertext Transfer Protocol

HTTPS – Hypertext Transfer Protocol Secure

JSON – JavaScript Object Notation

ORM – Object-Relational Mapping

PHP – Hypertext Preprocessor

SPA – Single-Page Application

URL – Uniform Resource Locator

ООП – Об'єктно-орієнтоване програмування

ВСТУП

В сучасному світі зростає значення використання інтернет-технологій у різних сферах життєдіяльності людини. Інтернет став невід'ємною складовою повсякденного життя людей, що дозволяє їм швидко та ефективно знаходити потрібну інформацію, здійснювати різноманітні операції та виконувати завдання, які раніше займали значну кількість часу та зусиль.

Обрана тема є актуальною у зв'язку із розвитком та збільшенням доступності технологій, відповідно і зростанням потреби у розширенні зони пошуку клієнтів, які потребують певної юридичної допомоги. Юридична допомога у наш час є невід'ємною частиною життя суспільства, оскільки усі права та обов'язки особи, певні правила співіснування регулюються саме юридичними приписами, які тим чи іншим чином порушуються, а тому, для захисту і поновлення порушених прав, й виникає потреба в існуванні компаній, що надають адвокатські послуги. Зважаючи на це, використання можливостей для розширення клієнтської бази шляхом пошуку потенційних клієнтів, за допомогою розміщення web-сторінки в інтернеті, є важливою умовою ефективної та успішної роботи фірми.

Однак, не всі адвокатські компанії мають достатній рівень інформаційної підтримки та доступності в Інтернеті, що може стати перешкодою для залучення нових клієнтів та розширення сфери юридичної практики. Тому, створення програмного комплексу з web-застосуном для надання адвокатських послуг, та web-сайтом реклами послуг фірми, може стати ефективним рішенням для покращення взаємодії з клієнтами та забезпечення швидкого й зручного доступу до інформації про послуги та професійні здібності працівників компанії.

У рамках даної дипломної роботи буде проведено дослідження та аналіз існуючих web-сайтів, що слугують платформою для пошуку клієнтів, розроблено концепцію web-застосунку для адвокатської компанії та здійснено реалізацію його функціоналу з використанням сучасних інструментів і технологій. Також будуть проведені тестування та оцінка ефективності розробленого web-застосунку.

Отримані результати дослідження та розробки будуть корисними для адвокатських компаній, які бажають розширити свою клієнтську базу через Інтернет, та забезпечити ефективну роботу з клієнтам за допомогою розробленого web-застосунку.

Метою дипломного проекту є розробка web-сайту для просування адвокатських послуг, та web-застосунку для взаємодії з базою клієнтської інформації в перевагу над звичними офісними способам.

РОЗДІЛ 1
ФУНКЦІОНУВАННЯ WEB-ЗАСТОСУНКУ

1.1. Веб браузер

Веб переглядач – це програмне забезпечення для комп'ютера, смартфону, планшету, ноутбука чи іншого електронного пристрою з підтриманою операційною системою, яка дає змогу його відкрити й у повній мірі ним користуватися, під'єднаного до глобальної мережі Інтернет, що дає можливість користувачу взаємодіяти з блоками тексту, малюнками, посиланнями на інші web-сторінки чи окремими розділами на поточній сторінці, на гіпертекстовій web-сторінці тощо.

Браузер відображає web-сторінки на екранах користувачів, логіка яких зосереджена на окремому віддаленому сервері. Основною ж його функцією є відображення завантаженої інформації по інтернет мережі з сервера та передача даних, отриманих від користувача у ході користування web-застосунком, назад до серверу для подальшої обробки та використання отриманої інформації.

Тобто, веб переглядач з'єднується з сервером по протоколу HTTP / HTTPS та отримує від нього документ, який надалі форматує для представлення користувачеві або ж намагається викликати зовнішнє програмне забезпечення, яке обробить отриманий документ, залежно від отриманого формату документа.

Адресування сторінок є ключовим елементом інтернету, а це забезпечується за допомогою URL, який інтерпретується як адреса, що починається з http: для протоколу HTTP. Однак багато навігаторів також підтримують інші типи URL та їх відповідні протоколи, такі як Gopher (ієрархічний протокол гіперпосилань), ftp: для протоколу перенесення файлів FTP, rtsp: для протоколу потоків реального часу RTSP, та HTTPS (HTTP Secure, що розширює HTTP за допомогою Secure Sockets

Layer SSL або Transport Layer Security TLS) HTTPS є особливо важливим протоколом, оскільки він забезпечує безпеку та захист конфіденційної інформації, наприклад: номерів кредитних карток чи паролів.

Окрім того, браузери можуть використовувати інші протоколи, такі як WebSocket, який забезпечує двостороннє зв'язку між клієнтом та сервером. Використання різних протоколів може різнитися залежно від призначення веб-сайту. Наприклад, протокол FTP є особливо важливим для веб-сайтів, які забезпечують доступ до великих обсягів даних або програмного забезпечення для завантаження.

Стосовно протоколів HTTP (Hypertext Transfer Protocol) та HTTPS (Hypertext Transfer Protocol Secure) варто зазначити, що вони є протоколами передачі даних по мережі Інтернет, проте не є однаковими, а це визначає наявність між ними декількох відмінностей. Основна відмінність між HTTP та HTTPS полягає в тому, що HTTPS забезпечує безпеку передачі даних через шифрування. При використанні HTTPS дані шифруються за допомогою SSL (Secure Socket Layer) або TLS (Transport Layer Security), що дозволяє захистити дані від прослуховування та зловживання з боку зловмисників.

Крім того, HTTPS використовує інші порти: порт 443 замість порту 80, який використовується для HTTP. Це дає змогу підтримувати безпеку передачі даних, навіть якщо HTTP трафік відкрито для перехоплення.

Формати документів, які веб браузер повинен обробляти без використання сторонніх програмних забезпечень, визначає консорціум W3C, що являють собою головну міжнародну організацію, діяльність якої полягає у розробці та впровадженні нових технологічних стандартів для Всесвітньої павутини.

До основних форматів отримуваних текстових файлів від серверу належать HTML та XHTML. Це стандартизовані мови розмітки документів для перегляду web-сторінок у браузері, які інтерпретують отриманий код з файлу у відображений інтерфейс на екрані монітору.

Найпопулярнішими браузерами будуть наступні:

- Google Chrome – безкоштовний веб браузер, розроблений компанією Google, на основі іншого браузеру, який має відкритий код, Chromium; Він є абсолютним лідером за популярністю у світі, займаючи біля 60% ринку;
- Mozilla Firefox – ще один безкоштовний веб-браузер з відкритим кодом, що розробляється організацією Mozilla. Firefox має високу швидкість роботи, стійкість та безпеку, тому користувачі довіряють йому, він займає більше 5% ринку;
- Safari – веб-браузер, що розробляється компанією Apple, доступний тільки для операційних систем macOS та iOS. Його популярність становить близько 15% ринку;
- Microsoft Edge – веб-браузер компанії Microsoft, який з'явився на зміну Internet Explorer. Edge має високу швидкість та стійкість, а також багато корисних функцій, зокрема і захист від шпигунства. Він займає більше 5% ринку.

Кожен з цих браузерів має свої переваги та недоліки, і кожен користувач може вибрати той з них, що найбільше відповідає потребам та вимогам особи. Незалежно від вибору браузера, Web-застосунки забезпечують безперервний та зручний доступ до різних сервісів та інформації, оскільки сучасним рішенням розробки Web-застосунків є те, що вони повинні бути крос браузерні, а це, в свою чергу, зробило їх невід'ємною частиною життя сучасної людини.

1.2. Web-застосунок

Web-застосунки дозволяють користувачам взаємодіяти зі змістом в Інтернеті у режимі реального часу, що робить їх надзвичайно популярними серед користувачів та бізнес-спільноти. Ці застосунки можуть бути розроблені для різноманітних платформ, таких як мобільні пристрої, планшети та комп'ютери, що дозволяє їх використання на будь-якому пристрої з доступом до Інтернету.

Нові технології та фреймворки дозволяють розробникам швидше створювати та підтримувати Web-застосунки. Наприклад, деякі з найпопулярніших фреймворків для розробки Web-додатків, такі як Laravel, React та Angular, пропонують широкі можливості для швидкої розробки високоякісних та ефективних додатків.

Крім того, Web-застосунки можуть бути підключені до інших сервісів та додатків, що дозволяє їм взаємодіяти з різними інформаційними системами. Наприклад, Web-застосунки можуть використовувати різноманітні API для отримання даних з інших додатків, зокрема соціальних мереж, електронної пошти тощо.

У зв'язку із розвитком технологій та зростанням популярності Інтернету, Web-застосунки стають все більш важливими для підтримки різноманітних бізнес-процесів та з метою забезпечення доступу до інформації для користувачів у всьому світі.

Однією з ключових переваг Web-застосунків є можливість автоматичного оновлення, що дозволяє користувачам завжди мати доступ до останніх версій та функцій програми без необхідності ручного оновлення. Оскільки всі зміни будуть застосовані до файлів на сервері, а це, відповідно до вищевказаного означає, що браузер просто інтерпретує отримані файли у гіпертекстові web-сторінки, тому користувачу не потрібно нічого самостійно оновлювати та завантажувати. Це полегшує підтримку та управління додатками для розробників та компаній, оскільки вони можуть вносити зміни та вдосконалення безпосередньо на сервері, а ті, в свою чергу, будуть негайно доступні для усіх користувачів.

Безпека є не менш важливим аспектом Web-застосунків. Розробники активно використовують різні методи захисту, такі як HTTPS, автентифікація користувачів, рівні доступу, а також технології, які допомагають виявляти та запобігати можливим атакам, зокрема SQL-ін'єкціям та XSS-атакам.

Web-застосунки також відрізняються своєю масштабованістю та гнучкістю. Завдяки хмарним технологіям, серед яких Amazon Web Services (AWS) та Google Cloud Platform (GCP), компанії можуть легко розширювати свої ресурси відповідно

до зростання кількості користувачів та запитів. Це забезпечує стабільну роботу додатків навіть під час пікових навантажень.

Крім того, адаптивний дизайн та сучасні web-застосунки стають все більш поширеними, що дозволяє користувачам отримувати практично однаковий досвід користування як на десктопних комп'ютерах, так і на мобільних пристроях. Це забезпечує зручність та універсальність використання web-застосунків незалежно від типу пристрою.

Наступною перевагою web-застосунків є відкриті стандарти й уніфіковані технології, що сприяють співпраці та інтеграції різних платформ та сервісів. Використання таких стандартів як HTML, CSS та JavaScript, дозволяє розробникам створювати сумісні рішення, що сприяють інноваційному розвитку в галузі.

Інтеграція з іншими технологіями, до яких відносять штучний інтелект, машинне навчання та блокчейн, відкриває нові можливості для розвитку web-застосунків та покращення їх функціональності. Використання цих технологій може допомогти компаніям створювати більш інтелектуальні та автоматизовані сервіси, що значно покращать рівень задоволеності користувачів.

Web-застосунки забезпечують можливість для міжнародної співпраці та комунікації, оминаючи мовні та культурні бар'єри. Це відкриває нові можливості для глобального бізнесу, освіти та культурного обміну між різними країнами та регіонами.

Враховуючи все вищезазначене, можна стверджувати, що web-застосунки відіграють важливу роль у сучасному цифровому світі та стають невід'ємною частиною життя багатьох людей і компаній. Їх використання забезпечує зручний та ефективний доступ до інформації і сервісів, забезпечуючи зв'язок між користувачами та організаціями.

1.3. Структура web-застосунків

Web-застосунки складаються з трьох основних компонентів, а саме:

- клієнтської частини (client-side);

- серверної частини (server-side);
- бази даних (database).

Клієнтська частина – це те, що користувачі бачать та з чим взаємодіють через власний встановлений браузер. Клієнтська частина складається з HTML-коду, CSS-стилів та JavaScript-скриптів.

HTML – це мова розмітки, яка використовується для створення структури та наповнення вмістом веб-сторінок. HTML-код відповідає за структуру та відображення контенту на сторінці. HTML-код є основою будь-якої веб-сторінки та дозволяє створювати зручну та логічну структуру сторінки, що буде легко зрозумілою користувачам. Крім того, HTML-код дозволяє забезпечити доступність веб-сторінок для людей з обмеженими можливостями, зокрема слабозорих, сліпих тощо.

CSS – це мова стилів, яка використовується для візуального оформлення web-сторінок. CSS-стилі відповідають за оформлення сторінки та зміну зовнішнього вигляду елементів на сторінці, базуючись на написаних розробником правилах. Один CSS-стиль може використовуватися для безлічі елементів на сторінці. Це дозволяє зберігати час та ресурси, оскільки стилі можуть бути повторно використані на цій ж сторінці або навіть на інших веб-сторінках. CSS-стилі також дозволяють створювати анімацію та інші візуальні супроводжувальні ефекти на сторінці, що робить веб-сторінки більш інтерактивними та цікавими для користувачів. CSS-стилі можуть бути динамічно змінені з використанням JavaScript, що дозволяє створювати більш складні та привабливі ефекти на сторінці. Оформлення веб-сторінок з використанням CSS-стилів дозволяє створювати структурований, зручний та логічний дизайн сторінки, яка буде легко зрозумілою, зручною у використанні користувачам.

JavaScript – це мова програмування, яка використовується для створення динамічних та інтерактивних веб-сторінок. Скрипти дозволяють створювати різноманітні інтерактивні ефекти та функції на сторінці, керувати вмістом та відображенням контенту користувача. JavaScript може взаємодіяти з клієнтською та

серверною частинами веб-застосунків. Це дозволяє створювати різноманітні ефекти на сторінці: переходи, зміна стилів, зміна зображень та інші.

Серверна частина – це той компонент web-застосунку, який забезпечує логіку роботи, обробку даних та взаємодію з базою даних. Найпопулярніші мови програмування для серверної частини веб-застосунків - це PHP, Python та Java.

- PHP є однією з найбільш популярних мов програмування для створення веб-застосунків та широко використовується при роботі з WordPress, Drupal й іншими системами управління контентом;
- Python використовують для створення веб-застосунків та машинного навчання; він також відомий своєю простотою та зрозумілістю коду;
- Java також використовується для розробки веб-застосунків, але більш поширено для розробки мобільних додатків та ігор.

Окрім наведених вище мов програмування, також існують нові технології, які дозволяють створювати веб-застосунки з використанням JavaScript на серверній стороні. Node.js є однією з найпопулярніших технологій для створення серверної частини веб-застосунків з використанням JavaScript. Це дозволяє використовувати JavaScript на серверній стороні та забезпечувати високу швидкість і ефективність роботи.

База даних є однією з основних складових веб-застосунку, що забезпечує зберігання, організацію та доступ до даних, необхідних для роботи та підтримки застосунку. База даних дозволяє веб-застосунку ефективно зберігати та обробляти велику кількість даних, а це забезпечує швидшу та ефективнішу роботу web-сайту. Існують різноманітні типи баз даних, але два серед них є найбільш популярними - це реляційні та NoSQL бази даних.

Реляційні бази даних зберігають дані у вигляді таблиць, пов'язаних між собою за допомогою ключів. Вони забезпечують повноцінну структуру для зберігання та організації використаних даних при роботі з web-додатком, що є надзвичайно важливим для великих організацій та компаній.

NoSQL бази даних, в свою чергу, не використовують табличну структуру для зберігання даних, а використовують документи, колекції та ключ-значення. Вони

забезпечують більш гнучкий підхід до зберігання та обробки даних, що робить їх ідеальним вибором для веб-застосунків з високою швидкістю обробки невеликих даних.

1.4. Огляд сучасного стану адвокатських послуг

Адвокатські послуги є невід'ємною складовою справедливого і безпечного життя у суспільстві. У сучасному світі, де правові норми та регуляції постійно змінюються, адвокати відіграють важливу роль у захисті прав та інтересів своїх клієнтів. Однак, як і в будь-якій іншій галузі, існують певні проблеми та виклики, з якими стикаються адвокати.

Однією з основних проблем є доступність адвокатських послуг. У багатьох країнах, зокрема в Україні, доступність адвокатських послуг є досить низькою через низьку правову обізнаність. Це може призвести до того, що потенційний клієнт не зможе отримати якісний захист своїх прав та інтересів, адже не зможе знайти кваліфікованого, досвідченого захисника. Клієнт просто не може дізнатися про те, що існує така фірма, яка займається подібними до його випадку справами, через те, що адвокатські офіси зазвичай надають консультації лише клієнтам, які звернулися до них особисто. Також ця проблема з'являється через локальну рекламу: у маленькому містечку люди будуть знати лише про кілька місцевих адвокатів саме через те, що не мають змоги ознайомитися з послугами та досягненнями правових фірм в сусідньому місті.

Додатковою проблемою, з якою стикаються адвокатські компанії, є брак довіри громадян до кваліфікованості адвоката. Це пов'язано зі скандальними випадками надання низької якості послуг, корупцією та іншими недоліками. Низький рівень довіри може негативно вплинути на сприйняття адвокатської професії громадськістю та призвести до зменшення популярності послуг, а також заплямування репутації адвокатури, оскільки у клієнта виникатиме негативне враження, своєрідна недовіра, яку зможе виправити лише прозоре портфоліо компанії, з усіма справами, над якими вона працювала та нині працює.

Саме тому впровадження інформаційних технологій у правову практику та повсякденне життя дає змогу розширити клієнтську базу, адже клієнти можуть переконатися у кваліфікованості працівників, ознайомитися з їхнім досвідом робіт, послугами, які надають фірми, а також безпосередньо підібрати саме ту компанію, що є найбільш обізнаною у їхньому випадку. Це, в свою чергу, допоможе клієнтам знайти кваліфікованого та досвідченого адвоката, а відповідним компаніям збільшити дохід за рахунок збільшення потоку справ.

1.5. Огляд ринку компаній що надають адвокатські послуги

Український ринок юридичних послуг є конкурентним та розвиненим. За даними Асоціації юристів України, існує близько 4000 юридичних фірм, які надають різні види юридичних послуг.

Однією з провідних компаній на ринку є Sayenko Kharenko, яка має широкий діапазон послуг та займається практикою у багатьох галузях, таких як: фінанси, податки, міжнародний бізнес тощо. Іншими провідними юридичними компаніями в Україні є Asters, Arzinger, Baker McKenzie Kyiv, DLA Piper Ukraine та інші. Багато з цих компаній мають партнерство з провідними європейськими та американськими юридичними компаніями, а також надають послуги на міжнародному рівні. На ринку юридичних послуг в Україні діють компанії, що спеціалізуються у певних сферах діяльності, зокрема IT, фармацевтика, енергетика тощо. Наприклад, компанія Baker McKenzie Kyiv має практику в галузі інформаційних технологій, в той час як компанія GOLAW спеціалізується на фармацевтиці та медичній галузі.

Отже, для створення конкурентної фірми потрібно створити спілку, яка буде включати в себе професійних, досвідчених та кваліфікованих адвокатів, що працюють у певній конкретній галузі, щоб у групі можна було охопити увесь або майже увесь діапазон галузей.

Також важливо стежити за новими тенденціями та розвитком технологій у галузі юридичних послуг. Наприклад, зараз на ринку дедалі більше компаній переходять до використання розумних технологій та штучного інтелекту, що значно

спрощує та прискорює процеси надання юридичних послуг. Фірми, які зможуть успішно впровадити ці нові технології, зможуть отримати конкурентну перевагу на ринку.

