

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «бакалавр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Інженерія програмного забезпечення»)

на тему:

**«Вебплатформа для подачі та обробки звернень
громадян»**

Виконав студент групи ІПЗ-22-4
ЧМУТ Дмитро Костянтинович

Керівник роботи:
ЄФРЕМОВ Юрій Миколайович

Рецензент:
ТАЛАВЕР Олег Володимирович

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

«ЗАТВЕРДЖУЮ»

Зав. каф. інженерії
програмного забезпечення
Тетяна ВАКАЛЮК
«28» лютого 2026 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **Чмут Дмитро Костянтинович**

Керівник роботи: **Єфремов Юрій Миколайович**

Тема роботи: **«Вебплатформа для подачі та обробки звернень громадян»,**
затверджена Наказом закладу вищої освіти від **«27» лютого 2026 р., №92/с**

Вихідні дані для роботи: **розробка вебплатформи для подачі та обробки**
звернень громадян.

Консультанти з бакалаврської кваліфікаційної роботи із зазначенням розділів,
що їх стосуються:

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Єфремов Ю.М.	28.02.2026р.	28.02.2026р
2	Єфремов Ю.М.	28.02.2026р	28.02.2026р
3	Єфремов Ю.М.	28.02.2026р	28.02.2026р

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи виконана на тему «Вебплатформа для подачі та обробки звернень громадян». Загальний обсяг роботи становить 85 сторінки, робота містить 31 рисунок та 25 таблиць.

Метою роботи є розробка вебплатформи для подачі та обробки звернень громадян, яка забезпечує зручний, структурований і прозорий механізм взаємодії між користувачами та системою, а також дозволяє контролювати процес виконання звернень. У роботі проведено аналіз предметної області, досліджено існуючі програмні рішення у сфері електронних звернень громадян, визначено їхні переваги та недоліки. На основі проведеного аналізу сформовано функціональні та нефункціональні вимоги до системи. Результатом роботи є створення вебплатформи для подачі та обробки звернень громадян із можливістю подальшого розширення функціоналу.

КЛЮЧОВІ СЛОВА: ВЕБ-ПЛАТФОРМА, ЗВЕРНЕННЯ ГРОМАДЯН, ОБРОБКА ЗАЯВОК, ЕЛЕКТРОННІ СЕРВІСИ, LARAVEL, PHP, БАЗА ДАНИХ, MVC, ORM.

					ІПЗ.КР.Б – 121 – 26 – ПЗ					
Змн.	Арк.	№ докум.	Підпис	Дата	Вебплатформа для подачі та обробки звернень громадян			Літ.	Арк.	Аркушів
Розроб.		Чмут Д.К.								
Керівник		Єфремов Ю.М.							3	94
								Житомирська політехніка, група ІПЗ-22-4		
Н. контр.										
Зав. каф.		Вакалюк Т.А.			Пояснювальна записка					

ABSTRACT

The explanatory note to the qualification work is devoted to the topic "Web Platform for Submitting and Processing Citizens' Requests". The total volume of the work is 85 pages, 31 including figures and 25 tables.

The purpose of the work is to develop a web platform for submitting and processing citizens' requests, which provides a convenient, structured, and transparent mechanism of interaction between users and the system, as well as allows monitoring the process of request execution. The work includes an analysis of the subject area, a study of existing software solutions in the field of electronic citizens' requests, and identification of their advantages and disadvantages. Based on the conducted analysis, functional and non-functional requirements for the system were formed. The result of the work is the creation of a web platform for submitting and processing citizens' requests with the possibility of further functional expansion.

KEYWORDS: WEB PLATFORM, CITIZENS' REQUESTS, REQUEST PROCESSING, ELECTRONIC SERVICES, LARAVEL, PHP, DATABASE, MVC, ORM.

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				4
Змн.	Арк.	№ докум.	Підпис	Дата		

ЗМІСТ

РЕФЕРАТ	3
ABSTRACT	4
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ.....	6
ВСТУП	7
РОЗДІЛ 1. АНАЛІЗ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ...	10
1.1 Постановка задачі	10
1.2 Аналіз аналогів програмного продукту	14
1.3 Обґрунтування вибору архітектури вебплатформи CityVoice	21
1.4 Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення	25
Висновки до першого розділу	32
РОЗДІЛ 2. ПРОЕКТУВАННЯ СИСТЕМИ.....	34
2.1. Визначення варіантів використання та об’єктної структури системи	34
2.2. Розробка бази даних системи.....	44
2.3. Проєктування та реалізація алгоритмів системи	51
Висновки до другого розділу.....	56
РОЗДІЛ 3 ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ ІЗ СИСТЕМОЮ	58
3.1 Порядок встановлення та налаштування параметрів системи.....	58
3.2 Структура інтерфейсу платформи	62
3.3 Тестування програмної системи.....	77
Висновки до третього розділу	80
ВИСНОВКИ.....	81
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	83
ДОДАТКИ.....	86
Додаток А	87
Додаток Б	89
Додаток В.....	90

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

API – Application Programming Interface (Інтерфейс програмування застосунків);

БД – База даних;

SQL – Structured Query Language

MVC – Model-View-Controller

ORM – Object-Relational Mapping

ER – Entity-Relationship

HTTP – HyperText Transfer Protocol

SPA – Single Page Application

UI – User Interface

PDF – Portable Document Format

UML – Unified Modeling Language

URL – Uniform Resource Locator

UX – User Experience

ПЗ – Програмне забезпечення

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		6

ВСТУП

Актуальність теми. Стрімкий розвиток інформаційно-комунікаційних технологій суттєво вплинув на всі сфери суспільного життя, зокрема на взаємодію громадян із органами державної влади та місцевого самоврядування. У сучасних умовах цифрової трансформації суспільства дедалі більшого значення набуває впровадження електронних сервісів, які забезпечують швидке, зручне та прозоре надання адміністративних послуг, підвищують ефективність роботи державних установ і сприяють активній участі громадян у вирішенні суспільно важливих питань.

Одним із важливих напрямів цифровізації є створення інформаційних систем, призначених для прийому, обробки та контролю виконання звернень громадян. Щоденно мешканці міст стикаються з великою кількістю проблем, пов'язаних із пошкодженням дорожнього покриття, аварійним станом комунальних мереж, несправністю вуличного освітлення, несанкціонованими сміттєзвалищами, благоустроєм територій, екологічними проблемами та іншими питаннями, що потребують оперативного реагування відповідальних служб. Ефективність вирішення таких проблем значною мірою залежить від швидкості передачі інформації, її достовірності, можливості контролю виконання звернення та забезпечення зворотного зв'язку між громадянином і відповідальною установою.

На практиці значна частина звернень подається за допомогою телефонних дзвінків, електронної пошти, соціальних мереж або усних повідомлень. Такі способи комунікації є недостатньо структурованими, ускладнюють централізований облік звернень, не забезпечують єдиного механізму контролю за їх виконанням та не дозволяють громадянам відстежувати поточний стан розгляду власних заяв. Крім того, інформація може дублюватися, втрачатися або надходити одночасно до різних структурних підрозділів, що негативно впливає на швидкість прийняття рішень та якість надання послуг.

Використання сучасних вебтехнологій дозволяє автоматизувати більшість процесів, пов'язаних із реєстрацією, обробкою, маршрутизацією та контролем

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		7

виконання звернень. Вебплатформи забезпечують цілодобовий доступ до сервісу незалежно від місця перебування користувача, дозволяють централізовано зберігати інформацію, автоматизувати обмін даними між користувачами та адміністраторами, а також реалізувати механізми сповіщення, модерації та аналітичної обробки накопичених даних.

Особливого значення такі системи набувають у процесі розвитку концепції електронного урядування та цифрової держави, де взаємодія між громадянином і владою здійснюється переважно засобами інформаційних технологій. Використання сучасних вебплатформ сприяє підвищенню відкритості діяльності органів влади, забезпечує прозорість процесу розгляду звернень, покращує рівень довіри громадян до державних інституцій та дозволяє оперативніше реагувати на актуальні проблеми громади.

Враховуючи зазначене, актуальним є створення сучасної вебплатформи, яка забезпечуватиме централізоване подання звернень громадян, можливість їх категоризації, модерації, контролю стану виконання, взаємодії між користувачами та адміністраторами, а також накопичення інформації для подальшого аналізу. Саме вирішенню цих завдань присвячена дана кваліфікаційна робота, у межах якої розроблено вебплатформу CityVoice.

Розроблена система орієнтована на забезпечення ефективної взаємодії між громадянами та відповідальними особами шляхом використання сучасних вебтехнологій. Вона дозволяє користувачам створювати звернення із зазначенням категорії, опису проблеми, фотографій та географічного розташування, переглядати статус їх опрацювання, брати участь в обговоренні, підтримувати звернення шляхом голосування та отримувати інформацію про результати їх розгляду. Для адміністраторів передбачено окрему панель керування, що забезпечує модерацію звернень, управління користувачами, категоріями, статусами та іншими компонентами системи.

Метою роботи є розробка вебплатформи для подання та обробки звернень громадян, яка забезпечує структурований процес створення, перегляду, редагування та контролю виконання звернень.

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		8

Для досягнення поставленої мети необхідно вирішити **такі завдання:**

- проаналізувати існуючі програмні рішення у сфері електронних звернень та визначити їх переваги й недоліки;
- сформулювати функціональні вимоги до вебплатформи;
- обґрунтувати вибір архітектури програмного забезпечення, технологій та інструментальних засобів розробки;
- спроектувати структуру бази даних, взаємодію основних модулів системи та програмну архітектуру;
- реалізувати функціональні можливості вебплатформи для роботи зі зверненнями громадян;
- провести тестування розробленої системи та оцінити її працездатність.

Об'єктом дослідження є процеси подання, реєстрації, обробки та контролю виконання звернень громадян.

Предметом дослідження є методи та засоби проєктування і реалізації вебплатформи для подання та обробки звернень громадян.

Під час виконання роботи використовувалися методи системного аналізу, об'єктно-орієнтованого проєктування, моделювання структури баз даних, аналізу програмних аналогів, проєктування архітектури програмного забезпечення, а також сучасні методи веброзробки із застосуванням фреймворку Laravel, мови програмування PHP, системи керування базами даних MySQL, HTML, CSS, JavaScript та інших програмних засобів.

Практичне значення отриманих результатів полягає у створенні працездатної вебплатформи, яка може бути використана як основа для впровадження інформаційної системи електронних звернень у діяльність органів місцевого самоврядування, комунальних підприємств або інших установ, що здійснюють прийом та обробку звернень громадян.

Структура кваліфікаційної роботи складається зі вступу, трьох розділів, висновків, списку використаних джерел та додатків.

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				9
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 1. АНАЛІЗ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

1.1 Постановка задачі

Метою розробки є створення вебплатформи CityVoice, яка забезпечить подання, обробку та контроль виконання звернень громадян щодо проблем міської інфраструктури, благоустрою територій, житлово-комунального господарства та інших питань місцевого значення. Програмний продукт повинен забезпечувати ефективну взаємодію між громадянами, модераторами та адміністраторами, а також спростувати процес реєстрації, опрацювання й моніторингу звернень у межах єдиного інформаційного середовища.

У сучасних умовах цифрової трансформації електронні сервіси стали невід'ємною складовою діяльності органів державної влади та місцевого самоврядування. Використання вебтехнологій дозволяє автоматизувати значну кількість адміністративних процесів, підвищити швидкість обробки інформації та забезпечити відкритість взаємодії між громадянами й відповідальними службами. Одним із важливих напрямів цифровізації є впровадження інформаційних систем, призначених для централізованого приймання, реєстрації, обробки та контролю виконання звернень громадян.

