Перший європейський університет
ННІ «Європейська школа бізнесу»
Кафедра інформаційних технологій

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
інформаційних технологій

(підпис, ПІБ)
«___» _____20___

КВАЛІФІКАЦІЙНА БАКАЛАВРСЬКА РОБОТА
(ПОЯСНЮВАЛЬНА ЗАПИСКА)
зі спеціальності 121 «Інженерія програмного забезпечення»

ТЕМА: «Розробка довідкової системи корупціонерів України»

Виконавець: Симоненко Дмитро Олександрович
(ПІБ)

Науковий керівник: Нестеренко Олександр Васильович, професор кафедри інформаційних технологій
(ПІБ)

Консультанти з окремих розділів пояснювальної записки:

(ПІБ) (ПІБ)

Нормоконтролер:

_____
(ПІБ)

КИЇВ — 2026

Перший європейський університет
ННІ «Європейська школа бізнесу»
Кафедра інформаційних технологій

Ступінь вищої освіти - перший (бакалаврський)
рівень Спеціальність 121 «Інженерія програмного забезпечення» Освітньо-професійна програма «Інженерія програмного забезпечення» Освітньо-кваліфікаційний рівень бакалавр

ЗАТВЕРДЖУЮ:
Професор кафедри інформаційних технологій ННІ «Європейська школа бізнесу»
Нестеренко О. В.
«____» _____2026 р.

ЗАВДАННЯ

НА ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ БАКАЛАВРСЬКОЇ РОБОТИ

Симоненко Дмитро Олександрович

1. Тема:
«Розробка довідкової системи корупціонерів України», затверджена наказом ректора від «24» грудня 2025 р. № 207-к.

2. Термін виконання роботи:
з «26» травня 2026 р. по «22» червня 2026 р.

3. Дата подання роботи на випускну кафедру:
«12» червня 2026 р.

4. Вихідні дані роботи: вихідні дані роботи:
нормативно-правові та довідкові матеріали щодо корупційних і пов'язаних з корупцією правопорушень; вимоги до побудови довідкових інформаційних систем; принципи проєктування вебзастосунків; PHP, Laravel, MySQL, Blade, HTML, CSS; середовище розробки Laravel Herd; структура бази даних, що містить таблиці осіб, регіонів, статей законодавства та записів правопорушень; реалізація CRUD-функціоналу, пошуку, фільтрації та статистичного модуля.

5. Зміст пояснювальної записки:
Вступ.
Розділ 1. Аналіз предметної області та існуючих інформаційних систем.
Розділ 2. Проєктування довідкової системи корупціонерів України.
Розділ 3. Реалізація та тестування довідкової системи.
Висновки.
Список використаних джерел.
Додатки.

6. Перелік обов'язкового графічного матеріалу: структурна схема системи; схема бази даних; ER-діаграма; UML-діаграма варіантів використання; блок-схема роботи системи; скріншоти інтерфейсу розробленого вебзастосунку.

7. Календарний план - графік виконання КБР

№ з/п	Завдання	Термін виконання	Відмітка про виконання
1	Вибір теми кваліфікаційної бакалаврської роботи, подання заяви на кафедру, затвердження теми та отримання індивідуального завдання	01.05.2026	
2	Підготовка вступу і розділу 1 кваліфікаційної бакалаврської роботи	04.05.2026	
3	Підготовка розділу 2 кваліфікаційної бакалаврської роботи	09.05.2026	
4	Підготовка розділу 3 кваліфікаційної бакалаврської роботи, висновків і списку використаних джерел	15.05.2026	
5	Подання студентом завершеної кваліфікаційної бакалаврської роботи науковому керівнику для перевірки на плагіат та оформлення відгуку	29.05.2026	
6	Попередній розгляд кваліфікаційної бакалаврської роботи на комісії від кафедри	12-13.06.2026	
7	Доопрацювання роботи, прийняття кафедрою рішення про допуск роботи до захисту, оформлення та зовнішнє рецензування	16-20.06.2026	
8	Захист кваліфікаційної бакалаврської роботи в ЕК і присвоєння випускнику кваліфікації	23-24.06.2026	

Студент _____

(підпис, ПІБ)

Керівник _____

(підпис, ПІБ)

РЕФЕРАТ

Тема: Розробка довідкової системи корупціонерів України
Кваліфікаційна бакалаврська робота зі спеціальності 121 «Інженерія програмного забезпечення», ОПП «Інженерія програмного забезпечення», Міжнародний європейський університет, 2025 р.

Загальний обсяг роботи 85 стор., складається зі вступу, трьох розділів, висновків, списку використаних джерел та додатків; містить 7 таблиць, 11 рисунків, 23 літературних джерел.

Ключові слова: довідкова система, корупційні правопорушення, вебзастосунок, база даних, Laravel, PHP, MySQL, CRUD, пошук, фільтрація, статистика.

Актуальність теми роботи: У сучасних умовах цифровізації державного управління та розвитку відкритих інформаційних ресурсів зростає потреба у створенні зручних програмних засобів для систематизації, зберігання та пошуку суспільно важливої інформації. Дані про корупційні та пов’язані з корупцією правопорушення потребують структурованого представлення, можливості швидкого пошуку, фільтрації за різними критеріями та аналітичного узагальнення. У зв’язку з цим розробка довідкової системи корупціонерів України є актуальним завданням, оскільки така система може бути використана як навчальний приклад побудови інформаційного вебзастосунку з базою даних, CRUD-функціоналом, пошуком, фільтрами та статистичним модулем.

Метою роботи є розробка веборієнтованої довідкової системи корупціонерів України, яка забезпечує зберігання, перегляд, додавання, редагування, видалення, пошук і фільтрацію записів про корупційні правопорушення, а також формування статистичної інформації.

Для досягнення поставленої мети у роботі вирішено такі завдання:

1. Проаналізовано предметну область та особливості побудови довідкових інформаційних систем.
2. Визначено функціональні вимоги до довідкової системи корупціонерів України.
3. Обґрунтовано вибір технологій для реалізації вебзастосунку.
4. Спроєктовано структуру бази даних із таблицями осіб, регіонів, статей законодавства та записів правопорушень.

5. Реалізовано вебзастосунок на основі PHP-фреймворку Laravel з використанням MySQL.
6. Реалізовано CRUD-функціональність для записів правопорушень та керування фізичними особами.
7. Додано пошук, фільтрацію за регіоном і статтею, детальний перегляд записів та статистичний модуль.
8. Проведено тестування основних функцій системи.

Об’єктом роботи є процес розробки веборієнтованих інформаційно-довідкових систем.

Предмет дослідження — методи, засоби та технології створення довідкової системи для обліку й пошуку інформації про корупційні та пов’язані з корупцією правопорушення.

Теоретичне та практичне значення одержаних результатів полягає у створенні навчального вебзастосунку, який демонструє принципи проєктування бази даних, реалізації зв’язків між сутностями, CRUD-операцій, пошуку, фільтрації та статистичної обробки інформації. Розроблена система може бути використана як демонстраційний програмний продукт для кваліфікаційної бакалаврської роботи, а також як основа для подальшого розширення функціоналу довідкових інформаційних систем.

Публікації, апробація: Розроблений програмний продукт протестовано в локальному середовищі Laravel Herd. Під час тестування перевірено роботу бази даних, сторінок перегляду, додавання, редагування та видалення записів, пошуку, фільтрації, керування особами та статистичного модуля.

ЗМІСТ

ВСТУП

У сучасних умовах розвитку інформаційного суспільства особливого значення набувають програмні засоби, що забезпечують швидкий доступ до структурованої, достовірної та зручно представленої інформації. Розвиток електронного урядування, відкритих даних, цифрових сервісів та онлайн-реєстрів сприяє перенесенню значної частини суспільно важливої інформації у цифрове середовище [10]. Це дає змогу підвищувати прозорість суспільних процесів, спрощувати пошук необхідних відомостей і забезпечувати зручну взаємодію користувачів з інформаційними ресурсами.

Однією з актуальних сфер, у якій доцільним є використання інформаційних систем, є облік та систематизація відомостей про корупційні та пов’язані з корупцією правопорушення. Корупція є складною соціально-правовою проблемою, що впливає на ефективність державного управління, рівень довіри громадян до органів влади та загальний розвиток суспільства [1]. Саме тому інформація про осіб, правопорушення, регіони, статті законодавства, рішення судів або інших уповноважених органів повинна бути впорядкованою, доступною для пошуку та зручною для аналізу.

Традиційне зберігання подібної інформації у вигляді окремих документів, таблиць або неструктурованих списків має низку недоліків. Зокрема, ускладнюється пошук потрібних записів, відсутня можливість швидкої фільтрації за регіоном або статтею, складно виконувати узагальнення даних і формувати статистичні показники. У зв’язку з цим виникає потреба у розробці спеціалізованої довідкової системи, яка дозволить зберігати дані у структурованому вигляді та забезпечить зручний інтерфейс для роботи з ними.

Актуальність теми кваліфікаційної бакалаврської роботи полягає у необхідності створення веборієнтованої довідкової системи, яка дозволяє здійснювати облік, пошук, фільтрацію, перегляд, додавання, редагування та видалення записів про корупційні правопорушення. Така система є прикладом практичного застосування сучасних технологій веброзробки, баз даних та принципів побудови інформаційних систем. Розроблений програмний продукт може бути використаний як навчальний і демонстраційний приклад побудови довідкової системи з пов’язаними сутностями, CRUD-функціональністю та аналітичним модулем.

Проблема створення довідкових інформаційних систем є актуальною для багатьох сфер діяльності, оскільки такі системи дозволяють організувати великі обсяги інформації, забезпечити її логічне

структурування та спростити доступ користувачів до потрібних даних [10; 15]. У контексті даної роботи особливу увагу приділено розробці системи, яка працює з кількома взаємопов’язаними сутностями: фізичними особами, регіонами України, статтями законодавства та записами правопорушень.

Для реалізації проєкту було обрано веборієнтований підхід, оскільки вебзастосунки є зручними для користувачів, не потребують встановлення окремого клієнтського програмного забезпечення та можуть працювати через браузер. Як основну технологічну платформу використано PHP-фреймворк Laravel, який надає зручні засоби для маршрутизації, роботи з базою даних, створення моделей, контролерів і шаблонів сторінок [19]. Для зберігання даних використано реляційну базу даних MySQL, а інтерфейс системи реалізовано з використанням Blade-шаблонів, HTML та CSS [20; 22; 23].

Метою кваліфікаційної бакалаврської роботи є розробка довідкової системи корупціонерів України у вигляді вебзастосунку, який забезпечує зберігання, перегляд, додавання, редагування, видалення, пошук, фільтрацію та статистичне узагальнення інформації про корупційні та пов’язані з корупцією правопорушення. Для досягнення поставленої мети необхідно вирішити такі завдання:

Проаналізувати предметну область та визначити особливості обліку інформації про корупційні правопорушення. Розглянути принципи побудови довідкових інформаційних систем і визначити вимоги до розроблюваного програмного продукту. Обґрунтувати вибір архітектури вебзастосунку та технологій реалізації. Спроєктувати структуру бази даних для зберігання інформації про осіб, регіони, статті законодавства та записи правопорушень. Реалізувати моделі, контролери, маршрути та сторінки вебзастосунку. Забезпечити CRUD-функціональність для записів правопорушень. Реалізувати функції додавання, перегляду та видалення фізичних осіб. Додати пошук і фільтрацію записів за основними критеріями. Реалізувати сторінку детального перегляду запису та статистичний модуль. Провести тестування основних функцій системи та перевірити коректність роботи з базою даних. Об’єктом дослідження є процес розробки веборієнтованих інформаційно-довідкових систем.

Предметом дослідження є методи, засоби та технології створення довідкової системи для обліку, пошуку, фільтрації та аналізу інформації про корупційні та пов’язані з корупцією правопорушення.

У процесі виконання роботи використано такі методи дослідження: аналіз предметної області — для визначення структури даних та основних функціональних вимог до системи; порівняльний аналіз технологій — для обґрунтування вибору засобів розробки; моделювання — для проєктування структури бази даних і зв’язків між сутностями; методи програмної інженерії — для реалізації вебзастосунку; тестування — для перевірки працездатності основних функцій системи.

Практичне значення роботи полягає у створенні функціонального вебзастосунку, який демонструє повний цикл розробки довідкової інформаційної системи: від аналізу предметної області та проєктування бази даних до реалізації інтерфейсу користувача, CRUD-операцій, пошуку, фільтрації та статистики. Розроблена система може бути використана як навчальний програмний продукт, демонстраційний проєкт для захисту кваліфікаційної роботи, а також як основа для подальшого розвитку більш складної інформаційної системи.

Структура кваліфікаційної бакалаврської роботи складається зі вступу, трьох розділів, висновків, списку використаних джерел і додатків. У першому розділі розглядаються теоретичні аспекти предметної області та особливості довідкових інформаційних систем. У другому розділі виконується проєктування системи, обґрунтовується вибір технологій та описується структура бази даних. У третьому розділі наведено реалізацію програмного продукту, описано основні модулі системи та результати тестування.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ІСНУЮЧИХ ІНФОРМАЦІЙНИХ СИСТЕМ

1.1. Поняття та призначення довідкових інформаційних систем

У сучасному інформаційному суспільстві значна частина даних створюється, зберігається та обробляється в електронному вигляді. Зростання обсягів інформації зумовлює необхідність використання спеціалізованих програмних засобів, які дозволяють упорядковувати дані, забезпечувати швидкий доступ до них, здійснювати пошук, фільтрацію та аналіз. Одним із таких засобів є довідкові інформаційні системи.

Довідкова інформаційна система — це програмний комплекс, призначений для зберігання, структурування, пошуку та надання користувачеві інформації з певної предметної області [10; 15]. Основною метою таких систем є забезпечення зручного доступу до впорядкованих даних. На відміну від звичайних текстових документів або електронних таблиць, довідкова система дозволяє працювати з інформацією більш ефективно, оскільки дані зберігаються у структурованому вигляді та можуть бути пов’язані між собою.

Довідкові системи застосовуються у різних сферах діяльності: освіті, медицині, державному управлінні, правовій сфері, бізнесі, кадровому обліку, архівах, бібліотеках, електронних каталогах та реєстрах. Вони можуть містити інформацію про осіб, організації, документи, нормативно-правові акти, події, товари, послуги або інші об’єкти. Головною особливістю таких систем є те, що користувач має можливість швидко знайти потрібні відомості за певними критеріями.

Основними функціями довідкових інформаційних систем є:
1. зберігання даних у структурованому вигляді;
2. додавання нових записів;
3. редагування наявної інформації;
4. видалення застарілих або помилкових записів;
5. пошук інформації за ключовими словами;
6. фільтрація даних за визначеними параметрами;
7. перегляд детальної інформації про окремий запис;
8. формування статистичних або аналітичних показників.

Важливим елементом довідкової системи є база даних [15]. Саме база даних забезпечує збереження інформації у вигляді таблиць, між якими можуть бути встановлені зв’язки. Наприклад, у системі можуть окремо зберігатися дані про осіб, регіони, категорії, документи або події, а потім ці дані можуть поєднуватися між собою через відповідні зв’язки. Такий підхід дозволяє уникати дублювання інформації та забезпечує цілісність даних.

Для користувача довідкова система повинна мати зрозумілий інтерфейс. Інтерфейс є посередником між користувачем і базою даних. Саме через інтерфейс користувач виконує основні дії: переглядає список записів, вводить пошуковий запит, обирає фільтри, відкриває сторінку детального перегляду, додає або редагує інформацію. Тому при розробці довідкової системи важливо враховувати не лише технічну частину, а й зручність використання.

Однією з переваг веборієнтованих довідкових систем є доступність через браузер [16; 17]. Користувачеві не потрібно встановлювати окреме програмне забезпечення на комп’ютер, достатньо мати доступ до вебсторінки. Це робить такі системи зручними для використання в навчальних, адміністративних та інформаційних цілях. Крім того, вебзастосунок легше масштабувати, оновлювати та підтримувати.

Довідкові інформаційні системи можуть бути як простими, так і складними [14; 16]. Проста система може містити лише список записів і базовий пошук. Складніші системи мають авторизацію користувачів, ролі доступу, розширені фільтри, аналітичні модулі, імпорт і експорт даних, інтеграцію з іншими сервісами. Вибір функціоналу залежить від мети розробки, потреб користувачів і предметної області.

У межах даної кваліфікаційної роботи довідкова система розглядається як вебзастосунок, що забезпечує облік відомостей про корупційні та пов’язані з корупцією правопорушення. Для такої системи важливими є структурованість інформації, можливість швидкого пошуку за особою, фільтрація за регіоном і статтею законодавства, а також перегляд детальної інформації про кожний запис. Додатково важливим є наявність статистичного модуля, який дозволяє отримати узагальнені дані щодо кількості записів, осіб, регіонів і правопорушень за певними категоріями.

Таким чином, довідкова інформаційна система є ефективним інструментом для організації та обробки структурованих даних. Її використання дозволяє підвищити зручність доступу до інформації, скоротити час пошуку, зменшити кількість помилок під час роботи з даними та забезпечити можливість аналітичного узагальнення. Саме тому розробка довідкової системи корупціонерів України є доцільною з погляду практичного застосування сучасних засобів веброзробки та баз даних.

1.2. Особливості обліку інформації про корупційні правопорушення

Облік інформації про корупційні та пов'язані з корупцією правопорушення має важливе значення для забезпечення прозорості суспільних процесів, контролю за дотриманням антикорупційного законодавства та формування довіри громадян до державних інституцій. Корупційні правопорушення належать до категорії суспільно небезпечних явищ, оскільки вони можуть негативно впливати на якість державного управління, ефективність використання бюджетних коштів, справедливість прийняття управлінських рішень та рівень правової культури в суспільстві [1].

У сучасних умовах облік таких відомостей доцільно здійснювати за допомогою електронних інформаційних систем. Це пояснюється тим, що дані про правопорушення мають складну структуру та можуть містити значну кількість пов'язаних елементів: інформацію про особу, місце роботи, посаду, регіон, статтю законодавства, дату рішення, номер рішення, суд або інший орган, вид відповідальності та опис обставин правопорушення. Зберігання таких даних у неструктурованому вигляді ускладнює пошук, аналіз і подальше використання інформації.

В Україні функціонує Єдиний державний реєстр осіб, які вчинили корупційні або пов'язані з корупцією правопорушення [2; 3]. На офіційному порталі НАЗК цей ресурс представлено як реєстр, що містить відомості про осіб, які вчинили корупційні або пов'язані з корупцією правопорушення, а також забезпечує можливості пошуку, отримання довідок, перегляду аналітики та використання відкритого API [3; 4].

Відповідно до Положення про Єдиний державний реєстр, реєстр є електронною базою даних, яка містить відомості про фізичних осіб, що вчинили корупційні або пов’язані з корупцією правопорушення, а також про юридичних осіб, до яких застосовано заходи кримінально-правового характеру у зв’язку з корупційним правопорушенням [2].

Наявність такого офіційного реєстру підтверджує актуальність створення навчальної довідкової системи, яка моделює основні принципи роботи з подібними даними. У межах кваліфікаційної роботи не ставиться мета дублювати офіційний державний ресурс або замінити його. Розроблена система має навчальний характер і демонструє підхід до побудови вебзастосунку, який працює зі структурованою інформацією про осіб, регіони, статті законодавства та записи правопорушень.