Крім того, на ринку послуг важливо зберігати високий рівень професійної етики та дотримуватися найвищих стандартів якості надання послуг. Це допомагає зберегти репутацію компанії та забезпечити досягнення поставлених клієнтами завдань, що, в свою чергу, призводить до збільшення клієнтського потоку і доходу компанії.

Оглянувши ринок компаній, які надають адвокатські послуги, можна дійти висновку, що створення власного web-сайту дозволить збільшити притік клієнтів у кілька разів, а це є позитивним результатом для власників даних компаній. Тому створення web-сайту для просування послуг власної компанії, є обов'язковим елементом відкриття власної фірми.

Проте порівнюючи web-сайти, які знаходяться в мережі інтернет, можна зіткнутися з великою різноманітністю дизайнерських рішень, які, на жаль, не можуть бути визнані найкращими.

Основним недоліком зазначених дизайнерських оформлень є значне перенасичення користувацького інтерфейсу інформацією, різноманітними блоками та спливаючими вікнами, які зазвичай не несуть жодної користі користувачу, окрім візуального шуму. Такі web-сайти не користуються значною популярністю, відповідно і втрачають клієнтів, оскільки їхній інтерфейс не є дружнім до користувача: особі досить важко орієнтуватися в інформації на сторінці, знаходити важливі для себе аспекти та повноцінно ознайомлюватися із послугами, які надає компанія. Клієнт не може переглянути та запевнитись в кваліфікованості працівників фірми, оскільки відсутні розділи з останніми справами компанії.

Тому створення сучасного, легкого та інтуїтивно зрозумілого дизайну web-сайту є основною частиною при розробці. Це дозволить заохотити клієнтів звертатися за послугами до даної фірми, оскільки клієнт зможе швидко розібратися у інтерфейсі, знайти потрібну для себе інформацію починаючи від контактів

компанії та спектру послуг, закінчуючи переліком останніх справ, у яких учасниками були саме адвокати даної фірми.

1.6. Розробка дизайну web-сайту

Розробка дизайну web-сайту є однією з головних складових процесу розробки. Головна мета дизайну - створення естетично приємного, зрозумілого та зручного для користувача інтерфейсу, який забезпечить зручну навігацію, доступність та відображення контенту.

Першим кроком у розробці дизайну web-сайту є створення концепції дизайну. На цьому етапі визначається загальний стиль та особливі елементи дизайну, а також обираються кольори та шрифт. Ключовою метою концепції дизайну є створення візуальної цілісності та забезпечення однакового стилю, настрою на усіх сторінках сайту, створивши при цьому щось унікальне і таке, що запам’ятається користувачеві. Далі необхідно розробити макети дизайну. Це може бути зроблено за допомогою спеціального програмного забезпечення для розробки макетів, таких як Adobe Photoshop або ж Figma. На цьому етапі визначається структура та розташування загальних елементів web-сайту: меню, кнопки, блоки тексту, зображення, посилання. Розробники також додають інші дизайнерські елементи, такі як графічні ефекти, анімація та трансформації елементів, аби забезпечити більш ефективну інтерактивність та заохотити користувача скористатися послугами web-сайту.

Останнім етапом у розробці дизайну web-сайту є тестування та відлагодження. Розробники перевіряють, чи правильно працює інтерфейс, чи забезпечує коректну роботу анімацій, переходів по розділам та доступність контенту. Відлагодження включає в себе: виправлення помилок, багів та за потребою додавання нових функцій, які допоможуть покращити користувацький досвід. У кінці процесу дизайну, дизайнери повинні перевірити, як виглядає сайт на різних пристроях та різних браузерах, аби переконатися, що дизайн працює правильно та виглядає однаково на усіх пристроях.

Розроблений дизайн повинен бути повністю адаптивним і забезпечувати коректне і повне відображення інформації незалежно від типу пристрою, за допомогою якого користувач переглядає web-сторінку, увесь контент повинен адаптуватися під розміри дисплею пристрою: чи це буде планшет, чи комп'ютер, мобільний телефон або телевізор. Інтерфейс повинен повністю забезпечувати інтуїтивну підтримку користувача, а саме можливість користувача, вперше зіткнувшись з меню, функціями чи іншими елементами web-сайту, відразу збагнути, як саме ним користуватися. Він не повинен бути перенаповнений непотрібною інформацією, яка лише заважає користувачу та втомлює око, створюючи візуальний шум. Дизайн має бути простим, але зі своєю певною унікальністю, відмінністю від усіх інших продуктів конкурентів.

Підсумовуючи, розробка дизайну є складним та важливим процесом, який вимагає великого досвіду та знань у галузі web-дизайну. Крім того, важливо пам'ятати, що потреби та очікування користувачів є пріоритетними, оскільки задоволення їхніх потреб - це ключовий фактор успіху будь-якого web-сайту, що принесе збільшення потоку клієнтів.

1.7. Обґрунтування створення web-сайту власної компанії

Створення web-сайту для власної компанії є обов'язковим кроком для розвитку та збільшення прибутку бізнесу. На підтримку даної думки слугуватимуть наступні обґрунтування:

Задання власної репутації – клієнт, який не може знайти web-сайт компанії або ж він є застарілим та невпорядкованим, отримає думку про низьку компетентність фірми, оскільки вона не може мати власну хорошу web-сторінку. Він сприйматиме це за недбале ставлення до діяльності. Відповідно, клієнти будуть віддавати перевагу компаніям, у яких якісний, функціональний та повноцінний web-сайт, оскільки це зазвичай свідчить про високу якість наданих послуг фірмою. Також відсутність власного web-сайту призведе до складнощів у просуванні компанії на ринку послуг.

Зручність – web-сайт робить бізнес більш масштабним, оскільки клієнти можуть дізнатися про фірму не лише з місцевої реклами, а й знайти сторінку в мережі Інтернет, що дає змогу розширити потік клієнтів за допомогою людей з інших міст. Оскільки клієнти можуть ознайомитися зі спектром послуг, практикою та кваліфікованістю працівників не залишаючи місця свого перебування, а просто відкривши інтернет сторінку компанії.

Конкуренція – більшість компаній, що надають певні послуги, мають хороші, структуровані web-сайти, тому відсутність власного сайту може призвести до того, що клієнт раніше знайде web-сайт конкурентів в Інтернеті, аніж дізнається про вашу компанію, яку рекламують лише місцеві оголошення, і скористається їхніми послугами. А це, в свою чергу, призведе до спаду потоку клієнтів, відповідно і доходу фірми.

Прибутковість – враховуючи усі попередні аргументи, можна дійти висновку, що web-сайт безпосередньо впливає на збільшення потоку клієнтів завдяки можливості охопити більшу аудиторію, відповідно і збільшує прибуток компанії. Відкритий доступ до інформації про послуги компанії допомагає залучити нових клієнтів, що лише позитивно впливає на фінанси.

Крім того, web-сайт може бути корисним інструментом для збору та аналізу даних про клієнтів та їхніх потреб. За допомогою аналітичних інструментів, які можуть бути вбудовані в web-сайт, компанія може отримувати цінні дані про поведінку своїх клієнтів на web-сторінці, що дозволяє покращити якість наданих послуг та оптимізувати маркетингові та рекламні стратегії, базуючись на найпопулярніших розділах.

Також, web-сайт дозволяє компанії взаємодіяти з клієнтами та отримувати від них зворотний зв’язок. Тобто на сторінках сайту можуть бути розміщені форми зворотного зв’язку, чат-боти, контакти менеджерів та інші інструменти, які дозволяють клієнтам легко зв’язатися з компанією та отримати необхідну інформацію, записатись на консультацію. Додатково на сайті можна розмістити блог або новини, що дозволяє компанії вести активну комунікацію з клієнтами та ділитися корисною інформацією про останні події, що відбулися у межах самої

фірми. На прикладі адвокатського бюро будуть відображені новини про останні справи, які вела фірма та посилання на їх рішення з єдиного державного реєстру судових рішень.

Отже, створення web-сайту власної фірми, в епоху цифрових технологій, є необхідністю. Рекламування та просування власних послуг через Інтернет стане одним із основ для побудови позитивної репутації, залучення нових клієнтів та збільшення прибутку компанії.

1.8. Функціональні та технологічні можливості web-сайту

Функціональні та технологічні можливості web-сайту кожної компанії можуть бути різними, залежно від конкретних потреб, бажань та цілей клієнтів. Однак в загальному можна виділити деякі можливості, які є обов'язковими для будь-якого web-сайту.

Однією із обов'язкових функцій web-сайту є надання користувачу інформації про компанію, її послуги та персонал. Це можуть бути опубліковані описи спектру послуг, які надає компанія, інформація про спеціалізацію працівників, в яких галузях вони мають досвід та компетенцію у наданні послуг. У випадку адвокатської компанії обов'язковими розділами є розділи із адвокатами компанії, переліком та коротким описом галузей, в яких вони працюють.

Аналізуючи наявні web-сайти адвокатських компаній, можна помітити що жодна сторінка не передбачає розділ з історією справ компанії, аби клієнт міг ознайомитись з рівнем успішності фірми: які рішення приймались у суді за участі цих адвокатів, з яким відсотком ці рішення були успішними для клієнтів даної фірми тощо. Цей розділ являтиме головне нововведення що відрізнятиме розроблений web-сайт від ринкових аналогів.

Ще однією важливою функцією web-сайту є забезпечення комунікації між клієнтом та менеджером для розгляду ситуації клієнта відповідним кваліфікованим адвокатом фірми, а також подальшим записом на консультацію, збір необхідних документів, супровід у суді тощо. На web-сайті це реалізовується за допомогою

форми зворотного зв’язку та блоку контактної інформації про те, як знайти адресу фірми на мапі, контактні номери телефонів менеджерів фірми, email. Крім того, можна встановити чат-бота або інші сучасні засоби онлайн-комунікації з клієнтом, які допоможуть швидко відповісти на популярні запитання клієнтів та надати йому певну інформацію, яка його цікавить, після перегляду послуг компанії. Також сюди можна віднести посилання на соціальні мережі чи месенджери, адже це забезпечить зручний та швидкий спосіб зв’язку з клієнтом та можливість надати первинну консультацію в онлайн-режимі.

Додатковою функцією для web-сайту масштабної міжрегіональної адвокатської компанії стане пошук адвокатів за регіоном клієнта, тобто клієнт зможе відфільтрувати список робітників за їх місцем знаходження, залежно від потрібного йому, та звернутися до менеджера з повідомленням про потребу саме у послугах цього адвоката.

Блок контактної інформації є невід’ємною частиною web-сайту, адже ця інформація допомагає клієнтам зв’язатися з компанією. На сторінці повинні бути вказані усі можливі засоби зв’язку, починаючи від контактних номерів телефонів, закінчуючи посиланнями на сторінки в соціальних мережах, де будуть публікуватися певні новини компанії, які отримуватимуть підписники.

Проте потрібно пам’ятати про дизайн та головне правило: «не перенасичувати сторінку контентом та непотрібним функціоналом», тобто необхідно аналізувати потреби користувачів та приймати рішення на основі зібраної статистики. Саме тому варто дотримуватися ідеї простого структурованого web-сайту.

Зі сторони адвокатської частини програмного комплексу, web-застосунок повинен містити можливості опрацьовувати базу даних клієнтів та містити можливості занотовувати інформацію, оскільки це є значно зручнішим, аніж звичайні традиційні варіанти. Наприклад: адвокат має можливість переглянути якісь певні нотатки по справі, перелік документів наданих клієнтом, встановити нагадування про певну подію. З розвитком технологій це стало набагато комфортніше використовувати у повсякденному житті, оскільки доступ до інтернету є майже всюди, а, наприклад, записник чи блокнот з нотатками по справі

завжди можна забути в офісі чи у житлі, також, через перенасичення інформацією, адвокат може забути про заплановану зустріч. Тому вирішенням даної проблеми стануть нагадування.

Web-застосунок працює з базою даних клієнтів, використовуючи різні технології та підходи до реалізації. Наприклад, мову запитів SQL для управління та створення реляційної бази даних, в якій кожна таблиця відображатиме різні аспекти інформації про клієнта, а саме: ім’я, прізвище, контактну інформацію, інформацію про замовлені послуги, стан справи та інше. Також є метод, який використовує NoSQL бази даних, що дозволяє зберігати та організовувати інформацію у форматі документу. Зв’язок забезпечується за допомогою інтерфейсу в окремо розробленому web-застосунку, від основного web-сайту з рекламою послуг, до якого доступ мають лише авторизовані адвокати фірми, які входять у систему під власними логінами та паролями. Ввівши правильні дані, адвокат потрапляє у інтерфейс, в якому відображаються усі його активні клієнти та певні нотатки з календарем і відміткою про заплановану зустріч чи консультацію.

1.9. Висновки до розділу 1

В Розділі 1 було розглянуто ключову структуру та визначення web-додатків та web-сайтів, їх дизайнерське оформлення, функціональні можливості, наведено аргументи для обов’язкового створення власного web-сайту компанії, проаналізовано ринок наявних web-сайтів адвокатських компаній. Висновки що випливають із підпунктів даного розділу включають в себе рекомендації щодо покращення та розробки індивідуального дизайну web-сайту, аналізу помилок сайтів – аналогів та їх вирішення при розробці, а також розширення функціональних можливостей для користувачів.

Метою кваліфікаційної роботи, що підкріплена наведеним вище обґрунтуванням, є розробка програмного комплексу з клієнтської частини (web-сайт) та адвокатської (web-додаток). Даний комплекс призначений для створення можливості просування надання адвокатських послуг компанією за допомогою web-

сайту, з новими функціями для клієнтів у вигляді можливості переглядати перелік нещодавніх справ, над якими працює фірма, так і повноцінного web-застосунку, в якому адвокати зможуть працювати над інформацією клієнтів що знаходиться в базі даних фірми. Процес розробки передбачає створення повноцінного крос-браузерного web-сайту з унікальним дизайном, який вирізняється своєю лаконічністю та запровадженням низки додаткових функцій, та web-застосунку, який можна використовувати на будь-якому пристрої за допомогою мережі Інтернет.

РОЗДІЛ 2
МЕТОДИ ТА ЗАСОБИ РЕАЛІЗАЦІЇ ПРОГРАМНОГО КОМПЛЕКСУ

Аналізуючи поставлену задачу та методи її вирішення, було вирішено розробити програмний комплекс на основі web-технологій, що включатиме до себе дві частини : клієнтську, та окрему частину розроблену під потреби адвокатів для роботи з базою даних клієнтів. Головною перевагою web-застосунків перед всіма іншими можливими варіантами є універсальність, яка виражається у можливості користуватись web-сайтом на будь-якому пристрої без цільового перенесення на конкретну операційну систему, оскільки браузер з його віртуальною машиною і є цільовою універсальною операційною системою, що в свою чергу дає змогу використовувати додаток на будь-якому пристрої що має змогу відкривати web-браузер. Також варто відзначити підтримуваність, яка виражається в тому, що для розширення функціоналу, виправлення помилок чи зміни інформації, оновлення web-сайту потрібно вносити лише на сервері, клієнту ж в свою чергу нічого робити не потрібно, оскільки браузер автоматично відобразить нову версію з серверу.

2.1. Вибір архітектури програмного комплексу

Цільовий програмний комплекс було вирішено побудувати на основі двох архітектур, в залежності від частини, клієнтської чи адвокатської. Для клієнтської частини використовуватиметься дволанкова архітектура, яка складається з наступних компонентів: клієнт та сервер. Схему даної архітектури зображено на рисунку 2.1.

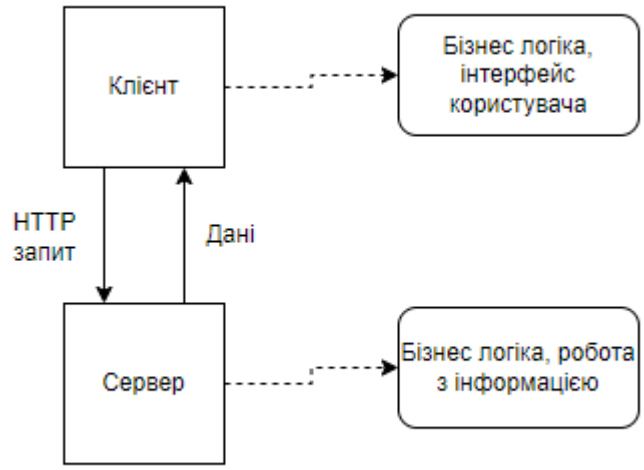

Рис. 2.1. Дволанкова архітектура клієнтської частини

В свою чергу, адвокатська частина web-додатку буде побудована на основі триланкової архітектури: різницею між дволанковою є лише те, що у зв’язок додається база даних, що відображено на рисунку 2.2.

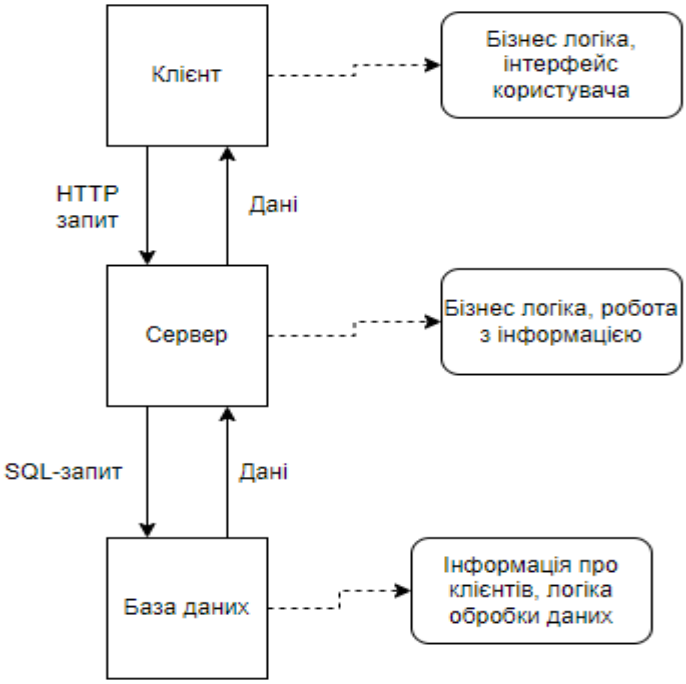

Рис. 2.2. Триланкова архітектура адвокатської частини

Користуючись web-сайтом, клієнт взаємодіє з реалізованим інтерфейсом, за допомогою якого й відбувається обробка даних користувача, виведення для перегляду опрацьованих результатів від серверу та інші дії. Web-сайти зазвичай надають клієнтам інтерфейс для взаємодії з різноманітними функціями та послугами: коли клієнт відкриває web-сторінку, він може бачити різні елементи інтерфейсу, зокрема і текстові поля, кнопки, меню, елементи форми тощо. Це дозволяє йому взаємодіяти з програмою, запущеною на сервері. Клієнт може ввести дані у текстові поля, вибрати певні параметри за допомогою перелічувальних списків або перемикачів, а також натиснути на кнопки для ініціювання певних дій на сервері. Ці дані та параметри надсилаються на сервер для обробки - на сервері розташована програма, яка отримує дані від клієнта, обробляє їх і повертає результати назад клієнту. Обробка даних може включати розрахунки, перевірку, збереження в базі даних, виведення інформації, отриманої з бази даних, або інші операції, які відповідають функціональності web-сайту. Отримані результати передаються клієнту, і вони можуть бути відображені на web-сторінці, або отримані скриптом для подальших дій над ними. Результати можуть бути представлені у вигляді тексту, таблиць, графіків або інших елементів, залежно від їх призначення. Цей процес взаємодії між клієнтом і сервером відбувається за допомогою комунікаційного протоколу, зокрема такого як HTTP, який дозволяє передавати дані між клієнтом і сервером через глобальну мережу Інтернет.