Попри активний розвиток електронного врядування, у багатьох громадах повідомлення про проблеми міської інфраструктури досі подаються за допомогою телефонних дзвінків, електронної пошти, соціальних мереж, месенджерів або особистих звернень. Такі способи комунікації не забезпечують єдиного підходу до реєстрації повідомлень, ускладнюють їх централізоване зберігання та не дозволяють здійснювати повноцінний контроль за процесом їх виконання.

Використання різних каналів взаємодії призводить до виникнення низки організаційних проблем. Частина звернень дублюється, окремі повідомлення можуть залишатися без належного опрацювання, а інформація нерідко надходить без достатнього опису або підтверджувальних матеріалів. Відсутність

		Чмут Д.К.			ІПЗ.КР.Б – 121 – 26 – ІЗ	Арк.
		Єфремов. Ю.М.				10
Змн.	Арк.	№ докум.	Підпис	Дата		

єдиної інформаційної бази ускладнює аналіз накопичених даних, визначення найбільш актуальних проблем громади та формування статистичних показників, необхідних для оцінювання ефективності роботи відповідальних служб.

Не менш важливою проблемою є недостатня прозорість процесу розгляду звернень. У більшості випадків заявники не мають можливості відстежувати стан виконання власного повідомлення, отримувати інформацію про прийняті рішення або контролювати виконання робіт. Крім того, відсутність відкритої взаємодії між користувачами призводить до повторного створення однакових звернень, що збільшує навантаження на відповідальні служби та знижує ефективність їх роботи.

З огляду на зазначені недоліки актуальним завданням є створення сучасної вебплатформи, яка забезпечить централізовану взаємодію між громадянами та органами місцевого самоврядування, автоматизує процес подання й опрацювання звернень та створить єдиний інформаційний простір для накопичення, систематизації й аналізу інформації про проблеми міської інфраструктури.

Основною перевагою запропонованого програмного продукту є централізоване зберігання інформації про всі зареєстровані звернення в єдиній базі даних. Завдяки цьому громадяни отримують можливість оперативно повідомляти про виявлені проблеми, контролювати процес їх розгляду, брати участь у громадському обговоренні та підтримувати найбільш актуальні ініціативи. Для відповідальних осіб вебплатформа створює єдиний інструмент керування зверненнями, який забезпечує їх систематизацію, зміну статусів виконання, формування статистичної інформації та контроль ефективності виконаних робіт.

Для досягнення поставленої мети необхідно виконати комплекс взаємопов'язаних завдань, що охоплюють усі етапи створення програмного продукту — від формування функціональних вимог до реалізації, тестування та підготовки системи до практичного використання [23]. На першому етапі необхідно визначити основні вимоги до вебплатформи, сформулювати перелік

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		11

функціональних можливостей та обґрунтувати архітектурні рішення, які забезпечать стабільну, безпечну й ефективну роботу системи.

Першочерговим завданням є створення підсистеми реєстрації та автентифікації користувачів. Вона повинна забезпечувати можливість створення облікового запису, авторизації, відновлення доступу до нього та розмежування прав доступу відповідно до ролі користувача. Реалізація зазначених функцій дозволить гарантувати захист персональних даних, запобігти несанкціонованому доступу до службових можливостей системи та забезпечити безпечну взаємодію всіх категорій користувачів.

Наступним етапом є реалізація функціоналу роботи зі зверненнями громадян. Користувачі повинні мати можливість створювати нові звернення, обирати категорію проблеми, додавати її опис, зазначати місце розташування, прикріплювати фотографії або інші інформаційні матеріали. Крім створення звернень, система повинна забезпечувати їх перегляд, редагування, пошук, сортування та фільтрацію за визначеними параметрами, що значно спростить процес роботи як для громадян, так і для відповідальних осіб.

Особливу увагу необхідно приділити підтримці життєвого циклу звернення. Кожне повідомлення повинно проходити послідовні етапи обробки, які супроводжуються зміною його статусу. Такий підхід забезпечить прозорість виконання робіт, дозволить користувачам відстежувати поточний стан розгляду звернення та своєчасно отримувати інформацію про результати його опрацювання.

Важливою складовою вебплатформи є реалізація механізмів взаємодії між користувачами. Програмний продукт повинен підтримувати можливість коментування звернень, голосування за найбільш актуальні проблеми, перегляд історії змін та надсилання інформаційних повідомлень про результати їх розгляду. Використання таких механізмів сприятиме активнішій участі громадян у вирішенні проблем місцевого значення, підвищить відкритість процесу їх розгляду та дозволить визначати найбільш пріоритетні звернення.

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				12
Змн.	Арк.	№ докум.	Підпис	Дата		

Окремим завданням є створення адміністративної частини системи. Адміністратори повинні отримати інструменти для керування користувачами, категоріями звернень, ролями доступу та іншими параметрами програмного продукту. Для модераторів необхідно реалізувати функціонал перевірки звернень, зміни їх статусів, контролю коректності опублікованої інформації та взаємодії з авторами повідомлень. Це забезпечить централізоване керування вебплатформою та підвищить ефективність її використання.

Під час розробки особлива увага повинна приділятися забезпеченню інформаційної безпеки. Програмний продукт має гарантувати захист персональних даних користувачів, безпечне зберігання облікової інформації, контроль прав доступу та захист від несанкціонованого втручання. Крім цього, необхідно забезпечити цілісність даних, стабільну обробку запитів і надійну роботу системи навіть за умови одночасної взаємодії значної кількості користувачів.

Наступним етапом є проєктування архітектури програмного забезпечення та вибір інструментальних засобів розробки. Архітектура системи повинна забезпечувати модульність, масштабованість і можливість подальшого розширення функціональних можливостей без суттєвої зміни вже реалізованих компонентів. Використання сучасних технологій веброзробки дозволить забезпечити ефективну взаємодію серверної та клієнтської частин, надійне зберігання інформації та зручність подальшого супроводу програмного продукту.

Завершальним етапом є тестування розробленої вебплатформи. Необхідно перевірити правильність роботи всіх функціональних модулів, коректність взаємодії між окремими компонентами системи, відповідність реалізованого функціоналу визначеним вимогам, а також усунути виявлені недоліки. Після завершення тестування програмний продукт повинен забезпечувати стабільну роботу, достатній рівень продуктивності та готовність до практичного використання.

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				13
Змн.	Арк.	№ докум.	Підпис	Дата		

У результаті виконання поставлених завдань має бути розроблена сучасна вебплатформа, яка забезпечить ефективну взаємодію між громадянами та органами місцевого самоврядування, автоматизує процес подання, обробки та контролю виконання звернень, а також створить єдиний інформаційний простір для накопичення, аналізу й систематизації інформації про проблеми міської інфраструктури. Використання розробленої системи сприятиме підвищенню прозорості роботи відповідальних служб, покращенню якості електронних сервісів і розвитку цифрової взаємодії між громадянами та місцевою владою.

1.2 Аналіз аналогів програмного продукту

Перед початком розробки вебплатформи доцільно провести аналіз існуючих інформаційних систем, призначених для взаємодії громадян з органами місцевого самоврядування. Дослідження аналогів дозволяє визначити сучасні підходи до організації процесу подання та обробки звернень, оцінити функціональні можливості вже реалізованих програмних продуктів, а також виявити їх сильні й слабкі сторони. Отримані результати є основою для формування вимог до розроблюваної вебплатформи та вибору функціональних можливостей, які доцільно реалізувати під час створення системи програмного продукту.

На сьогодні існує значна кількість програмних рішень, які забезпечують електронну взаємодію між громадянами та органами місцевого самоврядування. Частина таких систем орієнтована на подання звернень щодо проблем міської інфраструктури, інші призначені для підтримки електронних петицій або централізованого приймання повідомлень через контактні центри. Незважаючи на різне функціональне призначення, усі вони реалізують окремі механізми електронної взаємодії, що можуть бути використані під час проєктування власної інформаційної системи. Після проведення аналізу сервісів e-DEM «Відкрите місто

Для аналізу було обрано програмні рішення Єдиний вебпортал звернень громадян, e-DEM «Місцеві петиції» та Контактний центр 15-51, які найбільш

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				14
Змн.	Арк.	№ докум.	Підпис	Дата		

повно відповідають предметній області дипломної роботи та використовуються в діяльності органів місцевого самоврядування України.

Першим аналогом є сервіс «Урядовий контактний центр (1545) – Подати електронне звернення», який призначений для приймання, реєстрації та опрацювання електронних звернень громадян до органів виконавчої влади, органів місцевого самоврядування, державних підприємств та інших установ (Рис. 1.1). Основною метою сервісу є забезпечення офіційної взаємодії між громадянами та органами влади відповідно до вимог Закону України «Про звернення громадян» [1]. Використання вебпорталу дозволяє подавати звернення в електронній формі без необхідності особистого відвідування державних установ, що сприяє розвитку електронного урядування та підвищенню доступності державних послуг.

Після авторизації користувач може створити електронне звернення, обрати орган або установу, до якої воно адресоване, заповнити необхідні реквізити та надіслати повідомлення на розгляд. Система забезпечує централізовану реєстрацію звернень, їх облік, можливість відстеження етапів опрацювання та отримання офіційної відповіді після завершення розгляду. Усі звернення обробляються відповідно до чинного законодавства України та встановлених процедур роботи Урядового контактного центру.

До основних переваг сервісу належать офіційний статус державної інформаційної системи, можливість електронного подання звернень, централізований облік повідомлень, підтримка особистого кабінету користувача, можливість контролю стану розгляду звернення та відповідність вимогам чинного законодавства України. Крім того, сервіс забезпечує взаємодію

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		15

громадян із широким колом державних органів через єдину інформаційну платформу.

Рис.1.1 – Інтерфейс Єдиного вебпорталу звернень громадян

Другим аналогом є модуль «Місцеві петиції» платформи e-DEM, який призначений для електронної взаємодії між громадянами та органами місцевого самоврядування шляхом створення, підтримки та розгляду електронних петицій (Рис. 1.2). На відміну від сервісу «Відкрите місто», що орієнтований на повідомлення про конкретні проблеми міської інфраструктури, цей модуль використовується для подання ініціатив, пропозицій та колективних звернень щодо розвитку громади, удосконалення місцевого законодавства або вирішення суспільно важливих питань.

Платформа надає можливість зареєстрованим користувачам створювати нові петиції, визначати їх тему та зміст, а також підтримувати вже опубліковані ініціативи шляхом електронного голосування. Для кожної петиції відображається інформація про кількість зібраних голосів, необхідний поріг підтримки, термін збору голосів і поточний статус розгляду. Після досягнення встановленої кількості голосів петиція передається на офіційний розгляд органу

		Чмут Д.К.			ПІЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				16
Змн.	Арк.	№ докум.	Підпис	Дата		

місцевого самоврядування, який повинен надати відповідь у визначений законодавством строк.

До основних переваг сервісу належать відкритість процесу голосування, прозорість розгляду петицій, можливість залучення громадян до прийняття управлінських рішень, простий механізм підтримки ініціатив та централізоване зберігання інформації про всі подані петиції. Крім цього, система забезпечує публічний доступ до результатів голосування та офіційних відповідей органів місцевого самоврядування, що сприяє підвищенню довіри громадян до електронних сервісів.