Особливістю обліку інформації про корупційні правопорушення є необхідність забезпечення логічного зв’язку між різними типами даних. Наприклад, один запис правопорушення пов’язаний із конкретною особою, певним регіоном та відповідною статтею законодавства. Якщо всі ці дані зберігати в одній таблиці без нормалізації, це може призвести до дублювання інформації та складності оновлення записів. Тому під час проєктування доцільно розділяти дані на окремі сутності: «особи», «регіони», «статті» та «правопорушення» [12; 15].

Такий підхід дозволяє забезпечити цілісність інформації. Наприклад, регіони зберігаються в окремому довіднику, а в записі правопорушення використовується лише посилання на відповідний регіон. Аналогічно статті законодавства зберігаються окремо, що дає змогу уникнути повторного введення однакових назв і кодів статей. Дані про фізичних осіб також виділяються в окрему таблицю, оскільки одна особа теоретично може мати кілька пов’язаних записів правопорушень.

Ще однією важливою особливістю є необхідність швидкого пошуку інформації. У довідковій системі користувач повинен мати змогу знайти запис за прізвищем, ім’ям, по батькові, місцем роботи або посадою. Такий пошук є зручним для користувача, оскільки не вимагає точного знання всіх реквізитів запису. Додатково важливими є фільтри за регіоном та статтею законодавства, які дозволяють звузити список записів і швидко отримати потрібний результат.

Для обліку корупційних правопорушень важливо також передбачити можливість перегляду детальної інформації. У загальній таблиці доцільно відображати лише основні поля, такі як ПІБ особи, місце роботи, регіон, стаття, дата рішення та вид стягнення. Водночас на окремій сторінці детального перегляду можна розмістити повну інформацію: дату народження особи, посаду, номер рішення, суд або орган, опис правопорушення та опис відповідної статті. Це дозволяє не перевантажувати головну сторінку зайвими даними, але зберігати можливість доступу до повної інформації.

Окрему роль відіграє аналітична складова. Інформаційна система повинна не лише зберігати записи, а й надавати узагальнені статистичні показники. До таких показників можна віднести загальну кількість записів, кількість осіб, кількість статей у довіднику, кількість регіонів, а також розподіл правопорушень за регіонами, статтями та роками. Наявність статистичного модуля підвищує практичну цінність системи, оскільки дозволяє швидко оцінити загальну картину даних.

Під час розробки системи також необхідно враховувати питання коректності та відповідальності при роботі з персональними даними. У межах навчального проєкту доцільно використовувати тестові або умовні записи, щоб уникнути неправомірного використання персональної інформації реальних осіб. Такий підхід дозволяє продемонструвати функціональні можливості системи без порушення етичних і правових вимог.

Отже, облік інформації про корупційні правопорушення потребує структурованого підходу, використання бази даних, поділу інформації на взаємопов’язані сутності, реалізації пошуку, фільтрації, детального перегляду та статистичної обробки. Саме ці особливості були враховані під час розробки довідкової системи корупціонерів України, що є предметом даної кваліфікаційної бакалаврської роботи.

1.3. Аналіз існуючих інформаційних систем і реєстрів

У процесі проєктування довідкової системи корупціонерів України важливо проаналізувати наявні інформаційні ресурси, які вже використовуються для зберігання, пошуку та оприлюднення відомостей про корупційні або пов’язані з корупцією правопорушення. Такий аналіз дозволяє визначити основні функціональні можливості подібних систем, їх переваги та обмеження, а також сформувати вимоги до власного програмного продукту.

Основним офіційним інформаційним ресурсом у цій сфері є Єдиний державний реєстр осіб, які вчинили корупційні або пов’язані з корупцією правопорушення [3]. Цей реєстр функціонує як електронний портал, де міститься інформація про фізичних та юридичних осіб, які вчинили відповідні правопорушення. Адміністрування реєстру здійснює Національне агентство з питань запобігання корупції [4]. На офіційному порталі реєстру передбачено можливість пошуку в реєстрі, отримання особистої довідки фізичної або юридичної особи, перевірки довідки, перегляду інформації про реєстр та використання аналітичних можливостей.

Офіційний реєстр має важливе практичне значення, оскільки забезпечує централізоване зберігання відомостей про осіб, притягнутих до відповідальності за корупційні або пов’язані з корупцією правопорушення. Відповідно до інформації на порталі відкритих даних, з 04.02.2019 року формування і ведення цього реєстру здійснюється Національним агентством з питань запобігання корупції. На порталі відкритих даних також зазначено нормативні підстави функціонування реєстру, зокрема статті 11 і 59 Закону України «Про запобігання корупції» та відповідні нормативні акти [1; 2].

Структура даних, що використовується у відкритих наборах, демонструє складність предметної області [3]. Набір даних може містити реєстраційний запис, дату реєстрації, вид покарання або стягнення, ПІБ особи, місце роботи, статті кодексів, склад корупційного або пов’язаного з корупцією правопорушення, вид дисциплінарного стягнення, дані про юридичну особу, дату та номер судового рішення або наказу про накладення дисциплінарного стягнення. Це підтверджує необхідність використання структурованої бази даних і поділу інформації на окремі логічні сутності.

До переваг офіційного реєстру можна віднести централізованість, нормативну визначеність, наявність пошуку, можливість отримання довідок, а також зв’язок з офіційними державними даними. Такий ресурс орієнтований на практичне використання громадянами, установами, організаціями та іншими суб’єктами, яким необхідно перевірити інформацію щодо фізичних або юридичних осіб. Наявність такого реєстру є прикладом цифровізації антикорупційної сфери та підтверджує доцільність створення інформаційних систем для роботи з подібними даними [9; 10].

Разом з тим, офіційний реєстр має власну специфіку. Він є державним ресурсом і працює відповідно до визначених нормативних правил, тому його структура, інтерфейс та функціональність орієнтовані на офіційне використання. Для навчального проєкту доцільно створити спрощену довідкову систему, яка не дублює державний реєстр, а демонструє принципи побудови подібного програмного забезпечення: проєктування бази даних, створення зв’язків між таблицями, реалізацію пошуку, фільтрації, CRUD-операцій та статистичного модуля [5; 14; 15].

Окрім офіційного державного реєстру, у відкритому доступі існують інформаційні сервіси, які використовують відкриті державні дані для відображення відомостей у зручному для користувача вигляді. Прикладом такого сервісу є Opendatabot, який надає доступ до інформації про корупційні правопорушення та пояснює, що реєстр корупціонерів є списком людей, які вчинили корупційні або пов’язані з корупцією правопорушення. У сервісі також зазначається, що особа потрапляє до такого реєстру після притягнення до відповідальності за відповідне правопорушення.

Подібні сервіси мають перевагу у вигляді зручного інтерфейсу, швидкого доступу до даних та інтеграції з іншими відкритими реєстрами. Водночас вони часто залежать від джерел відкритих даних, їх актуальності, доступності та формату оновлення. Такі системи більше орієнтовані на кінцевого користувача, тоді як у межах кваліфікаційної роботи основний акцент робиться на розробці власної архітектури, моделі даних і функціональних модулів вебзастосунку.

Аналіз існуючих інформаційних систем показує, що ключовими функціями для роботи з даними про корупційні правопорушення є пошук, фільтрація, перегляд детальної інформації, структуроване зберігання даних і формування аналітики. Саме ці можливості були взяті за основу під час розробки власної довідкової системи. У створеному програмному продукті реалізовано список записів, сторінку детального перегляду, додавання, редагування та видалення записів, керування фізичними особами, пошук за ПІБ, місцем роботи або посадою, фільтрацію за регіоном і статтею, а також сторінку статистики.

На відміну від офіційних державних ресурсів, розроблена система має навчальний характер і використовує тестові записи. Її призначення полягає не в офіційному оприлюдненні даних про реальних осіб, а в демонстрації процесу створення інформаційно-довідкового вебзастосунку. Такий підхід дозволяє безпечно відпрацювати всі основні етапи розробки: від аналізу предметної області та проєктування бази даних до реалізації інтерфейсу, CRUD-функціональності та тестування.

Таким чином, аналіз існуючих інформаційних систем і реєстрів свідчить про актуальність теми роботи та доцільність розробки власної довідкової системи. Офіційні реєстри демонструють практичну потребу в централізованому обліку даних про корупційні правопорушення, а відкриті сервіси показують важливість зручного інтерфейсу і швидкого пошуку. Власна система, розроблена в межах кваліфікаційної роботи, поєднує базові принципи таких ресурсів у навчальному програмному продукті, орієнтованому на демонстрацію навичок веброзробки, роботи з базами даних і побудови інформаційних систем.

1.4. Постановка задачі розробки довідкової системи

На основі аналізу предметної області та існуючих інформаційних систем можна визначити основну задачу кваліфікаційної роботи — розробити веборієнтовану довідкову систему, яка забезпечує структуроване зберігання, пошук, фільтрацію, перегляд, додавання, редагування, видалення та аналітичне узагальнення інформації про корупційні та пов’язані з корупцією правопорушення [5; 14].

Розроблювана система повинна мати навчально-прикладний характер і демонструвати принципи побудови сучасного вебзастосунку з використанням бази даних. Вона не є офіційним державним реєстром і не призначена для публікації юридично значущих відомостей про реальних осіб. Основна мета системи полягає у відтворенні логіки роботи довідкового інформаційного ресурсу, який працює зі структурованими записами, пов’язаними сутностями та основними операціями керування даними [15; 16].

У межах розробки необхідно створити програмний продукт, який дозволяє користувачеві переглядати список записів про правопорушення, відкривати детальну інформацію про кожний запис, додавати нові записи, редагувати наявні дані, видаляти непотрібні записи, здійснювати пошук і використовувати фільтри. Крім цього, система повинна мати окрему сторінку статистики, яка відображає узагальнені показники щодо кількості записів, осіб, регіонів, статей законодавства та розподілу правопорушень за окремими критеріями.

Для реалізації довідкової системи необхідно визначити основні сутності предметної області. До них належать фізичні особи, регіони, статті законодавства та записи правопорушень. Кожна з цих сутностей повинна зберігатися в окремій таблиці бази даних [12; 15]. Такий підхід дозволяє уникати дублювання інформації, забезпечувати логічні зв’язки між даними та підвищувати зручність подальшої обробки інформації.

Сутність «особа» повинна містити такі дані: прізвище, ім’я, по батькові, дату народження, місце роботи та посаду. Ці дані використовуються для ідентифікації фізичної особи в межах навчальної системи та відображаються у списку записів і на сторінці детального перегляду. Окреме зберігання осіб дозволяє додавати нові записи правопорушень для вже наявних осіб без повторного введення однакової інформації.

Сутність «регіон» призначена для зберігання назв областей України та міста Києва. Вона використовується як довідник, за яким можна фільтрувати записи правопорушень. Це дозволяє користувачеві швидко знайти записи, пов’язані з певною адміністративно-територіальною одиницею.

Сутність «стаття законодавства» повинна містити код статті, її назву та короткий опис. Такий довідник потрібен для систематизації правопорушень за правовою ознакою. У системі користувач може фільтрувати записи за статтею, а на сторінці детального перегляду бачити не лише код статті, а й її назву та опис.

Основною сутністю системи є «запис правопорушення». Вона поєднує особу, регіон і статтю законодавства, а також містить додаткові дані: дату рішення, номер рішення, назву суду або органу, вид стягнення та опис правопорушення. Саме ця сутність є центральною в базі даних і забезпечує роботу основного функціоналу довідкової системи.

Функціональні вимоги до розроблюваної системи можна сформулювати таким чином [14; 16]:

1) система повинна відображати список записів правопорушень у вигляді таблиці;
2) система повинна забезпечувати детальний перегляд окремого запису;
3) система повинна дозволяти додавати нові записи правопорушень;
4) система повинна дозволяти редагувати наявні записи;
5) система повинна забезпечувати видалення записів;
6) система повинна дозволяти додавати нових фізичних осіб;
7) система повинна відображати список фізичних осіб;
8) система повинна дозволяти видаляти фізичних осіб;
9) система повинна підтримувати пошук за ПІБ, місцем роботи та посадою;
10) система повинна підтримувати фільтрацію записів за регіоном і статтею законодавства;
11) система повинна мати сторінку статистики;
12) система повинна працювати з реляційною базою даних;
13) система повинна мати зрозумілий вебінтерфейс користувача.

Окрім функціональних вимог, до системи висуваються й нефункціональні вимоги. Вебзастосунок повинен мати простий і зрозумілий інтерфейс, бути зручним для користувача, забезпечувати

коректну роботу з формами введення даних, виводити повідомлення про успішне виконання дій, а також зберігати цілісність інформації в базі даних. Система повинна бути побудована таким чином, щоб її можна було надалі розширювати, наприклад, додати авторизацію користувачів, ролі доступу, імпорт даних або експорт результатів пошуку.

Для реалізації поставленої задачі було обрано архітектуру вебзастосунку на основі шаблону MVC [18]. Такий підхід дозволяє розділити логіку роботи програми на три основні частини: модель, представлення та контролер. Моделі відповідають за роботу з даними, контролери обробляють запити користувача, а представлення відповідають за відображення інформації у браузері. Використання такої архітектури сприяє кращій організації коду та спрощує подальшу підтримку системи.

Як технологічну основу для реалізації системи обрано PHP-фреймворк Laravel [19]. Він забезпечує зручні інструменти для створення маршрутів, контролерів, моделей, міграцій бази даних та шаблонів сторінок. Для зберігання даних використано MySQL, оскільки ця система керування базами даних добре підходить для роботи з реляційною структурою та підтримує зв’язки між таблицями [15; 20]. Інтерфейс користувача реалізовано за допомогою Blade-шаблонів, HTML і CSS.

Під час розробки передбачається створення таких основних модулів системи: модуль перегляду записів, модуль детального перегляду, модуль додавання та редагування записів, модуль видалення, модуль керування особами, модуль пошуку й фільтрації, а також модуль статистики. Кожен із цих модулів виконує окрему функцію, але разом вони формують єдину довідкову систему.

Очікуваним результатом виконання роботи є функціональний вебзастосунок, який дозволяє працювати з інформацією про корупційні правопорушення в межах навчальної довідкової системи. Розроблений продукт повинен демонструвати навички проєктування бази даних, створення вебінтерфейсу, реалізації CRUD-операцій, роботи з пов’язаними таблицями, пошуку, фільтрації та статистичного узагальнення даних.

Таким чином, постановка задачі передбачає розробку повноцінної інформаційно-довідкової системи, яка поєднує базу даних, вебінтерфейс, функції керування записами та аналітичний модуль. Реалізація такої системи дозволяє практично застосувати знання з програмної інженерії, баз даних, веброзробки та проєктування інформаційних систем.

1.5. Висновки до розділу 1

У першому розділі кваліфікаційної бакалаврської роботи було розглянуто теоретичні та практичні аспекти предметної області, пов’язаної з розробкою довідкової системи корупціонерів України. Визначено, що довідкові інформаційні системи є важливим інструментом для зберігання, структурування, пошуку та аналізу даних у різних сферах діяльності.

Було встановлено, що інформація про корупційні та пов’язані з корупцією правопорушення має складну структуру та потребує впорядкованого зберігання. Саме тому для роботи з такими відомостями доцільно використовувати базу даних із взаємопов’язаними таблицями.

У процесі аналізу існуючих інформаційних систем і реєстрів було визначено, що в Україні вже функціонують офіційні електронні ресурси для обліку відповідних відомостей. Наявність таких систем підтверджує актуальність теми роботи та практичну потребу у створенні програмних засобів для роботи з такою інформацією.

На основі проведеного аналізу було сформульовано задачу розробки власної довідкової системи, яка повинна забезпечувати перегляд, додавання, редагування та видалення даних, пошук, фільтрацію, детальний перегляд записів і формування статистичних показників.

Також було визначено основні сутності майбутньої системи: фізичні особи, регіони, статті законодавства та записи правопорушень. Такий поділ дає змогу побудувати логічну структуру бази даних, уникнути дублювання інформації та забезпечити коректні зв’язки між даними.

Отже, результати першого розділу підтверджують доцільність розробки веборієнтованої довідкової системи корупціонерів України. Проведений аналіз став основою для подальшого проєктування архітектури системи, вибору технологій, створення структури бази даних та реалізації програмного продукту.

РОЗДІЛ 2. ПРОЄКТУВАННЯ ДОВІДКОВОЇ СИСТЕМИ КОРУПЦІОНЕРІВ УКРАЇНИ

2.1. Вибір архітектури вебзастосунку

Після аналізу предметної області та визначення основних вимог до довідкової системи наступним етапом є вибір архітектури програмного продукту. Архітектура вебзастосунку визначає загальну структуру системи, спосіб взаємодії її компонентів, принципи обробки запитів користувача, роботу з базою даних та формування інтерфейсу [16; 17].

Для розробки довідкової системи корупціонерів України було обрано веборієнтовану архітектуру [16]. Такий підхід є доцільним, оскільки вебзастосунок доступний через браузер і не потребує встановлення окремої клієнтської програми на комп'ютер користувача. Це спрощує використання системи, полегшує її подальше оновлення та дозволяє централізовано зберігати й обробляти дані.

Розроблена система має клієнт-серверну архітектуру [17]. У такій архітектурі клієнтська частина відповідає за відображення інтерфейсу користувача у браузері, а серверна частина виконує обробку запитів, взаємодіє з базою даних і повертає користувачу сформовані сторінки. Користувач надсилає запит через вебінтерфейс, сервер обробляє його за допомогою відповідного маршруту та контролера, отримує або змінює дані в базі даних, після чого повертає результат у вигляді HTML-сторінки.

Такий підхід дозволяє чітко розділити відповідальність між клієнтською та серверною частинами системи. Клієнтська частина не виконує складної обробки даних, а лише забезпечує зручну взаємодію користувача з вебінтерфейсом. Основна бізнес-логіка, перевірка даних, робота з моделями та звернення до бази даних виконуються на сервері. Це підвищує надійність системи та спрощує її подальшу підтримку.

Загальну схему роботи довідкової системи наведено на рисунку 2.1. Вона відображає послідовність взаємодії користувача з вебзастосунком: від надсилання HTTP-запиту через веббраузер до обробки його маршрутами, контролером, моделлю, базою даних MySQL і повернення сформованої HTML-відповіді користувачеві.

Рисунок 2.1 – Загальна схема роботи довідкової системи

Як видно з рисунка 2.1, клієнтська частина системи представлена користувачем і веббраузером, через який надсилаються запити до серверної частини. Серверна частина реалізована на основі Laravel і містить маршрути, контролери, моделі Eloquent ORM, базу даних MySQL та Blade-шаблони. Така структура забезпечує обробку запитів користувача, взаємодію з базою даних і формування вебінтерфейсу.

Для організації логіки системи використано архітектурний шаблон MVC [18]. Назва MVC походить від трьох основних компонентів: Model, View, Controller. Такий підхід дозволяє розділити програмний код на логічні частини, кожна з яких виконує окрему функцію [18].

Компонент **Model** відповідає за роботу з даними та взаємодію з базою даних. У межах розробленої системи моделями є Person, Region, Article та Offense. Вони відповідають відповідним таблицям бази даних і забезпечують доступ до інформації про фізичних осіб, регіони, статті законодавства та записи правопорушень.