Головною ланкою в даних архітектурах програмного комплексу виступає сервер, що забезпечує роботу усієї системи. Коли сервер отримує HTTP запит від веб-браузера користувача, він обробляє цей запит і генерує відповідь, яка містить дані для відображення на веб-сторінці. Основний процес виглядає наступним чином:

- Отримання запиту: Сервер отримує HTTP запит від браузера клієнта, даний запит містить інформацію про дію, яку користувач хоче виконати, наприклад: отримання даних сторінки, відправлення форми, завантаження файлу;

- Маршрутизація: Сервер використовує маршрутизацію для визначення того, який обробник події, тобто функція чи метод, повинен бути викликаний для обробки отриманого запиту від web-переглядача;
- Обробка запиту: Обробник, отримавши запит, виконує відповідні описані у ньому дії. Це може включати доступ до бази даних, виконання певних обчислень або взаємодію з іншими зовнішніми сервісами, зокрема надсилання листа на пошту;
- Генерація відповіді: Після обробки запиту сервер генерує відповідь, яка містить дані, що повинні бути відображені на веб-сторінці. Це може бути HTML-код бажаної сторінки, JSON-дані для обробки чи виведення певної інформації з бази, зображення або інші інформаційні ресурси;
- Надсилання відповіді: Сформована відповідь відправляється назад до браузера за допомогою того ж HTTP-протоколу. Вона містить заголовки, що включають метадані про відповідь, а також тіло, що містить саму інформацію про сформовану відповідь від серверу;
- Відображення веб-сторінки: Браузер отримує відповідь від сервера і використовує отримані дані для відображення інформації на веб-сторінці. Він інтерпретує HTML, CSS та JavaScript код, щоб побудувати інтерфейс користувача і відобразити отримані дані від серверу на екрані пристрою користувача.

Цей процес відбувається швидко, автоматично і практично непомітно для користувача при взаємодії з web-сайтом.

2.2. Опис інструментів для розробки web-застосунку

Оскільки програмний комплекс включає в себе дві різні частини, то й кожну з них було реалізовано з використанням різних технологій з метою створення повноцінного web-застосунку під потреби адвокатської фірми, зокрема у просуненні та рекламі власних послуг за допомогою мережі Інтернет, та створенні бази даних клієнтів, з можливістю вести записи щодо кожного з них окремо.

Для розробки різних рівнів даного комплексу було використано наступний набір технологій.

Клієнтський рівень:

- HTML: Використовується для створення структури блоків, в яких відображається інформація на веб-сторінці. З бібліотекою font-awesome для відображення значків;
- CSS: Використовується для стилізації, оформлення веб-сторінок та надання їм зовнішнього вигляду. З бібліотекою Bootstrap для адвокатської частини web-застосунку;
- JavaScript: Використовується для програмування логіки веб-сторінок, динамічних ефектів та певної взаємодії з користувачем.

Серверний рівень:

- Серверна платформа Open Server: Це набір серверного програмного забезпечення на основі веб-серверу Apache;
- Apache: Веб-сервер, який використовується для обробки запитів веб-клієнтів на основі відправлених даних та доставки оброблених відповідей на них;
- PHP: Серверна мова програмування, яка використовується для обробки запитів, генерації динамічного контенту та взаємодії з базою даних;
- Фреймворк Laravel: Це популярний PHP-фреймворк, який допомагає швидко розробляти веб-додатки, забезпечує структуру, готові компоненти та інструменти для роботи з базами даних і маршрутизацією;
- AJAX : Ця технологія дозволяє взаємодіяти з сервером асинхронно, без перезавантаження всієї веб-сторінки. Вона використовує JavaScript для відправки та отримання даних з сервера без зміни стану сторінки.

Рівень бази даних:

- MySQL: Це популярна відкрита реляційна база даних, яка використовується для зберігання та управління даними, з якими взаємодіє користувач відповідно до отриманого від нього запиту.

Для розробки самого ж коду було використано популярний та доступний текстовий редактор «Visual Studio Code», адже:

1. Visual Studio Code є повністю безкоштовним;
2. Має вбудований сервіс різноманітних розширень, які є дуже зручними, наприклад: Live Server, який дає змогу локально захостити web-сайт;
3. Легко налаштовується та має синхронізацію за акаунтом;
4. Має інтегровану систему роботи з GitHub;
5. Підтримується на багатьох операційних системах;
6. Має підтримку роботи з кількома файлами одночасно в одному вікні за допомогою розділення його на рівні частини;
7. Є портативним через свій розмір, який з встановленими кількома розширеннями становить лише 660 мега байт.

2.3. Гіпертекстова мова розмітки HTML

Гіпертекстова мова розмітки HTML використовується для побудування структури та відображення web-сторінок, з його допомогою визначаються різні елементи, починаючи від параграфів, секцій, елементів меню, зображень, закінчуючи посиланнями, списками тощо.

Структура web-сайту складається з шапки, головної центральної інформації та з нижньої частини. Шапка сайту, так званий header, не змінюється протягом переходів по різним розділам сайту; вона однакова, і зазвичай містить навігаційне меню, логотип компанії та короткі контактні дані. Головна центральна інформація, тобто main, відрізняється за кожним розділом сайту - вона є динамічною та змінюється залежно від розділу, до якого перейшов користувач. Нижня частина web-сайту, тобто footer, схожа за принципом до шапки, оскільки теж є незмінною

незалежно від розділу, на якому знаходиться користувач: чи буде це головна сторінка, чи розділ практики. Footer ж зазвичай містить короткий опис інформації про компанію, корисні посилання та повторення контактної інформації. Для прикладу даного розподілення на блоки, на рисунках 2.3 та 2.4 можна побачити структуру головної сторінки, а саме розділення на header, footer та main.

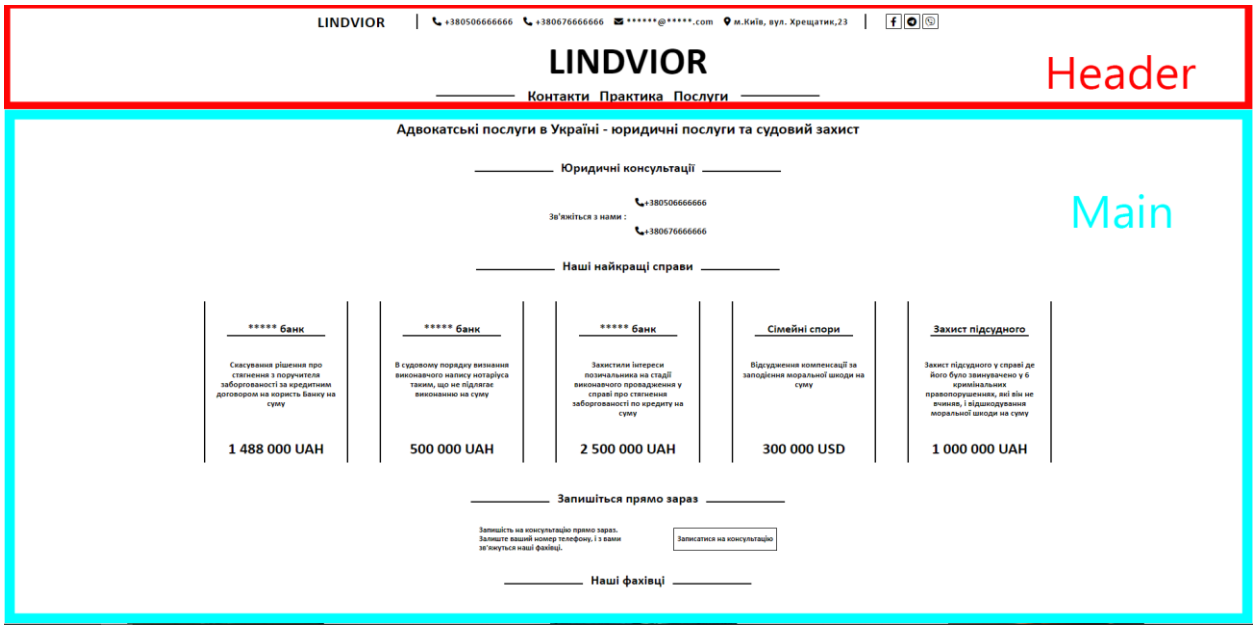

Рис. 2.3. Структура головної сторінки web-сайту

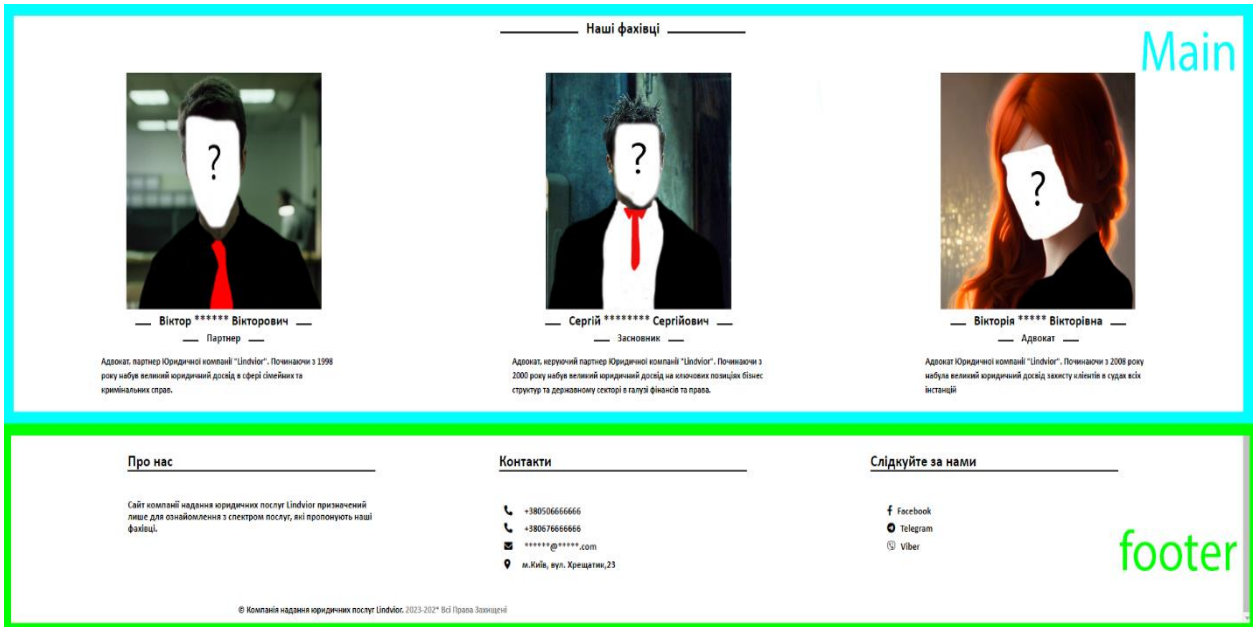

Рис. 2.4. Структура головної сторінки web-сайту

HTML для побудови web-сайтів використовує спеціальні теги, кожен з яких визначає певний елемент на сторінці. Теги являють собою певну розмітку, яка оточує необхідну для відображення інформацію і надає їй спеціальну функцію чи певне значення. Наприклад: для параграфу використовується тег <p>, для заголовку теги <h1…h6>, для зображень тег <img>. Теги також мають власні атрибути, яким розробник задає певні значення, від яких відображення вмісту на web-сторінці може змінюватись. Наприклад: атрибут “title” для згаданого вище тегу зображення <img>, що задає текст, який буде відображено користувачу при наведені на картинку, приклад тексту зображено на рисунку 2.5.

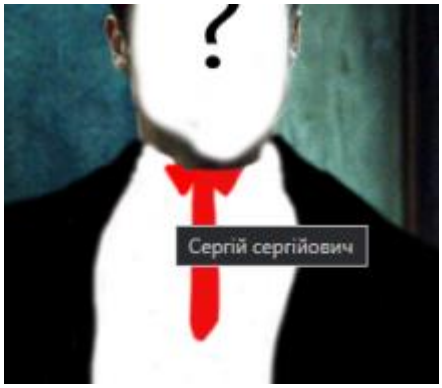

Рис. 2.5. Відображення значення записаного у атрибуті title

Головним та загальним тегом, який в себе і вміщує решту тегів, що будуть відображені на сторінці, являє тег <html>: він означає початок та кінець HTML-документу і являє собою корінний елемент структури сторінки.

Тег має дочірні елементи, такі як <head> та <body>. У <head> розміщуються усі необхідні метадані сторінки: заголовок сторінки, посилання на інші файли, які включаються до даної web-сторінки, файли стилів. Також там розміщуються усі необхідні метатеги для пошукових систем, тобто для опису сторінки та ключових слів. Дані теги допомагають пошуковим системам краще індексувати web-сторінку та просувати її вище у результатах пошуку. У тезі <body> розміщується основний вміст сторінки: текст, форми, зображення та інші елементи, які будуть відображені користувачу, тобто сама структура web-сайту.

При розробці було використано також інтернет бібліотеку іконок, таких як font-awesome. Дана бібліотека є найпопулярнішою для використання web-іконок у проекті, вона надає велику колекцію векторних іконок, які можна частково змінювати під власні потреби проекту, а також практично повністю змінювати при преміум плані підписки. Саме підключення бібліотеки до проекту не займає й п’яти хвилин, оскільки потрібно просто додати тег <link> у секції <head> з унікальним сформованим посиланням у кабінеті користувача. А після підключення, іконки можна вставляти в будь-якому місці сторінки за допомогою елементу <i> чи , задаючи клас “fas” “far” в залежності від типу іконки, і, записавши додаткові класи, налаштовувати вигляд іконок або застосовувати додаткові ефекти.

Дані web-іконки є досить важливими при створені дизайну web-сторінки, оскільки вони додають інформаційної наповненості, а це дозволяє клієнту, який навіть не дочитав усю інформацію, інтуїтивно зрозуміти, до чого відноситься дана інформація. Прикладом може бути блок контактної інформації, що зображено на рисунку 2.6.

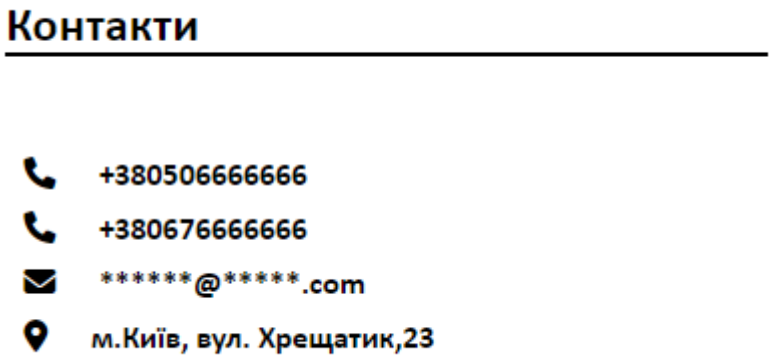

Рис. 2.6. Відображення іконок font-awesome на web-сайті

2.4. Каскадна мова стилів CSS

Кожен web-сайт повинен виділятись своїм унікальним дизайном, який зацікавить користувача залишатись на ньому, та дізнаватись про усю інформацію,

яка на ньому присутня. Для форматування відображення структури web-сайту, сформованої за допомогою HTML-коду у web-браузері, використовують саме CSS.

CSS - це каскадні таблиці стилів, тобто це мова для запису стильового оформлення web-сторінок. CSS визначає зовнішній вигляд елементів HTML, тобто шрифт, колір, розмір, позиціонування, відступи та інші різні візуальні аспекти. CSS використовують для розділення стилю відображення елементів від структури web-сторінки, тобто від HTML, що в свою чергу дає змогу змінювати вигляд елементів сторінки без зміни HTML-коду, а це дозволяє розробнику легко змінювати стиль та вигляд всієї web-сторінки.

Дані таблиці працюють на принципах селекторів - певних правил або наборів правил, які будуть описувати властивості відображення елементів сторінки відповідно до заданих параметрів. Самі ж правила описуються за допомогою селекторів – частина CSS правила, яка повідомляє браузеру, до якого елементу чи елементів web-сторінки потрібно застосувати певні стильові властивості. Формат селектору виглядає наступним чином.

Селектор { властивість: значення властивості; властивість: значення властивості; … }

- Селектор вказує на елемент чи групу елементів, до яких будуть застосовані стилі, тобто селектор може бути ім'ям будь якого тегу, класом, ідентифікатором, псевдокласом або комбінацією цих елементів, у разі чого властивості будуть застосовані до всієї обраної групи елементів;
- Властивість являє собою атрибут, який буде змінюватись в залежності від потрібного відображення на сторінці, наприклад: "background-color" відповідає за колір фону, "font-size" відповідає за розмір шрифту, "color" за колір шрифту, а "font-family" за сам стиль шрифту;
- Значення властивості встановлює конкретну характеристику, яку має набути властивість при відображені на web-сторінці, наприклад: "color: red" встановить червоний колір тексту при відображені елементу у web-

браузері, "height: 12px" встановить висоту блоку у розмір 12 пікселів, і т.д.

Варто відмітити, що CSS також дає змогу забезпечити адаптивність web-сторінки під різні дисплеї пристроїв, зокрема телефонів, планшетів, моніторів. Це збільшує обсяг залучення клієнтів, оскільки вони мають змогу ознайомитися з інформацією, розміщеною на сайті, з будь-якого пристрою.

При розробці адвокатської частини програмного комплексу було також використано популярну бібліотеку стилів Bootstrap. Дана бібліотека надає набір певних готових компонентів, стилів та рішень для швидкого розроблення web-інтерфейсу. Bootstrap дає змогу розробникам використовуючи заздалегідь заготовлені та визначені класи, структури компонентів, створювати різноманітні рішення для користувацького інтерфейсу без необхідності писати багато власних CSS-стилів. Проте використання Bootstrap має також ряд недоліків, таких як:

- Схожість дизайну: За замовчуванням, бібліотека вже має чітко визначений стиль компонентів, який в даний час є доволі впізнаваним, а це, в свою чергу, призводить до того, що багато web-сайтів, які використовують Bootstrap, мають досить схожий вигляд. Тому для розробки саме клієнтської частини проекту і створення власного дизайну web-сторінки дана бібліотека не використовувалась;
- Недостатня гнучкість компонентів: Хоч у бібліотеці й зібрано велику кількість готових стилів, вона не може задовільнити усі потреби при розробці, оскільки існують певні обмеження у налаштуванні вигляду або функціоналу компонентів, а тому доводиться дописувати власні правила CSS селекторів;
- Великий розмір: Другою причиною використання Bootstrap лише у адвокатській частині інтерфейсу став значний розмір бібліотеки. Оскільки вона включає в себе багато стилів, компонентів, це призводить до зменшення швидкості завантаження сторінки, що особливо помітно при роботі з повільним інтернет-з’єднанням;

- Сучасність: Нажаль Bootstrap підтримує не всі версії web-браузерів (статистику відображено на рисунку 2.7), тому користувач на застарілій версії web-браузеру може зіткнутися з багами, некоректним або ж взагалі не відповідним відображенням web-сторінки;
- Залежність від бібліотеки: При використанні Bootstrap для розробки проект стає повністю залежним від даної бібліотеки, а це означає, що у разі, якщо в майбутньому буде вирішено змінити стиль, дизайн, або перейти до використання іншої стильової бібліотеки, це вимагатиме величезних змін у коді проекту. Також при оновлені версії Bootstrap потрібно перевіряти та адаптувати код під нові стандарти.