Водночас сервіс має і певні недоліки. Його функціональність обмежується виключно роботою з електронними петиціями та не передбачає можливості оперативного повідомлення про локальні проблеми міської інфраструктури. У системі відсутні механізми прикріплення геолокації до звернення, зміни статусів виконання окремих робіт, коментування процесу їх виконання та взаємодії між користувачами під час опрацювання конкретної проблеми. Через це сервіс не може повністю задовольнити потреби громадян щодо швидкого інформування відповідальних служб про поточні проблеми громади.

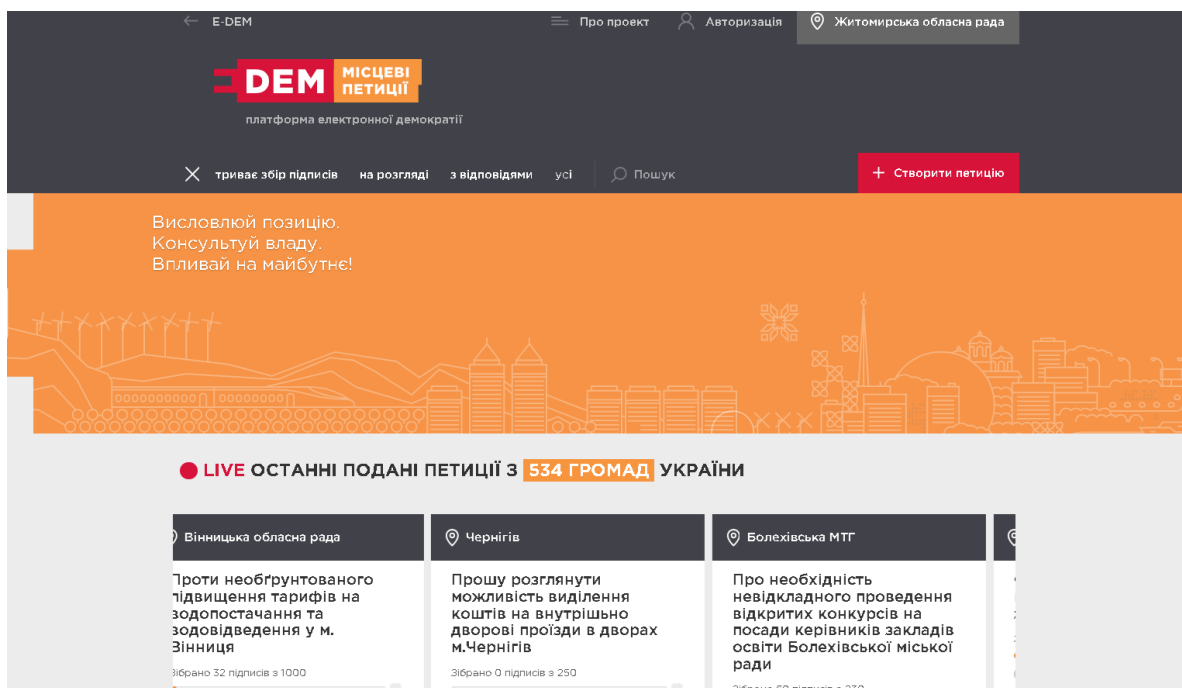


Рис. 1.2 – Інтерфейс сервісу e-DEM «Місцеві петиції»

		Чмут Д.К.			ПІЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				17
Змн.	Арк.	№ докум.	Підпис	Дата		

Третім аналогом є Контактний центр «15-51», який використовується органами місцевого самоврядування для централізованого приймання, реєстрації та опрацювання звернень громадян ([Рис.1.3](#)). Основним призначенням сервісу є забезпечення оперативної взаємодії між мешканцями міста та комунальними підприємствами, виконавчими органами влади й іншими відповідальними установами. За допомогою контактного центру користувачі можуть повідомляти про аварійні ситуації, проблеми благоустрою, житлово-комунального господарства, дорожньої інфраструктури, транспорту, освітлення та інші питання, що потребують реагування відповідних служб.

Звернення можуть подаватися через вебпортал, мобільний застосунок, телефонну гарячу лінію або інші канали комунікації. Після реєстрації кожному повідомленню присвоюється унікальний номер, за допомогою якого заявник може відстежувати стан його розгляду. У процесі опрацювання звернення відповідальний виконавець отримує необхідну інформацію, після чого змінює статус виконання та вносить результати виконаних робіт. Такий підхід дозволяє систематизувати роботу зі зверненнями та забезпечити контроль за їх виконанням.

До основних переваг контактного центру належать централізований облік звернень, підтримка декількох каналів подання повідомлень, можливість контролю статусу виконання заявки, розподіл звернень між відповідальними службами та накопичення статистичної інформації щодо результатів їх розгляду. Використання єдиної інформаційної системи дозволяє підвищити ефективність взаємодії між громадянами та виконавчими органами, а також скоротити час реагування на повідомлення про проблеми міської інфраструктури.

Водночас сервіс має і певні недоліки. Значна частина функціональних можливостей орієнтована насамперед на внутрішню роботу операторів і відповідальних служб, тоді як інструменти взаємодії між користувачами залишаються обмеженими. У більшості реалізацій відсутня можливість коментування звернень, підтримки повідомлень іншими користувачами, формування рейтингу актуальних проблем або ведення відкритого громадського

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				18
Змн.	Арк.	№ докум.	Підпис	Дата		

обговорення. Крім цього, функціональність контактних центрів може відрізнятися залежно від конкретної громади, що ускладнює використання єдиного підходу до організації роботи зі зверненнями.

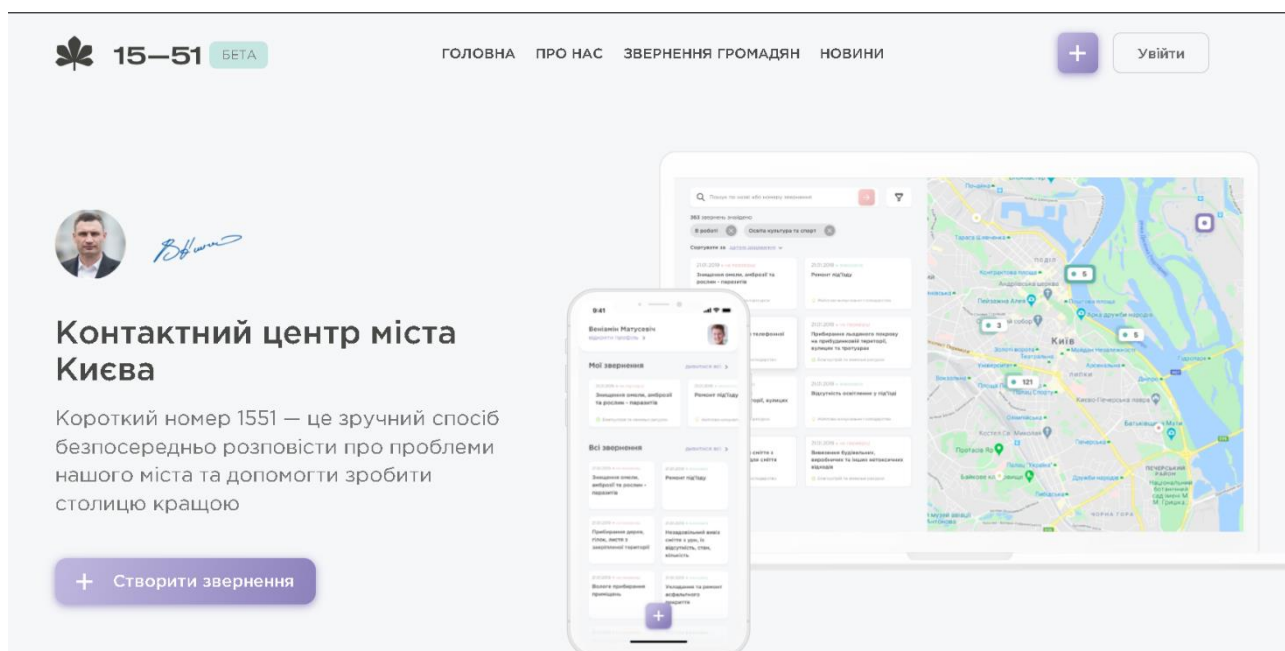


Рис 1.3 – Інтерфейс вебпорталу Контактного центру «15-51»

Після проведення аналізу Єдиного вебпорталу звернень громадян, e-DEM «Місцеві петиції» та Контактного центру «15-51» було визначено основні функціональні можливості, які доцільно врахувати під час розробки вебплатформи CityVoice. Насамперед до них належать можливість електронного подання звернень, централізоване зберігання інформації, контроль статусу виконання, підтримка рольової моделі користувачів, а також використання зручного інтерфейсу для взаємодії між громадянами та відповідальними службами.

Разом із тим проведений аналіз показав, що існуючі програмні рішення мають певні функціональні обмеження. Частина сервісів спеціалізується виключно на роботі із зверненнями, інші — лише на електронних петиціях або прийманні повідомлень через контактні центри. Крім того, у більшості аналогів недостатньо реалізовані сучасні механізми взаємодії між користувачами, персоналізація інформації, можливість відкритого обговорення проблем та інтеграція всіх необхідних функцій у межах єдиної вебплатформи.

		Чмут Д.К.			ПІЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				19
Змн.	Арк.	№ докум.	Підпис	Дата		

Після проведення аналізу Єдиного вебпорталу звернень громадян, e-DEM «Місцеві петиції» та Контактного центру 15-51 було визначено їх основні функціональні можливості, переваги й недоліки. На основі отриманих результатів виконано порівняння розглянутих програмних рішень за критеріями, що є найбільш важливими під час розробки системи. Результати порівняння наведено в (Табл. 1.1).

Таблиця 1.1.

Порівняння аналогів вебплатформи звернень громадян

Характеристика	Єдиний вебпортал звернень громадян	e-DEM «Місцеві петиції»	Контактний центр 15-51	Розроблювана вебплатформа CityVoice
Реєстрація та авторизація користувачів	+	+	+	+
Подання звернень	+	—	+	+
Додавання фотографій	+	—	+	+
Категоризація звернень	+	—	+	+
Відображення статусів звернень	+	—	+	+
Коментування звернень	—	+	—	+
Голосування	—	+	—	+
Сповіщення користувачів	+	—	+	+
Розмежування ролей користувачів	+	—	—	+
Панель адміністратора та модератора	+	—	—	+

Проведене порівняння показало, що кожна із розглянутих систем реалізує окремі функції, необхідні для організації електронної взаємодії між громадянами та органами державної влади або місцевого самоврядування. Водночас жодне з проаналізованих програмних рішень не забезпечує комплексного підходу до подання, опрацювання та контролю виконання звернень громадян у межах єдиної інформаційної системи.

З огляду на результати проведеного аналізу під час розробки вебплатформи було прийнято рішення поєднати переваги існуючих аналогів і доповнити їх сучасними функціями, що забезпечують більш ефективну взаємодію між користувачами та відповідальними особами. До таких функцій належать коментування звернень, голосування за актуальні проблеми, система сповіщень, гнучке керування ролями користувачів, а також окремі панелі адміністратора й модератора. Реалізація зазначених можливостей дозволить підвищити ефективність роботи системи, забезпечити прозорість процесу розгляду звернень і покращити взаємодію між громадянами та органами місцевого самоврядування.