Компонент **View** відповідає за відображення інформації користувачу. У системі представлення реалізовані за допомогою Blade-шаблонів. Саме ці файли формують сторінки списку записів, детального перегляду, додавання, редагування, списку осіб і статистики. Завдяки шаблонам можна зручно поєднувати HTML-код з динамічними даними, які надходять із контролера.

Компонент **Controller** відповідає за обробку запитів користувача. Контролери приймають запити, звертаються до моделей, виконують необхідну логіку та передають дані у представлення. У розробленій системі основну логіку роботи із записами правопорушень реалізовано в OffenseController, а роботу з фізичними особами — у PersonController.

Застосування MVC-архітектури має кілька переваг. По-перше, вона забезпечує зрозумілу структуру проєкту. По-друге, спрощує подальшу підтримку і розширення системи. По-третє, дозволяє окремо працювати з логікою, даними та інтерфейсом, не змішуючи їх в одному файлі. Це особливо важливо для інформаційних систем, у яких передбачено багато операцій з даними.

У межах довідкової системи користувач взаємодіє із системою через вебсторінки. Основними сценаріями роботи є перегляд списку записів, пошук, фільтрація, відкриття детальної інформації, додавання нового запису, редагування, видалення, додавання нової особи, перегляд списку осіб і перегляд статистики. Кожна з цих дій обробляється відповідним маршрутом і методом контролера.

Загальний процес роботи системи можна описати так: користувач відкриває сторінку у браузері, Laravel приймає HTTP-запит, система маршрутизації визначає, який контролер має обробити цей запит, контролер звертається до відповідної моделі, модель отримує або змінює дані в базі MySQL, після чого контролер передає результат у Blade-шаблон, який формує HTML-сторінку для користувача.

Важливим елементом архітектури є база даних. Вона використовується для зберігання всієї основної інформації системи. Дані розділено на окремі таблиці відповідно до сутностей предметної області: people, regions, articles та offenses. Таблиця offenses є центральною, оскільки вона пов’язує між собою особу, регіон і статтю законодавства.

Така структура дозволяє забезпечити логічну організацію даних і уникнути дублювання інформації.

Для взаємодії з базою даних у Laravel використовується ORM Eloquent [19]. Завдяки цьому робота з таблицями здійснюється через моделі, що робить код зрозумілішим і зручнішим. Наприклад, запис правопорушення може бути пов'язаний з особою, регіоном і статтею через відповідні методи моделей. Це дозволяє легко отримувати повну інформацію про запис без написання складних SQL-запитів у кожному окремому випадку.

Обрана архітектура також передбачає можливість подальшого розширення системи. У майбутньому до неї можна додати авторизацію користувачів, розмежування прав доступу, імпорт даних із файлів, експорт результатів пошуку, розширену аналітику або підключення до зовнішніх джерел даних. Завдяки використанню Laravel і MVC-структури такі зміни можна впроваджувати поступово, не порушуючи загальну логіку роботи системи.

Таким чином, для реалізації довідкової системи корупціонерів України було обрано клієнт-серверну вебархітектуру з використанням шаблону MVC. Такий підхід є доцільним для інформаційної системи з базою даних, оскільки забезпечує чіткий поділ логіки, зручну організацію коду, можливість масштабування та підтримку основних операцій роботи з даними.

2.2. Обґрунтування вибору технологій розробки

Вибір технологій розробки є важливим етапом створення будь-якого програмного продукту, оскільки саме від нього залежить зручність реалізації функціоналу, швидкість розробки, можливість подальшого розширення системи, стабільність роботи та простота супроводу [14; 16]. Для створення довідкової системи корупціонерів України було обрано набір технологій, який дозволяє реалізувати повноцінний вебзастосунок із базою даних, CRUD-функціональністю, пошуком, фільтрацією та статистичним модулем.

Основною мовою програмування для серверної частини системи було обрано **PHP** [21]. Ця мова широко використовується для створення вебзастосунків і має значну кількість готових інструментів, бібліотек та

фреймворків. PHP добре підходить для роботи з HTML-сторінками, обробки форм, взаємодії з базами даних і побудови динамічних вебсайтів. Крім того, PHP підтримується більшістю хостинг-провайдерів, що робить розроблені на ньому застосунки зручними для подальшого розгортання.

Для реалізації серверної логіки було використано фреймворк Laravel [19]. Laravel є сучасним PHP-фреймворком, який забезпечує зручну структуру проєкту, підтримує архітектурний шаблон MVC, має вбудовані механізми маршрутизації, роботи з базами даних, шаблонізації, валідації даних і міграцій. Використання Laravel дозволяє прискорити розробку, зменшити кількість однотипного коду та організувати проєкт у зрозумілому вигляді.

Однією з важливих переваг Laravel є підтримка ORM Eloquent [19]. ORM дозволяє працювати з таблицями бази даних через моделі, не використовуючи пряме написання SQL-запитів у кожному випадку. У розробленій системі це використовується для роботи з моделями Person, Region, Article та Offense.

Для зберігання інформації було обрано реляційну систему керування базами даних MySQL [15; 20]. Вона є поширеною, стабільною та зручною для використання у вебпроєктах. У межах розробленої системи база даних містить таблиці people, regions, articles та offenses, між якими встановлено логічні зв’язки. Це дозволяє забезпечити цілісність даних і зменшити дублювання інформації.

Для створення структури бази даних використано механізм міграцій Laravel [19]. Міграції дозволяють описувати таблиці бази даних у вигляді PHP-коду та виконувати їх створення або зміну через консольні команди. У розробленому застосунку за допомогою міграцій було створено таблиці для регіонів, статей законодавства, фізичних осіб і записів правопорушень.

Для заповнення бази початковими тестовими даними використано seeders. Вони дозволяють автоматично додавати до бази даних початкові записи, необхідні для перевірки роботи системи.

У межах розробки були створені seeders для регіонів України, статей законодавства, тестових фізичних осіб і записів правопорушень. Це спростило процес тестування системи та дозволило одразу перевірити роботу списку записів, фільтрів, пошуку і статистики.

Для створення сторінок інтерфейсу користувача використано шаблонізатор **Blade** [19; 22; 23]. Blade є вбудованою системою шаблонів Laravel, яка дозволяє поєднувати HTML-код із динамічними даними, переданими з контролера. За допомогою Blade реалізовано головну сторінку зі списком записів, сторінку детального перегляду, сторінки додавання та редагування запису, сторінку додавання особи, список осіб і сторінку статистики.

Для оформлення інтерфейсу використано **HTML** і **CSS**. HTML відповідає за структуру сторінок, а CSS — за зовнішній вигляд елементів інтерфейсу. У системі було реалізовано простий і зрозумілий дизайн із використанням карток, таблиць, форм, кнопок, повідомлень про успішні дії та адаптивних елементів. Такий підхід дозволяє зробити систему зручною для користувача і водночас не перевантажувати її зайвими візуальними компонентами.

Для локальної розробки використано середовище **Laravel Herd**. Воно забезпечує запуск PHP, Laravel, Composer, Node.js і локального вебсервера, що значно спрощує налаштування середовища розробки. Завдяки Herd проєкт можна запускати локально через адресу типу corruption-reference-system.test, що зручно під час розробки та тестування.

Для керування залежностями PHP використовується **Composer**. Він дозволяє встановлювати та оновлювати пакети, необхідні для роботи Laravel-проєкту. Для роботи з фронтенд-залежностями використовується **npm**, хоча в межах даної системи основний інтерфейс реалізовано переважно за допомогою звичайних HTML і CSS без складних JavaScript-фреймворків.

У процесі розробки також використовувався редактор коду **Visual Studio Code**, який забезпечує зручну роботу з файлами проєкту, підсвічування синтаксису, пошук по файлах та можливість встановлення додаткових розширень для роботи з PHP і Laravel. Для керування версіями може використовуватися **Git**, що дозволяє зберігати історію змін у проєкті та контролювати процес розробки.

Обраний технологічний стек має кілька важливих переваг. По-перше, він дозволяє реалізувати повноцінний вебзастосунок із серверною логікою та базою даних. По-друге, Laravel забезпечує чітку структуру проєкту й підтримку MVC-архітектури. По-третє, MySQL дозволяє ефективно працювати зі структурованими даними. По-четверте, Blade, HTML і CSS дають можливість швидко створити зрозумілий інтерфейс користувача.

Також важливо, що обрані технології є поширеними та добре документованими. Це спрощує пошук інформації під час розробки, полегшує вирішення технічних проблем і дає можливість у майбутньому розширювати систему. Наприклад, до проєкту можна додати авторизацію, ролі користувачів, імпорт даних із файлів, API або складніший аналітичний модуль.

Таким чином, для розробки довідкової системи корупціонерів України було обрано технології PHP, Laravel, MySQL, Blade, HTML і CSS. Такий набір засобів є доцільним для створення навчального вебзастосунку з базою даних, оскільки забезпечує зручну розробку, структуровану організацію коду, підтримку основних операцій з даними та можливість подальшого розвитку системи.

2.3. Проєктування бази даних

Одним із ключових етапів розробки довідкової системи корупціонерів України є проєктування бази даних. Оскільки система призначена для зберігання структурованої інформації про осіб, регіони, статті законодавства та записи правопорушень, необхідно створити таку структуру даних, яка забезпечить логічність, цілісність і зручність подальшої обробки інформації.

Для реалізації системи було обрано реляційну модель бази даних [15]. У такій моделі дані зберігаються у вигляді таблиць, між якими встановлюються зв’язки [15; 20]. Реляційний підхід є доцільним для даної предметної області, оскільки інформація про правопорушення має чітку структуру та складається з кількох взаємопов’язаних сутностей. Наприклад, один запис правопорушення пов’язаний із конкретною особою, певним регіоном і відповідною статтею законодавства.

Основними сутностями бази даних розробленої системи є:

1. фізичні особи;
2. регіони;
3. статті законодавства;
4. записи правопорушень.

На основі цих сутностей було створено відповідні таблиці бази даних: people, regions, articles та offenses [12; 15]. Кожна таблиця відповідає окремій групі даних і має власне призначення в системі. Загальну структуру бази даних і взаємозв'язки між основними таблицями наведено на рисунку 2.2 [11; 12].

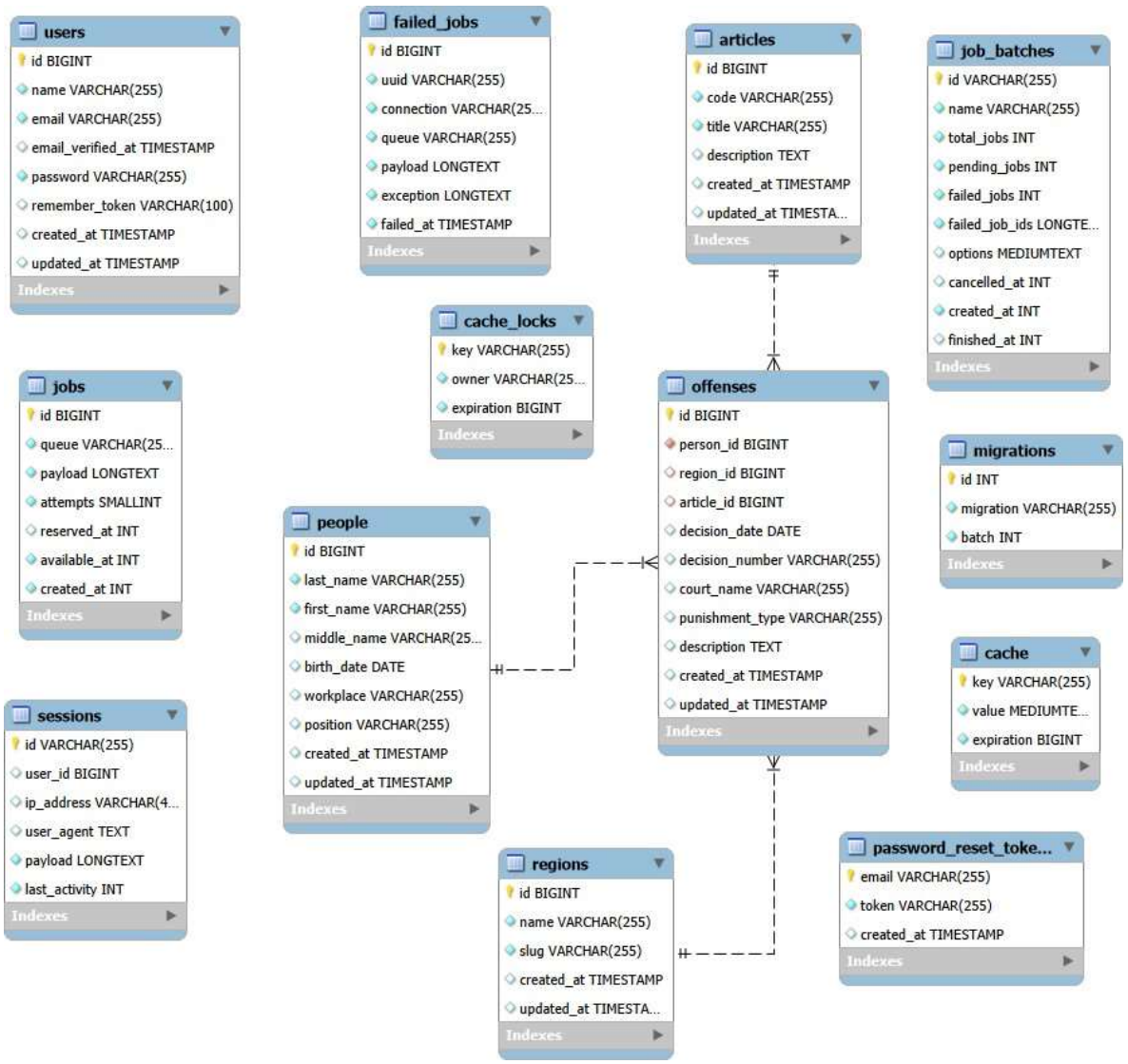

Рисунок 2.2 – ER-діаграма бази даних довідкової системи

Як видно з рисунка 2.2, центральною таблицею бази даних є таблиця offenses, яка містить записи правопорушень. Вона пов’язана з таблицями people, regions та articles за допомогою зовнішніх ключів. Така структура дозволяє зберігати дані у нормалізованому вигляді, уникати дублювання інформації та забезпечувати коректну роботу пошуку, фільтрації й статистичного модуля.

Таблиця people призначена для зберігання інформації про фізичних осіб. Вона містить поля для прізвища, імені, по батькові, дати народження, місця роботи та посади. Виділення інформації про осіб в окрему таблицю є доцільним, оскільки одна особа може бути пов’язана з одним або кількома записами правопорушень. Крім того, такий підхід дозволяє уникнути повторного введення однакової інформації про особу в кожному записі.
Основні поля таблиці people наведено в таблиці 2.1.

Поле	Тип даних	Призначення
id	BIGINT	Унікальний ідентифікатор особи
last_name	VARCHAR	Прізвище особи
first_name	VARCHAR	Ім’я особи
middle_name	VARCHAR, NULL	По батькові особи
birth_date	DATE, NULL	Дата народження
workplace	VARCHAR, NULL	Місце роботи
position	VARCHAR, NULL	Посада
created_at	TIMESTAMP	Дата створення запису
updated_at	TIMESTAMP	Дата оновлення запису

Таблиця 2.1 – Основні поля таблиці people

Таблиця regions використовується як довідник регіонів України. Вона містить назву регіону та службове поле slug, яке може використовуватися для ідентифікації регіону в системі. Такий довідник потрібен для фільтрації записів правопорушень за адміністративно-територіальною ознакою. Основні поля таблиці regions наведено в таблиці 2.2.

Поле	Тип даних	Призначення
id	BIGINT	Унікальний ідентифікатор регіону
name	VARCHAR	Назва регіону
slug	VARCHAR	Унікальна системна назва регіону
created_at	TIMESTAMP	Дата створення запису
updated_at	TIMESTAMP	Дата оновлення запису

Таблиця 2.2 – Основні поля таблиці regions

Таблиця articles призначена для зберігання інформації про статті законодавства, за якими класифікуються правопорушення. Вона містить код статті, її коротку назву та опис. Виділення статей в окремий довідник дає змогу уникнути дублювання однакової інформації та забезпечити зручну фільтрацію записів. Основні поля таблиці articles наведено в таблиці 2.3.

Поле	Тип даних	Призначення
id	BIGINT	Унікальний ідентифікатор статті
code	VARCHAR	Код статті законодавства
title	VARCHAR	Назва або короткий зміст статті
description	TEXT, NULL	Опис статті
created_at	TIMESTAMP	Дата створення запису
updated_at	TIMESTAMP	Дата оновлення запису

Таблиця 2.3 – Основні поля таблиці articles

Центральною таблицею бази даних є таблиця offenses, яка містить записи правопорушень [15]. Вона зберігає записи про правопорушення та поєднує між собою особу, регіон і статтю законодавства. Для цього в таблиці використовуються зовнішні ключі person_id, region_id та article_id. Крім цих полів, таблиця містить дату рішення, номер рішення, назву суду або органу, вид стягнення та опис правопорушення. Основні поля таблиці offenses наведено в таблиці 2.4.

Поле	Тип даних	Призначення
id	BIGINT	Унікальний ідентифікатор запису правопорушення
person_id	BIGINT	Посилання на особу з таблиці people
region_id	BIGINT, NULL	Посилання на регіон з таблиці regions
article_id	BIGINT, NULL	Посилання на статтю з таблиці articles
decision_date	DATE, NULL	Дата рішення
decision_number	VARCHAR, NULL	Номер рішення
court_name	VARCHAR, NULL	Назва суду або органу
punishment_type	VARCHAR, NULL	Вид стягнення або відповідальності
description	TEXT, NULL	Опис правопорушення
created_at	TIMESTAMP	Дата створення запису
updated_at	TIMESTAMP	Дата оновлення запису

Таблиця 2.4 – Основні поля таблиці offenses

Основні зв’язки між таблицями бази даних наведено в таблиці 2.6.

Зв’язок	Опис
people — offenses	одна особа може мати багато записів правопорушень
regions — offenses	один регіон може бути пов’язаний з багатьма записами
articles — offenses	одна стаття може бути пов’язана з багатьма записами

Таблиця 2.6 – Основні зв’язки між таблицями бази даних

Зв’язок між таблицями people і offenses реалізовано за принципом «один до багатьох». Це означає, що одна фізична особа може бути пов’язана з кількома записами правопорушень, але кожен запис правопорушення належить лише одній особі. У Laravel такий зв’язок реалізовано за допомогою методів hasMany() у моделі Person та belongsTo() у моделі Offense.

Зв’язок між таблицями regions і offenses також має тип «один до багатьох». Один регіон може бути вказаний у багатьох записах, але кожен запис правопорушення пов’язаний лише з одним регіоном. Аналогічний зв’язок використовується між таблицями articles і offenses.

Важливою особливістю бази даних є використання зовнішніх ключів. Вони забезпечують цілісність даних і не дозволяють створити запис правопорушення з посиланням на неіснуючу особу, регіон або статтю. Для поля person_id використано каскадне видалення. Це означає, що якщо особу буде видалено з таблиці people, усі пов’язані з нею записи правопорушень також будуть видалені. Такий підхід дозволяє уникнути «завислих» записів без пов’язаної особи.