Рис. 2.7. Підтримка браузерами бібліотеки Bootstrap

Зважаючи на всі недоліки наведені вище, при розробці клієнтської частини web-додатку, яка рекламує послуги компанії, було використано лише власно розроблений CSS-код, що дозволило досягти кількох важливих переваг, а саме:

- Унікальний дизайн: Відсутність Bootstrap дає змогу створити повністю унікальний дизайн, відповідний до ідентичності компанії та послуг, які вона надає;
- Індивідуальний підхід: Використання власного CSS-коду дало змогу створити особливий інтерфейс, який підкреслює особливості дизайну компанії, у власних кольорах, стилях, що дає змогу виділитись серед

фірм-конкурентів. І також дозволило створити власний адаптивний дизайн в залежності від формату дисплею пристрою клієнта;

- Швидке завантаження: Відмова від сторонньої бібліотеки дала змогу уникнути завантаження непотрібного коду користувачами, що позитивно позначилось на швидкості завантаження web-сторінки навіть при повільному інтернет-з’єднанні.

2.5. Мова програмування JavaScript

JavaScript – це динамічна, об’єктно-орієнтована одно поточна мова програмування, реалізована стандартом ECMAScript, що найчастіше використовується саме для створення сценаріїв web-сторінок, надання певної логіки елементам на сторінці, обробки певних подій чи певної взаємодії з користувачем, керування браузером, а також змінювати структуру та зовнішній вигляд web-сторінки.

JavaScript має низку властивостей об'єктно-орієнтованої мови, але завдяки концепції прототипів, підтримка об'єктів в ній відрізняється від традиційних мов ООП. Крім того, JavaScript має кілька властивостей, притаманних функціональним мовам, – функції як об'єкти першого класу, об'єкти як списки, каррінг, анонімні функції, замикання – що додає мові додаткову гнучкість.

JavaScript містить декілька десятків вбудованих об'єктів, які поділяються на групи:

- фундаментальні (Object, Function, Boolean, Symbol);
- помилки (група об'єктів Error);
- числа та дати (Number, BigInt, Math Date);
- текстові (String, RegExp);
- індексовані (група об'єктів Array);
- ключові (Map, Set, WeakMap, WeakSet);
- для роботи з структурованими даними (ArrayBuffer, Atomics, DataView, JSON);

- абстрактні (Promise, Generator).

Крім того, JavaScript містить набір вбудованих операцій, що керують логікою виконання програм. Синтаксис JavaScript в основному відповідає синтаксису мови Java (тобто, зрештою, успадкований від C), але спрощений у порівнянні з ним, щоб зробити мову сценаріїв легкою для вивчення. Так, наприклад, декларація змінної не містить її типу, властивості також не мають типів, а декларація функції може знаходитися в тексті програми після неї. Тобто на прикладі з коду розробленого web-сайту :

const popupClose = document.getElementById('popupClose');

Коли ми декларуємо (оголошуємо) змінну, ми пишемо лише якого вона типу, динамічна тобто let / var, чи константна, тобто const. Ми не вказуємо жодного типу даних для змінної, що є глобальною відмінністю від інших мов програмування, де обов’язково потрібно вказувати тип даних int, string, bool. JavaScript в цьому плані є гнучким та автоматично адаптує змінну під потрібний тип даних.

JavaScript також підтримує ряд важливих можливостей:

- Робота з API: JavaScript часто використовують для роботи з API, завдяки вбудованим функціям, наприклад: “fetch()”, скрипт може отримати дані з серверу, або ж навпаки відправити їх на сервер, що дає змогу створити web-додаток, який буде взаємодіяти з сторонніми сервісами;
- Взаємодія з користувачем: Завдяки різним механізмам, таким як: анімація, валідація полів форми, натискання миші, наведення на елемент, JavaScript дозволяє створювати багатофункціональні та інтерактивні інтерфейси при взаємодії користувача з елементами web-сайту;
- Розробка web-додатків на основі SPA: JavaScript дозволяє динамічно змінювати контент, тобто додавати, приховувати, редагувати, видаляти певний вміст сторінки без перезавантаження сторінки, що являється основою для розробки одно сторінкового web-сайту.

JavaScript має вбудовану підтримку взаємодії з сервером за допомогою технології AJAX, вона дозволяє отримувати дані з сервера та оновлювати вміст

асинхронно та без перезавантаження усієї web-сторінки. Принцип роботи AJAX полягає в тому, що JavaScript відправляє асинхронні HTTP-запити до серверу та отримує відповідь в фоновому режимі, після чого оброблює отриману інформацію або ж оновлює вміст сторінки. Основні етапи роботи AJAX полягають у наступному.

Створюється об’єкт XMLHttpRequest, за допомогою JavaScript, який використовується для самої взаємодії з сервером. Цей об’єкт дозволяє відправити певні запити до серверу та отримувати відповіді на них. Після створення даного об’єкту налаштовується запит до серверу, встановлюючи метод запиту до URL-адреси серверу, GET або ж POST, та обробник подій на отриману інформацію в результаті запиту. Після цього запит відправляється асинхронно, що означає, що сам JavaScript не очікує відповідь сервера, а продовжує виконання скрипту, на сервер за допомогою методу “send()”, після обробки інформації та отриманого запиту сервером він формує відповідні нові дані та відправляє їх назад до JavaScript-коду, у якому викликається обробник подій, який, в свою чергу, проводить певні маніпуляції над отриманою інформацією. Такі маніпуляції можуть включати в себе оновлення вмісту сторінки, відображення даних, сповіщення про помилку або виконання інших дій залежно від відповіді серверу. Саму схему роботи зображено на рисунку 2.8.

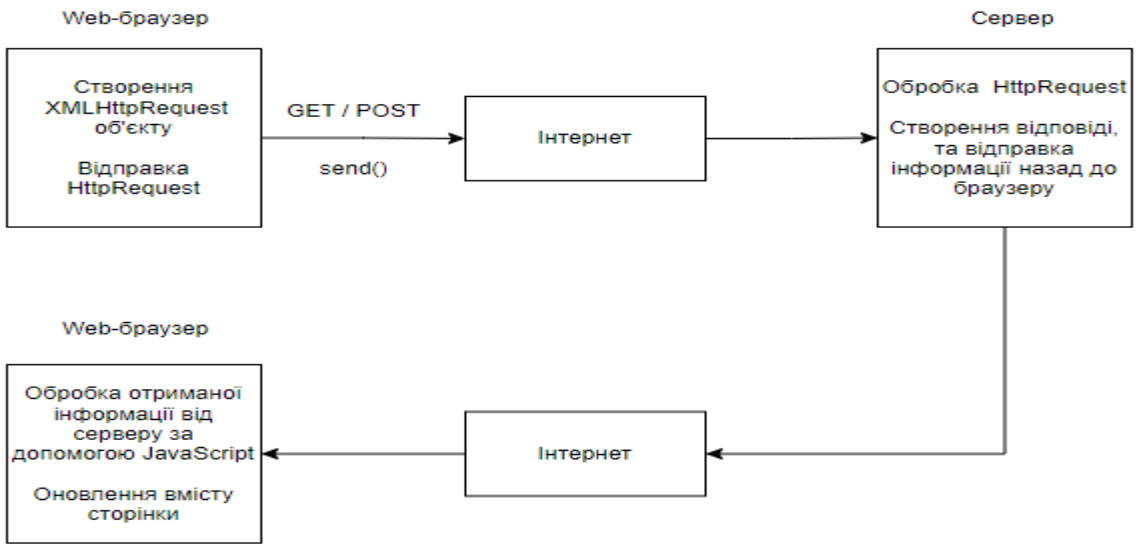

Рис. 2.8. Схема роботи AJAX-запиту

Для взаємодії з сервером використовуються різні методи та об’єкти з JavaScript, проте цей функціонал можна розширити і спростити роботу з AJAX-запитами за допомогою бібліотеки jQuery, яка надає додаткові функції, методи та певні утиліти для покращення роботи web-додатку.

JavaScript бібліотека jQuery надає зручний та простий спосіб виконання AJAX-запитів. Вона інкапсулює складність взаємодії з об'єктом XMLHttpRequest та надає простий інтерфейс для виконання різних типів запитів. Наприклад, основними методами виконання AJAX-запитів є:

$.ajax(), основний метод, який дозволяє виконувати будь-який тип AJAX-запиту. За допомогою цього методу можна налаштувати основні параметри запиту, такі як: URL, метод запиту GET чи POST, дані, тип очікуваної відповіді від серверу, обробники подій на отриману інформацію та інші параметри.

$.get() та $.post() – методи, що дозволяють виконувати GET- та POST-запити відповідно. Вони мають більш спрощений синтаксис написання, який дозволяє вказати лише URL-адресу серверу, що передаються на сервер та обробники подій для відповіді від серверу.

2.6. Серверна мова програмування PHP

Аналізуючи потреби адвокатів при роботі з клієнтами, було вирішено розробити окрему частину програмного комплексу у вигляді web-застосунку, в якому працівник фірми зможе працювати з власною базою даних клієнтів, вести певні записи по їх справах та супутній інформації, такої як надані документи чи примітки по роботі. Суть web-додатку полягає у тому, що адвокат авторизується під власним обліковим записом, може на календарі переглянути заплановані події, додати нові чи видалити застарілі. Даний календар є спільним для усієї фірми, оскільки можуть існувати спільні справи або колеги можуть нагадати про заплановану подію один одному, якщо у когось з них наразі немає можливості переглянути календар. В окремому розділі web-додатку адвокат може переглянути список власних клієнтів та справ, закріплених за ними, додати нового клієнта після

консультації, відредагувати інформацію по вже існуючому чи просто переглянути інформацію, яка є по справі клієнта.

Для реалізації даного web-застосунку було обрано мову програмування PHP з використанням популярного фреймворку Laravel.

PHP — це мова програмування загального призначення, яку спеціально розробили під створення web-додатків, основним призначенням якої є генерація динамічного вмісту web-сторінки та взаємодія з сервером і базою даних, на основі запиту від користувача.

Синтаксис у PHP є досить схожим на мови програмування Java, C та Perl. Він є досить легким у розумінні та вивченні, що робить його доволі доступним для початківців. Починаючи з п’ятої версії PHP, мова почала підтримувати повноцінне об’єктно-орієнтоване програмування з використанням класів, об’єктів, спадкування, поліморфізму та всіма іншими концепціями ООП для структурування та розробки коду, як і у інших об’єктно-орієнтованих мовах програмування, зокрема C# чи Java.

Особливістю даної мови також є те, що PHP-код можна вставляти безпосередньо в HTML-сторінку, що дозволяє поєднати статичний та динамічний вміст web-сторінки. Оскільки мова PHP інтерпретується web-сервером, на відміну від JavaScript, у чистий HTML-код, який передається безпосередньо на сторону клієнта у web-браузер і користувач не бачить самого PHP-коду, що покращує безпеку web-додатку.

PHP має вбудовану підтримку для роботи з HTTP-протоколом, сесіями та cookies, що полегшує взаємодію з базами даних, авторизацією користувача, роботу з XML та іншими ресурсами. Також слід відмітити вбудовану широку підтримку роботи з різними базами даними, реляційними моделями: MySQL, PostgreSQL, SQLite, а також нереляційними моделями, як MongoDB.

PHP є дуже гнучкою та потужною мовою програмування, спеціалізованою на розробці web-додатків, яка має широкий спектр функцій та можливостей, що робить її популярним вибором для реалізації різних типів проектів. Проте функціонал можна розширити за допомогою фреймворків, які полегшують та прискорюють

розробку, прикладом чого є вже використаний при розробці адвокатської частини проекту «Laravel».

Laravel являє собою один із найпопулярніших фреймворків для розробки web-додатків на основі мови PHP, оскільки він відчутно розширює спектр функціональних можливостей та має комфортний інтерфейс для ефективної і швидкої розробки web-застосунків. Особливостями та перевагами даного фреймворку є:

Аутентифікація та авторизація: Laravel надає зручний та простий механізм Auth для реалізації системи авторизації та аутентифікації користувачів. Даний механізм користується зв'язком з базою даних та перевіряє валідність введених даних при авторизації або ж заповнює таблицю бази даних з обліковими даними нового користувача. Auth має вбудовану підтримку ролей, аутентифікації та дозволів, що спрощує розробку захисту доступу до функцій та ресурсів web-застосунку. Дана функція є особливо важливою, оскільки база даних клієнтів зберігає інформацію про кожного з них, і дана інформація повинна бути надійно захищена від сторонніх.

Механізм Auth, можливий завдяки вбудованій ORM-бібліотеці Eloquent, яка спрощує взаємодію з базою даних, дозволяючи розробнику працювати з базою даних, використовуючи об'єктно-орієнтований підхід, замість звичних прямих SQL-запитів до бази даних. Це полегшує роботу з даними та покращує читабельність, зрозумілість і підтримуваність коду web-додатку. При цьому Laravel надає механізм міграцій та сідерів, які дозволяють легко створювати, оновлювати та керувати схемою бази даних. За допомогою сідерів фреймворк наповнює базу даних початковими значеннями, а за допомогою самих міграцій можна змінювати структуру бази даних прямо з коду, що значно спрощує розгортання та масштабування web-додатків.

Також важливим є механізм шаблонів Blade, який дозволяє розробнику легко створювати та керувати сторінками: він надає зручний синтаксис вставки фрагментів коду, циклів, умов чи інших операцій в HTML-шаблонах, що у сумі створює потужний засіб розробки. Оскільки у web-сторінки можна включати окремо

описаний елемент і просто підключати його в потрібному місці, то для редагування даного елементу на усіх сторінках потрібно лише змінити сам файл з даним елементом, а зміни з’являться на усіх сторінках де підключений даний елемент. На допомогу механізму шаблонів слугує широка система маршрутизації, яка дозволяє визначати URL-шляхи та організовувати контролери на них. Тобто система дає можливість налаштувати механізм обробки запитів користувачів за допомогою контролерів по якомусь конкретному URL-шляху.

2.7. Система керування базами даних MySQL

MySQL — являє собою вільну систему керування реляційним типом баз даних, яка надає широкий спектр можливостей для організації, зберігання, редагування та інших дій над даними. Система включає в себе як саму базу даних, так і компактний комплексний багато потоковий сервер, що забезпечує надійність, високу швидкість та простоту використання бази даних.

Дані у MySQL зберігаються під виглядом окремих таблиць, що дозволяє досягати високої ефективності та гнучкості при роботі з ними. Таблиці є самостійними або взаємозв’язаними за допомогою відношень, що дозволяє виконувати зв’язані та комплексні запити для отримання даних з декількох таблиць одночасно.

Запити виконуються на мові SQL, що використовується для доступу до бази даних. За допомогою запитів розробник може змінювати, видаляти, створювати нові таблиці, запитувати на дані з таблиць в бази даних. Дана мова є стандартною для роботи з усіма реляційними типами баз даних, забезпечуючи одноманітний та зрозумілий спосіб взаємодії з структурою та інформацією.

Дана система складається з двох частин, а саме серверної та клієнтської. Робота базується на тому, що клієнтські програми, такі як PHP-скрипти, посилають на сервер MySQL SQL-запити за допомогою механізму мережевих засобів, тобто сокетів. Сервер, отримавши запит, обробляє і запам’ятовує отриманий результат, потім надсилає відповідь з результатом обробки назад до клієнтського скрипту, де

отримана інформація вже виводиться користувачу або ж використовується для подальших обчислень. Схему роботи відображено на рисунку 2.9.

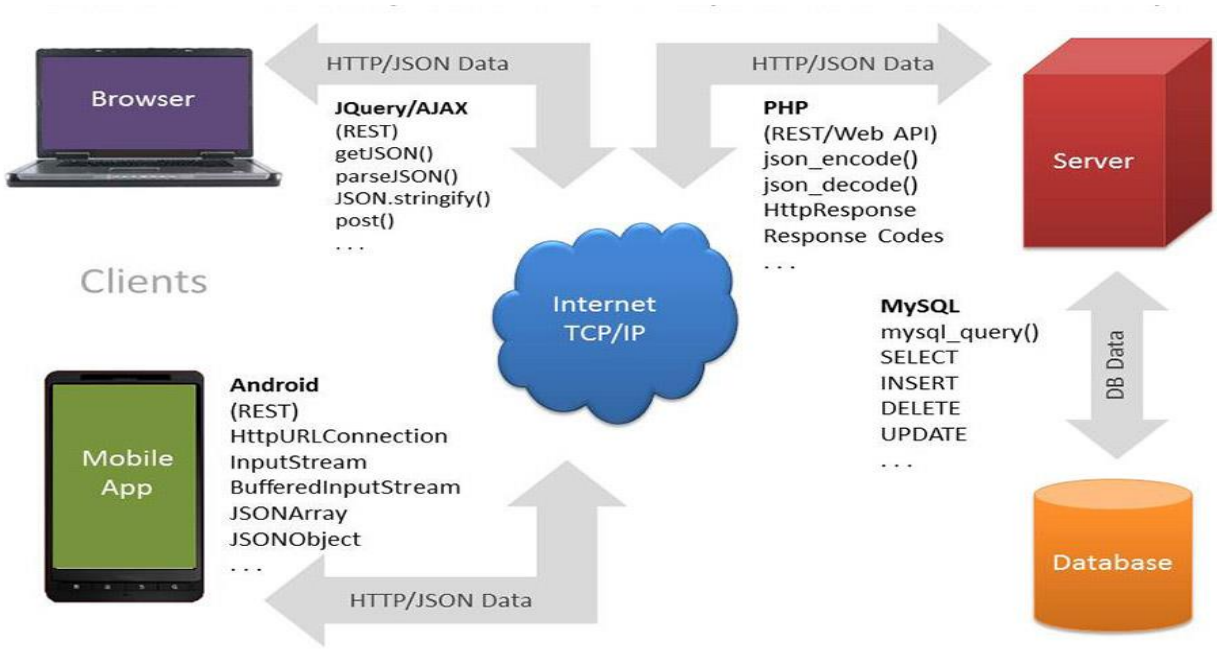

Рис. 2.9. Схема роботи MySQL-серверу

2.8. Опис бази даних

Весь вміст адвокатської частини програмного комплексу web-додатку, а також основні дані - інформація про клієнтів, про заплановані події та облікові записи адвокатів, - зберігаються в розробленій базі даних. Концептуальна модель бази даних зображена на рисунку 2.10.