1.3 Обґрунтування вибору архітектури вебплатформи CityVoice

Вибір архітектури є одним із найважливіших етапів проєктування програмного забезпечення, оскільки саме архітектурні рішення визначають структуру системи, принципи взаємодії між її компонентами, продуктивність, масштабованість, надійність та зручність подальшого супроводу. Для

Розроблюваної вебплатформи, призначеної для подання, обробки та контролю виконання звернень громадян, архітектура повинна забезпечувати стабільну роботу системи, швидку обробку запитів, централізоване зберігання інформації та можливість подальшого розширення функціональних можливостей.

Монолітна архітектура ([Рис.1.4](#)) передбачає реалізацію всіх функціональних компонентів програмної системи у межах єдиного програмного застосунку. У такій архітектурі інтерфейс користувача, бізнес-логіка та рівень

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		21

доступу до даних працюють як єдина система та використовують спільну базу даних.

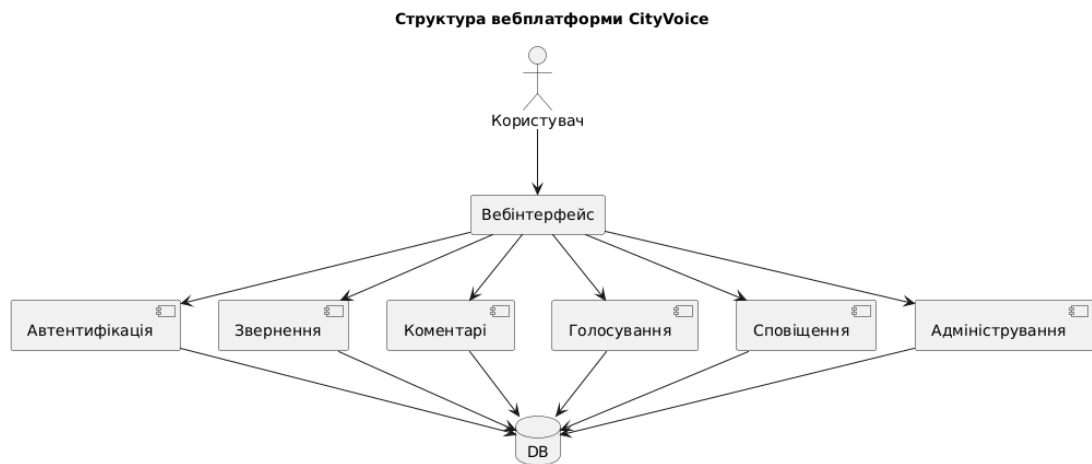


Рис. 1.4 – Діаграма монолітної архітектури

Основними перевагами монолітної архітектури є простота реалізації, централізоване керування програмним кодом, відсутність складної взаємодії між окремими сервісами, зручність тестування та розгортання системи. Завдяки використанню єдиної кодової бази значно спрощується процес розробки, а також забезпечується висока швидкість взаємодії між компонентами без використання мережових викликів.

До недоліків монолітної архітектури належать складність масштабування окремих компонентів, підвищена залежність між модулями та можливість впливу помилки в одному компоненті на роботу всієї системи. Проте для інформаційних систем середнього масштабу зазначені обмеження не є критичними й можуть бути мінімізовані шляхом використання модульної структури програмного коду.

Сервісно-орієнтована архітектура (SOA) ([Рис. 1.5](#)) ґрунтується на розподілі системи на окремі сервіси, кожен із яких реалізує певну бізнес-функцію. Обмін даними між сервісами здійснюється через стандартизовані протоколи та інтеграційний шар, що забезпечує взаємодію незалежних компонентів системи.



Рис. 1.5 – Діаграма сервісно-орієнтованої архітектури

Використання SOA забезпечує високу модульність програмного забезпечення, можливість інтеграції з іншими інформаційними системами та повторне використання окремих сервісів. Такий підхід широко застосовується під час розробки великих корпоративних інформаційних систем, які об'єднують значну кількість незалежних підсистем.

Разом із тим сервісно-орієнтована архітектура характеризується підвищеною складністю реалізації, необхідністю використання додаткових механізмів інтеграції, збільшенням витрат на супровід та адміністрування системи. Найбільш доцільним є використання такого підходу під час створення великих корпоративних інформаційних систем, які об'єднують значну кількість незалежних сервісів та потребують складної інтеграції між окремими компонентами.

Мікросервісна архітектура ([Рис. 1.6](#)) передбачає поділ програмної системи на незалежні сервіси, кожен із яких виконує окрему бізнес-функцію та може розроблятися, розгортатися й масштабуватися незалежно від інших компонентів.

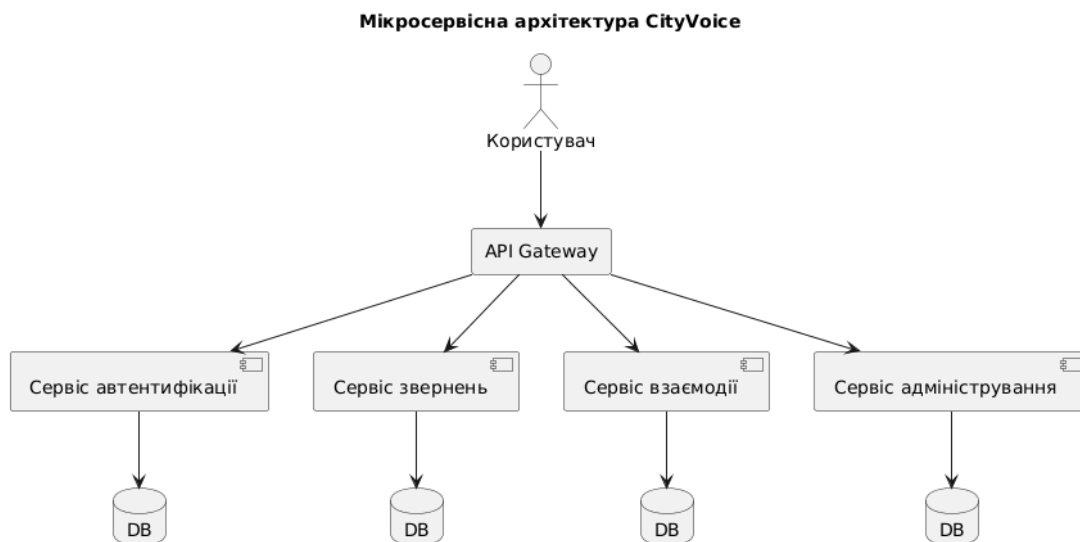


Рис. 1.6 – Діаграма мікросервісної архітектури

Основними перевагами мікросервісної архітектури є висока масштабованість, ізоляція відмов окремих компонентів, можливість незалежного оновлення сервісів та ефективне використання обчислювальних ресурсів за умов значного навантаження.

Водночас такий підхід потребує використання складної серверної інфраструктури, організації взаємодії між сервісами, реалізації механізмів синхронізації даних, моніторингу та забезпечення відмовостійкості. Крім цього, збільшується складність тестування, адміністрування та підтримки програмного забезпечення. Незважаючи на значні переваги щодо масштабованості та відмовостійкості, використання мікросервісної архітектури найбільш доцільне для великих високонавантажених інформаційних систем, які потребують незалежного масштабування окремих компонентів.

Проведений аналіз основних архітектурних підходів дозволив визначити найбільш доцільне рішення для реалізації розроблюваної вебплатформи. З урахуванням функціональних вимог, масштабу проєкту та особливостей предметної області було обрано монолітну архітектуру з модульною організацією.

Розроблювана вебплатформа містить взаємопов'язані функціональні модулі, серед яких модулі автентифікації користувачів, керування зверненнями, категоріями, коментарями, голосуванням, сповіщеннями, а також панелі

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				24
Змн.	Арк.	№ докум.	Підпис	Дата		

адміністратора й модератора. Реалізація зазначених компонентів у межах єдиного програмного застосунку спрощує їх взаємодію, полегшує розробку, тестування та подальший супровід програмного забезпечення [30].

Використання єдиної бази даних гарантує цілісність інформації про користувачів, звернення, коментарі, голосування та інші об'єкти предметної області. Крім того, відсутність необхідності організації міжсервісної взаємодії позитивно впливає на продуктивність системи, спрощує її розгортання та зменшує витрати на адміністрування.

Важливою перевагою обраної архітектури є можливість подальшого розвитку програмного продукту. У разі збільшення кількості користувачів або розширення функціональних можливостей окремі модулі можуть бути модернізовані чи винесені в окремі сервіси без необхідності повного перепроєктування всієї системи.

Таким чином, монолітна архітектура з модульною організацією є оптимальним рішенням для розроблюваної вебплатформи [25], оскільки повністю відповідає її функціональним вимогам, забезпечує високу продуктивність, простоту реалізації та супроводу, а також створює передумови для подальшого розвитку й масштабування програмного продукту.

1.4 Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення

Одним із важливих етапів розробки програмного забезпечення є вибір інструментальних засобів, які забезпечують реалізацію функціональних вимог, стабільність роботи системи та можливість її подальшого розвитку. Від правильного вибору мов програмування, фреймворків, систем керування базами даних та допоміжних програмних засобів залежить продуктивність вебплатформи, рівень її безпеки, зручність супроводу й ефективність процесу розробки.

Під час створення вебплатформи було обрано сучасний стек технологій, який дозволяє реалізувати необхідний функціонал, забезпечити взаємодію між

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				25
Змн.	Арк.	№ докум.	Підпис	Дата		

користувачами та серверною частиною, а також організувати надійне зберігання й обробку даних. Вибір кожного програмного засобу здійснювався з урахуванням функціональних вимог до системи, особливостей предметної області, сумісності між окремими компонентами та можливості їх подальшого використання під час модернізації програмного продукту.

Для реалізації серверної частини вебплатформи було використано фреймворк Laravel [2], що працює на основі мови програмування PHP [3]. Зберігання інформації здійснюється за допомогою реляційної системи керування базами даних MySQL [4], а для створення користувацького інтерфейсу застосовано HTML [13], CSS [11], Bootstrap [7] та JavaScript [12, 13]. Крім того, під час розробки використовувалися допоміжні інструменти, серед яких шаблонізатор Blade, менеджер залежностей Composer [8] та система контролю версій Git, що забезпечують ефективну організацію процесу розробки й підтримки програмного забезпечення.

Під час розробки серверної частини вебплатформи було обрано фреймворк Laravel [2], який є одним із найпоширеніших сучасних PHP-фреймворків із відкритим вихідним кодом. Він реалізує архітектурний шаблон Model–View–Controller (MVC), що забезпечує чіткий розподіл програмного коду між рівнями представлення, бізнес-логіки та роботи з даними [24]. Такий підхід спрощує розробку програмного забезпечення, підвищує читабельність програмного коду та полегшує його подальший супровід і модернізацію.

Однією з головних переваг є наявність великої кількості вбудованих інструментів, які дозволяють значно скоротити час розробки вебзастосунків. До них належать система маршрутизації, механізм автентифікації користувачів, підтримка middleware, засоби валідації даних, система міграцій бази даних, шаблонізатор Blade, ORM Eloquent, а також вбудовані механізми захисту від найбільш поширених вебзагроз. Використання зазначених компонентів дозволяє реалізувати основний функціонал програмного продукту без необхідності розробки великої кількості допоміжного програмного коду.

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				26
Змн.	Арк.	№ докум.	Підпис	Дата		

Важливою перевагою є підтримка механізму ORM Eloquent, який забезпечує об'єктно-реляційне відображення даних. Завдяки цьому взаємодія з базою даних здійснюється через програмні моделі без необхідності постійного використання SQL-запитів. Такий підхід підвищує зрозумілість програмного коду, спрощує його підтримку та зменшує ймовірність виникнення помилок під час роботи з даними.