Для полів region_id і article_id використовується можливість встановлення значення NULL у разі видалення відповідного регіону або статті. Це дозволяє зберегти сам запис правопорушення навіть тоді, коли довідкова інформація була змінена або видалена. Такий підхід є гнучкішим і дозволяє уникнути втрати основних даних.

Під час проєктування бази даних було враховано принцип нормалізації [12; 15]. Дані розподілено між окремими таблицями відповідно до їх змісту. Це дозволяє зменшити дублювання інформації, спростити оновлення даних і забезпечити логічну структуру системи. Наприклад, назва регіону або статті законодавства зберігається лише один раз у відповідному довіднику, а в записі правопорушення використовується лише посилання на цей запис.

Для створення таблиць у Laravel було використано міграції [19]. Міграції дають змогу описати структуру бази даних у вигляді PHP-коду, а потім створити таблиці за допомогою команди php artisan migrate. Це забезпечує контрольованість змін у структурі бази даних і дає можливість повторно створити її на іншому середовищі розробки.

Для заповнення бази початковими даними були створені seeders. Зокрема, було реалізовано заповнення таблиці регіонів України, таблиці статей законодавства, тестових осіб і тестових записів правопорушень. Це

дозволило швидко перевірити роботу основних функцій системи: списку записів, пошуку, фільтрації, детального перегляду та статистики.

Загальна структура бази даних дозволяє ефективно виконувати основні операції системи. Наприклад, при відкритті головної сторінки система отримує записи з таблиці offenses разом із пов'язаними даними з таблиць people, regions та articles. При використанні пошуку виконується перевірка полів особи, таких як прізвище, ім'я, по батькові, місце роботи та посада. При фільтрації використовуються поля region_id і article_id.

Таким чином, спроєктована база даних відповідає функціональним вимогам довідкової системи та забезпечує зберігання структурованої інформації про осіб, регіони, статті законодавства і правопорушення. Використання реляційної моделі, зовнішніх ключів, нормалізації та зв'язків між таблицями дозволяє забезпечити цілісність даних і зручність подальшої роботи з інформацією.

2.4. Опис основних модулів системи

Розроблена довідкова система корупціонерів України складається з кількох функціональних модулів, кожен із яких відповідає за окрему частину роботи вебзастосунку. Поділ системи на модулі дозволяє краще організувати програмний код, спростити підтримку проєкту та забезпечити зрозумілу логіку взаємодії користувача із системою [14; 16]. Основні варіанти використання довідкової системи з боку користувача наведено на рисунку 2.3 [11].

Рисунок 2.3 – UML-діаграма варіантів використання довідкової системи

Основними модулями розробленої системи є:

1. модуль перегляду записів;
2. модуль детального перегляду;
3. модуль додавання записів;

4. модуль редагування записів;
5. модуль видалення записів;
6. модуль керування фізичними особами;
7. модуль пошуку та фільтрації;
8. модуль статистики.

Модуль перегляду записів є основним модулем системи, оскільки саме з нього користувач починає роботу з вебзастосунком. На головній сторінці відображається таблиця із записами правопорушень. Для кожного запису виводиться ПІБ особи, місце роботи, посада, регіон, стаття законодавства, дата рішення, вид стягнення та кнопка переходу до детальної інформації. Такий формат подання даних є зручним, оскільки користувач одразу бачить основну інформацію про записи.

Для отримання даних модуль перегляду звертається до таблиці offenses і завантажує пов’язані дані з таблиць people, regions та articles. Це дозволяє відображати на головній сторінці не лише ідентифікатори пов’язаних записів, а й зрозумілу для користувача інформацію: ПІБ особи, назву регіону та код статті законодавства. У Laravel така взаємодія реалізована за допомогою зв’язків моделей і методу with().

Модуль детального перегляду призначений для відображення повної інформації про окремий запис правопорушення. На відміну від головної сторінки, де показується лише скорочена інформація, сторінка деталей містить розширені відомості: дату народження особи, місце роботи, посаду, регіон, статтю, назву статті, дату рішення, номер рішення, назву суду або органу, вид стягнення, опис правопорушення та опис статті законодавства. Це дозволяє користувачеві отримати повну інформацію без перевантаження головної таблиці.

Модуль додавання записів реалізує можливість створення нового запису правопорушення.

Для цього користувач відкриває форму, у якій обирає особу, регіон і статтю законодавства, а також вводить дату рішення, номер рішення, назву суду або органу, вид стягнення та опис правопорушення. Після надсилання форми дані проходять перевірку, а потім зберігаються в таблиці offenses.

Під час додавання запису важливим є використання валідації. Валідація дозволяє перевірити, чи правильно заповнені поля форми. Наприклад, поле вибору особи є обов'язковим, оскільки запис правопорушення не може існувати без пов'язаної особи. Поля регіону та статті є необов'язковими, але якщо вони заповнені, система перевіряє, чи існують відповідні записи в таблицях regions і articles.

Модуль редагування записів дозволяє змінювати вже наявні дані. Користувач може відкрити сторінку детального перегляду запису, натиснути кнопку «Редагувати» і перейти до форми редагування. У цій формі автоматично підставляються поточні значення полів. Після внесення змін користувач натискає кнопку збереження, система перевіряє дані та оновлює відповідний запис у базі даних.

Модуль редагування є важливим, оскільки під час роботи з інформаційними системами можуть виникати помилки у введених даних або потреба в їх уточненні. Наприклад, користувач може змінити дату рішення, номер рішення, вид стягнення або опис правопорушення. Завдяки цьому система залишається гнучкою і придатною для підтримки актуальності даних.

Модуль видалення записів забезпечує можливість вилучення непотрібних або помилкових записів правопорушень. Видалення виконується зі сторінки детального перегляду. Перед виконанням операції користувач отримує підтвердження, що зменшує ризик випадкового видалення. Після підтвердження запис видаляється з таблиці offenses, а користувач повертається на головну сторінку зі списком записів.

Модуль керування фізичними особами відповідає за додавання, перегляд і видалення осіб. У системі реалізовано окрему сторінку списку осіб, на якій відображаються ПІБ, дата народження, місце роботи, посада та кількість пов'язаних записів правопорушень. Також передбачена можливість додавання нової особи через окрему форму. Після додавання особа автоматично може бути використана при створенні нового запису правопорушення.

Видалення особи має особливість: якщо особа пов’язана із записами правопорушень, то разом із нею можуть бути видалені й відповідні записи. Тому на сторінці списку осіб передбачено попередження про наслідки такої дії. Це є важливим елементом зручності й безпеки користувацької взаємодії.

Модуль пошуку та фільтрації дозволяє швидко знаходити потрібну інформацію серед записів. Пошук реалізовано за такими полями: прізвище, ім’я, по батькові, місце роботи та посада особи. Це дозволяє користувачеві вводити частину ПІБ або назву організації й отримувати відповідні результати. Такий пошук є зручним, оскільки користувачеві не потрібно знати всі точні реквізити запису.

Фільтрація реалізована за двома основними критеріями: регіоном і статтею законодавства. Фільтр за регіоном дозволяє переглядати записи, пов’язані з конкретною областю або містом Києвом. Фільтр за статтею дозволяє вибрати записи, що належать до певної категорії правопорушень. Пошук і фільтри можуть використовуватися разом, що підвищує точність отриманих результатів.

Модуль статистики призначений для аналітичного узагальнення даних, які зберігаються в системі. На сторінці статистики відображається загальна кількість записів правопорушень, кількість фізичних осіб, кількість регіонів у довіднику та кількість статей законодавства. Крім того, система показує розподіл записів за регіонами, статтями та роками.

Статистичний модуль підвищує практичну цінність системи, оскільки дозволяє не лише зберігати записи, а й аналізувати їх. Наприклад, користувач може побачити, в яких регіонах найбільше записів, які статті найчастіше зустрічаються або за які роки внесено найбільше даних. У системі ці показники відображаються у вигляді інформаційних карток і горизонтальних індикаторів.

Крім основних модулів, у системі реалізовано повідомлення про успішне виконання дій. Після додавання, редагування або видалення запису користувач бачить відповідне повідомлення. Це покращує взаємодію із системою, оскільки користувач отримує підтвердження, що його дія була виконана успішно.

Усі модулі системи взаємодіють між собою через маршрути, контролери, моделі та представлення. Маршрути визначають, який запит користувача має бути оброблений. Контролери виконують логіку обробки запиту. Моделі забезпечують доступ до бази даних. Представлення формують сторінки, які бачить користувач у браузері. Такий підхід відповідає архітектурі MVC і забезпечує впорядковану структуру проєкту.

Таким чином, розроблена довідкова система має модульну структуру, що охоплює основні функції роботи з даними: перегляд, додавання, редагування, видалення, пошук, фільтрацію, керування особами та статистичний аналіз [14; 16]. Такий набір модулів дозволяє розглядати систему як повноцінний навчальний вебзастосунок із базою даних і практичною реалізацією основних принципів програмної інженерії.

2.5. Проєктування інтерфейсу користувача

Інтерфейс користувача є важливою складовою будь-якої інформаційної системи, оскільки саме через нього користувач взаємодіє з програмним продуктом [16; 17]. Навіть за наявності правильно спроєктованої бази даних і стабільної серверної логіки система буде малоефективною, якщо її інтерфейс є складним, незрозумілим або незручним. Тому під час розробки довідкової системи корупціонерів України особливу увагу було приділено простоті, логічності та зручності користувацького інтерфейсу.

Основною вимогою до інтерфейсу системи є зрозумілість для користувача [14; 16]. Оскільки система має довідковий характер, користувач повинен швидко знаходити потрібну інформацію, не витрачаючи багато часу на вивчення структури сайту. Для цього сторінки системи побудовано за простим принципом: у верхній частині розміщується заголовок сторінки, нижче — основні кнопки дій, а в центральній частині — таблиці, форми або блоки з інформацією.

Головна сторінка системи призначена для перегляду списку записів правопорушень. Вона містить назву системи, короткий опис призначення, блок із кнопками переходу до списку осіб, статистики та форми додавання нового запису. Основна частина сторінки представлена таблицею, у якій відображаються ключові дані: ПІБ особи, місце роботи, посада, регіон,

стаття законодавства, дата рішення та вид стягнення. Також для кожного запису передбачено кнопку «Детальніше», яка відкриває сторінку повної інформації.

Для підвищення зручності роботи з головною сторінкою реалізовано форму пошуку та фільтрації. Поле пошуку дозволяє вводити ПІБ, місце роботи або посаду. Поруч розташовані випадаючі списки для фільтрації за регіоном і статтею законодавства. Кнопки «Знайти» та «Скинути» дозволяють відповідно застосувати фільтри або повернутися до повного списку записів. Таке розташування елементів є зручним, оскільки всі інструменти пошуку знаходяться безпосередньо над таблицею.

Сторінка детального перегляду запису призначена для відображення повної інформації про правопорушення. На ній дані згруповано у кілька логічних блоків: інформація про особу, інформація про правопорушення, опис правопорушення та опис статті. Такий поділ дозволяє користувачеві швидко орієнтуватися у великому обсязі інформації. Для повернення до списку передбачено окрему кнопку, а також доступні кнопки редагування і видалення запису.

Форма додавання нового запису побудована таким чином, щоб користувач послідовно заповнював основні поля. Спочатку обирається фізична особа, регіон і стаття законодавства, після чого вводяться дата рішення, номер рішення, назва суду або органу, вид стягнення та опис правопорушення. Біля поля вибору особи передбачено кнопку «Додати нову особу», що дозволяє створити особу безпосередньо під час додавання правопорушення. Це підвищує зручність роботи, оскільки користувачеві не потрібно вручну переходити в інший розділ системи.

Форма редагування запису має структуру, подібну до форми додавання. Основною відмінністю є те, що поля автоматично заповнюються наявними даними обраного запису. Це дозволяє користувачеві швидко внести необхідні зміни без повторного введення всієї інформації. Після збереження змін система повертає користувача на сторінку детального перегляду та відображає повідомлення про успішне оновлення запису.

Сторінка додавання нової особи містить поля для введення прізвища, імені, по батькові, дати народження, місця роботи та посади.

Обов’язковими є поля прізвища та імені, інші поля можуть бути заповнені за потреби. Після збереження нова особа додається до бази даних і може бути використана при створенні нового запису правопорушення.

Сторінка списку осіб призначена для перегляду всіх фізичних осіб, які зберігаються в системі. Вона містить таблицю з ПІБ, датою народження, місцем роботи, посадою та кількістю пов’язаних записів правопорушень. Також на цій сторінці передбачено можливість видалення особи. Оскільки видалення особи може призвести до видалення пов’язаних із нею записів, на сторінці виводиться попередження для користувача.

Сторінка статистики має аналітичне призначення. Вона містить інформаційні картки із загальною кількістю записів, кількістю осіб, регіонів і статей законодавства. Крім того, на сторінці відображається розподіл записів за регіонами, статтями та роками. Для наочності використано горизонтальні індикатори, які показують частку кожної категорії від загальної кількості записів.

Для оформлення інтерфейсу використано єдину стилістику: світлий фон сторінки, темну шапку, білі картки з тінню, округлені кути, таблиці з чіткими межами, кнопки різних типів і кольорові повідомлення [22; 23]. Такий дизайн є простим, але водночас достатньо сучасним і зрозумілим. Він не перевантажує користувача зайвими елементами й акцентує увагу на основному змісті.

У системі використано кілька типів кнопок. Основні дії, такі як додавання або збереження, оформлено темними кнопками. Допоміжні дії, наприклад повернення, скидання фільтрів або перехід до іншої сторінки, оформлено світло-сірими кнопками. Для небезпечних дій, таких як видалення, використовується червона кнопка. Це дозволяє користувачеві швидко розрізняти типи дій та зменшує ризик випадкових помилок.

Особливу увагу приділено повідомленням про результат виконання дій. Після додавання, редагування або видалення запису система відображає зелене повідомлення про успішне виконання операції. Такі повідомлення покращують користувацький досвід, оскільки користувач отримує підтвердження, що його дія була виконана.

Для запобігання випадковому видаленню даних використано діалогове підтвердження. Перед видаленням запису або особи система запитує користувача, чи дійсно він хоче виконати цю дію.

Це є важливим елементом безпеки інтерфейсу, оскільки операція видалення є незворотною.

Інтерфейс системи також має елементи адаптивності [22; 23]. Для менших екранів передбачено зміну розташування елементів форм і кнопок, а таблиці можуть прокручуватися горизонтально. Це дозволяє користуватися системою не лише на великих екранах комп'ютерів, а й на пристроях із меншою шириною екрана.

Під час проєктування інтерфейсу було враховано принцип послідовності. Усі сторінки мають схожу структуру: заголовок, блок дій, основний вміст і кнопки навігації. Завдяки цьому користувач швидко звикає до логіки системи та може легко переходити між сторінками.

Таким чином, інтерфейс довідкової системи корупціонерів України було спроєктовано з урахуванням простоти, зрозумілості та зручності використання. Основні елементи інтерфейсу відповідають функціональному призначенню системи та забезпечують ефективну роботу з даними. Реалізовані сторінки дозволяють переглядати записи, виконувати пошук і фільтрацію, додавати та редагувати інформацію, керувати особами й аналізувати статистичні показники.

2.6. Висновки до розділу 2

У другому розділі кваліфікаційної бакалаврської роботи було виконано проєктування довідкової системи корупціонерів України. На основі результатів аналізу предметної області було визначено загальну архітектуру системи, обґрунтовано вибір технологій розробки, спроєктовано структуру бази даних, описано основні функціональні модулі та розглянуто підхід до створення інтерфейсу користувача.

Для реалізації програмного продукту було обрано веборієнтовану клієнт-серверну архітектуру. Такий підхід є доцільним для довідкової системи, оскільки дає змогу працювати із застосунком через браузер, централізовано зберігати дані та забезпечувати зручний доступ до функціоналу системи. Для організації програмної логіки було використано архітектурний шаблон MVC, який дозволяє розділити систему на моделі, представлення та контролери.

У межах розділу було обґрунтовано вибір технологій розробки. Для серверної частини обрано мову програмування PHP і фреймворк Laravel,

який забезпечує зручну маршрутизацію, роботу з контролерами, моделями, міграціями та шаблонами. Для зберігання даних використано MySQL, що є доцільним вибором для системи з реляційною структурою даних. Інтерфейс користувача реалізовано за допомогою Blade-шаблонів, HTML і CSS.

Особливу увагу було приділено проєктуванню бази даних. Було визначено основні сутності системи: фізичні особи, регіони, статті законодавства та записи правопорушень. На основі цих сутностей було спроєктовано таблиці people, regions, articles та offenses. Центральною таблицею є offenses, яка пов’язує між собою особу, регіон і статтю законодавства. Така структура забезпечує логічну організацію даних, зменшує дублювання інформації та підтримує цілісність зв’язків між сутностями.

Також було визначено основні модулі системи: модуль перегляду записів, модуль детального перегляду, модуль додавання, редагування та видалення записів, модуль керування фізичними особами, модуль пошуку й фільтрації, а також модуль статистики. Кожен із цих модулів виконує окрему функцію, а разом вони формують цілісну інформаційно-довідкову систему.

Під час проєктування інтерфейсу користувача було враховано принципи простоти, зрозумілості та послідовності. Основні сторінки системи мають єдину стилістику, містять зрозумілі кнопки дій, таблиці, форми введення, повідомлення про успішне виконання операцій і підтвердження перед видаленням даних. Це дозволяє зробити роботу із системою зручною навіть для користувача без спеціальної технічної підготовки.

Отже, у другому розділі було сформовано технічну основу майбутнього програмного продукту. Результати проєктування дали змогу перейти до практичної реалізації системи, створення бази даних, моделей, контролерів, маршрутів, сторінок інтерфейсу та тестування функціональності, що буде розглянуто у третьому розділі.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ДОВІДКОВОЇ СИСТЕМИ

3.1. Налаштування середовища розробки

Практична реалізація довідкової системи корупціонерів України розпочалася з налаштування середовища розробки. Для створення вебзастосунку було обрано набір інструментів, які забезпечують роботу серверної частини, бази даних, керування залежностями та редагування програмного коду [14; 16].

Основним середовищем для локального запуску Laravel-проєкту було використано **Laravel Herd**. Це середовище дозволяє швидко налаштувати роботу PHP, Composer, Laravel, Node.js і локального вебсервера без необхідності складного ручного конфігурування. Завдяки цьому проєкт можна запускати локально через домен виду:
http://corruption-reference-system.test

Використання Laravel Herd спростило процес розробки, оскільки більшість необхідних інструментів були доступні одразу після встановлення. Після налаштування середовища було перевірено роботу основних команд:
php -v
composer -V
laravel -V
node -v
npm -v
git –v
Ці команди дозволили переконатися, що PHP, Composer, Laravel Installer, Node.js, npm і Git встановлені та доступні в системі.

Для створення нового Laravel-проєкту було використано команду [19]:
laravel new corruption-reference-system

Під час створення проєкту було обрано варіант без starter kit, оскільки система розроблялася вручну з власною структурою сторінок, маршрутів, контролерів і шаблонів. Це дало можливість краще контролювати процес розробки та реалізувати саме той функціонал, який потрібен для кваліфікаційної роботи.