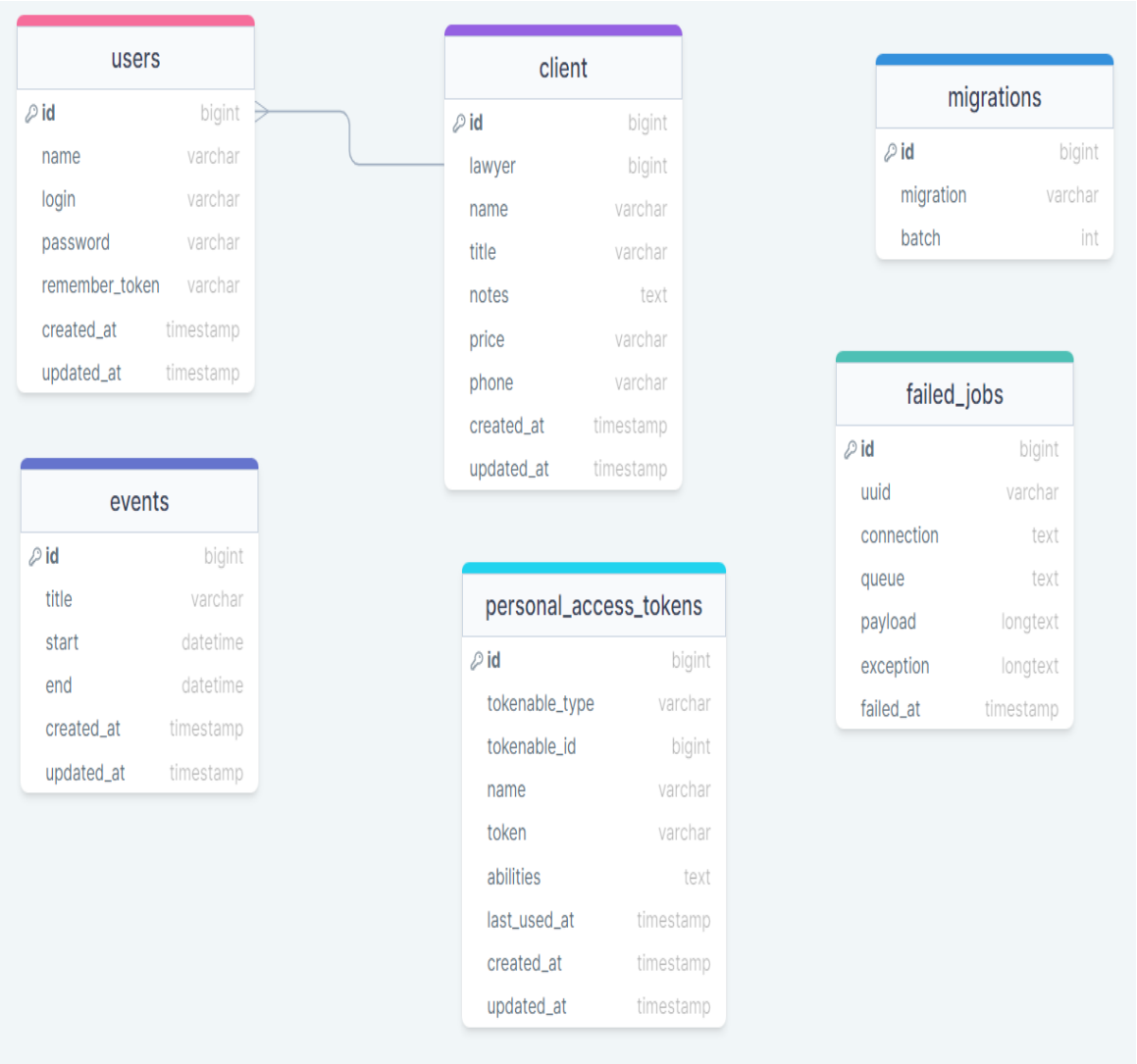

Рис. 2.10. Концептуальна модель бази даних

Дана база даних складається із шести таблиць, а саме : «user», «client», «events», «migrations», «failed_jobs» та «personal_access_tokens». Таблиці «migrations», «failed_jobs» та «personal_access_tokens» створені автоматично, оскільки для розробки даного web-додатку використовувався фреймворк Laravel, який, для повноцінного власного функціонування, автоматично їх створює при виконанні першої міграції. Поля у таблицях «created_at» та «updated_at» також створено автоматично за стандартом міграцій Laravel. Вони відповідають за дату та час створення або ж редагування запису у базі даних відповідно.

Структури розроблених таблиць матимуть наступний вигляд (таблиці 2.1-2.3):

Таблиця 2.1

Структура таблиці «user»

Ім’я поля	**Тип поля**	**Опис поля**
id	bigInt	Первинний ключ
name	varchar	ПІБ адвоката
login	varchar	Логін до облікового запису адвоката
password	varchar	Пароль до облікового запису адвоката
remember_token	varchar	Токен запам’ятовування облікового запису

Дана таблиця існує для створення облікового запису адвоката, за даними якого користувач авторизується в web-додатку, адже без власного кабінету та аутентифікації користувач не матиме жодного доступу до бази даних. Саме по даній таблиці при авторизації механізм фреймворку Auth звіряє введені дані користувачем у поля форми. Він автоматично під час запиту звіряє введений логін та пароль зі значеннями, що збережені у таблиці адвокатів. Якщо ж введені дані збігаються, то користувач отримує повний доступ до захищених ресурсів та функціоналу додатку.

Таблиця 2.2

Структура таблиці «client»

Ім’я поля	**Тип поля**	**Опис поля**
id	bigInt	Первинний ключ
lawyer	bigInt	ID адвоката що відповідає клієнту
name	varchar	ПІБ клієнта
title	varchar	Заголовок до справи клієнта
notes	text	Відомості по справі клієнта, по наданим документам, та статусу оплати послуг
price	varchar	Ціна послуг наданих клієнту
phone	varchar	Контактний номер телефону клієнта

Дана таблиця створена для ведення записів про клієнтів адвоката. Таблиця має зв’язок із попередньою, тобто таблицею «user», по ID облікового запису адвоката, що створив запис у базі даних про конкретного клієнта. Користувач під час роботи може видаляти записи про клієнта, редагувати вже внесені записи, переглядати їх по запиту до бази даних та, за допомогою форми на web-сторінці, створювати записи про нового клієнта.

Таблиця 2.3

Структура таблиці «event»

Ім’я поля	**Тип поля**	**Опис поля**
id	bigInt	Первинний ключ
title	varchar	Заголовок запланованої події
Start	datetime	Початок події
end	datetime	Кінець події

Таблиця створена для повноцінного функціонування календарю із запланованими подіями в особистому кабінеті користувача. Поля «start» та «end» слугують для коректного відображення події по днях.

2.9. Висновки до розділу 2

В розділі 2 було розглянуто основний набір технологій, використаних для розробки програмного комплексу на основі обраних архітектур. Було описано основні принципи роботи даних технологій, їх певні переваги та недоліки, а також їх супровідні бібліотеки та фреймворки. Розроблено та описано базу даних клієнтів з обліковими записами адвокатів та записами їх зустрічей.

В результаті програмний комплекс став повноцінним web-рішенням на основі базових технологій, таких як HTML, CSS та JavaScript, а також розробленою серверною частиною на мові програмування PHP з популярним фреймворком Laravel, який забезпечив зв’язок з базою даних MySQL.

РОЗДІЛ 3
РЕАЛІЗАЦІЯ ПРОГРАМНОГО КОМПЛЕКСУ

3.1. Розробка та налагодження web-застосунку

Як зазначено у попередньому розділі, адвокатська частина програмного комплексу розроблена на основі фреймворку Laravel та на базі мови програмування PHP відповідно. Для роботи даного фреймворку потрібно встановити кілька програм та пакетів, а саме:

Сама мова програмування PHP - для коректної роботи фреймворку встановлена версія повинна бути рівною 7.4 або новішою. PHP також потрібно встановити локально на комп'ютер або ж на сервер. Проте для запуску проекту потрібен ще web-сервер на основі Apache чи Nginx. Зазвичай встановлений PHP йде додатковим модулем разом із сервером, прикладом чого є програма OpenServer, а тому локально встановлювати його на власний пристрій не є потребою. Крім web-серверу, дана програма у комплекті має також необхідну базу даних для роботи Laravel, таку як MySQL чи SQLite.

Наступною обов'язковою умовою для створення проекту на основі Laravel є пакетний менеджер PHP — Composer. Даний менеджер використовується для управління залежностями та встановлення усіх необхідних пакетів для роботи Laravel. Після його встановлення у терміналі з'являється можливість звертатися до нього безпосередньо за допомогою ключового слова «composer», де потрібно встановити командний рядковий інтерфейс Laravel CLI за допомогою команди в терміналі «composer global require laravel/installer». Це дасть змогу користуватися командою «laravel» в тому ж самому терміналі для створення та керування Laravel проектом. Далі, після встановлення усіх попередніх пакетів, потрібно завантажити

сам Laravel локально до певного проекту. Для цього в терміналі потрібно прописати наступну команду: «php composer.phar create-project --prefer-dist laravel/laravel APP_NAME» , де APP_NAME – назва майбутнього проекту.

Оскільки робота Laravel з FrontEnd частиною проекту базується на механізмі Mix, що забезпечує збирання та оптимізацію ресурсів, тобто об’єднання JavaScript та CSS файлів в один або кілька компільованих файлів для зменшення їх розміру та покращення продуктивності проекту, то для його коректної роботи потрібно також встановити пакетний менеджер NPM на платформі Node.js. Він дозволяє легко встановлювати, оновлювати та керувати залежностями проекту, наприклад: фреймворки, бібліотеки чи інші пакети, необхідні для розробки проекту. Встановлюється даний пакет разом з Node.js, тому потрібно встановити лише його, й тоді у терміналі з’явиться можливість використовувати пакетний менеджер за допомогою ключового слова «npm». У самому ж терміналі в корінній папці проекту після встановлення потрібно прописати «npm i», після чого почнеться завантаження усіх необхідних та прописаних залежностей проекту з файлу «package.json». Після завершення завантаження усіх необхідних файлів, у терміналі потрібно прописати команду «npm run watch», внаслідок чого і запуститься механізм mix, який скомпілює потрібні CSS та JavaScript файли, в яких й потрібно писати, власне, сам код проекту.

Після усіх маніпуляцій та налаштувань, файлова структура проекту матиме наступний вигляд, який зображено на рисунку 3.1. В ході розробки також буде створено кілька файлів самого розробленого проекту. Наприклад: у папці «resources» — «views» будуть створені шаблони відображення контенту сторінки «blade». Дані шаблони являють собою файли з HTML структурою web-сайту, з синтаксисом Laravel, як підключення елементів з окремого файлу, використання циклів та умов.

Рис. 3.1. Файлова структура початкового проекту адвокатської частини

Для роботи «blade» шаблонів Laravel використовує механізм маршрутизації, тобто за певною специфічною URL-адресою користувача направляє до відповідної HTML структури, наприклад: у розробленому проекті при переході на головну URL-адресу, тобто для адреси, що закінчується на «/», користувача автоматично, якщо він не авторизований, направляє на файл шаблону, у якому описана форма з полями вводу облікового запису.

Також механізм маршрутизації по URL-адресі може звертатися по механізму Laravel, що зветься контролери. Контролери використовуються для обробки різних запитів; вони є класами, що містять методи, які відповідають за обробку конкретних подій та запитів для певного URL-шляху. Коли користувач звертається до налаштованої URL-адреси, Laravel знаходить відповідний маршрут та передає запит вказаному контролеру, в якому відбувається обробка запиту та повернення відповіді. Дані запити зазвичай мають тип GET або ж POST. Саме за допомогою створеного та розробленого контролеру «LoginController», web-застосунок здійснює обробку запиту на авторизацію користувача, використовуючи механізм Auth.

Підключення до бази даних відбувається автоматично за допомогою глобального файлу налаштувань «.env», в якому вказується ім’я бази даних та дані облікового запису, під яким web-додаток підключається для виконання запитів чи маніпуляції над даними у базі.

Для роботи вищезгаданих контролерів, які взаємодіють з базою даних, також необхідна спеціально створена модель елементу за допомогою функціонального файлу «artisan» та команди у терміналі «php artisan make:model MODEL_NAME», де MODEL_NAME - назва потрібної моделі для контролера. В моделях описуються поля, які можна заповнювати при створені нового запису у базі, а також спеціальна властивість «$table», яка визначає якій таблиці у базі даних відповідає модель. За допомогою моделей розробник легко взаємодіє з базою даних, може виконувати запити, зберігати, оновлювати та видаляти їх.

На рисунку 3.2 зображено діаграму використання, тобто всіх можливих функцій та вимог користувача web-застосунку, для комфортної роботи, які потрібно реалізувати при розробці.

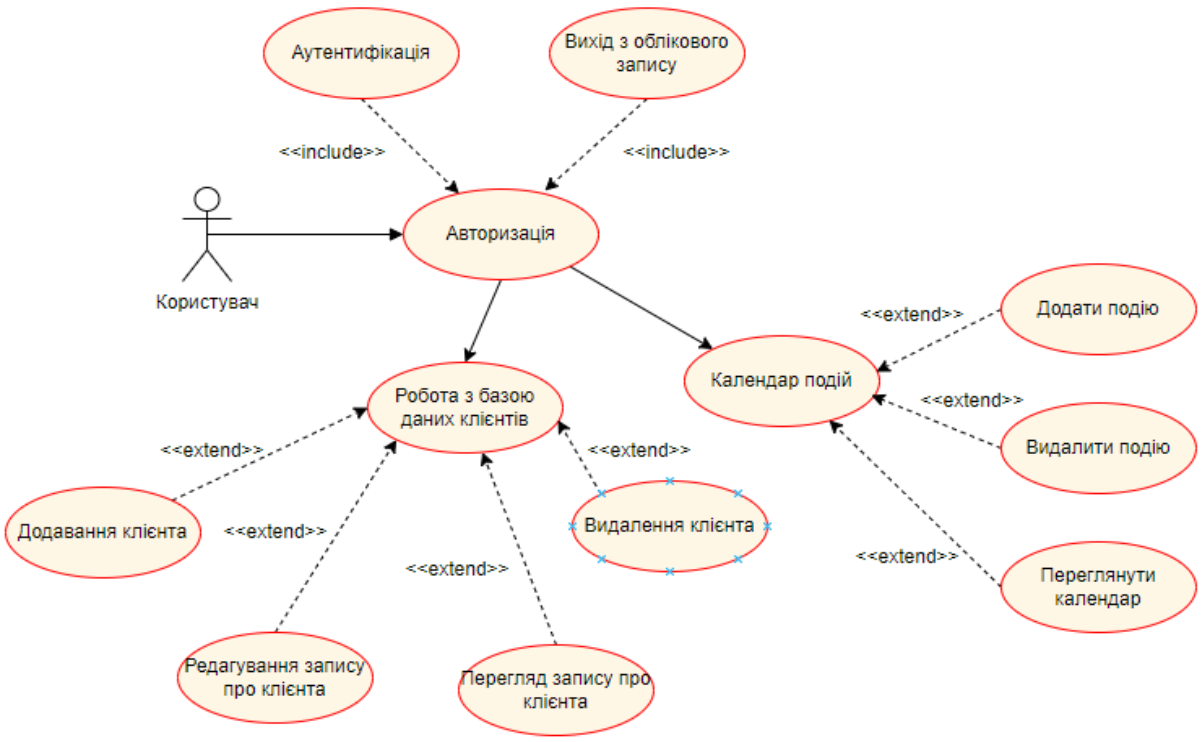

Рис. 3.2. Діаграма варіантів використання для web-застосунку з надання адвокатських послуг

3.2. Структура та функції клієнтської частини програмного комплексу

Другою частиною програмного комплексу є web-сайт, призначений для збільшення потоку клієнтів шляхом реклами власних послуг в мережі Інтернет. Дана web-сторінка просуває послуги, які надає фірма - потенційний клієнт, при знаходженні сторінки за запитом про юридичні послуги, може ознайомитися з спектром послуг та тем, на яких спеціалізуються адвокати фірми. Він може переглянути останні справи, над якими працювали фахівці та записатись на первинну консультацію за допомогою форми зворотного зв’язку, яка розташована у кожному розділі web-сайту. Відповідний лист із текстом, введеним у форму, прийде менеджеру на пошту, після чого він передає інформацію відповідному адвокату, який, у свою чергу, зв’язується із клієнтом для консультації. На рисунку 3.3 зображено діаграму звернення клієнта до фірмі, після перегляду web-сайту в мережі Інтернет і задоволені всіх своїх вимог до адвоката та фірми в цілому, клієнт заповнює форму зворотного зв’язку, в якій описує свою ситуацію, менеджер отримує повідомлення з даним текстом на пошту, та повідомляє спеціаліста, який в свою чергу зв’язується з клієнтом та записує його на консультацію.

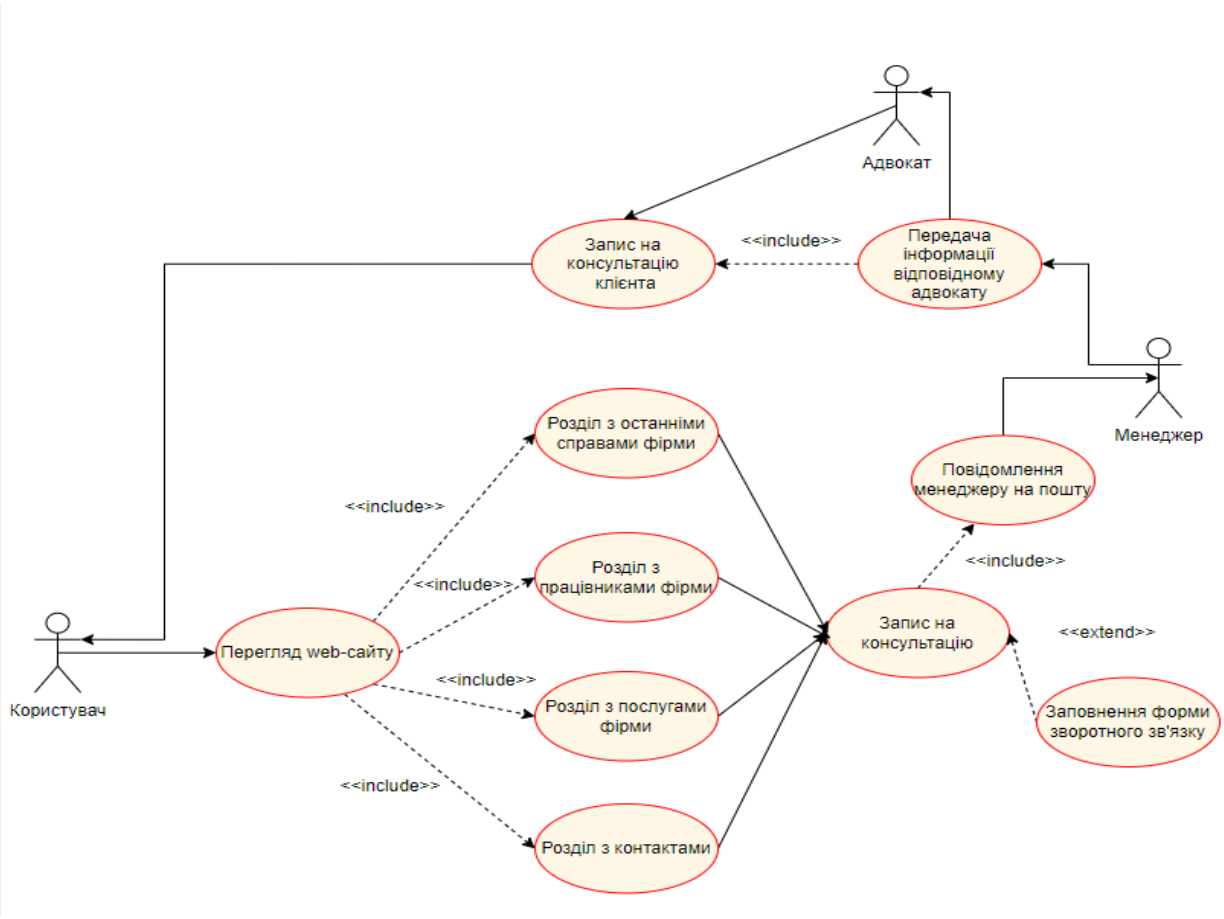

Рис. 3.3. Діаграма варіантів використання при користуванні web-сайту клієнтом

Як зазначено у попередньому розділі, даний web-сайт побудований на основних технологіях HTML, CSS та JavaScript, тому й структура проекту має стандартний вигляд з розподіленням відповідних форматів файлів по окремим папкам, що відображено на рисунку 3.4.