Для розроблюваної вебплатформи використання є доцільним також завдяки вбудованим механізмам забезпечення безпеки. Фреймворк підтримує захист від SQL-ін'єкцій, міжсайтового виконання сценаріїв (XSS), підробки міжсайтових запитів (CSRF), а також забезпечує безпечне зберігання паролів користувачів із використанням сучасних алгоритмів хешування. Наявність цих механізмів дозволяє підвищити рівень захисту персональних даних та забезпечити безпечну роботу вебплатформи.

У розроблюваному програмному продукті використовується для реалізації всіх основних функціональних можливостей системи, серед яких реєстрація та автентифікація користувачів, створення й опрацювання звернень, керування категоріями та статусами, додавання коментарів, голосування за звернення, надсилання сповіщень, а також реалізація функціоналу адміністратора й модератора. Використання єдиного фреймворку для реалізації всіх компонентів системи забезпечує узгодженість програмного коду, спрощує інтеграцію між окремими модулями та створює умови для подальшого розвитку вебплатформи.

Основною мовою програмування серверної частини вебплатформи було обрано PHP [3], яка є однією з найпоширеніших мов розробки вебзастосунків. Її популярність зумовлена простотою використання, високою продуктивністю, широкою підтримкою з боку спільноти розробників та сумісністю з більшістю сучасних вебсерверів і систем керування базами даних. Крім того, саме він є основою для роботи фреймворку Laravel, що забезпечує повну сумісність між усіма компонентами програмного забезпечення.

Використання дозволяє ефективно реалізовувати серверну логіку вебплатформи, обробляти HTTP-запити, виконувати перевірку введених даних,

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				27
Змн.	Арк.	№ докум.	Підпис	Дата		

взаємодіяти з базою даних та формувати динамічний вміст вебсторінок. Мова підтримує об'єктно-орієнтоване програмування, що дає можливість створювати модульні програмні рішення, повторно використовувати програмний код і спрощує подальшу підтримку програмного продукту.

Важливою перевагою сучасних версій є підвищена продуктивність, оптимізоване використання оперативної пам'яті та підтримка сучасних мовних конструкцій, які дозволяють створювати безпечні й ефективні вебзастосунки. Завдяки цьому розроблювана вебплатформа здатна швидко обробляти запити користувачів, забезпечувати стабільну роботу та підтримувати одночасну взаємодію значної кількості користувачів.

Для зберігання інформації у вебплатформі використовується реляційна система керування базами даних MySQL [4], яка є одним із найбільш поширених рішень для створення вебзастосунків. Її вибір зумовлений високою швидкістю, надійністю, підтримкою механізмів забезпечення цілісності даних та повною інтеграцією з фреймворком Laravel.

Реляційна модель даних дозволяє ефективно організувати взаємозв'язки між основними сутностями предметної області, серед яких користувачі, звернення, категорії, коментарі, голосування, сповіщення та інші службові дані. Використання зовнішніх ключів забезпечує підтримку зв'язків між таблицями, а механізм транзакцій гарантує коректне виконання операцій під час додавання, редагування чи видалення інформації.

Для взаємодії з базою даних використовується ORM Eloquent, яка входить до складу Laravel. Це дозволяє працювати з даними за допомогою програмних моделей без необхідності написання великої кількості SQL-запитів. Такий підхід підвищує продуктивність розробки, покращує читабельність програмного коду та значно спрощує його подальший супровід.

У розроблюваній вебплатформі MySQL [4] використовується для централізованого зберігання інформації про користувачів, звернення, категорії, коментарі, результати голосування, повідомлення та інші дані, необхідні для

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				28
Змн.	Арк.	№ докум.	Підпис	Дата		

функціонування системи. Це забезпечує швидкий доступ до інформації, її цілісність і можливість ефективної обробки великих обсягів даних.

Для реалізації клієнтської частини вебплатформи використовуються сучасні вебтехнології HTML [10], CSS [11], Bootstrap [7] та JavaScript [12, 13], які забезпечують створення зручного, адаптивного та функціонального користувацького інтерфейсу. Використання зазначених технологій дозволяє забезпечити коректне відображення вебсторінок у сучасних браузерах, підвищити зручність роботи користувачів та покращити загальне сприйняття програмного продукту.

Мова розмітки HTML [10] використовується для формування структури вебсторінок і відображення інформації користувачам. За її допомогою реалізовано сторінки реєстрації та авторизації, створення й перегляду звернень, особистий кабінет користувача, адміністративну панель, форми введення даних та інші елементи інтерфейсу. Використання семантичних елементів HTML [10] сприяє підвищенню зрозумілості структури сторінок, покращує їх доступність і забезпечує коректну роботу в різних веббраузерах.

Для оформлення вебінтерфейсу використовується CSS [11], який забезпечує стилізацію сторінок, налаштування кольорової гами, шрифтів, відступів, анімації та інших елементів дизайну. Застосування каскадних таблиць стилів дозволяє відокремити логіку відображення від структури сторінки, що спрощує внесення змін до дизайну без необхідності модифікації HTML-коду [10].

З метою прискорення процесу розробки та забезпечення адаптивності інтерфейсу використовується фреймворк Bootstrap [7]. Він містить велику кількість готових компонентів інтерфейсу, серед яких навігаційні панелі, кнопки, таблиці, форми, картки, модальні вікна та інші елементи. Крім цього, фреймворк реалізує адаптивну сітку, завдяки якій вебплатформа коректно відображається на персональних комп'ютерах, планшетах і мобільних пристроях без необхідності створення окремих версій інтерфейсу.

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				29
Змн.	Арк.	№ докум.	Підпис	Дата		

Для реалізації інтерактивних можливостей вебплатформи використовується JavaScript [12, 13]. За допомогою цієї мови програмування забезпечується динамічна взаємодія користувача зі сторінками без необхідності їх повного перезавантаження, перевірка коректності введених даних, відображення повідомлень, керування окремими елементами інтерфейсу та виконання інших клієнтських сценаріїв. Використання JavaScript [12, 13] підвищує швидкість взаємодії користувача із системою, покращує зручність роботи та робить інтерфейс більш сучасним і функціональним.

Під час розробки вебплатформи також використовувалися допоміжні програмні засоби, які забезпечують ефективну організацію програмного коду, спрощують процес розробки та полегшують подальшу підтримку програмного забезпечення. До таких засобів належать шаблонізатор Blade, менеджер залежностей Composer [8] та система контролю версій Git.

Шаблонізатор Blade, який входить до складу фреймворку Laravel, використовується для формування користувацького інтерфейсу вебплатформи. Він дозволяє поєднувати HTML-розмітку [10] з серверною логікою, використовуючи спеціальні директиви та шаблони. Завдяки цьому значно зменшується дублювання програмного коду, а спільні елементи інтерфейсу, такі як меню навігації, верхня панель, підвал сторінки та інші компоненти, можуть повторно використовуватися на різних сторінках вебплатформи. Такий підхід спрощує внесення змін до дизайну та підвищує зручність супроводу програмного продукту.

Для керування бібліотеками та зовнішніми пакетами використовується Composer [8] — стандартний менеджер залежностей для PHP. Його застосування дозволяє автоматизувати процес встановлення, оновлення та керування сторонніми компонентами, необхідними для роботи вебплатформи. Крім того, він забезпечує автоматичне завантаження класів і підтримує єдину структуру програмного проєкту, що позитивно впливає на організацію програмного коду та спрощує процес його подальшого розвитку.

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				30
Змн.	Арк.	№ докум.	Підпис	Дата		

Для керування вихідним кодом застосовується система контролю версій Git, яка забезпечує збереження історії змін, контроль версій програмного забезпечення та можливість повернення до попередніх станів проєкту. Використання Git дозволяє впорядкувати процес розробки, контролювати внесені зміни та значно спрощує подальшу модернізацію програмного продукту. Крім цього, система контролю версій є ефективним засобом резервного зберігання програмного коду та підтримує можливість командної розробки [16]. Для зберігання репозиторіїв та організації командної роботи використовувалась платформа GitHub [17].

Як основне середовище розробки використовувався редактор Visual Studio Code [18], який забезпечує підсвічування синтаксису, налагодження коду та інтеграцію з системою контролю версій Git.

Для стабільної роботи розроблюваної вебплатформи необхідно забезпечити відповідні програмні та апаратні ресурси як на стороні сервера, так і на стороні користувача. Оскільки система реалізована як монолітний вебзастосунок із використанням фреймворку Laravel та системи керування базами даних, вимоги до серверного обладнання є помірними та не потребують використання високопродуктивної спеціалізованої інфраструктури.

Для розгортання серверної частини вебплатформи рекомендується використовувати сервер із сучасним багатоядерним процесором, оперативною пам'яттю обсягом не менше 4 ГБ та SSD-накопичувачем обсягом від 40 ГБ. Такі характеристики забезпечують стабільну роботу вебсервера, виконання серверної логіки, обробку запитів користувачів і взаємодію з базою даних без суттєвого зниження продуктивності.

Для роботи серверної частини необхідно встановити операційну систему, яка підтримує виконання PHP-застосунків, вебсервер Apache або Nginx, інтерпретатор PHP [3], систему керування базами даних MySQL [4], а також менеджер залежностей Composer [8]. Сукупність зазначених програмних компонентів забезпечує коректне функціонування вебплатформи та можливість її подальшого супроводу.

		Чмут Д.К.			ПІЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				31
Змн.	Арк.	№ докум.	Підпис	Дата		

На стороні користувача спеціальних вимог до апаратного забезпечення не передбачається. Для роботи із вебплатформою достатньо персонального комп'ютера, ноутбука, планшета або смартфона, підключеного до мережі Інтернет, із встановленим сучасним веббраузером, що підтримує сучасні вебстандарті. Завдяки використанню адаптивного інтерфейсу система забезпечує коректне відображення сторінок на пристроях із різними розмірами екранів та різними операційними системами.

Використання веборієнтованої архітектури дозволяє користувачам працювати із системою без встановлення додаткового програмного забезпечення. Для доступу до функціональних можливостей вебплатформи достатньо перейти за відповідною вебадресою через браузер, що значно спрощує використання системи та забезпечує її доступність для широкого кола користувачів.

Висновки до першого розділу

У першому розділі було проаналізовано предметну область та обґрунтовано актуальність розробки вебплатформи для подання й опрацювання звернень громадян щодо проблем міської інфраструктури. Визначено основні функціональні вимоги до системи, сформульовано мету розробки та окреслено перелік завдань, необхідних для її реалізації.

Проведений аналіз існуючих програмних рішень дозволив визначити їх функціональні можливості, переваги та недоліки. На основі отриманих результатів сформовано вимоги до розроблюваної вебплатформи CityVoice, яка поєднує механізми подання звернень, коментування, голосування, зміни статусів, надсилання сповіщень, а також засоби адміністрування й модерації в межах єдиної інформаційної системи.

У результаті аналізу сучасних архітектурних підходів було обґрунтовано вибір монолітної архітектури з модульною організацією, яка найбільш повно відповідає функціональним вимогам проєкту, забезпечує простоту реалізації, ефективність супроводу та можливість подальшого розвитку програмного продукту.