Після створення проєкту було виконано встановлення необхідних залежностей і первинну збірку ресурсів. Для цього використовувалися команди:

npm install
npm run build

Для роботи з базою даних було використано **MySQL Server** і програму **MySQL Workbench** [20]. MySQL Workbench забезпечує зручний графічний інтерфейс для створення баз даних, перегляду таблиць і перевірки збережених записів. У межах розробки було створено базу даних з назвою:
corruption_reference_system

Після створення бази даних було виконано налаштування підключення Laravel до MySQL [19; 20]. Для цього у файлі .env було вказано параметри підключення:

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=corruption_reference_system
DB_USERNAME=root
DB_PASSWORD=********

Пароль у файлі .env відповідає паролю користувача root, який використовується для підключення до MySQL. Після зміни налаштувань було очищено кеш конфігурації Laravel за допомогою команди:

php artisan config:clear

Для перевірки правильності підключення до бази даних було виконано команду міграції:

php artisan migrate

У результаті Laravel створив стандартні службові таблиці, після чого стало можливим створення власних таблиць для довідкової системи.

Для редагування програмного коду використовувався редактор **Visual Studio Code**. Він дозволяє зручно працювати зі структурою Laravel-проєкту, відкривати файли контролерів, моделей, міграцій, маршрутів і Blade-шаблонів. Основні файли, які редагувалися під час розробки, розміщені в таких каталогах:

app/Models
app/Http/Controllers
database/migrations
database/seeders
resources/views
routes/web.php

Після налаштування середовища вебзастосунок було відкрито у браузері. У разі роботи через Laravel Herd сайт доступний за адресою:
http://corruption-reference-system.test

Також для запуску проєкту можна використовувати вбудований сервер Laravel:
php artisan serve

У такому випадку застосунок відкривається за адресою:
http://127.0.0.1:8000

Таким чином, на етапі налаштування середовища було встановлено та перевірено всі необхідні інструменти для розробки: Laravel Herd, PHP, Composer, Laravel, Node.js, npm, Git, MySQL Server, MySQL Workbench і Visual Studio Code. Після цього було створено Laravel-проєкт, налаштовано підключення до бази даних і перевірено працездатність початкової конфігурації вебзастосунку.

3.2. Реалізація бази даних і моделей

Після налаштування середовища розробки наступним етапом стала реалізація бази даних і моделей вебзастосунку. База даних є основою довідкової системи, оскільки саме в ній зберігаються всі відомості про фізичних осіб, регіони, статті законодавства та записи правопорушень. Для роботи з базою даних у проєкті використано MySQL, а для створення таблиць — механізм міграцій Laravel [19; 20].

Спочатку було створено базу даних з назвою:
corruption_reference_system

Після цього у файлі .env було налаштовано підключення Laravel до цієї бази даних. Основними параметрами підключення є тип бази даних, адреса сервера, порт, назва бази, ім’я користувача та пароль. Після збереження налаштувань було виконано команду очищення конфігурації, щоб Laravel застосував нові параметри підключення.

Для створення таблиць використовувалися міграції. Міграція в Laravel — це файл, у якому описується структура таблиці бази даних [19]. Завдяки міграціям можна створювати, змінювати та відтворювати структуру бази даних за допомогою консольних команд. Це зручно, оскільки структура бази зберігається в коді проєкту.

У межах розробки було створено такі основні міграції:
create_regions_table
create_articles_table
create_people_table
create_offenses_table

Таблиця regions призначена для зберігання довідника регіонів України. Вона містить назву регіону та унікальне службове поле slug. Така таблиця використовується для фільтрації записів правопорушень за регіоном.

Структура таблиці regions була реалізована так:

```
Schema::create('regions', function (Blueprint $table) {
    $table->id();
    $table->string('name');
    $table->string('slug')->unique();
    $table->timestamps();
});
```

Таблиця articles використовується для зберігання статей законодавства. У ній міститься код статті, її назва та опис. Ця таблиця є довідником, який використовується під час створення або фільтрації записів правопорушень.
Структура таблиці articles:

```
Schema::create('articles', function (Blueprint $table) {
    $table->id();
    $table->string('code')->unique();
    $table->string('title');
    $table->text('description')->nullable();
    $table->timestamps();
});
```

Таблиця people призначена для зберігання інформації про фізичних осіб. Вона містить прізвище, ім’я, по батькові, дату народження, місце роботи та посаду. У Laravel модель має назву Person, але таблиця в базі даних називається people, що відповідає стандартному підходу фреймворку до множини назв.
Структура таблиці people:

```
Schema::create('people', function (Blueprint $table) {
    $table->id();
    $table->string('last_name');
    $table->string('first_name');
    $table->string('middle_name')->nullable();
    $table->date('birth_date')->nullable();
    $table->string('workplace')->nullable();
    $table->string('position')->nullable();
    $table->timestamps();
});
```

Центральною таблицею системи є offenses. Вона зберігає записи правопорушень і містить зовнішні ключі для зв’язку з таблицями people, regions та articles. Крім цього, таблиця містить дату рішення, номер рішення, назву суду або органу, вид стягнення та опис правопорушення.
Структура таблиці offenses:

```
Schema::create('offenses', function (Blueprint $table) {
   $table->id();
   $table->foreignId('person_id')
      ->constrained('people')
      ->cascadeOnDelete();
   $table->foreignId('region_id')
      ->nullable()
      ->constrained('regions')
      ->nullOnDelete();
   $table->foreignId('article_id')
      ->nullable()
      ->constrained('articles')
      ->nullOnDelete();
   $table->date('decision_date')->nullable();
   $table->string('decision_number')->nullable();
   $table->string('court_name')->nullable();
   $table->string('punishment_type')->nullable();
   $table->text('description')->nullable();
   $table->timestamps();
});
```

У цій таблиці поле person_id є обов’язковим, оскільки кожен запис правопорушення повинен бути пов’язаний з конкретною особою. Для цього поля використано каскадне видалення. Це означає, що при видаленні особи автоматично видаляються всі пов’язані з нею записи правопорушень. Для полів region_id і article_id дозволено значення NULL, що дає змогу зберегти запис навіть у разі видалення відповідного регіону або статті.

Після створення міграцій було виконано команду:

php artisan migrate

У результаті всі таблиці були створені в базі даних MySQL.

Для роботи з таблицями були створені відповідні моделі Laravel:

Region

Article

Person

Offense

Модель Region відповідає таблиці regions. У ній визначено поля, які можна масово заповнювати, а також зв’язок із записами правопорушень.

```
class Region extends Model
{
   protected $fillable = [
      'name',
      'slug',
   ];
   public function offenses()
   {
      return $this->hasMany(Offense::class);
   }
}
```

Модель Article відповідає таблиці articles і містить зв’язок із записами правопорушень.

```
class Article extends Model
{
   protected $fillable = [
      'code',
      'title',
      'description',
   ];
   public function offenses()
   {
      return $this->hasMany(Offense::class);
   }
}
```

Модель Person відповідає таблиці people. Оскільки назва моделі в однині, а таблиця має назву people, у моделі явно вказано назву таблиці. Також визначено зв’язок із записами правопорушень.

```
class Person extends Model
{
   protected $table = 'people';
   protected $fillable = [
      'last_name',
      'first_name',
      'middle_name',
      'birth_date',
      'workplace',
      'position',
   ];
   public function offenses()
   {
      return $this->hasMany(Offense::class);
   }
}
```

Модель Offense є центральною моделлю системи [19]. Вона відповідає таблиці offenses і містить зв’язки з моделями Person, Region та Article.

```
class Offense extends Model
{
   protected $fillable = [
      'person_id',
      'region_id',
      'article_id',
      'decision_date',
      'decision_number',
      'court_name',
      'punishment_type',
      'description',
   ];
   public function person()
```

```
    {
        return $this->belongsTo(Person::class);
    }
    public function region()
    {
        return $this->belongsTo(Region::class);
    }
    public function article()
    {
        return $this->belongsTo(Article::class);
    }
}
```

Завдяки таким зв'язкам система може легко отримувати пов'язані дані [19]. Наприклад, для запису правопорушення можна отримати інформацію про особу, регіон і статтю законодавства без написання складних SQL-запитів. Це значно спрощує реалізацію списку записів, сторінки детального перегляду та статистики.

Для початкового наповнення бази даних було створено seeders. Вони дозволяють автоматично додавати тестові дані до таблиць. У проєкті було створено такі seeders:

RegionSeeder
ArticleSeeder
PersonSeeder
OffenseSeeder

RegionSeeder заповнює таблицю regions областями України та містом Києвом. ArticleSeeder додає приклади статей законодавства, пов'язаних із корупційними та адміністративними правопорушеннями. PersonSeeder створює тестових фізичних осіб, а OffenseSeeder додає тестові записи правопорушень, пов'язуючи осіб із регіонами та статтями.

Для запуску seeders використовувалися команди:

php artisan db:seed --class=RegionSeeder
php artisan db:seed --class=ArticleSeeder
php artisan db:seed --class=PersonSeeder
php artisan db:seed --class=OffenseSeeder

Після виконання цих команд база даних була наповнена тестовими даними, що дозволило перевірити роботу основного функціоналу системи.

Реалізована база даних забезпечує зберігання всіх основних даних, необхідних для роботи довідкової системи. Завдяки використанню реляційних зв'язків система може коректно відображати повну інформацію про кожний запис, виконувати пошук і фільтрацію, формувати статистику та підтримувати операції додавання, редагування і видалення [15; 20].

Окрему роль у реалізації бази даних відіграють зовнішні ключі, які забезпечують зв'язок між основними таблицями системи. Завдяки їм кожен запис правопорушення пов'язується з конкретною особою, регіоном та статтею законодавства, що дозволяє уникнути неузгодженості даних.

Також важливим є використання моделей Eloquent, які спрощують роботу з базою даних на рівні програмного коду. Через моделі система отримує доступ до пов'язаних записів без необхідності постійного написання складних SQL-запитів.

Крім того, використання seeders дозволило швидко наповнити базу початковими тестовими даними для перевірки роботи системи. Це спростило тестування основних функцій вебзастосунку, зокрема перегляду записів, пошуку, фільтрації та формування статистичних показників.

Таким чином, реалізація бази даних і моделей стала основою для подальшої розробки функціональних модулів системи. Використання міграцій, моделей Eloquent, зв'язків між таблицями та seeders дозволило створити структуровану, зрозумілу й придатну для розширення основу вебзастосунку.

3.3. Реалізація основного функціоналу системи

Після створення бази даних і моделей було реалізовано основний функціонал довідкової системи корупціонерів України. До основного функціоналу належать перегляд списку записів, детальний перегляд окремого запису, додавання, редагування та видалення записів правопорушень, а також додавання і видалення фізичних осіб. Реалізація цих можливостей забезпечує роботу системи як повноцінного вебзастосунку з базовими CRUD-операціями [14; 16].

Для організації взаємодії користувача із системою було створено маршрути у файлі routes/web.php [19]. Маршрути визначають, які URL-адреси доступні в системі та які методи контролерів відповідають за їх обробку. Основні маршрути системи пов’язані з контролерами OffenseController та PersonController [18; 19].

Основні маршрути для роботи із записами правопорушень мають такий вигляд:

```
Route::get('/', [OffenseController::class, 'index'])->name('offenses.index');
Route::get('/offenses/create', [OffenseController::class, 'create'])-
>name('offenses.create');
Route::post('/offenses', [OffenseController::class, 'store'])-
>name('offenses.store');
Route::get('/offenses/{offense}', [OffenseController::class, 'show'])-
>name('offenses.show');
Route::get('/offenses/{offense}/edit', [OffenseController::class, 'edit'])-
>name('offenses.edit');
Route::put('/offenses/{offense}', [OffenseController::class, 'update'])-
>name('offenses.update');
Route::delete('/offenses/{offense}', [OffenseController::class, 'destroy'])-
>name('offenses.destroy');
```

Головна сторінка системи реалізована через метод index() контролера OffenseController. Цей метод отримує записи правопорушень із бази даних разом із пов’язаними даними про особу, регіон і статтю законодавства. Для цього використовується метод with(), який дозволяє завантажити зв’язані моделі [19].

```
$offenses = Offense::with(['person', 'region', 'article'])
    ->latest()
    ->paginate(10);
```

Зовнішній вигляд головної сторінки зі списком записів наведено на рисунку 3.1.

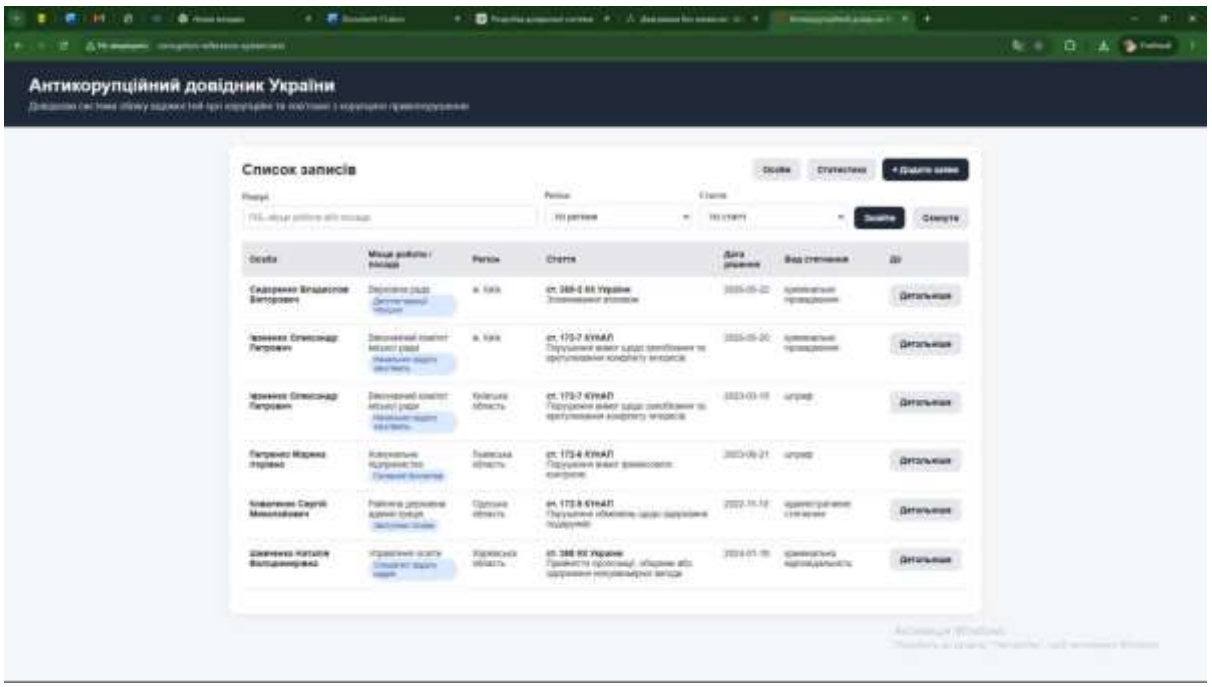

Рисунок 3.1 – Головна сторінка довідкової системи

Отримані дані передаються у Blade-шаблон offenses.index, де відображаються у вигляді таблиці. У таблиці користувач бачить ПІБ особи, місце роботи, посаду, регіон, статтю, дату рішення, вид стягнення та кнопку переходу до детальної інформації. Для зручності список записів розділено на сторінки за допомогою пагінації.

Детальний перегляд запису реалізовано через метод show(). Він отримує конкретний запис правопорушення та завантажує пов’язані дані:

```
public function show(Offense $offense)
{
    $offense->load(['person', 'region', 'article']);
    return view('offenses.show', compact('offense'));
}
```

Сторінка детального перегляду містить повну інформацію про запис. Дані згруповано у логічні блоки: інформація про особу, інформація про правопорушення, опис правопорушення та опис статті. Така структура дозволяє користувачу зручно переглядати розширені дані без

перевантаження головної таблиці. Приклад сторінки детального перегляду запису правопорушення наведено на рисунку 3.2.

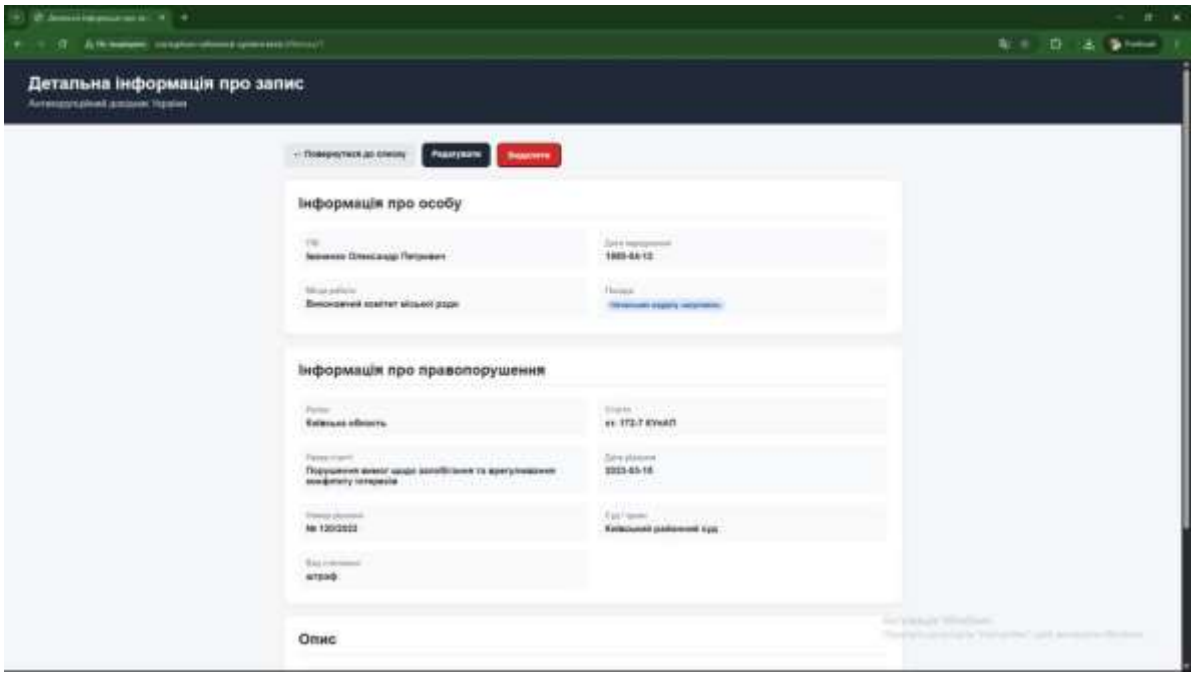

Рисунок 3.2 – Сторінка детального перегляду запису правопорушення

Функція додавання нового запису реалізована через два методи: create() і store(). Метод create() відкриває форму додавання та передає до неї списки осіб, регіонів і статей законодавства:

```
public function create()
{
   $people = Person::orderBy('last_name')->get();
   $regions = Region::orderBy('name')->get();
   $articles = Article::orderBy('code')->get();
   return view('offenses.create', compact('people', 'regions', 'articles'));
}
```

Форму додавання нового запису правопорушення наведено на рисунку 3.3.