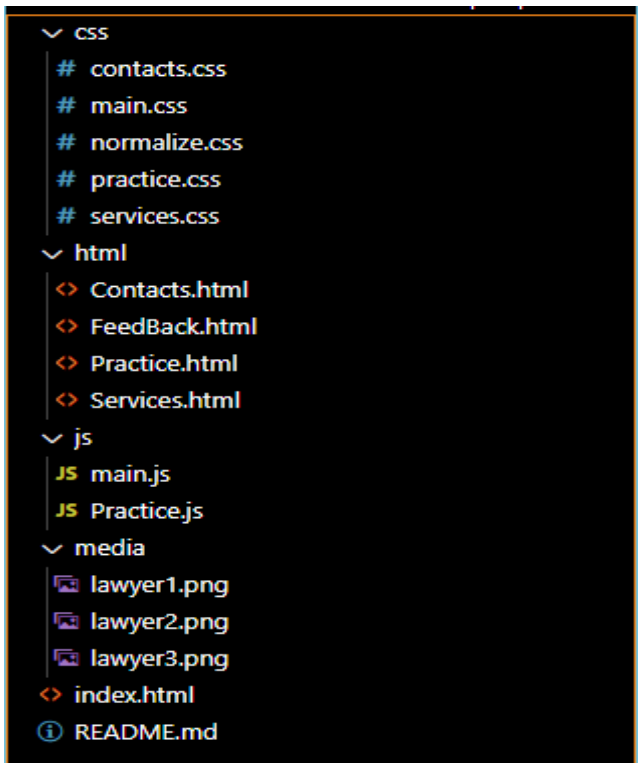

Рис. 3.4. Файлова структура розробленого проекту клієнтської частини

Папки CSS, HTML, JS відповідають за конкретне розширення файлу проекту. Головним файлом є «index.html», який містить структуру головної сторінки web-сайту. Решта файлів з розділами web-сайту знаходяться у відповідній папці HTML, де файли містять свою структуру розділу:

- Contacts.html — містить структуру розділу з контактами фірми та гугл-картою фізичного місця розташування фірми.
- FeedBack.html — це структура розділу, на який клієнта перенаправляє після заповнення форми зворотного зв’язку.
- Practice.html — відповідає за розділ з останніми справами фірми, вміст даного розділу доповнюється за допомогою JavaScript-файлу «Practice.js», в якому міститься масив об’єктів, що заповнюється за шаблоном менеджером і заповнюється на web-сторінці динамічно в залежності від вмісту масиву.
- Services.html — розділ web-сайту, на якому клієнт ознайомлюється з послугами, які надають спеціалісти, та з темами, в яких вони найкраще розуміються.

Стосовно файлів з розширенням «.css», то файли мають відповідні назви до своїх розділів, лише файл з назвою «main.css» є головним файлом стилів та підключається до усіх розділів, так само як і файл з JavaScript-кодом «main.js».

3.3. Результат розробки та тестування функцій при роботі з програмним комплексом

Результатом розробки клієнтської частини програмного комплексу є повноцінний web-сайт просування послуг компанії, головна сторінка якого зображена на рисунку 3.5.

Рис. 3.5 — Частина головної web-сторінки

На web-сторінці клієнт може ознайомитись з адвокатами фірми та їх спеціалізацією, що зображено на рисунку 3.6.

Рис. 3.6. Розділ web-сторінки з працівниками фірми

Навігація клієнта по розділам web-сайту здійснюється за допомогою навігаційного меню у верхній частині кожної сторінки. Меню відображено на рисунку 3.7.

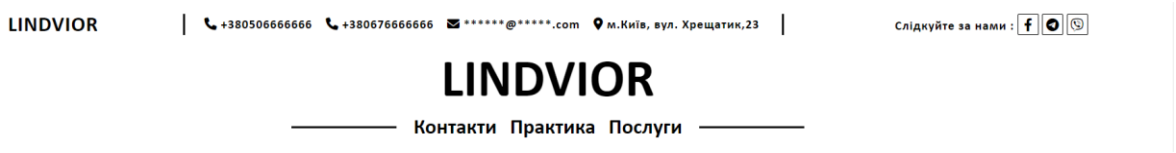

Рис. 3.7. Навігаційне меню web-сайту

Клієнт, при натисканні на відповідний пункт меню, буде направлений на інший розділ web-сайту, наприклад: при натисканні на пункт «Контакти» клієнта буде направлено на розділ з контактами фірми, що зображено на рисунку 3.8.

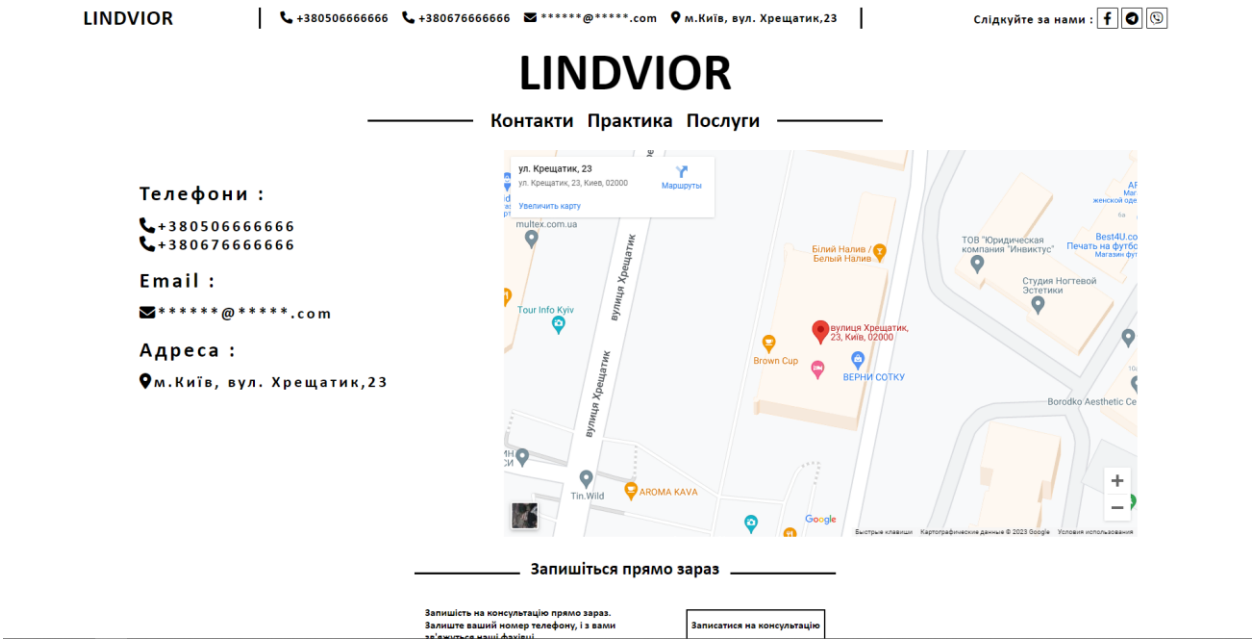

Рис. 3.8. Розділ web-сайту з контактами компанії

Також на сайті присутні посилання на соціальні мережі фірми за відповідними кнопками, до прикладу, список з посиланнями у футері, що зображено на рисунку 3.9.

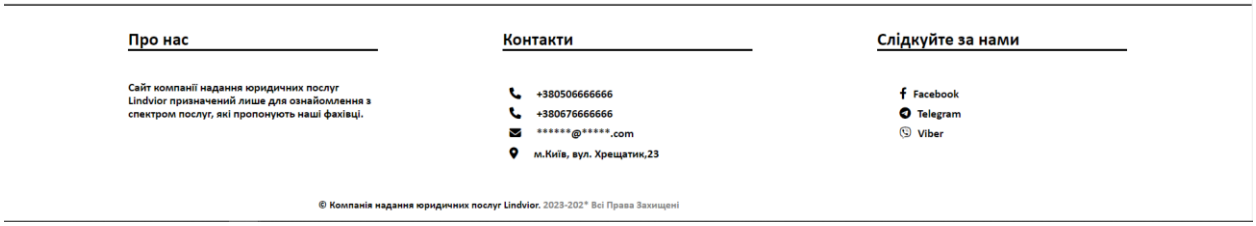

Рис. 3.9. Футер web-сайту з посиланнями на соціальні мережі та чат у Viber фірми

У розділі «Послуги» клієнт може оглянути перелік справ та тем, у яких фахівці є обізнаними та мають практику. Частину розробленого розділу зображено на рисунку 3.10.

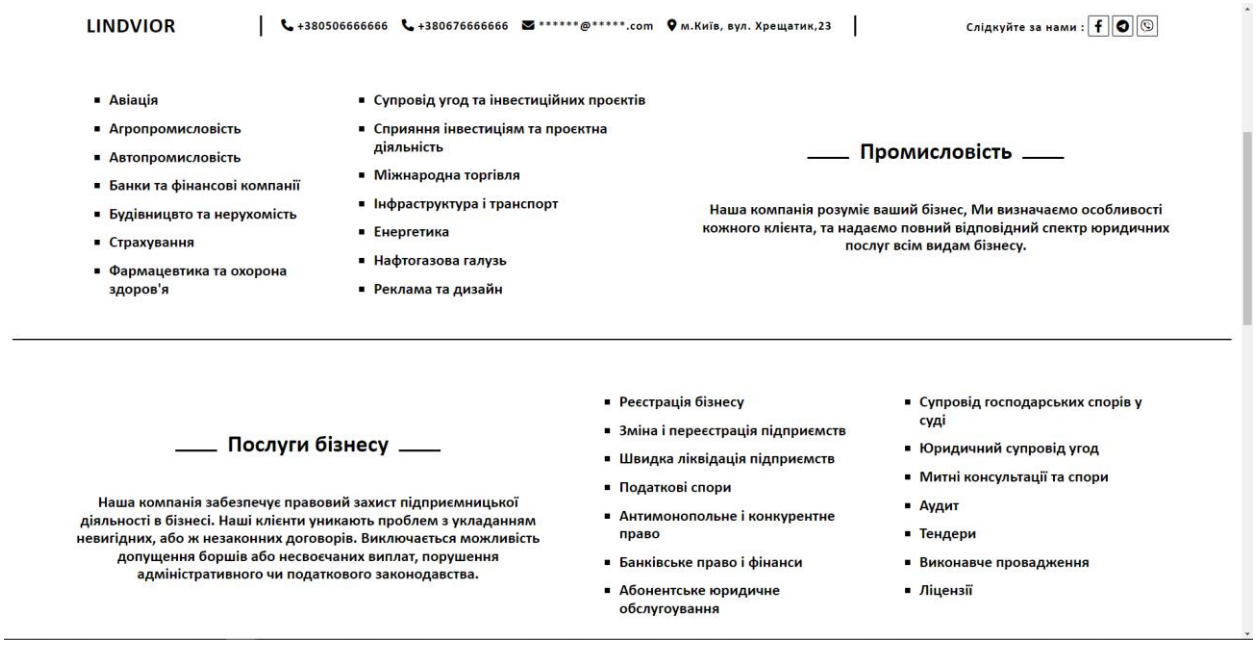

Рис. 3.10. Частина розділу з темами послуг компанії

Розділ «Практика» відобразить перед потенційним клієнтом список з останніми справами фірми. Список справ, як було зазначено раніше у даній роботі, є динамічним та залежить від наповнення масиву, тобто, якщо додати нові записи у масив, зміни автоматично відобразяться на web-сторінці, що можна побачити на рисунку 3.11 та рисунку 3.12.

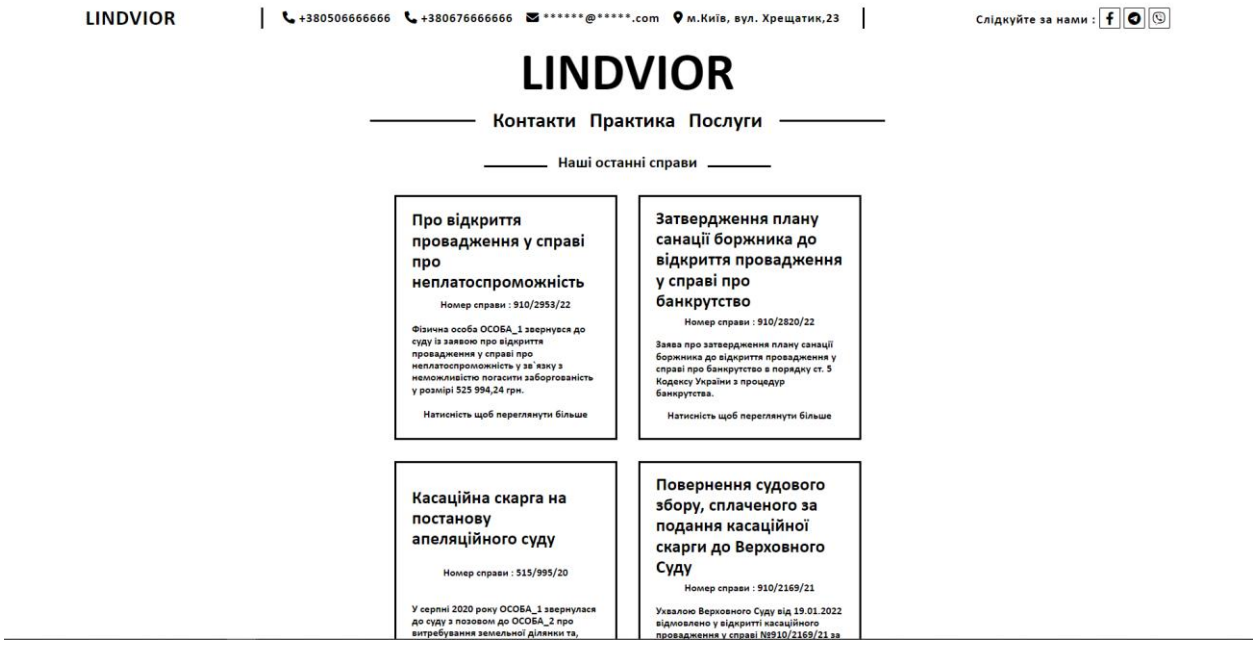

Рис. 3.11. Розділ з справами компанії до змін у масиві

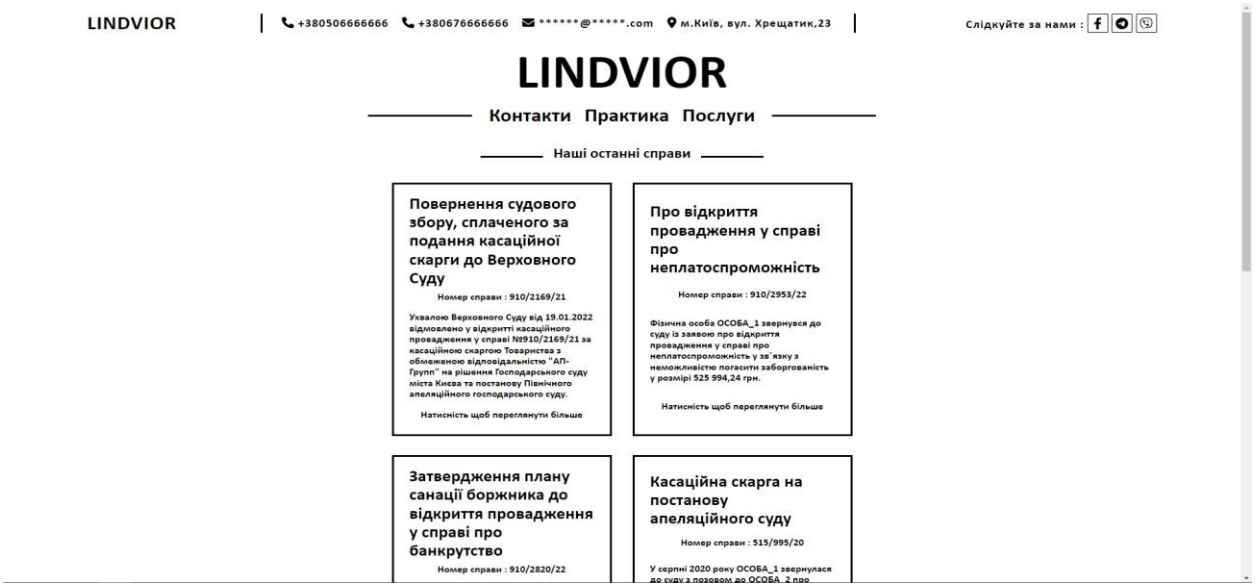

Рис. 3.12. Розділ з справами після змін у масиві

Як можна спостерігати, після додавання нового запису у масив зі справами, перший запис змістився на одну позицію. Дані записи, при натисканні, також містять посилання на справу у Єдиному державному реєстрі судових рішень, що дозволяє потенційному клієнту ознайомитися із рішенням суду, яке стало результатом справи, у якій брав участь адвокат фірми.

Після ознайомлення з фірмою та її послугами, потенційний клієнт може звернутися одразу за контактними номерами телефонів, а може, наприклад, скористатися формою зворотного зв’язку, яка є у кожному розділі web-сайту. Форма є спливаючою та відображається при натисканні на відповідну кнопку «Записатися на консультацію», що зображена на рисунку 3.13.

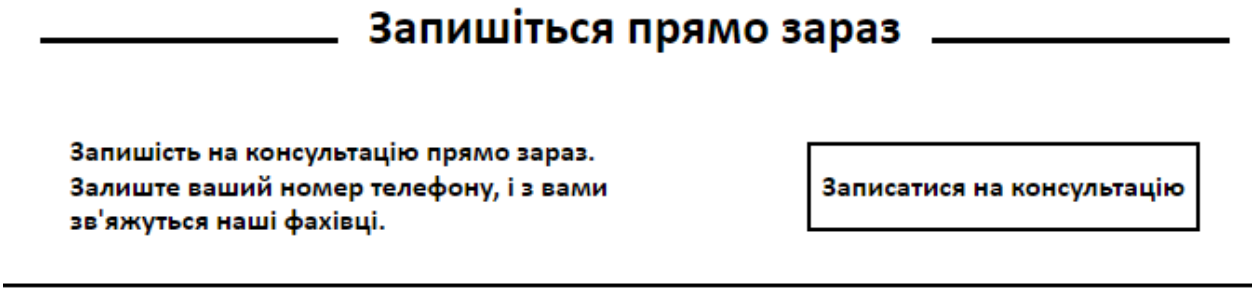

Рис. 3.13. Частина з кнопкою для запису на консультацію

Після натискання даної кнопки перед користувачем з’являється форма, яку потрібно заповнити. Форма до заповнення зображена на рисунку 3.14, після заповнення на рисунку 3.15 відповідно.