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				32
Змн.	Арк.	№ докум.	Підпис	Дата		

Також було обґрунтовано вибір інструментальних засобів розробки, серед яких фреймворк Laravel, мова програмування PHP, система керування базами даних MySQL, а також HTML, CSS, Bootstrap і JavaScript, для реалізації користувацького інтерфейсу. Використання зазначених технологій забезпечує створення сучасної, продуктивної та безпечної вебплатформи, здатної ефективно підтримувати процес подання, опрацювання й контролю виконання звернень громад

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		33

РОЗДІЛ 2. ПРОЕКТУВАННЯ СИСТЕМИ

2.1 Визначення варіантів використання та об'єктної структури системи

Проектування вебплатформи CityVoice розпочинається з визначення основних сценаріїв використання, які описують взаємодію користувачів із системою. Аналіз таких сценаріїв дає змогу сформулювати функціональні вимоги до вебплатформи, визначити права доступу для різних категорій користувачів і встановити послідовність виконання основних операцій під час роботи із системою.

У вебплатформі передбачено чотири основні ролі: гість, користувач, модератор та адміністратор. Кожна із зазначених ролей має визначений набір функціональних можливостей відповідно до покладених на неї обов'язків. Гість може переглядати загальнодоступну інформацію та виконувати реєстрацію або авторизацію. Авторизований користувач отримує можливість створювати звернення, додавати фотографії, залишати коментарі, голосувати за звернення та отримувати сповіщення. Модератор здійснює перевірку звернень, зміну їх статусів і формування звітів, тоді як адміністратор відповідає за керування користувачами, ролями, категоріями та загальне адміністрування вебплатформи.

Для наочного відображення взаємодії користувачів із системою побудовано діаграму варіантів використання, наведену на ([Рис. 2.1](#)).

Основні варіанти використання вебплатформи охоплюють процеси реєстрації та авторизації користувачів, перегляду та пошуку звернень, створення нових звернень, додавання фотографій, коментування, голосування та отримання сповіщень. Для службових ролей передбачено можливість зміни статусів звернень, перегляду архіву, формування PDF-звітів, а також керування користувачами, ролями та категоріями.

		Чмут Д.К.			ІПЗ.КР.Б – 121 – 26 – ІЗ	Арк.
		Єфремов. Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		34

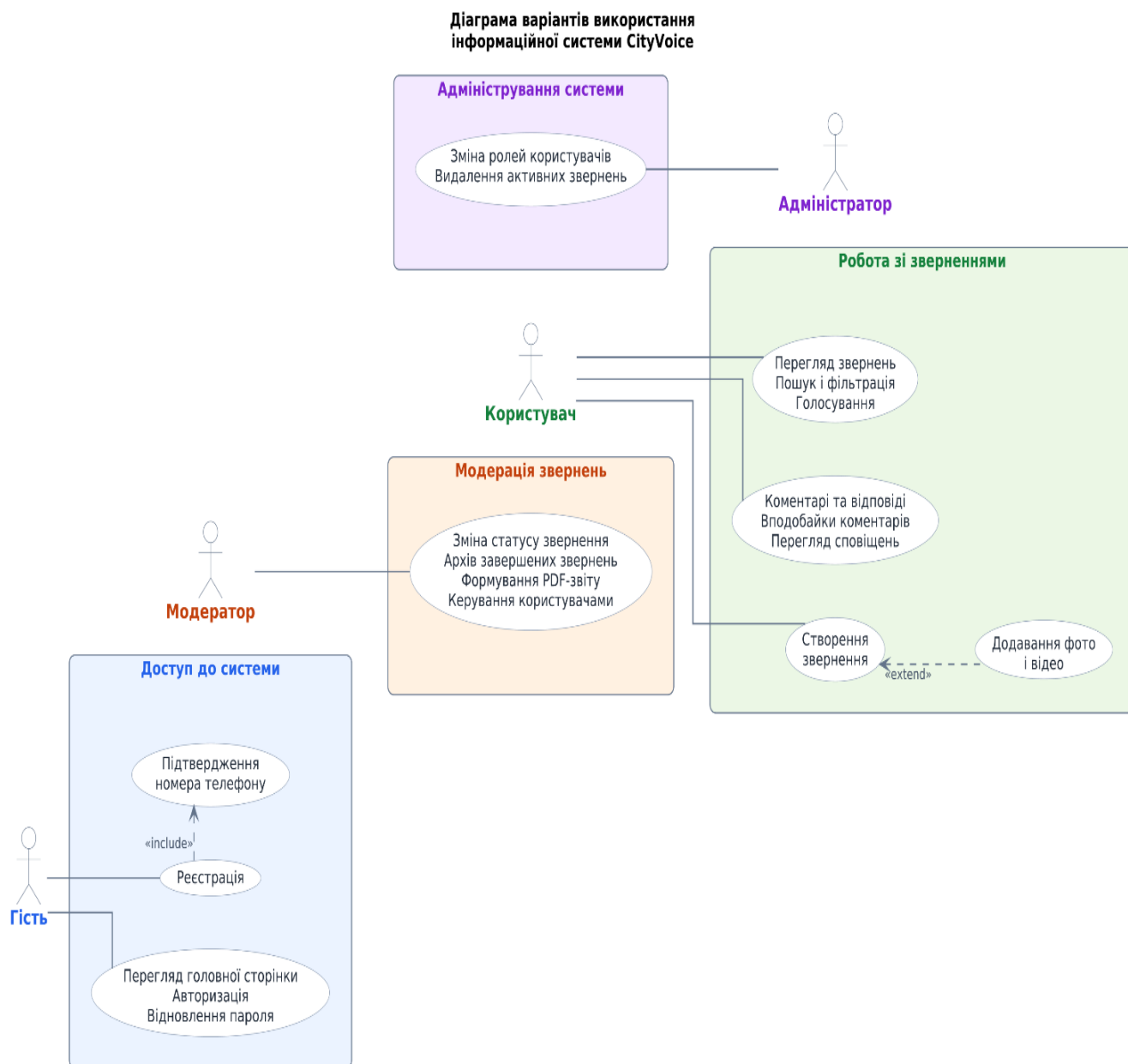


Рис.2.1 – Діаграма варіантів використання вебплатформи

На діаграмі відображено взаємозв'язок між ролями користувачів та основними сценаріями роботи вебплатформи. Вона наочно демонструє, які операції доступні гостю, авторизованому користувачу, модератору та адміністратору, а також дозволяє оцінити розподіл функціональних можливостей між окремими ролями.

Детальніший перелік доступних операцій для кожної ролі наведено у матриці доступу ([Табл. 2.1](#)).

Таблиця 2.1

Матриця доступу учасників до основних функцій

Функція	Гість	Користувач	Модератор	Адміністратор
Перегляд головної сторінки	+	+	+	+
Реєстрація та вхід	+	—	—	—
Створення громадського звернення	—	+	—	—
Перегляд списку звернень	—	+	+	+
Пошук і фільтрація	—	+	+	+
Голосування	—	+	—	—
Коментарі та відповіді	—	+	—	—
Вподобайки коментарів	—	+	—	—
Зміна статусу звернення	—	—	+	+
Перегляд архіву завершених звернень	—	—	+	+
Формування PDF-звіту	—	—	+	+
Видалення незавершеного звернення через панель	—	—	—	+
Видалення звичайного користувача	—	—	+	+
Видалення модератора	—	—	—	+
Зміна ролі User/Moderator	—	—	—	+
Самостійне видалення облікового запису	—	+	—	—

Умовні позначення: «+» — функція доступна для відповідної ролі; «—» — функція недоступна.

Наведена матриця демонструє розподіл функціональних можливостей між різними категоріями користувачів. Кожна роль має визначений набір прав доступу, що відповідає її призначенню та забезпечує безпечну й ефективну

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				36
Змн.	Арк.	№ докум.	Підпис	Дата		

взаємодію із системою. На основі визначених функціональних можливостей сформовано основні сценарії використання вебплатформи, опис яких наведено далі.

Таблиця 2.2

Реєстрація та підтвердження телефону

Назва	Реєстрація та підтвердження телефону
Основні актори	Гість, користувач
Короткий опис	Гість вводить прізвище, ім'я, по батькові, email, український номер телефону та пароль. Після перевірки унікальності система створює обліковий запис, виконує вхід і формує шестизначний код підтвердження.
Результат	Номер телефону підтверджено, а користувач отримав доступ до захищених розділів.

Реєстрація забезпечує створення нового облікового запису користувача та надання доступу до функціональних можливостей вебплатформи ([Табл. 2.2](#)). Після успішного введення необхідних даних інформація зберігається у базі даних, а користувач отримує можливість виконати авторизацію.

Таблиця 2.3

Авторизація користувача

Поле	Значення
Назва	Авторизація користувача
Основні актори	Користувач, модератор, адміністратор
Передумова	Обліковий запис уже створений.
Короткий опис	Учасник вводить email і пароль. Система перевіряє облікові дані, створює сесію та надає доступ до функцій відповідно до ролі.
Результат	Виконано вхід до системи.

Авторизація забезпечує ідентифікацію користувача та надання доступу до функцій системи відповідно до його ролі ([Табл. 2.3](#)). Після успішної перевірки

облікових даних користувач переходить до особистого кабінету або відповідної панелі керування.

Таблиця 2.4

Перегляд головної сторінки

Поле	Значення
Назва	Авторизація користувача
Основні актори	Гість, користувач, модератор, адміністратор
Короткий опис	Користувач відкриває головну сторінку вебплатформи та переглядає інформацію про її призначення, останні звернення, статистику та навігаційні елементи.
Результат	Відображено головну сторінку вебплатформи CityVoice.

Перегляд головної сторінки є початковим сценарієм взаємодії користувача з вебплатформою (Табл. 2.4). На головній сторінці відображається основна інформація про систему, останні звернення громадян, елементи навігації

Після ознайомлення з інформацією користувач може перейти до перегляду окремих звернень або скористатися іншими функціями вебплатформи відповідно до своїх потреб. Це забезпечує зручний початок роботи із системою та швидкий доступ до її основних можливостей. Інтерфейс головної сторінки розроблено з урахуванням простоти та зручності використання.

Таблиця 2.5

Перегляд звернень

Поле	Значення
Назва	Перегляд звернень
Основні актори	Гість, користувач, модератор, адміністратор
Короткий опис	Перегляд списку звернень, їх опису, категорії, статусу та іншої інформації.
Результат	Відображено перелік звернень.

Перегляд звернень дозволяє користувачам ознайомитися з інформацією про вже зареєстровані проблеми, їх статус, категорію та місце розташування (Табл. 2.5).

Це забезпечує відкритість інформації та сприяє контролю за процесом опрацювання звернень.

Таблиця 2.6

Створення звернення

Поле	Значення
Назва	Створення звернення
Основні актори	Користувач
Короткий опис	Користувач створює нове звернення, заповнює необхідні поля, зазначає категорію, адресу та за потреби додає фотографії.

Створення звернення є основною функціональною можливістю вебплатформи (Табл. 2.6). Під час формування звернення користувач зазначає необхідну інформацію про проблему та, за потреби, додає фотографії для її наочного підтвердження.

Таблиця 2.7

Коментування звернення

Поле	Значення
Назва	Коментування звернення
Основні актори	Користувач
Короткий опис	Користувач залишає коментар до звернення для уточнення інформації або обговорення проблеми.
Результат	Коментар успішно додано.

Коментування забезпечує можливість обговорення звернень між користувачами та уточнення додаткової інформації щодо виявленої проблеми (Табл. 2.7). Це сприяє більш повному опису ситуації та підвищує ефективність її подальшого опрацювання.