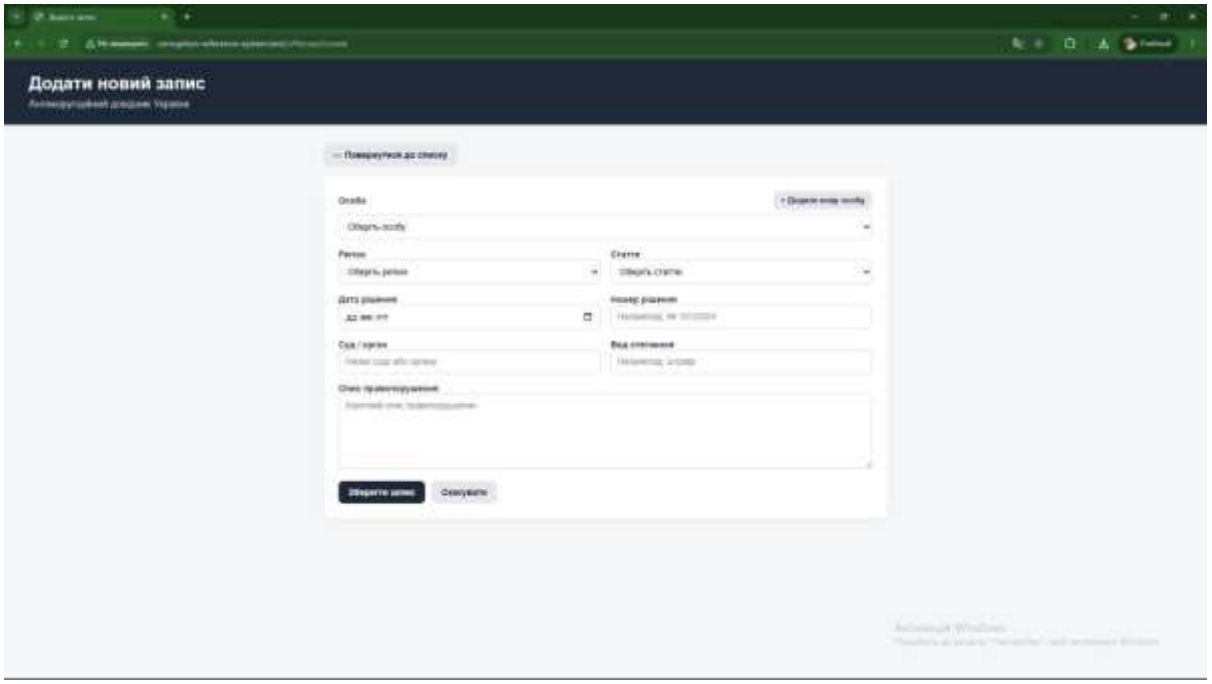

Рисунок 3.3 – Форма додавання нового запису правопорушення

У формі користувач обирає особу, регіон і статтю, а також заповнює дату рішення, номер рішення, назву суду або органу, вид стягнення й опис правопорушення. Після надсилання форми дані обробляються методом store().

Перед збереженням дані проходять валідацію:

```
$validated = $request->validate([
    'person_id' => ['required', 'exists:people,id'],
    'region_id' => ['nullable', 'exists:regions,id'],
    'article_id' => ['nullable', 'exists:articles,id'],
    'decision_date' => ['nullable', 'date'],
    'decision_number' => ['nullable', 'string', 'max:255'],
    'court_name' => ['nullable', 'string', 'max:255'],
    'punishment_type' => ['nullable', 'string', 'max:255'],
    'description' => ['nullable', 'string'],
]);
```

Валідація необхідна для запобігання збереженню некоректних даних. Наприклад, поле person_id є обов’язковим, оскільки запис правопорушення повинен бути пов’язаний з конкретною особою. Якщо користувач заповнив поля регіону або статті, система перевіряє, чи існують такі записи в базі даних.

Після успішної перевірки створюється новий запис:
$offense = Offense::create($validated);

Після збереження користувача перенаправляє на сторінку детального перегляду нового запису, де відображається повідомлення про успішне додавання.

Редагування запису реалізовано через методи edit() та update(). Метод edit() відкриває форму редагування і передає до неї поточний запис, а також списки осіб, регіонів і статей:

```
public function edit(Offense $offense)
{
   $people = Person::orderBy('last_name')->get();
   $regions = Region::orderBy('name')->get();
   $articles = Article::orderBy('code')->get();
   return view('offenses.edit', compact('offense', 'people', 'regions',
'articles'));
}
```

Форму редагування запису правопорушення наведено на рисунку 3.4.

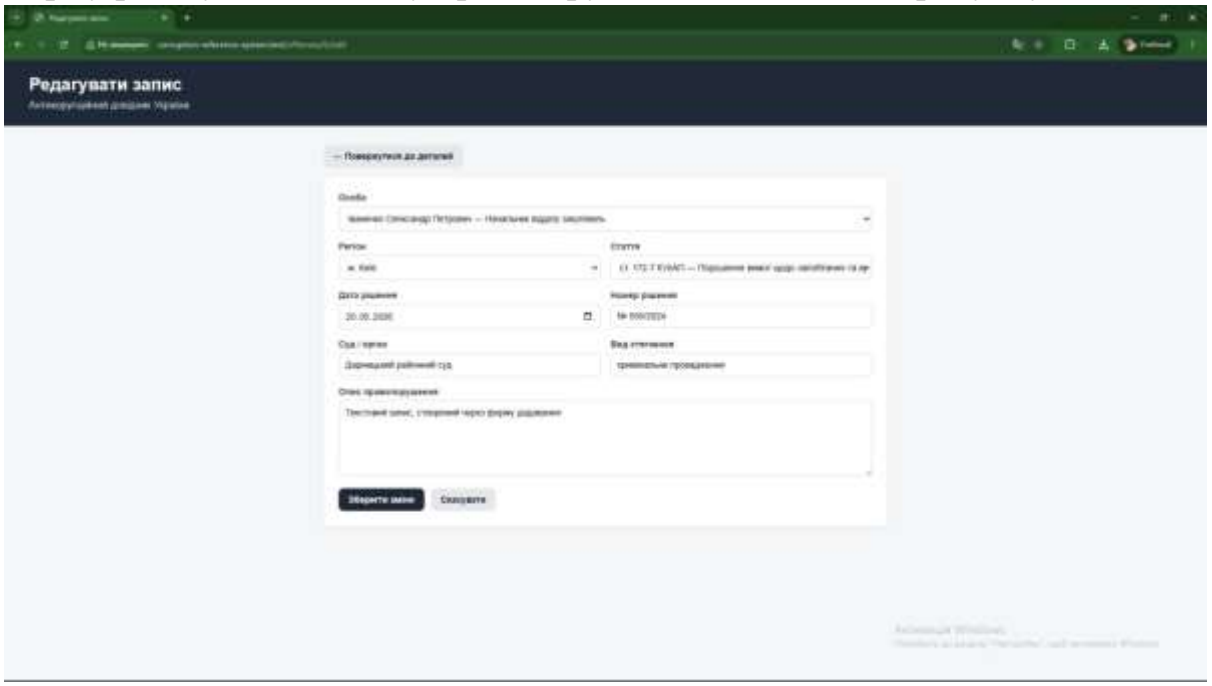

Рисунок 3.4 – Форма редагування запису правопорушення

У формі редагування поля автоматично заповнюються поточними значеннями. Це дозволяє користувачеві швидко змінити лише необхідні дані. Після надсилання форми метод update() виконує валідацію і оновлює запис у базі даних:

```
$offense->update($validated);
```

Після оновлення користувач повертається на сторінку детального перегляду запису та бачить повідомлення про успішне збереження змін.

Видалення запису реалізовано через метод destroy():

```
public function destroy(Offense $offense)
{
   $offense->delete();
   return redirect()
      ->route('offenses.index')
      ->with('success', 'Запис успішно видалено.');
}
```

На сторінці детального перегляду розміщено кнопку «Видалити». Перед видаленням користувач отримує повідомлення з підтвердженням дії. Це зменшує ризик випадкового видалення даних. Після підтвердження запис видаляється, а користувач повертається на головну сторінку.

Окрім керування записами правопорушень, у системі реалізовано функціонал роботи з фізичними особами. Для цього було створено PersonController. Він відповідає за відображення списку осіб, створення нової особи та видалення особи.

Реалізація окремого контролера для роботи з фізичними особами дозволила відокремити логіку керування особами від логіки роботи із записами правопорушень [18; 19]. Такий підхід відповідає принципам MVC-архітектури та спрощує подальшу підтримку програмного коду.

Важливим елементом цього функціоналу є можливість додавання нової особи безпосередньо під час роботи із системою. Завдяки цьому користувач може спочатку створити запис про фізичну особу, а потім використати її під час додавання нового запису правопорушення. Форму додавання нової фізичної особи наведено на рисунку 3.5.

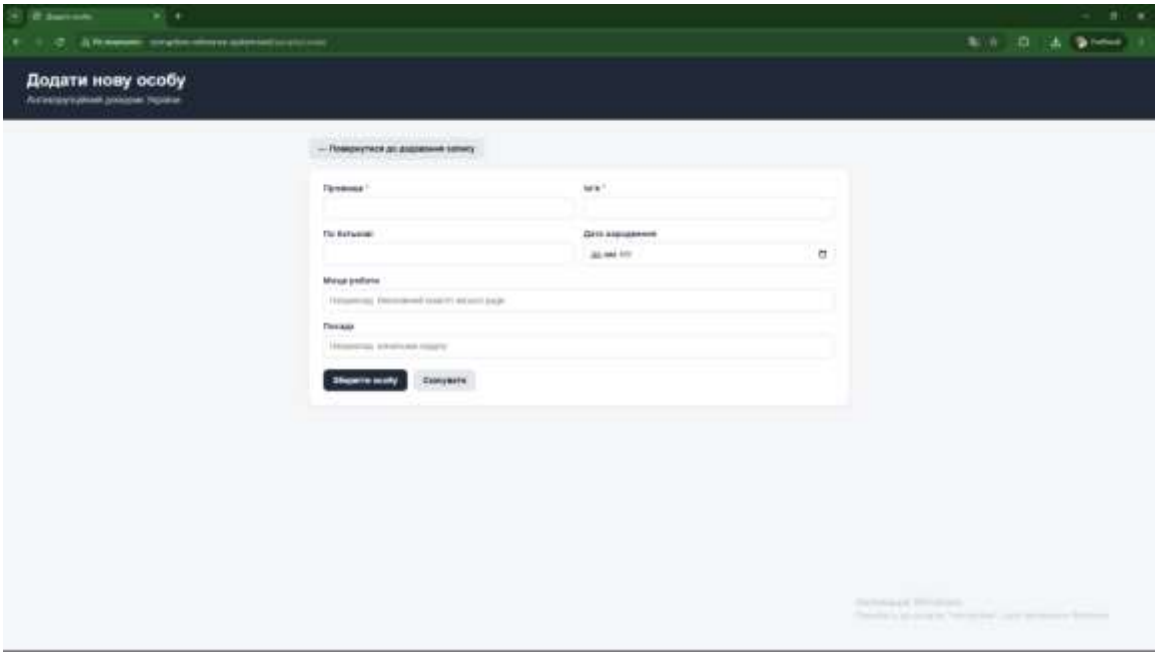

Рисунок 3.5 – Форма додавання нової фізичної особи

Сторінка списку осіб доступна за маршрутом:

Route::get('/people', [PersonController::class, 'index'])->name('people.index');

У методі `index()` виконується отримання осіб разом із кількістю пов'язаних записів правопорушень:

```
$people = Person::withCount('offenses')
    ->orderBy('last_name')
    ->paginate(10);
```

Застосування методу withCount('offenses') дозволяє не лише отримати список фізичних осіб, а й одразу підрахувати кількість записів правопорушень, пов'язаних із кожною особою [19]. Це зручно для користувача, оскільки на сторінці списку осіб відображається не тільки персональна інформація, а й кількість відповідних записів у системі.

Сортування за полем last_name забезпечує впорядковане відображення осіб за прізвищем. Такий підхід полегшує пошук потрібної особи у списку та робить сторінку більш зручною для перегляду.

Пагінація за допомогою методу paginate(10) дозволяє розділити великий список осіб на окремі сторінки. Це запобігає перевантаженню інтерфейсу та покращує швидкість роботи сторінки при збільшенні кількості записів у базі даних.

Отримані дані передаються до Blade-шаблону сторінки списку осіб, де відображаються у вигляді таблиці. У таблиці користувач бачить ПІБ особи, дату народження, місце роботи, посаду, кількість пов’язаних записів і доступні дії для керування записами.

Сторінку списку фізичних осіб наведено на рисунку 3.6.

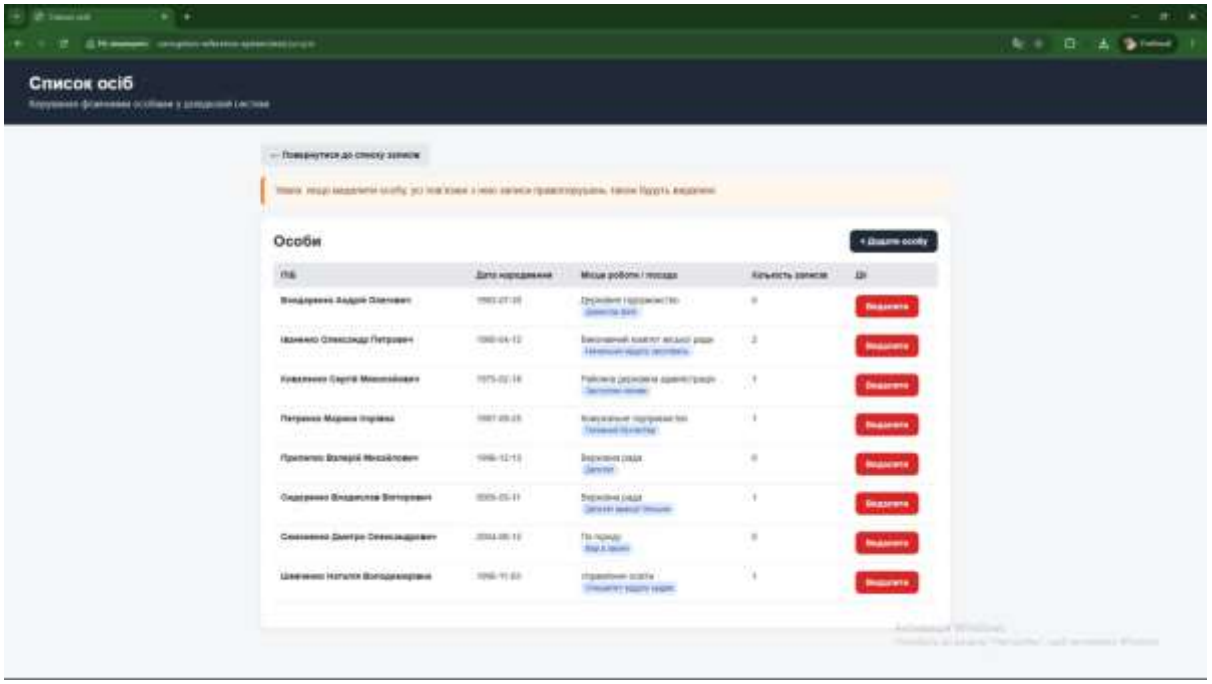

Рисунок 3.6 – Сторінка списку фізичних осіб

Це дозволяє на сторінці списку осіб відображати не лише персональні дані, а й кількість записів, пов’язаних із кожною особою.

Додавання нової особи реалізовано через форму, яка містить поля для прізвища, імені, по батькові, дати народження, місця роботи та посади. Після збереження нової особи користувача автоматично повертає на сторінку додавання нового запису правопорушення. Нова особа одразу може бути вибрана в полі «Особа», що робить процес створення запису зручнішим.

Видалення особи реалізовано через метод destroy() у PersonController. Оскільки між таблицями people і offenses встановлено каскадне видалення, при видаленні особи також видаляються всі пов’язані з нею записи правопорушень. Тому на сторінці списку осіб передбачено попередження про наслідки цієї операції.

Для покращення взаємодії з користувачем у системі реалізовано повідомлення про успішне виконання дій.

Наприклад, після додавання, редагування або видалення запису користувач бачить зелене повідомлення з відповідним текстом. Це дозволяє користувачеві переконатися, що дія була виконана успішно.

Таким чином, у системі реалізовано основний функціонал, необхідний для роботи довідкової інформаційної системи. Користувач може переглядати записи, відкривати детальну інформацію, додавати нові записи, редагувати та видаляти їх, а також працювати зі списком фізичних осіб. Реалізований функціонал відповідає вимогам, сформульованим на етапі постановки задачі, і забезпечує базову роботу програмного продукту.

3.4. Реалізація пошуку, фільтрації та статистики

Окрім базових CRUD-операцій, у довідковій системі корупціонерів України було реалізовано додаткові функції, які підвищують зручність роботи з даними. До таких функцій належать пошук записів, фільтрація за регіоном і статтею законодавства, а також формування статистичної інформації [15; 16]. Ці можливості є важливими для довідкової системи, оскільки дозволяють швидко знаходити потрібні записи та отримувати узагальнене уявлення про дані, що зберігаються в базі.

Пошук реалізовано на головній сторінці системи. Приклад роботи пошуку та фільтрації записів наведено на рисунку 3.7. Користувач може ввести у відповідне поле прізвище, ім’я, по батькові, місце роботи або посаду особи. Після натискання кнопки «Знайти» система виконує запит до бази даних і відображає тільки ті записи, які відповідають пошуковому запиту [15; 20].

Реалізація пошуку саме на головній сторінці є зручною для користувача, оскільки не потребує переходу до окремого розділу системи. Користувач одразу бачить поле пошуку, може швидко ввести потрібний запит і отримати відфільтрований список записів у тій самій таблиці.