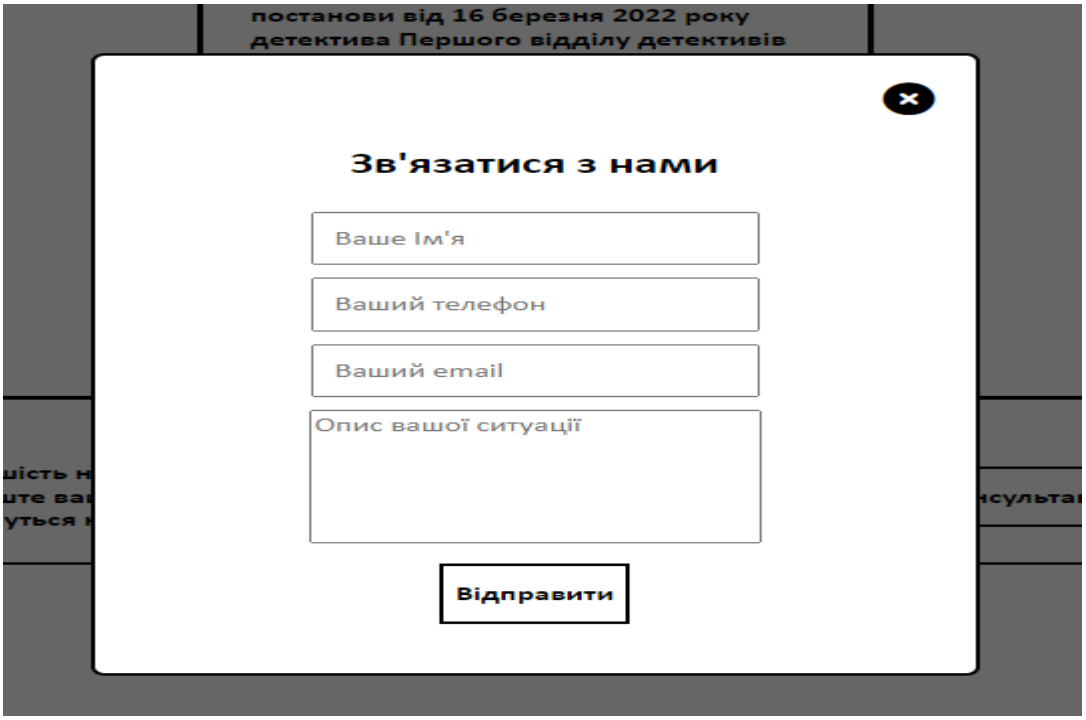

Рис. 3.14. Форма зворотного зв’язку до заповнення користувачем

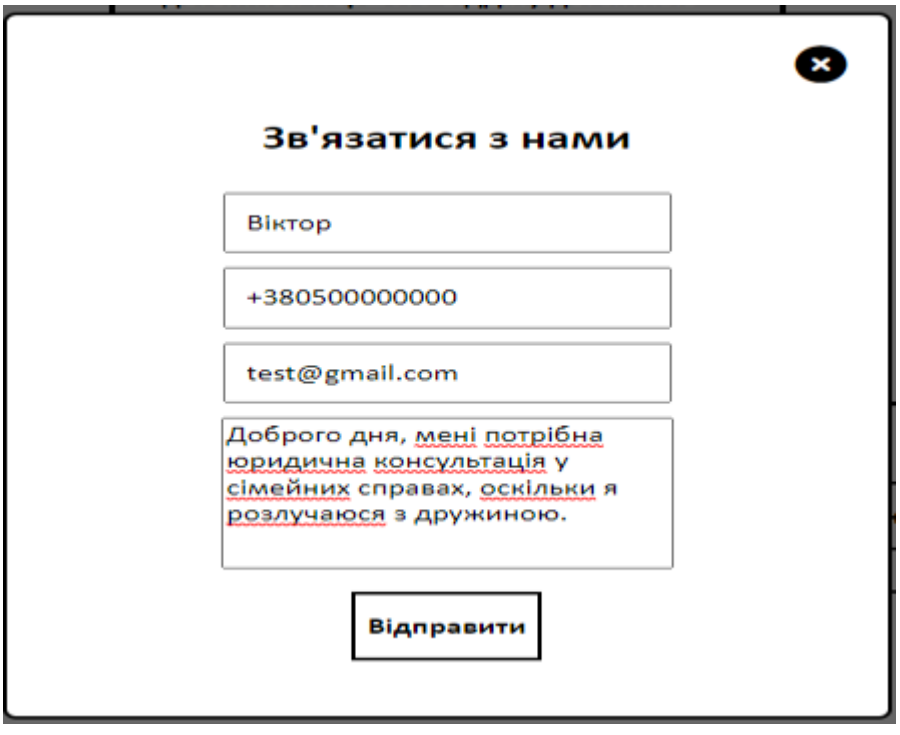

Рис. 3.15. Форма зворотного після заповнення користувачем

Після натискання користувачем кнопки «відправити», користувача направляє до сторінки подяки за звернення до фірми, що зображено на рисунку 3.16.

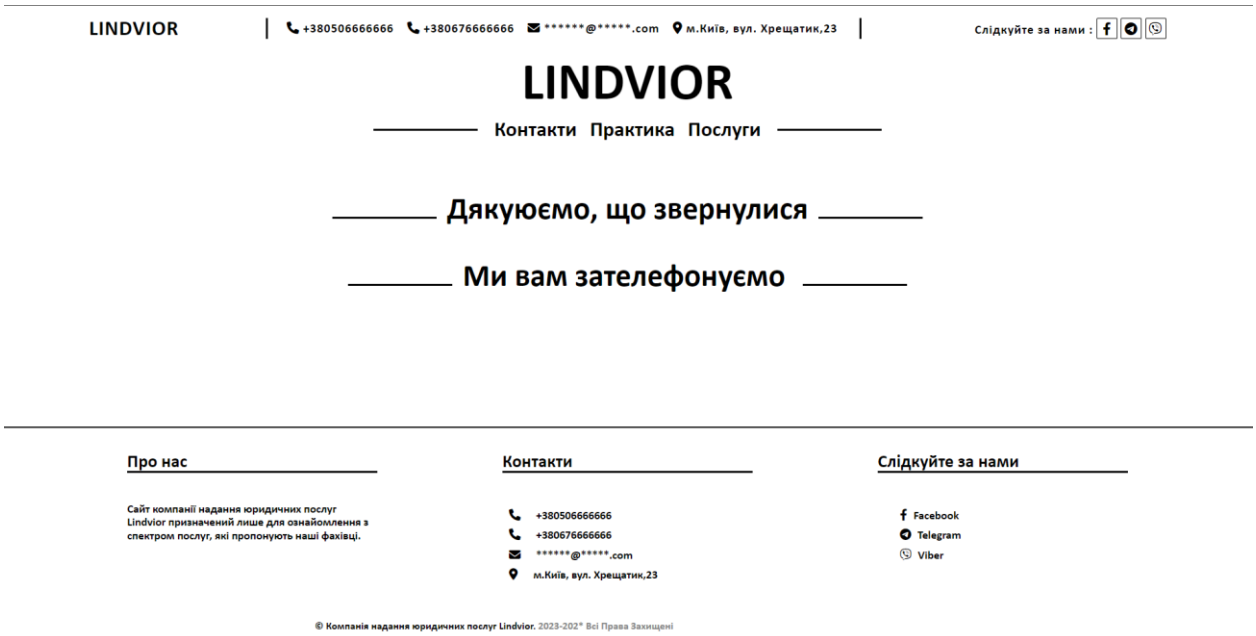

Рис. 3.16. Сторінка з подякою користувачу за звернення до компанії

В цей же час менеджеру фірми на пошту приходить повідомлення із введеним текстом користувача у форму з рисунку 3.15. Приклад отриманого повідомлення зображено на рисунку 3.17.

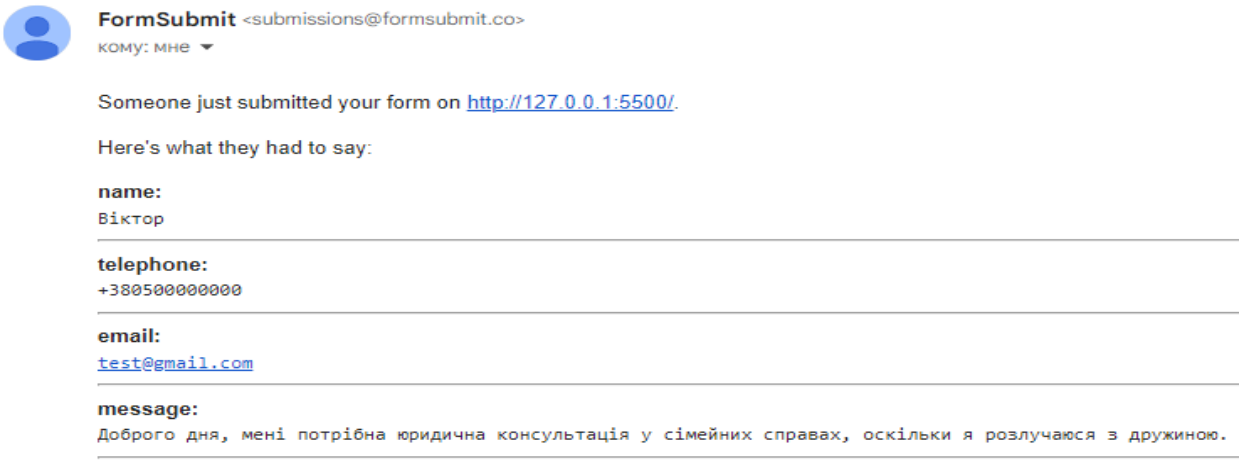

Рис. 3.17. Повідомлення з текстом з форми зворотного зв’язку у менеджера на пошті

Результатом розробки адвокатської частини програмного комплексу є web-застосунок, у якому користувач, тобто адвокат, може взаємодіяти із базою даних власних клієнтів.

При відкритті головної сторінки, якщо користувач не авторизований, механізм маршрутизації направляє його на шаблон з формою авторизації, що зображена на рисунку 3.18.

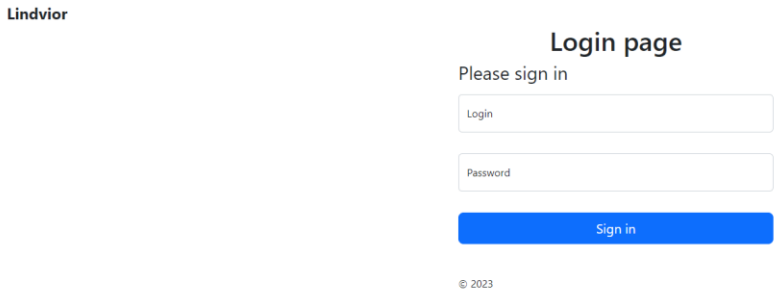

Рис. 3.18. Форма авторизації користувача

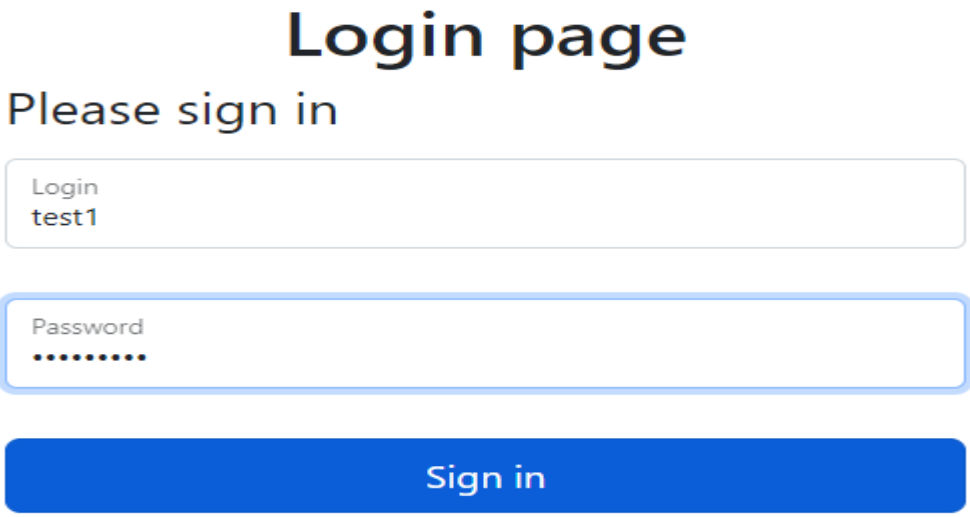

Рис. 3.19. Заповнена форма авторизації

Форма створює запит з введеними даними та передає їх логін контролеру, який, в свою чергу, перевіряє введені дані з наявними у базі даних. Якщо ж такий

обліковий запис у базі даних відсутній, то користувач повідомляється помилкою, яку зображено на рисунку 3.20.

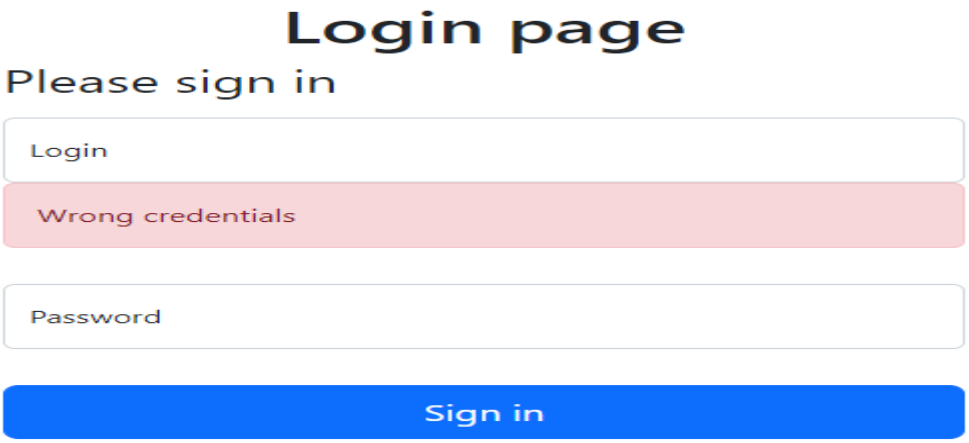

Рис. 3.20. Повідомлення про неправильно введені дані

Оскільки це корпоративний web-додаток, то реєстрація в ньому відсутня. Обліковий запис адвоката фірми створюється за допомогою механізму seed, який вносить потрібні дані, такі як логін, пароль, ПІБ. Після запуску механізму заповнення бази даних створиться новий запис у відповідній табличці «users», прикладом чого слугує обліковий запис для Сергія Сергійовича, на рисунку 3.21. При цьому пароль зберігається у вигляді hash-коду.

id	name	login	password	remember_token	created_at	updated_at
1	Serhiy Serhiyovich	LawLindvior1	$2y$10$uWRs/uq2l/7G1Tq8w57k8ert.Qg.LcTwaquszO5OxY0...	NULL	NULL	NULL

Рис. 3.21. Запис у базі даних з обліковим записом адвоката

Коли ж користувач вводить правильні дані, які містяться у базі даних, то він потрапляє до приватної захищеної частини web-додатку, де може користуватися усіма його функціями. Перед користувачем з’являється календар із запланованими зустрічами та справами фірми, де він може: створити запис про нову зустріч, видалити застарілий запис або той, що відмінився, та за потреби переглянути запис іншого адвоката. Дану сторінку відображено на рисунку 3.22.

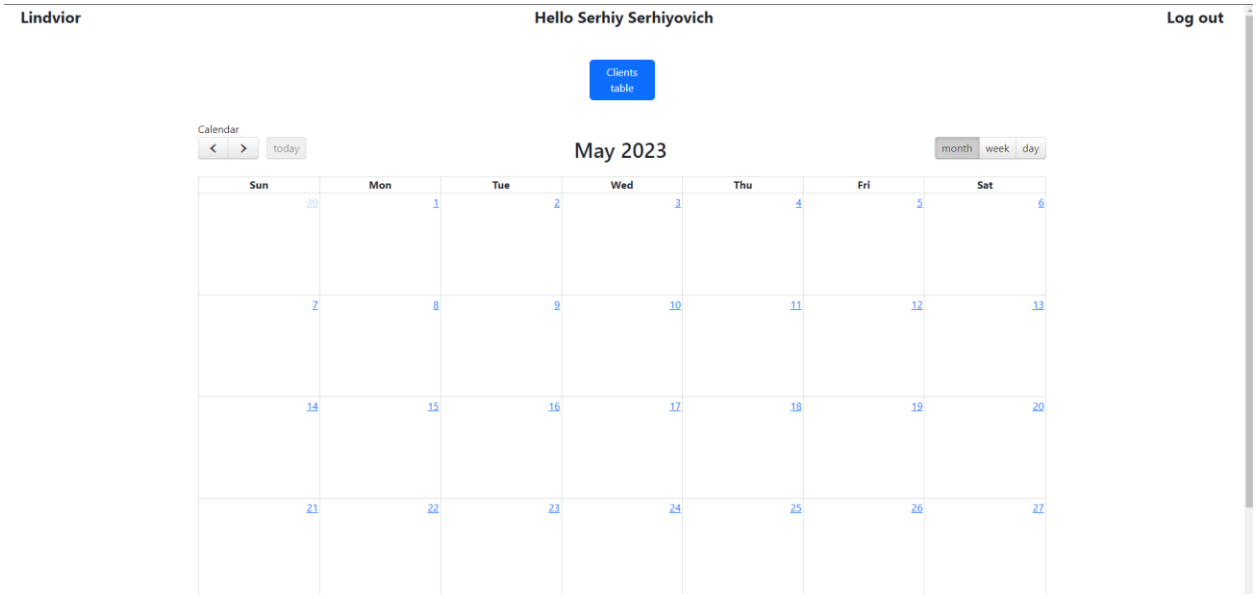

Рис. 3.22. Сторінка web-додатку з календарем запланованих зустрічей

При натисканні на комірку дня адвокат створює новий запис, ввівши необхідну інформацію про подію у відповідному вікні, зображеному на рисунку 3.23, після чого подія запишеться до бази даних та відобразиться на сторінці, що відображено на рисунках 3.24 та 3.25 відповідно.