Таблиця 2.8

Голосування за звернення

Поле	Значення
Назва	Голосування за звернення
Основні актори	Користувач
Передумова	Користувач авторизований і підтвердив телефон.
Короткий опис	Користувач указує назву, опис проблеми та категорію. Система перевіряє введені дані, присвоює статус «Нове» і зберігає запис у базі даних.
Результат	Нове звернення з'явилося в загальному списку й особистому кабінеті автора.

Механізм голосування дозволяє користувачам підтримувати звернення, які вони вважають найбільш актуальними ([Табл. 2.8](#)). Кількість отриманих голосів може використовуватися як один із критеріїв визначення пріоритетності розгляду проблем. Інформація про кількість голосів відображається безпосередньо на сторінці звернення та оновлюється після взаємодії користувачів із системою. Це дає можливість оцінити рівень підтримки кожного звернення та сприяє більшій відкритості процесу його розгляду. Використання такого механізму підвищує активність користувачів і заохочує їх брати участь у вирішенні актуальних проблем громади.

Таблиця 2.9

Зміна статусу звернення

Поле	Значення
Назва	Додавання фото та відео
Основні актори	Модератор, адміністратор
Короткий опис	Відповідальна особа змінює статус звернення відповідно до етапу його розгляду.
Результат	Статус звернення успішно оновлено.

Зміна статусу звернення здійснюється відповідальною особою після перевірки інформації або виконання відповідних робіт ([Табл. 2.9](#)). Завдяки цьому користувачі можуть відстежувати поточний стан розгляду власних звернень

Таблиця 2.10

Формування PDF-звіту

Поле	Значення
Назва	Формування PDF-звіту
Основні актори	Модератор, адміністратор
Короткий опис	Формування звіту про звернення у форматі PDF.
Результат	PDF-звіт успішно сформовано.

Формування звітів у форматі PDF забезпечує можливість експорту інформації про звернення для подальшого аналізу, друку або передачі відповідальним особам ([Табл. 2.10](#)). Це спрощує документування результатів роботи вебплатформи.

Таблиця 2.11

Керування користувачами

Поле	Значення
Назва	Керування користувачами
Основні актори	Адміністратор
Короткий опис	Адміністратор переглядає, редагує, блокує або видаляє облікові записи користувачів.
Результат	Дані користувачів успішно оновлено.

Функція керування користувачами призначена для адміністрування облікових записів та контролю доступу до функціональних можливостей системи ([Табл. 2.11](#)). Адміністратор може переглядати інформацію про користувачів, змінювати їхні ролі та виконувати інші адміністративні операції.

Наведені варіанти використання визначають основні сценарії взаємодії користувачів із вебплатформою та дозволяють сформувати перелік основних

сутностей предметної області. Кожна функція системи пов'язана з відповідними об'єктами, які використовуються під час зберігання та обробки інформації.

Для відображення структури предметної області, взаємозв'язків між її об'єктами та їхніх основних властивостей побудовано діаграму класів доменних моделей, наведену на (Рис. 2.2).

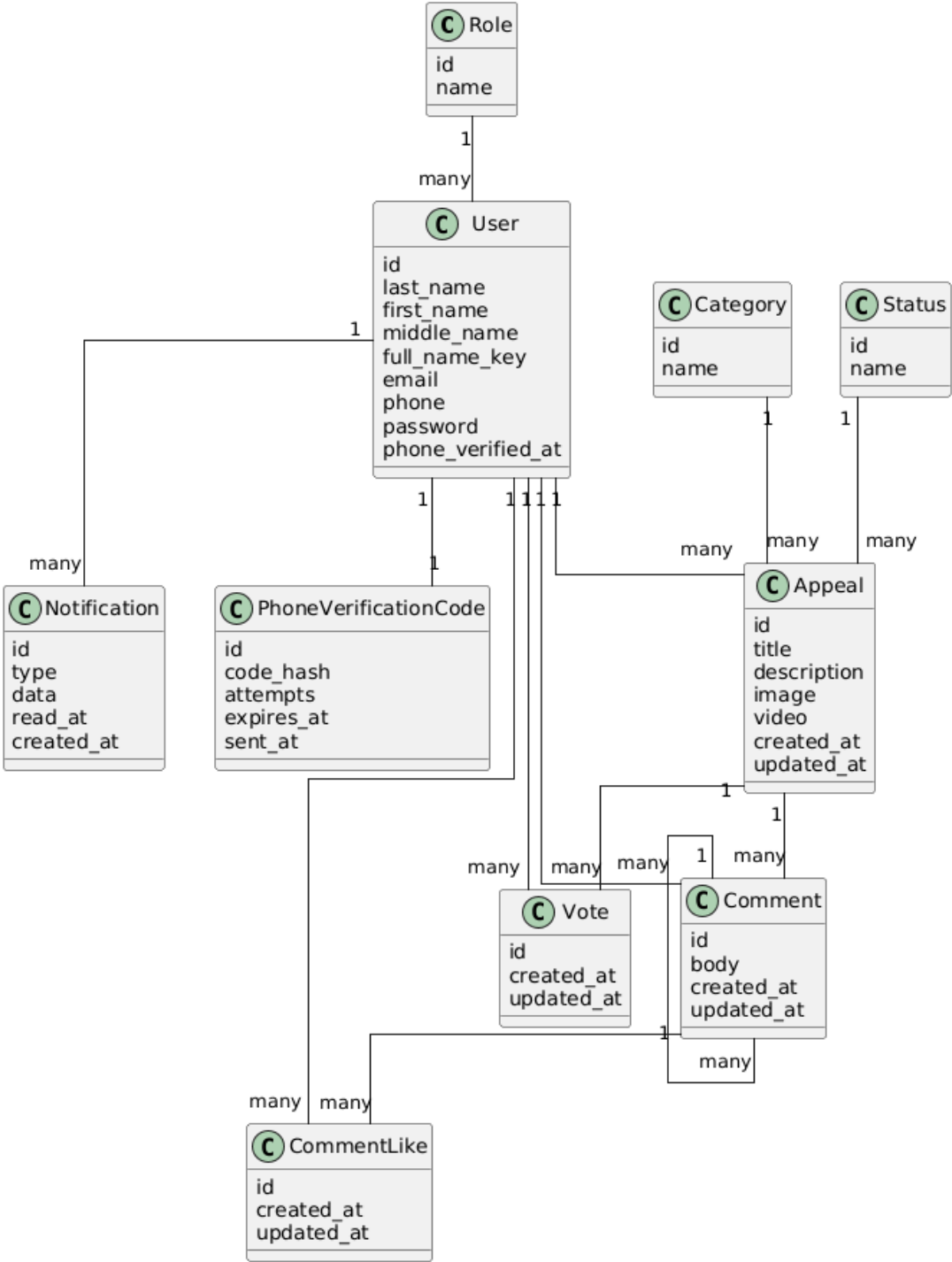


Рис.2.2 – Діаграма класів доменних моделей вебплатформи

Діаграма класів відображає основні сутності вебплатформи та взаємозв'язки між ними. Центальною сутністю системи є клас Appeal, який описує звернення користувача та містить інформацію про назву, опис проблеми, прикріплені медіафайли, автора, категорію та поточний статус.

Клас User відповідає за зберігання облікових і персональних даних користувача. Він пов'язаний із класом Role, який визначає рівень доступу користувача до функціональних можливостей системи. Залежно від ролі користувач може створювати звернення, залишати коментарі, голосувати або виконувати службові дії.

Для групування звернень за тематикою використовується клас Category, а для визначення етапу опрацювання звернення — клас Status. Таке розділення дозволяє уникнути дублювання текстових значень у зверненнях і забезпечує більш зручне керування довідковими даними.

Клас Comment використовується для зберігання коментарів до звернень і відповідей користувачів. Для реалізації механізму підтримки звернень застосовується клас Vote, який фіксує факт голосування користувача за конкретне звернення. Окремо виділено клас CommentLike, який забезпечує можливість вподобання коментарів.

Клас Notification призначений для зберігання системних сповіщень, що надсилаються користувачам після важливих подій, зокрема зміни статусу звернення або появи нового коментаря. Клас PhoneVerificationCode використовується для підтвердження номера телефону користувача під час реєстрації.

Таким чином, побудована діаграма класів відображає основні доменні моделі вебплатформи та логіку їх взаємодії. Структура класів розроблена відповідно до принципів об'єктно-орієнтованого проектування та загальноприйнятих патернів [26, 29]. Характеристику основних сутностей, їх призначення та роль у структурі системи наведено в (Табл. 2.12). Визначені класи забезпечують реалізацію ключових функціональних можливостей

		Чмут Д.К.			ІПЗ.КР.Б – 121 – 26 – ІЗ	Арк.
		Єфремов. Ю.М.				43
Змн.	Арк.	№ докум.	Підпис	Дата		

системи, зокрема реєстрацію користувачів, подання звернень, коментування, голосування, зміну статусів і надсилання сповіщень.

Таблиця 2.12

Характеристика основних сутностей

Сутність	Призначення
Role	Зберігає назву ролі User, Moderator або Admin і визначає рівень доступу користувача.
User	Містить облікові й персональні дані, номер телефону, ознаку його підтвердження та посилання на роль.
Appeal	Представляє звернення громадянина з назвою, описом, категорією, статусом, фото та відео.
Category	Групує звернення за тематикою та використовується у фільтрації.
Status	Визначає поточний стан звернення: Нове, В роботі, Завершено або Відхилено.
Vote	Зв'язує користувача зі зверненням і фіксує його підтримку.
Comment	Зберігає коментар або відповідь і підтримує зв'язок із користувачем-адресатом.
CommentLike	Фіксує одну вподобайку користувача до конкретного коментаря.
PhoneVerificationCode	Зберігає хеш одноразового коду, строк дії та кількість спроб.
Notification	Зберігає системні повідомлення про коментар, відповідь або зміну статусу.

2.2 Розробка бази даних системи

База даних є одним із найважливіших компонентів вебплатформи CityVoice, оскільки забезпечує централізоване зберігання інформації про користувачів, звернення, категорії, коментарі, голосування та інші об'єкти предметної області [21]. Від правильної організації структури бази даних

залежать продуктивність системи, цілісність інформації та швидкість виконання запитів.

Для реалізації вебплатформи було використано реляційну систему керування базами даних. Вибір цієї СУБД обумовлений її високою продуктивністю, підтримкою механізмів транзакцій, зовнішніх ключів, індексування та широким використанням під час розробки сучасних вебзастосунків. Використання MySQL [4] також повністю інтегрується з фреймворком Laravel через ORM Eloquent, що значно спрощує роботу з даними.

Структура бази даних сформована відповідно до функціональних вимог вебплатформи та побудованої діаграми класів. Основними сутностями є користувачі, ролі, звернення, категорії, статуси, коментарі, голосування, вподобання коментарів, сповіщення та коди підтвердження телефонних номерів. Кожна таблиця відповідає окремому об'єкту предметної області та містить лише необхідні для його опису атрибути.

Між таблицями реалізовано зв'язки типу «один до багатьох» та «один до одного» [22], що забезпечують логічну структуру даних і підтримують їхню цілісність. Для встановлення взаємозв'язків використовуються зовнішні ключі, які гарантують узгодженість інформації між таблицями та виключають появу некоректних записів.

Для візуального представлення структури бази даних побудовано ER-діаграму [28], наведену на ([Рис. 2.3](#)).