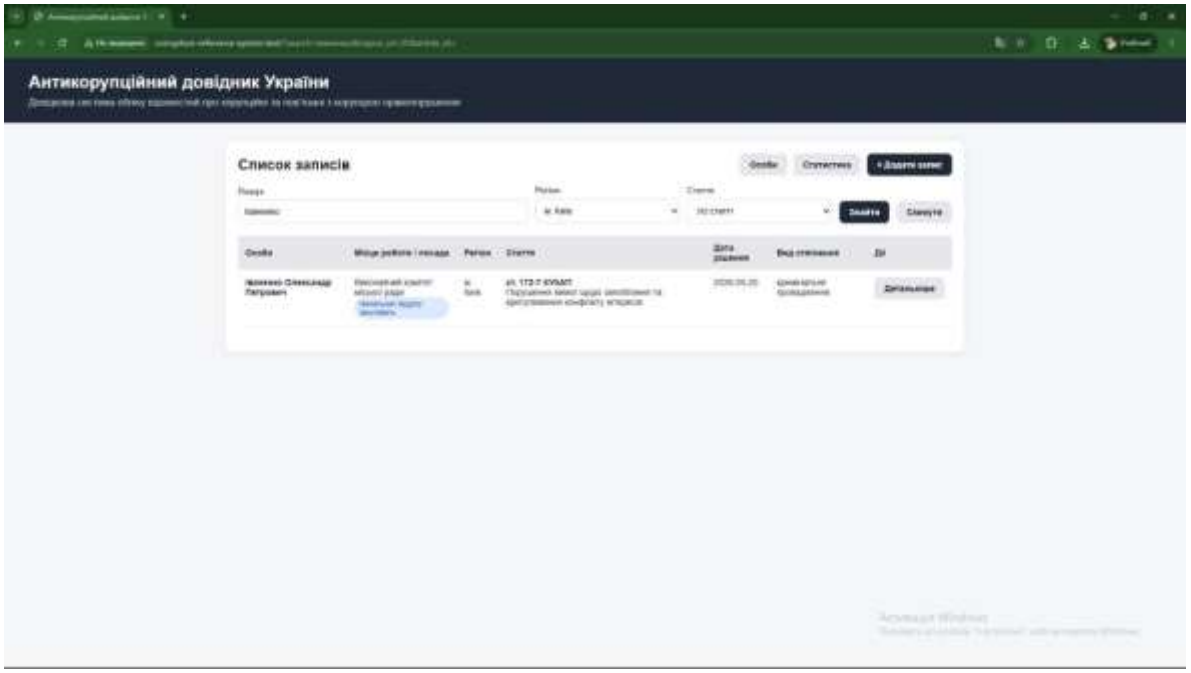

Рисунок 3.7 – Приклад роботи пошуку та фільтрації записів

У контролері OffenseController пошук реалізовано в методі index(). Спочатку формується базовий запит до таблиці offenses із завантаженням пов’язаних моделей:

$query = Offense::with(['person', 'region', 'article']);

Далі перевіряється, чи було введено пошуковий запит. Якщо поле search заповнене, система виконує пошук у пов’язаній таблиці people:

```
if ($request->filled('search')) {
    $search = $request->search;

    $query->whereHas('person', function ($q) use ($search) {
        $q->where('last_name', 'like', "%{$search}%")
          ->orWhere('first_name', 'like', "%{$search}%")
          ->orWhere('middle_name', 'like', "%{$search}%")
          ->orWhere('workplace', 'like', "%{$search}%")
          ->orWhere('position', 'like', "%{$search}%");
    });
}
```

У цьому фрагменті використано метод whereHas(), який дозволяє виконувати пошук не лише в основній таблиці offenses, а й у пов’язаній таблиці people [19]. Це є важливим, оскільки основні пошукові поля — ПІБ, місце роботи та посада — зберігаються саме в таблиці фізичних осіб.

Фільтрація записів реалізована за двома параметрами: регіоном і статтею законодавства. Для цього на головній сторінці передбачено два випадаючі списки. Перший список містить регіони, другий — статті законодавства. Якщо користувач обирає певний регіон або статтю, система додає відповідну умову до запиту.

Фільтр за регіоном реалізовано так:

```
if ($request->filled('region_id')) {
   $query->where('region_id', $request->region_id);
}
```

Фільтр за статтею реалізовано аналогічно:

```
if ($request->filled('article_id')) {
   $query->where('article_id', $request->article_id);
}
```

Пошук і фільтри можуть використовуватися одночасно [15; 16]. Наприклад, користувач може ввести прізвище особи та додатково обрати регіон або статтю законодавства. У такому разі система сформує запит з урахуванням усіх заданих умов і відобразить лише ті записи, які відповідають одночасно всім критеріям.

Після застосування пошуку й фільтрів результати виводяться з пагінацією. Для того щоб параметри пошуку та фільтрів зберігалися при переході між сторінками, використовується метод withQueryString() [19]:

```
$offenses = $query
   ->latest()
   ->paginate(10)
   ->withQueryString();
```

Це дозволяє користувачеві переглядати результати пошуку на кількох сторінках без втрати обраних параметрів.

Для реалізації форми пошуку та фільтрації у Blade-шаблоні offenses.index використано метод GET. Це означає, що параметри пошуку передаються через URL-адресу. Такий підхід є зручним, оскільки

користувач може бачити активні параметри запиту, а також оновлювати сторінку без втрати результатів пошуку.

Форма пошуку має такий вигляд:

```
<form method="GET" action="{{ route('offenses.index') }}"
class="filters">
   <input
      type="text"
      name="search"
      placeholder="ПІБ, місце роботи або посада"
      value="{{ request('search') }}"
   >
   <select name="region_id">
      <option value="">Усі регіони</option>
      @foreach($regions as $region)
         <option value="{{ $region->id }}" @selected(request('region_id') ==
$region->id)>
            {{ $region->name }}
         </option>
      @endforeach
   </select>
   <select name="article_id">
      <option value="">Усі статті</option>
      @foreach($articles as $article)
         <option value="{{ $article->id }}" @selected(request('article_id') ==
$article->id)>
            {{ $article->code }}
         </option>
      @endforeach
   </select>
   <button type="submit">Знайти</button>
</form>
```

У формі використовується функція request(), яка дозволяє зберігати введені користувачем значення після виконання пошуку. Наприклад, якщо користувач ввів прізвище або обрав регіон, ці значення залишаються у формі після оновлення сторінки. Це покращує зручність роботи із системою.

Окремим елементом системи є статистичний модуль. Його призначення полягає у формуванні узагальненої інформації про дані, що містяться в системі. Статистика доступна на окремій сторінці за маршрутом /statistics. Сторінку статистики довідкової системи наведено на рисунку 3.8.

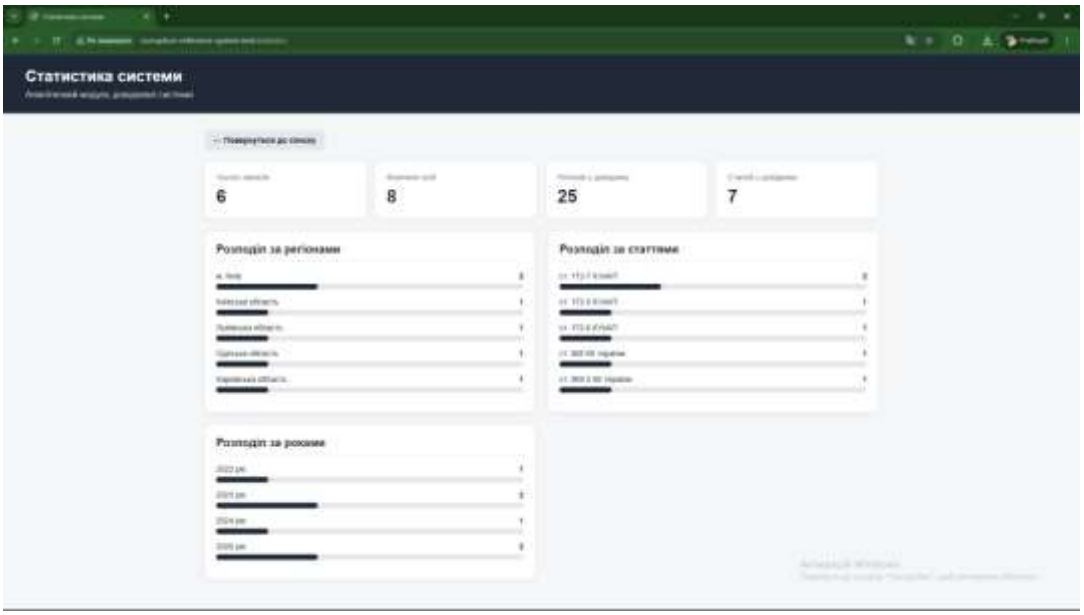

Рисунок 3.8 – Сторінка статистики довідкової системи

У контролері для цього створено метод statistics():

```
public function statistics()
{
    $totalOffenses = Offense::count();
    $totalPeople = Person::count();
    $totalRegions = Region::count();
    $totalArticles = Article::count();

    $offensesByRegion = Region::withCount('offenses')
        ->having('offenses_count', '>', 0)
        ->orderByDesc('offenses_count')
```

```
        ->get();
    $offensesByArticle = Article::withCount('offenses')
        ->having('offenses_count', '>', 0)
        ->orderByDesc('offenses_count')
        ->get();
    $offensesByYear = Offense::select(
            DB::raw('YEAR(decision_date) as year'),
            DB::raw('COUNT(*) as total')
        )
        ->whereNotNull('decision_date')
        ->groupBy('year')
        ->orderBy('year')
        ->get();
    return view('offenses.statistics', compact(
        'totalOffenses',
        'totalPeople',
        'totalRegions',
        'totalArticles',
        'offensesByRegion',
        'offensesByArticle',
        'offensesByYear'
    ));
}
```

У цьому методі формується кілька груп статистичних даних. По-перше, обчислюється загальна кількість записів правопорушень, фізичних осіб, регіонів і статей законодавства. По-друге, формується розподіл правопорушень за регіонами. По-третє, формується розподіл за статтями законодавства. По-четверте, система підраховує кількість записів за роками на основі поля decision_date.

Для підрахунку кількості записів за регіонами та статтями використовується метод withCount() [19]. Він дозволяє отримати кількість пов’язаних записів для кожного регіону або статті. Наприклад:

```
$offensesByRegion = Region::withCount('offenses')
  ->having('offenses_count', '>', 0)
  ->orderByDesc('offenses_count')
  ->get();
```

Цей запит отримує лише ті регіони, які мають хоча б один пов’язаний запис правопорушення, і сортує їх за спаданням кількості записів.

Для підрахунку записів за роками використовується SQL-функція YEAR(), яка отримує рік із дати рішення [20]:

```
$offensesByYear = Offense::select(
    DB::raw('YEAR(decision_date) as year'),
    DB::raw('COUNT(*) as total')
  )
  ->whereNotNull('decision_date')
  ->groupBy('year')
  ->orderBy('year')
  ->get();
```

Це дозволяє відобразити, скільки записів припадає на кожен рік. Така статистика є корисною для загального аналізу динаміки внесених записів.

На сторінці статистики дані подано у вигляді інформаційних карток і горизонтальних індикаторів. Інформаційні картки показують загальні значення: кількість записів, кількість осіб, кількість регіонів і кількість статей. Розподіли за регіонами, статтями та роками відображаються у вигляді списку з числовим значенням і візуальною смугою.

Для обчислення ширини смуги використовується відсоток від загальної кількості записів:

```
$percent = $totalOffenses > 0
  ? round(($region->offenses_count / $totalOffenses) * 100)
  : 0;
```

Після цього значення передається в CSS-властивість width:

Такий спосіб візуалізації дозволяє зробити статистику більш наочною без використання додаткових JavaScript-бібліотек або графічних фреймворків.

Реалізація пошуку, фільтрації та статистики значно підвищує практичну цінність системи. Пошук дозволяє швидко знаходити потрібні записи, фільтри допомагають звузити результати за конкретними критеріями, а статистика дає можливість отримати узагальнену інформацію про стан бази даних.

Таким чином, у довідковій системі реалізовано не лише базові операції додавання, перегляду, редагування та видалення, а й додаткові інструменти для ефективної роботи з інформацією. Це робить систему зручнішою для користувача та дозволяє розглядати її як повноцінний навчальний вебзастосунок із довідковим і аналітичним функціоналом.

3.5. Тестування довідкової системи

Після реалізації основного функціоналу довідкової системи корупціонерів України було проведено тестування її роботи. Метою тестування є перевірка коректності функціонування основних модулів системи, правильності взаємодії з базою даних, працездатності форм, пошуку, фільтрації, статистики, а також перевірка зручності користувацького інтерфейсу [16; 17].

Тестування здійснювалося в локальному середовищі розробки Laravel Herd. Вебзастосунок відкривався у браузері за адресою:
http://corruption-reference-system.test

Також працездатність системи могла перевірятися через вбудований сервер Laravel за адресою:

http://127.0.0.1:8000

Перед початком тестування було перевірено підключення до бази даних MySQL, виконання міграцій і наявність тестових даних у таблицях regions, articles, people та offenses [19; 20]. Для цього використовувалися команди Laravel і перевірка таблиць через MySQL Workbench. Після виконання seeders база даних була наповнена тестовими записами, що дозволило перевірити роботу системи в умовах наявності даних.

Тестування проводилося за такими основними напрямами:

1. перевірка відкриття головної сторінки;
2. перевірка відображення списку записів;
3. перевірка сторінки детального перегляду;
4. перевірка додавання нового запису;
5. перевірка редагування запису;
6. перевірка видалення запису;
7. перевірка додавання нової особи;
8. перевірка списку осіб і видалення особи;
9. перевірка пошуку;
10. перевірка фільтрації;
11. перевірка сторінки статистики;
12. перевірка повідомлень про успішні дії.

На першому етапі було перевірено відкриття головної сторінки системи. У результаті тестування встановлено, що головна сторінка відкривається коректно, заголовок системи відображається правильно, таблиця із записами завантажується з бази даних, а всі пов’язані дані — особа, регіон і стаття — виводяться у зрозумілому для користувача вигляді.

Далі було перевірено роботу сторінки детального перегляду. Для цього на головній сторінці було натиснуто кнопку «Детальніше» біля одного із записів. Система відкрила сторінку з повною інформацією про обраний запис. Було перевірено відображення ПІБ особи, дати народження, місця роботи, посади, регіону, статті законодавства, дати рішення, номера рішення, суду або органу, виду стягнення та опису. Усі дані відображалися відповідно до інформації, збереженої в базі даних.

Наступним етапом було протестовано додавання нового запису правопорушення. Користувач відкривав форму додавання, обирав особу, регіон і статтю, заповнював поля дати рішення, номера рішення, суду або органу, виду стягнення та опису. Після натискання кнопки «Зберегти запис» система зберігала новий запис у таблиці offenses і перенаправляла користувача на сторінку детального перегляду створеного запису. Під час тестування встановлено, що новий запис успішно додається до бази даних і відображається в загальному списку.

Було також перевірено роботу валідації форми додавання [19]. Якщо користувач не обирав обов'язкове поле «Особа», система не дозволяла зберегти запис і відображала повідомлення про помилку. Це підтверджує, що перевірка введених даних працює коректно.

Функція редагування запису була перевірена шляхом відкриття сторінки детального перегляду та натискання кнопки «Редагувати». Після цього відкривалася форма редагування, у якій поля були заповнені поточними даними запису. Після внесення змін і натискання кнопки «Зберегти зміни» система оновлювала запис у базі даних і повертала користувача на сторінку детального перегляду. Перевірка показала, що зміни зберігаються коректно.

Функція видалення запису була перевірена на сторінці детального перегляду. Після натискання кнопки «Видалити» система виводила діалогове підтвердження. Після підтвердження запис видалявся з таблиці offenses, а користувач перенаправлявся на головну сторінку. У списку записів видалений елемент більше не відображався, що підтверджує правильність роботи функції видалення.

Окремо було протестовано роботу з фізичними особами. Система дозволяє додавати нову особу через форму, у якій вводяться прізвище, ім'я, по батькові, дата народження, місце роботи та посада. Після збереження особа додається до таблиці people і стає доступною для вибору під час створення нового запису правопорушення. Під час тестування було підтверджено, що нова особа успішно зберігається та відображається у списку осіб.

Також було перевірено сторінку списку осіб. На цій сторінці відображаються ПІБ, дата народження, місце роботи, посада та кількість пов’язаних записів правопорушень. Під час тестування встановлено, що кількість пов’язаних записів підраховується коректно. Додатково було перевірено видалення особи. Перед видаленням система виводить попередження про те, що пов’язані записи також можуть бути видалені. Після підтвердження особа видаляється з бази даних.

Пошук було протестовано шляхом введення в поле пошуку різних значень: прізвища, імені, місця роботи та посади. Наприклад, при введенні прізвища система відображала лише ті записи, які пов’язані з відповідною особою. Було встановлено, що пошук працює коректно та відсіює записи, які не відповідають запиту користувача.

Фільтрацію було перевірено за регіоном і статтею законодавства. Після вибору певного регіону система відображала лише записи, пов’язані з ним. Аналогічно після вибору статті законодавства відображалися тільки записи з відповідною статтею. Також було перевірено одночасне використання пошуку та фільтрів. Результати відповідали заданим критеріям.

Сторінка статистики була протестована окремо. Було перевірено відображення загальної кількості записів, кількості фізичних осіб, кількості регіонів і кількості статей законодавства. Також було перевірено розподіл записів за регіонами, статтями та роками. Під час тестування встановлено, що статистичні дані формуються відповідно до інформації, яка зберігається в базі даних.

Окремо було перевірено коректність повідомлень, які система відображає після виконання основних дій. Після додавання, редагування або видалення запису користувач отримує відповідне повідомлення про успішне виконання операції. Це підтвердило, що система не лише правильно обробляє дані, а й забезпечує зрозумілий зворотний зв’язок для користувача. Результати тестування основних функцій довідкової системи наведено в таблиці 3.1.

№	Функція системи	Очікуваний результат	Фактичний результат	Статус
1	Відкриття головної сторінки	Сторінка відкривається, таблиця записів відображається	Сторінка відкривається коректно	Успішно
2	Детальний перегляд запису	Відображається повна інформація про запис	Дані відображаються правильно	Успішно
3	Додавання запису	Новий запис зберігається в базі даних	Запис додано та показано в системі	Успішно
4	Валідація форми додавання	Без вибору особи запис не зберігається	Система показує помилку	Успішно
5	Редагування запису	Дані запису оновлюються	Зміни збережено	Успішно
6	Видалення запису	Запис видаляється після підтвердження	Запис видалено зі списку	Успішно
7	Додавання особи	Нова особа додається до бази даних	Особа з’являється у списку	Успішно
8	Видалення особи	Особа видаляється після підтвердження	Особу видалено	Успішно
9	Пошук	Відображаються записи за введеним запитом	Пошук працює коректно	Успішно
10	Фільтр за регіоном	Відображаються записи вибраного регіону	Фільтр працює	Успішно
11	Фільтр за статтею	Відображаються записи вибраної статті	Фільтр працює	Успішно
12	Сторінка статистики	Відображаються узагальнені показники	Статистика формується коректно	Успішно
13	Повідомлення про успіх	Після дії показується повідомлення	Повідомлення відображається	Успішно

Таблиця 3.1 – Результати тестування функціональності довідкової системи

За результатами тестування можна зробити висновок, що основні функції довідкової системи працюють коректно. Система забезпечує перегляд, додавання, редагування та видалення записів, роботу з фізичними особами, пошук, фільтрацію та формування статистики. Виявлені під час розробки дрібні помилки інтерфейсу були усунені, зокрема було виправлено відображення полів у формах та додано повідомлення про успішне виконання дій.

Таким чином, проведене тестування підтвердило працездатність розробленого вебзастосунку та відповідність його основним функціональним вимогам [16; 17]. Система може бути використана як навчальний програмний продукт, що демонструє реалізацію довідкової інформаційної системи з базою даних, CRUD-операціями, пошуком, фільтрацією та статистичним модулем.

3.6. Висновки до розділу 3

У третьому розділі кваліфікаційної бакалаврської роботи було розглянуто практичну реалізацію та тестування довідкової системи корупціонерів України. На основі результатів проєктування, виконаного у другому розділі, було створено вебзастосунок із використанням PHP-фреймворку Laravel, бази даних MySQL, Blade-шаблонів, HTML і CSS.

На початковому етапі було налаштовано середовище розробки. Для локального запуску проєкту використано Laravel Herd, що дозволило швидко організувати роботу PHP, Composer, Laravel, Node.js та локального вебсервера. Також було створено базу даних corruption_reference_system, налаштовано підключення до MySQL через файл .env і перевірено працездатність міграцій.