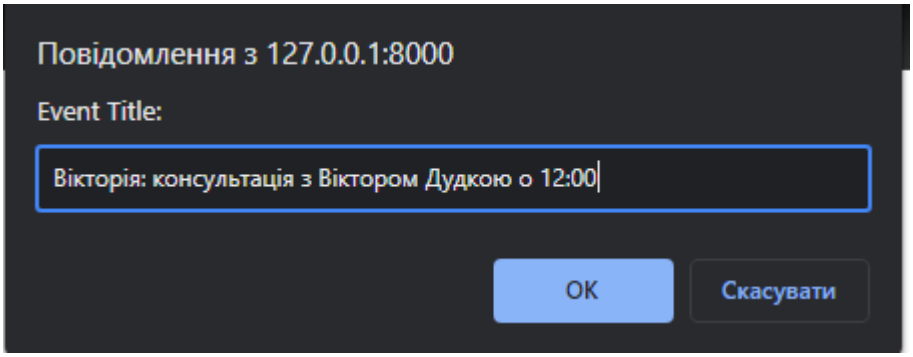

Рис. 3.23. Заповнення заголовку події з інформацією про неї

id	title	start	end	created_at	updated_at
1	Вікторія: консультація з Віктором Дудкою о 12:00	2023-06-07 00:00:00	2023-06-08 00:00:00	NULL	NULL

Рис. 3.24. Запис у базі даних про подію

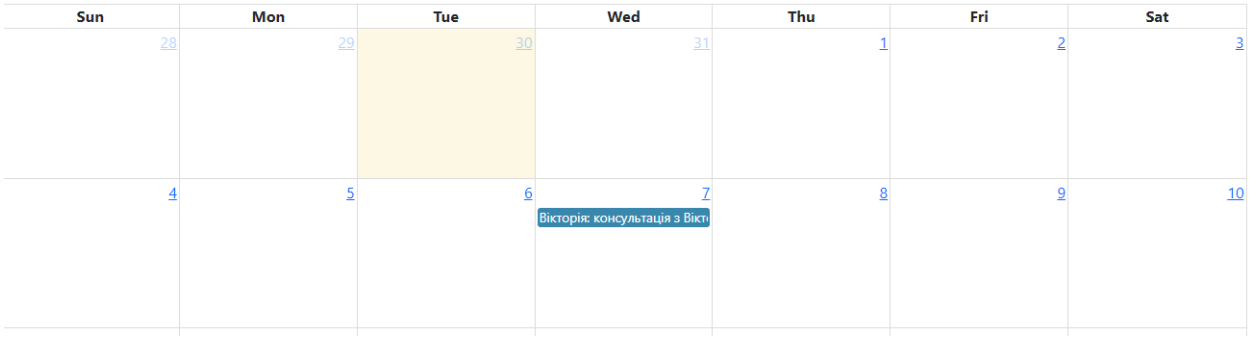

Рис. 3.25. Відображення події у календарі

При натисканні кнопки «Clients table» адвоката направляє на таблицю з усіма записами його клієнтів. В даному розділі web-додатку користувач може всіляко взаємодіяти з записами у базі даних, зокрема редагувати відомості у справі або змінити ціни послуг, номер телефону тощо. Приклад даних маніпуляцій відображено на рисунку 3.26.

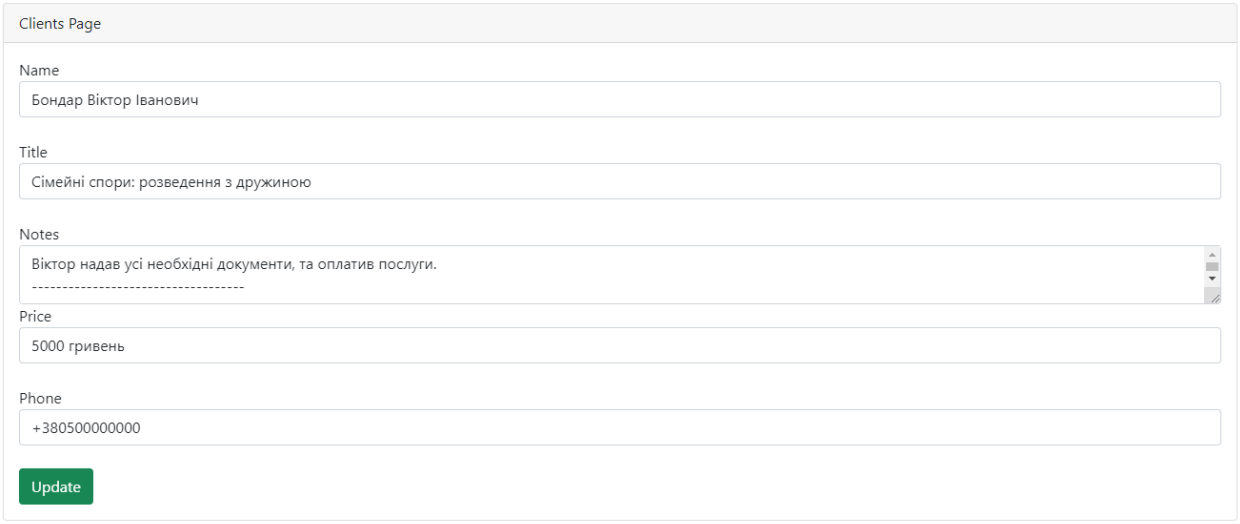

Рис. 3.26. Модифікація запису про клієнта

Також варто відзначити, що адвокати не мають доступу до записів один одного, оскільки у користувача відображаються лише ті записи, які він створив власноруч. Це відображено на рисунках 3.27 та 3.28, де на рисунку 3.27 відображено таблицю клієнтів адвоката Сергія, а на рисунку 3.28, при авторизації адвоката Віктора, запис відсутній.

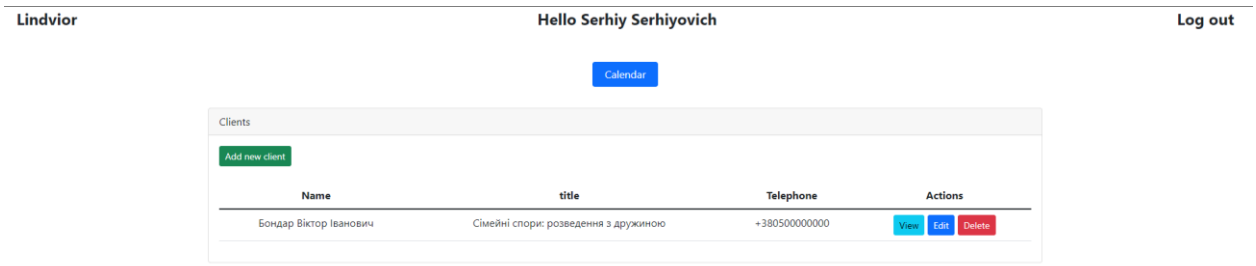

Рис. 3.27. Запис у базі даних адвоката Сергія

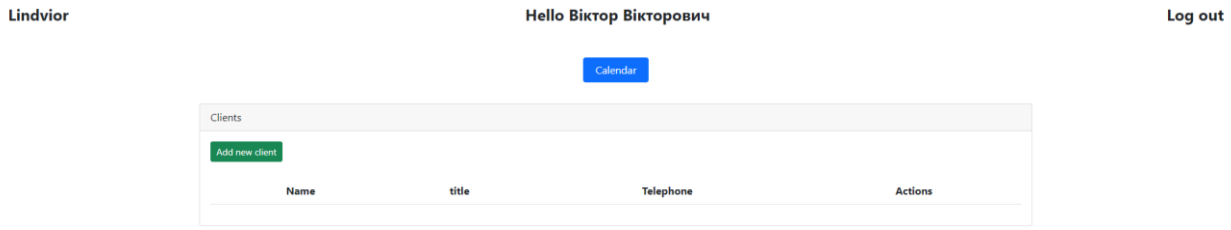

Рис. 3.28. Відсутній запис при авторизації з облікового запису Віктора

Адвокат під час консультації вносить свого клієнта до бази даних за допомогою відповідної кнопки «Add new client». При натисканні перед користувачем з’являється відповідна форма з полями, зображеними на рисунку 3.29.

Рис. 3.29. Форма створення запису про нового клієнта

Після заповнення форми даними, як відображено на рисунку 3.30, і натисканні кнопки «Save», у базі даних створиться новий запис про клієнта, а у таблиці клієнтів адвоката з’явиться відповідна інформація, що зображено на рисунку 3.31.

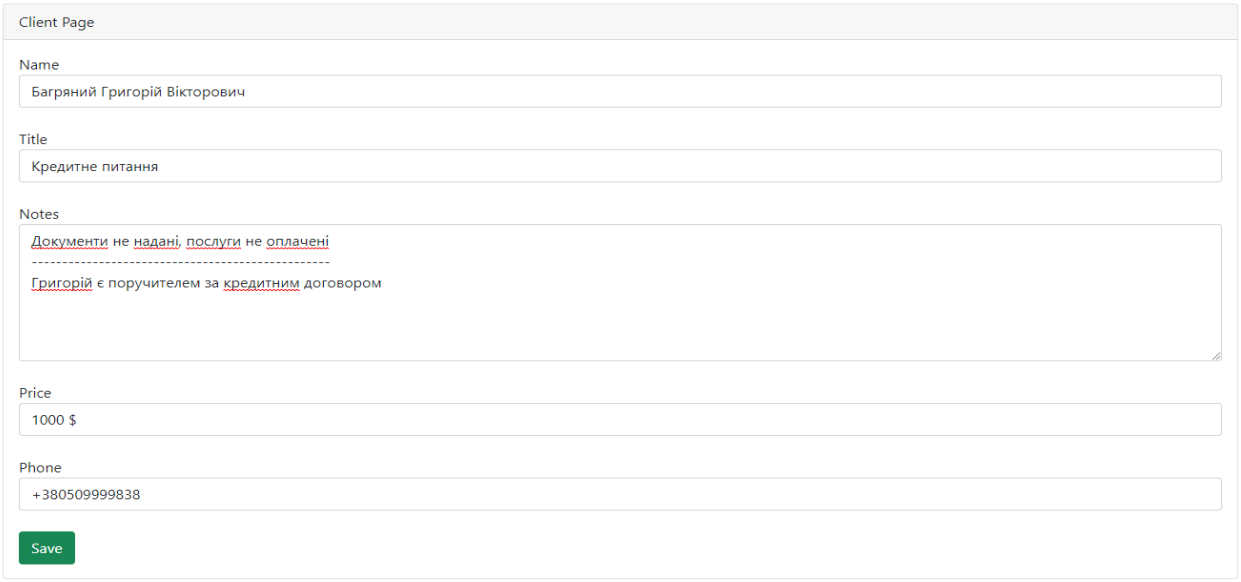

Рис. 3.30. Створення запису про нового клієнта

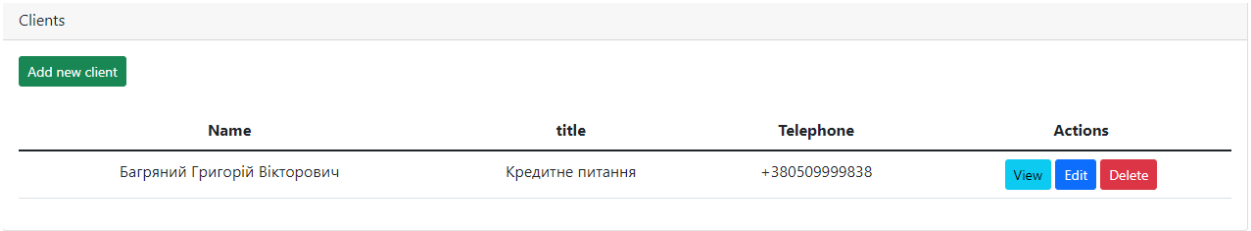

Рис. 3.31. Відображення клієнтів користувача

Надалі користувач може редагувати запис, як було наведено вище, а також переглядати інформацію за допомогою кнопки «View», як зображено на рисунку 3.32, та видаляти повністю запис про даного клієнта за допомогою кнопки «Delete», що розміщена напроти відповідного клієнта, при такій потребі.

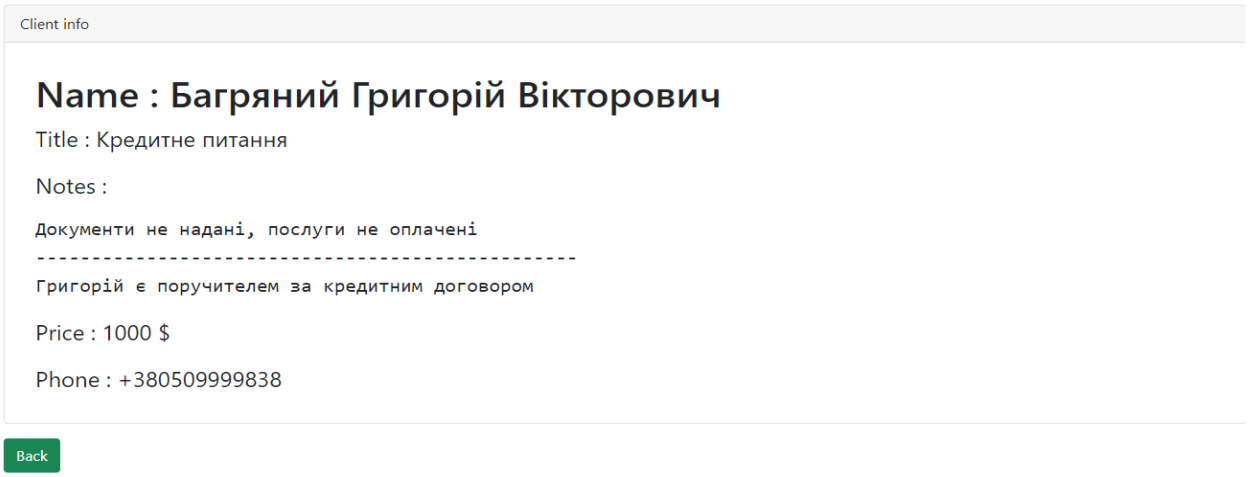

Рис. 3.32. Відображення інформації про клієнта

3.4. Висновки до розділу 3

В даному розділі було відображено результат розробки програмного комплексу у вигляді двох частин: web-сайту для реклами послуг фірми через мережу Інтернет та web-застосунку для обслуговування клієнтів, а також реалізації певних додаткових функцій для маніпуляції з інформацією про них у базі даних.

Web-сайт розроблено з унікальним дизайном та адаптацією під усі дисплеї пристроїв, що дає змогу охопити якомога більшу аудиторію потенційних клієнтів.

Web-застосунок, в свою ж чергу, повністю задовольняє потреби адвоката при надані послуг клієнтам: він має захищений доступ до власних записів у базі даних, до календаря з планами фірми та до управління інформацією по справам, де він може вести певні нотатки.

ВИСНОВКИ

У дипломному проекті було розглянуто основні відомості про web-технології та розроблено програмний комплекс на їхній основі. Даний комплекс включає в себе дві частини, кожна з яких має відповідне призначення. Клієнтська частина слугує для пошуку та розширення клієнтської бази. Потенційний клієнт може знайти даний ресурс в мережі Інтернет та після ознайомлення з послугами фірми, останніми справами, і фахівцями фірми, може звернутися до компанії за допомогою форми зворотного зв'язку. Після цього менеджер отримає відповідне повідомлення та проінформує адвоката про потенційного клієнта. Адвокат ж, в свою чергу, зв'яжеться із клієнтом та призначить консультацію. Саме для консультацій та комфортного ведення справи і було розроблено другу частину, що являє собою web-застосунок з надання послуг клієнтам.

Користувач даного web-застосунку має можливість створювати записи про заплановані події у календарі фірми, створювати записи про власних клієнтів під час або після консультацій, вести нотатки по справі та відомостям про клієнта, а також видаляти та редагувати дані записи у базі даних. При цьому усі функції є захищеними від сторонніх осіб шляхом авторизації та аутентифікації, механізм яких не дає користувачу без відомостей відповідного облікового запису, який створений у базі даних, користуватися будь-якими функціями web-застосунку.

Отже, дана робота покращила знання та практичні навички, здобуті протягом бакалаврського курсу навчання, які стали основою для розробки програмного комплексу та web-застосунку в цілому. За допомогою здобутих знань про HTML, CSS та JavaScript було розроблено клієнтський web-сайт, який став доповненням до web-застосунку про обслуговування клієнтів, розробленого на основі знань про бази даних та мови програмування PHP, що у сумі створило повноцінний програмний комплекс для адвокатської фірми.

СПИСОК БІБЛІОГРАФІЧНИХ ПОСИЛАНЬ ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Гайна Г. А. Основи проектування баз даних : навч. посіб. для студ. вищ. навч. закл., що навчаються за напрямом підгот. 0804 "Комп"ютерні науки" / Г. А. Гайна. – Київ : Кондор, 2008. – 200 с.
2. Сучасний підручник з JavaScript [Електронний ресурс] – Режим доступу: https://uk.javascript.info/ (дата звернення: 20.05.2023). – Назва з екрана.
3. Український веб-довідник [Електронний ресурс] – Режим доступу: https://css.in.ua/ (дата звернення: 24.05.2023). – Назва з екрана.
4. Ab M. MySQL: administrator's guide / MySQL Ab. – [S. l.] : MySQL Press, 2005. – 696 p.
5. Carfantan J. PHP & MySQL / Jean Carfantan. – 6th ed. – Paris : Micro Application, 2012. – 431 p.
6. Krug S. Don't make me think, revisited: a common sense approach to web usability / Steve Krug. – [S. l.] : New Riders, 2014. – 216 p.
7. Duckett J. HTML & CSS: design and build websites / Jon Duckett. – [S. l. : s. n.], 2014. – 490 p.
8. Freeman E. Head first HTML and CSS: a learner's guide to creating standards-based web pages / Eric Freeman, Elisabeth Robson. – [S. l.] : O'Reilly Media, Incorporated, 2012. – 768 p.
9. Hu Y. Self master HTML CSS javascript: HTML CSS javascript beginner guide / Yang Hu. – [S. l.] : Independently Published, 2019. – 119 p.
10. Kozyra N. Learning go web development / Nathan Kozyra. – [S. l.] : Packt Publishing - ebooks Account, 2016. – 136 p.
11. LaGrone B. HTML5 and CSS3 responsive web design cookbook / Benjamin LaGrone. – [S. l.] : Packt Publishing, 2013. – 204 p.
12. MacGregor A. Magento PHP developer's guide / Allan MacGregor. – [S. l.] : Packt Publishing, 2013. – 256 p.

13. Puntambekar A. A. Web based application development: using PHP / Anuradha A. Puntambekar. – [S. l.] : 9789333223881, 2020. – 128 p.

14. Shklar L. Web application architecture: principles, protocols and practices / Leon Shklar, Rich Rosen. – [S. l.] : Wiley & Sons, Incorporated, John, 2011. – 440 p.

15. Sinha S. Beginning laravel: build websites with laravel 5.8 / Sanjib Sinha. – [S. l.] : Apress, 2019. – 422 p.

16. Stauffer M. Laravel : up & running: A framework for building modern PHP apps / Matt Stauffer. – [S. l.] : O'Reilly Media, 2019. – 544 p.

17. Stubbs T. Web design basics / Todd Stubbs. – Cambridge, Mass : Course Technology, 2003. – 204 p.

18. Tidwell J. Designing interfaces: patterns for effective interaction design / Jenifer Tidwell, Charles Brewer, Aynne Valencia. – [S. l.] : O'Reilly Media, Incorporated, 2020. – 500 p.

19. W3Schools українською [Електронний ресурс] – Режим доступу: https://w3schoolsua.github.io/ (дата звернення: 01.06.2023). – Назва з екрана.