У межах реалізації бази даних було створено основні таблиці системи: regions, articles, people та offenses. Для цього використано механізм міграцій Laravel. Таблиці було спроєктовано відповідно до сутностей предметної області: регіони, статті законодавства, фізичні особи та записи правопорушень. Центральною таблицею стала таблиця offenses, яка пов’язує записи правопорушень з особами, регіонами та статтями законодавства.

Для роботи з даними було створено моделі Region, Article, Person та Offense. У моделях реалізовано зв’язки між сутностями, зокрема зв’язки типу «один до багатьох» між особами й записами правопорушень, регіонами й записами, статтями й записами. Це дозволило зручно отримувати пов’язані дані та відображати їх у вебінтерфейсі.

У системі було реалізовано основний CRUD-функціонал для записів правопорушень. Користувач може переглядати список записів, відкривати детальну інформацію про окремий запис, додавати нові записи, редагувати наявні та видаляти непотрібні записи. Крім цього, реалізовано роботу з фізичними особами: додавання нової особи, перегляд списку осіб і видалення особи.

Окремо було реалізовано пошук і фільтрацію. Пошук виконується за прізвищем, ім’ям, по батькові, місцем роботи та посадою особи. Фільтрація доступна за регіоном і статтею законодавства. Завдяки цьому користувач може швидко знаходити потрібні записи серед усіх даних системи.

Також було створено статистичний модуль, який відображає загальну кількість записів, кількість фізичних осіб, регіонів і статей законодавства, а також розподіл записів за регіонами, статтями та роками. Наявність цього модуля дозволяє не лише зберігати інформацію, а й виконувати її базове аналітичне узагальнення.

У процесі реалізації було створено основні сторінки вебзастосунку: головну сторінку зі списком записів, сторінку детального перегляду, форму додавання запису, форму редагування запису, сторінку додавання особи, список осіб і сторінку статистики. Для покращення користувацького досвіду реалізовано повідомлення про успішне виконання дій, підтвердження перед видаленням і адаптивне відображення окремих елементів інтерфейсу.

Після завершення розробки було проведено тестування системи. Перевірено відкриття сторінок, відображення записів, додавання, редагування та видалення даних, роботу пошуку, фільтрів, статистики, а також коректність взаємодії з базою даних. Результати тестування показали, що основні функції системи працюють коректно та відповідають поставленим вимогам.

Важливим результатом реалізації є те, що всі основні модулі системи працюють як єдиний вебзастосунок. Дані, які додаються через форми, зберігаються в базі даних і одразу відображаються на відповідних сторінках системи, що підтверджує коректну взаємодію між контролерами, моделями, представленнями та базою даних.

Також у процесі розробки було враховано зручність користувача під час роботи із системою. Інтерфейс побудовано таким чином, щоб користувач міг швидко перейти до потрібної дії: перегляду списку записів, додавання нового запису, роботи з особами або перегляду статистики.

Отже, у третьому розділі було реалізовано функціональний вебзастосунок, який демонструє основні принципи побудови довідкової інформаційної системи. Розроблена система забезпечує роботу з базою даних, підтримує CRUD-операції, пошук, фільтрацію, керування особами та статистичний аналіз. Це підтверджує досягнення практичної мети кваліфікаційної бакалаврської роботи.

ВИСНОВКИ

У кваліфікаційній бакалаврській роботі було виконано розробку довідкової системи корупціонерів України. У процесі виконання роботи було проаналізовано предметну область, визначено основні вимоги до системи, спроєктовано структуру бази даних, реалізовано вебзастосунок і проведено тестування його основних функцій.

У першому розділі було розглянуто поняття та призначення довідкових інформаційних систем. Встановлено, що такі системи є ефективним засобом для зберігання, структурування, пошуку та аналізу інформації. Вони дозволяють організувати великі обсяги даних у зручному вигляді, забезпечити швидкий доступ до потрібних записів і підвищити ефективність роботи користувача з інформацією.

Також було розглянуто особливості обліку інформації про корупційні та пов’язані з корупцією правопорушення. Визначено, що така інформація має складну структуру та включає відомості про фізичних осіб, регіони, статті законодавства, судові рішення, види стягнень і описи правопорушень. Це обґрунтовує необхідність використання структурованої бази даних із взаємопов’язаними таблицями.

У процесі аналізу існуючих інформаційних систем і реєстрів було встановлено, що в Україні функціонують офіційні ресурси для обліку відомостей про осіб, які вчинили корупційні або пов’язані з корупцією правопорушення. Наявність таких ресурсів підтверджує актуальність теми роботи та доцільність створення навчальної довідкової системи, яка демонструє принципи побудови подібного програмного продукту.

У другому розділі було виконано проєктування довідкової системи. Для реалізації системи було обрано веборієнтовану клієнт-серверну архітектуру з використанням шаблону MVC. Такий підхід дозволив розділити логіку системи на моделі, представлення та контролери, що спростило організацію коду й забезпечило можливість подальшого розширення проєкту.

Було обґрунтовано вибір технологій розробки. Для серверної частини використано мову програмування PHP і фреймворк Laravel, для зберігання даних — MySQL, для створення інтерфейсу — Blade-шаблони, HTML і CSS.

Обраний набір технологій дозволив реалізувати повноцінний вебзастосунок із базою даних, CRUD-функціональністю, пошуком, фільтрацією та статистичним модулем.

Під час проєктування бази даних було визначено основні сутності системи: фізичні особи, регіони, статті законодавства та записи правопорушень. На основі цих сутностей було створено таблиці people, regions, articles та offenses. Центральною таблицею стала таблиця offenses, яка поєднує записи правопорушень з особами, регіонами та статтями законодавства.

У третьому розділі було описано практичну реалізацію системи. Спочатку було налаштовано середовище розробки, створено Laravel-проєкт, підключено базу даних MySQL і виконано міграції. Після цього було реалізовано моделі, зв’язки між таблицями, контролери, маршрути та Blade-шаблони для основних сторінок системи.

У результаті розробки було створено вебзастосунок, який забезпечує перегляд списку записів правопорушень, детальний перегляд окремого запису, додавання, редагування та видалення записів. Також реалізовано додавання нової особи, перегляд списку осіб і видалення особи. Для зручності користувача було додано повідомлення про успішне виконання дій і підтвердження перед видаленням даних.

Окремо було реалізовано пошук і фільтрацію. Пошук виконується за ПІБ, місцем роботи та посадою особи. Фільтрація доступна за регіоном і статтею законодавства. Це дозволяє користувачеві швидко знаходити потрібні записи й працювати з великим обсягом інформації більш ефективно.

Також у системі було створено статистичний модуль. Він відображає загальну кількість записів, кількість фізичних осіб, регіонів і статей законодавства, а також розподіл записів за регіонами, статтями та роками. Наявність статистики підвищує практичну цінність системи, оскільки дозволяє не лише зберігати дані, а й виконувати їх базове аналітичне узагальнення.

Після завершення реалізації було проведено тестування системи. Перевірено відкриття сторінок, відображення записів, додавання, редагування та видалення даних, роботу пошуку, фільтрації, статистики та взаємодію з базою даних. Результати тестування показали, що основні

функції системи працюють коректно та відповідають поставленим вимогам.

Таким чином, мету кваліфікаційної бакалаврської роботи досягнуто. Було розроблено довідкову систему корупціонерів України у вигляді вебзастосунку, який забезпечує зберігання, перегляд, додавання, редагування, видалення, пошук, фільтрацію та статистичне узагальнення інформації про корупційні та пов’язані з корупцією правопорушення.

Практичне значення розробленої системи полягає в тому, що вона демонструє повний цикл створення інформаційно-довідкового вебзастосунку: від аналізу предметної області та проєктування бази даних до реалізації функціональних модулів і тестування. Розроблений програмний продукт може бути використаний як навчальний і демонстраційний приклад, а також як основа для подальшого розвитку більш складної інформаційної системи.

Під час виконання роботи було підтверджено доцільність використання веборієнтованого підходу для створення довідкових інформаційних систем. Такий формат забезпечує зручний доступ до системи через браузер, спрощує роботу користувача з даними та дозволяє централізовано зберігати інформацію в базі даних.

Розроблена система демонструє практичне застосування архітектурного шаблону MVC, який дозволяє розділити логіку роботи вебзастосунку на моделі, контролери та представлення. Це забезпечує впорядковану структуру програмного коду, полегшує його підтримку та створює основу для подальшого розширення функціональності.

Особливе значення у роботі має реалізація зв’язків між таблицями бази даних, оскільки саме вони забезпечують коректне поєднання інформації про фізичних осіб, регіони, статті законодавства та записи правопорушень. Завдяки цьому система може відображати повну інформацію про кожний запис, виконувати фільтрацію та формувати статистичні показники.

У перспективі систему можна розширити шляхом додавання авторизації користувачів, розмежування прав доступу, імпорту та експорту даних, інтеграції з відкритими джерелами інформації, журналу дій користувачів і розширеного аналітичного модуля. Це дозволить підвищити функціональність, безпеку та практичну цінність програмного продукту.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Закон України «Про запобігання корупції» від 14.10.2014 № 1700-VII.
2. Положення про Єдиний державний реєстр осіб, які вчинили корупційні або пов’язані з корупцією правопорушення.
3. Єдиний державний реєстр осіб, які вчинили корупційні або пов’язані з корупцією правопорушення. Національне агентство з питань запобігання корупції.
4. Національне агентство з питань запобігання корупції. Офіційний вебсайт.
5. Нестеренко О. В. Основи програмної інженерії: методичні рекомендації для здобувачів спеціальності 121 «Інженерія програмного забезпечення». Міжнародний європейський університет, 2025.
6. Нестеренко О. В. Групова динаміка і комунікації: навчально-методичні матеріали. Міжнародний європейський університет, 2022.
7. Nesterenko O. The Teams Information Model for Software Engineering Development Processes. Міжнародний європейський університет, 2021.
8. Нестеренко О. В. Порівняльний аналіз підходів до вдосконалення освітнього процесу в галузі програмної інженерії. Міжнародний європейський університет, 2022.
9. Нестеренко О. В. Інформаційні технології, цифровізація та штучний інтелект: виклики для бізнесу та менеджменту. Розвиток економіки та бізнес-адміністрування: наукові течії та рішення: збірник тез доповідей V Міжнародної науково-практичної конференції. Київ, 2024. С. 74–75.
10. Нестеренко О. В., Федоров В., Яцук П. П., Чабан В. П. Технологічний розвиток електронного урядування як інноваційна стратегія держави. Наука, технології, інновації, 2025.
11. Болілий Р. О., Шевчук Б. В. Використання UML діаграм в програмуванні. Освіта і наука – 2023: матеріали науково-звітної студентської конференції УДУ імені Михайла Драгоманова. Київ, 2023.
12. Рудник С. М., Шевчук Б. В. Проектування реляційних баз даних. Освіта і наука – 2023: матеріали науково-звітної студентської конференції УДУ імені Михайла Драгоманова. Київ, 2023.

13. Литвин В. С., Шевчук Б. В. Інноваційні розробки в галузі баз даних та їх вплив на сучасне суспільство. Освіта і наука – 2023: матеріали науково-звітної студентської конференції УДУ імені Михайла Драгоманова. Київ, 2023.
14. Бородкіна І., Бородкін Г. Інженерія програмного забезпечення: посібник для студентів вищих навчальних закладів. Київ, 2018. 230 с.
15. Silberschatz A., Korth H. F., Sudarshan S. Database System Concepts. 7th ed. New York: McGraw-Hill Education, 2019.
16. Sommerville I. Software Engineering. 10th ed. Pearson, 2016.
17. Pressman R. S., Maxim B. R. Software Engineering: A Practitioner’s Approach. 9th ed. McGraw-Hill Education, 2020.
18. Fowler M. Patterns of Enterprise Application Architecture. Addison-Wesley, 2002.
19. Laravel Documentation. The PHP Framework for Web Artisans.
20. MySQL Documentation. MySQL Reference Manual.
21. PHP Manual. Офіційна документація мови PHP.
22. MDN Web Docs. HTML: HyperText Markup Language.
23. MDN Web Docs. CSS: Cascading Style Sheets.

ДОДАТКИ

ДОДАТОК А
Фрагменти програмного коду окремих модулів системи

```
public function index(Request $request)
{
    $query = Offense::with(['person', 'region', 'article']);
    if ($request->filled('search')) {
        $search = $request->search;
        $query->whereHas('person', function ($q) use ($search) {
            $q->where('last_name', 'like', "%{$search}%")
                ->orWhere('first_name', 'like', "%{$search}%")
                ->orWhere('middle_name', 'like', "%{$search}%")
                ->orWhere('workplace', 'like', "%{$search}%")
                ->orWhere('position', 'like', "%{$search}%");
        });
    }
    if ($request->filled('region_id')) {
        $query->where('region_id', $request->region_id);
    }
    if ($request->filled('article_id')) {
        $query->where('article_id', $request->article_id);
    }
    $offenses = $query
        ->latest()
        ->paginate(10)
        ->withQueryString();
    $regions = Region::orderBy('name')->get();
    $articles = Article::orderBy('code')->get();
    return view('offenses.index', compact('offenses', 'regions', 'articles'));
}
```

Лістинг А.1 – Фрагмент контролера OffenseController для виведення списку записів, пошуку та фільтрації

```
public function store(Request $request)
{
    $validated = $request->validate([
        'person_id' => ['required', 'exists:people,id'],
        'region_id' => ['nullable', 'exists:regions,id'],
        'article_id' => ['nullable', 'exists:articles,id'],
        'decision_date' => ['nullable', 'date'],
        'decision_number' => ['nullable', 'string', 'max:255'],
        'court_name' => ['nullable', 'string', 'max:255'],
        'punishment_type' => ['nullable', 'string', 'max:255'],
        'description' => ['nullable', 'string'],
    ]);
    $offense = Offense::create($validated);
    return redirect()
        ->route('offenses.show', $offense)
        ->with('success', 'Запис успішно додано.');
}
```

Лістинг А.2 – Фрагмент методу додавання нового запису правопорушення

```
public function statistics()
{
    $totalOffenses = Offense::count();
    $totalPeople = Person::count();
    $totalRegions = Region::count();
    $totalArticles = Article::count();
    $offensesByRegion = Region::withCount('offenses')
        ->having('offenses_count', '>', 0)
        ->orderByDesc('offenses_count')
        ->get();
    $offensesByArticle = Article::withCount('offenses')
        ->having('offenses_count', '>', 0)
        ->orderByDesc('offenses_count')
```

```
        ->get();
    $offensesByYear = Offense::select(
            DB::raw('YEAR(decision_date) as year'),
            DB::raw('COUNT(*) as total')
        )
        ->whereNotNull('decision_date')
        ->groupBy('year')
        ->orderBy('year')
        ->get();
    return view('offenses.statistics', compact(
        'totalOffenses',
        'totalPeople',
        'totalRegions',
        'totalArticles',
        'offensesByRegion',
        'offensesByArticle',
        'offensesByYear'
    ));
}
```

Лістинг А.3 – Фрагмент методу формування статистичних показників системи

```
public function index()
{
    $people = Person::withCount('offenses')
        ->orderBy('last_name')
        ->paginate(10);
    return view('people.index', compact('people'));
}
public function store(Request $request)
{
    $validated = $request->validate([
        'last_name' => ['required', 'string', 'max:255'],
        'first_name' => ['required', 'string', 'max:255'],
        'middle_name' => ['nullable', 'string', 'max:255'],
        'birth_date' => ['nullable', 'date'],
        'workplace' => ['nullable', 'string', 'max:255'],
```

```
            'position' => ['nullable', 'string', 'max:255'],
        ]);
        Person::create($validated);
        return redirect()
            ->route('people.index')
            ->with('success', 'Особу успішно додано.');
    }
    public function destroy(Person $person)
    {
        $person->delete();
        return redirect()
            ->route('people.index')
            ->with('success', 'Особу успішно видалено.');
    }
```

Лістинг А.4 – Фрагмент контролера PersonController для роботи з фізичними особами

```
class Offense extends Model
{
    protected $fillable = [
        'person_id',
        'region_id',
        'article_id',
        'decision_date',
        'decision_number',
        'court_name',
        'punishment_type',
        'description',
    ];
    public function person()
    {
        return $this->belongsTo(Person::class);
    }
    public function region()
    {
        return $this->belongsTo(Region::class);
    }
    public function article()
```

```
    {
        return $this->belongsTo(Article::class);
    }
}
```

Лістинг А.5 – Модель Offense та зв’язки з іншими сутностями системи

```
Route::get('/', [OffenseController::class, 'index'])
    ->name('offenses.index');
Route::get('/offenses/create', [OffenseController::class, 'create'])
    ->name('offenses.create');
Route::post('/offenses', [OffenseController::class, 'store'])
    ->name('offenses.store');
Route::get('/offenses/{offense}', [OffenseController::class, 'show'])
    ->name('offenses.show');
Route::get('/offenses/{offense}/edit', [OffenseController::class, 'edit'])
    ->name('offenses.edit');
Route::put('/offenses/{offense}', [OffenseController::class, 'update'])
    ->name('offenses.update');
Route::delete('/offenses/{offense}', [OffenseController::class, 'destroy'])
    ->name('offenses.destroy');
Route::get('/people', [PersonController::class, 'index'])
    ->name('people.index');
Route::get('/people/create', [PersonController::class, 'create'])
    ->name('people.create');
Route::post('/people', [PersonController::class, 'store'])
    ->name('people.store');
Route::delete('/people/{person}', [PersonController::class, 'destroy'])
    ->name('people.destroy');
Route::get('/statistics', [OffenseController::class, 'statistics'])
    ->name('offenses.statistics');
```

Лістинг А.6 – Фрагмент маршрутизації довідкової системи

ДОДАТОК Д
Результати тестування системи

№	Функція системи	Очікуваний результат	Фактичний результат	Статус
1	Відкриття головної сторінки	Сторінка відкривається, таблиця записів відображається	Сторінка відкривається коректно	Успішно
2	Детальний перегляд запису	Відображається повна інформація про запис	Дані відображаються правильно	Успішно
3	Додавання запису	Новий запис зберігається в базі даних	Запис додано та показано в системі	Успішно
4	Валідація форми додавання	Без вибору особи запис не зберігається	Система показує помилку	Успішно
5	Редагування запису	Дані запису оновлюються	Зміни збережено	Успішно
6	Видалення запису	Запис видаляється після підтвердження	Запис видалено зі списку	Успішно
7	Додавання особи	Нова особа додається до бази даних	Особа з’являється у списку	Успішно
8	Видалення особи	Особа видаляється після підтвердження	Особу видалено	Успішно
9	Пошук	Відображаються записи за введеним запитом	Пошук працює коректно	Успішно
10	Фільтр за регіоном	Відображаються записи вибраного регіону	Фільтр працює	Успішно
11	Фільтр за статтею	Відображаються записи вибраної статті	Фільтр працює	Успішно
12	Сторінка статистики	Відображаються узагальнені показники	Статистика формується коректно	Успішно
13	Повідомлення про успіх	Після дії показується повідомлення	Повідомлення відображається	Успішно

Таблиця Д.1 – Результати тестування функціональності довідкової системи