Діаграма бази даних демонструє структуру основних таблиць вебплатформи, їхні поля, первинні та зовнішні ключі, а також зв'язки між таблицями. Центральною таблицею є *appeals*, у якій зберігається інформація про звернення громадян. Вона пов'язана з таблицями *users*, *categories* і *statuses*, що дозволяє визначити автора звернення, його категорію та поточний стан опрацювання.

Для детального аналізу структури бази даних доцільно окремо розглянути кожную таблицю, її основні поля, типи даних та призначення. Такий підхід

		Чмут Д.К.			ІПЗ.КР.Б – 121 – 26 – ІЗ	Арк.
		Єфремов. Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		45

дозволяє повніше охарактеризувати структуру бази даних вебплатформи CityVoice та обґрунтувати вибір її складових.

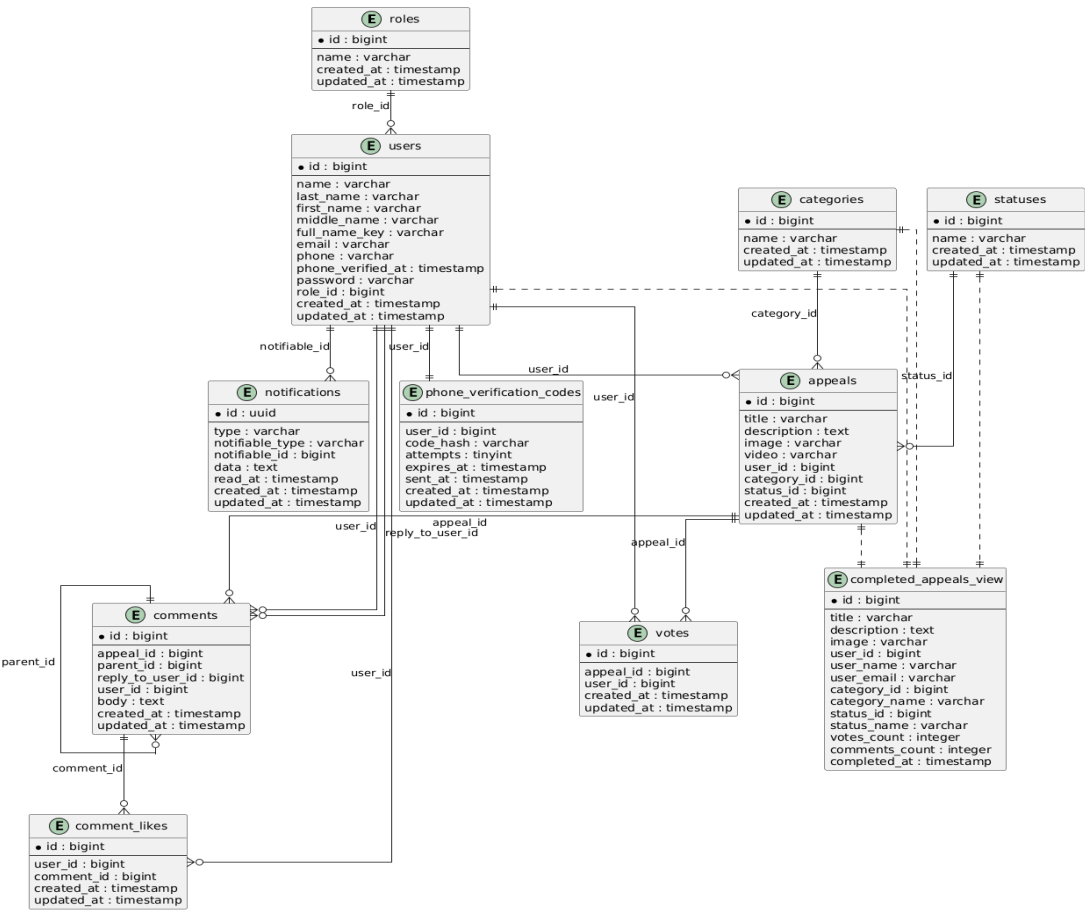


Рис 2.3 – ER-діаграма бази даних вебплатформи CityVoice

Таблиці comments, votes і comment_likes забезпечують громадську взаємодію користувачів зі зверненнями. Таблиця notifications використовується для зберігання системних повідомлень, а phone_verification_codes — для підтвердження номера телефону під час реєстрації.

Таблиця 2.13

Roles

Поле	Тип	Призначення
id	BIGINT	Унікальний ідентифікатор ролі
name	VARCHAR(255)	Назва ролі користувача
created_at	TIMESTAMP	Дата створення запису
updated_at	TIMESTAMP	Дата останнього оновлення запису

Таблиця roles використовується для зберігання ролей користувачів системи ([Табл. 2.13](#)). Вона забезпечує розмежування прав доступу між звичайними користувачами, модераторами та адміністраторами.

Таблиця 2.14

Users

Поле	Тип	Призначення
id	BIGINT	Унікальний ідентифікатор користувача
role_id	BIGINT	Ідентифікатор ролі користувача
last_name	VARCHAR(255)	Прізвище користувача
first_name	VARCHAR(255)	Ім'я користувача
middle_name	VARCHAR(255)	По батькові користувача
full_name_key	VARCHAR(255)	Службове поле для пошуку
email	VARCHAR(255)	Адреса електронної пошти
phone	VARCHAR(20)	Номер телефону
phone_verified_at	TIMESTAMP	Дата підтвердження номера
password	VARCHAR(255)	Хешований пароль
created_at	TIMESTAMP	Дата створення
updated_at	TIMESTAMP	Дата оновлення

У таблиці Users також зберігаються дані, необхідні для автентифікації користувача та підтвердження його контактної інформації ([Табл. 2.14](#)). Поле phone_verified_at використовується для фіксації факту підтвердження номера телефону, а пароль зберігається у вигляді хешованого значення, що підвищує рівень безпеки персональних даних, забезпечує захист облікового запису від несанкціонованого доступу та відповідає сучасним вимогам щодо безпечного зберігання конфіденційної інформації.

Таблиця 2.15

Categories

Поле	Тип	Призначення
id	BIGINT	Унікальний ідентифікатор
name	VARCHAR(255)	Назва категорії
created_at	TIMESTAMP	Дата створення
updated_at	TIMESTAMP	Дата оновлення

Таблиця categories використовується для групування звернень за тематикою ([Табл. 2.15](#)).

Таблиця 2.16

Statuses

Поле	Тип	Призначення
id	BIGINT	Унікальний ідентифікатор
name	VARCHAR(255)	Назва статусу
created_at	TIMESTAMP	Дата створення
updated_at	TIMESTAMP	Дата оновлення

Таблиця statuses використовується для зберігання можливих станів звернень ([Табл. 2.16](#)).

Таблиця 2.17

Appeals

Поле	Тип	Призначення
id	BIGINT	Унікальний ідентифікатор
user_id	BIGINT	Автор звернення
category_id	BIGINT	Категорія
status_id	BIGINT	Поточний статус
title	VARCHAR(255)	Назва
description	TEXT	Опис

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				48
Змн.	Арк.	№ докум.	Підпис	Дата		

продовж. табл. 2.17

image	VARCHAR(255)	Зображення
video	VARCHAR(255)	Відео
created_at	TIMESTAMP	Дата створення
updated_at	TIMESTAMP	Дата оновлення

Таблиця `appeals` є центральною таблицею бази даних вебплатформи ([Табл. 2.17](#)). У ній зберігається інформація про звернення користувачів, включаючи автора, категорію, статус та мультимедійні матеріали.

Таблиця 2.18

Comments

Поле	Тип	Призначення
id	BIGINT	Унікальний ідентифікатор
appeal_id	BIGINT	Звернення
user_id	BIGINT	Автор
parent_id	BIGINT	Батьківський коментар
reply_to_user_id	BIGINT	Користувач-відповідь
body	TEXT	Текст
created_at	TIMESTAMP	Дата створення
updated_at	TIMESTAMP	Дата оновлення

Поля `parent_id` та `reply_to_user_id` використовуються для реалізації вкладених відповідей між користувачами, що дозволяє підтримувати структуру обговорення ([Табл. 2.18](#)). Такий підхід забезпечує зручне відображення коментарів, спрощує навігацію в діалогах та робить взаємодію користувачів більш зрозумілою. Усі коментарі прив'язуються до конкретного звернення та автора, що забезпечує цілісність інформації в базі даних.

Таблиця 2.19

Votes

Поле	Тип	Призначення
id	BIGINT	Унікальний ідентифікатор
appeal_id	BIGINT	Звернення
user_id	BIGINT	Користувач
created_at	TIMESTAMP	Дата створення
updated_at	TIMESTAMP	Дата оновлення

Таблиця votes реалізує механізм голосування за звернення ([Табл. 2.19](#)).

Таблиця 2.20

Comment_likes

Поле	Тип	Призначення
id	BIGINT	Унікальний ідентифікатор
user_id	BIGINT	Користувач
comment_id	BIGINT	Коментар
created_at	TIMESTAMP	Дата створення
updated_at	TIMESTAMP	Дата оновлення

Таблиця comment_likes використовується для збереження вподобань коментарів ([Табл. 2.20](#)).

Таблиця 2.21

Notifications

Поле	Тип	Призначення
id	CHAR(36)	Унікальний ідентифікатор
type	VARCHAR(255)	Тип повідомлення
notifiable_type	VARCHAR(255)	Тип отримувача
notifiable_id	BIGINT	Ідентифікатор отримувача

продовж. табл. 2.21

data	TEXT	Дані повідомлення
read_at	TIMESTAMP	Дата прочитання
created_at	TIMESTAMP	Дата створення
updated_at	TIMESTAMP	Дата оновлення

Таблиця notifications призначена для зберігання системних повідомлень користувачів ([Табл. 2.21](#)).

Таблиця 2.22

Phone_verification_code

Поле	Тип	Призначення
id	BIGINT	Унікальний ідентифікатор
user_id	BIGINT	Користувач
code_hash	VARCHAR(255)	Хешований код
attempts	TINYINT	Кількість спроб
expires_at	TIMESTAMP	Час завершення дії
sent_at	TIMESTAMP	Час відправлення
created_at	TIMESTAMP	Дата створення
updated_at	TIMESTAMP	Дата оновлення

Таблиця phone_verification_codes використовується для підтвердження номера телефону користувача та підвищення безпеки процесу реєстрації ([Табл. 2.22](#)).

2.3 Проєктування та реалізація алгоритмів системи

При розробці вебплатформи CityVoice одним із найважливіших етапів є проєктування алгоритмів функціонування системи. Саме алгоритми визначають порядок взаємодії користувачів із програмним забезпеченням, послідовність виконання операцій, правила обробки даних та логіку переходу між окремими станами звернень. Від правильності їх побудови залежить стабільність роботи

вебплатформи, швидкість обробки запитів користувачів та надійність функціонування всієї системи.

У розробленій вебплатформі алгоритми реалізують основні бізнес-процеси, пов'язані зі створенням звернень громадян, їх подальшим опрацюванням представниками органів місцевого самоврядування, зміною статусів звернень, а також інформуванням користувачів про результати розгляду. Формалізація цих процесів за допомогою UML-діаграм [15], побудованих за допомогою PlantUML [14], діяльності та діаграми станів дозволяє наочно представити логіку роботи системи та значно спростити її програмну реалізацію.

Особливістю вебплатформи CityVoice є взаємодія декількох категорій користувачів. У системі працюють звичайні користувачі, представники органів місцевого самоврядування та адміністратори. Кожна категорія користувачів має власний набір дозволених операцій, тому алгоритми враховують перевірку прав доступу, валідацію введених даних та контроль допустимих переходів між станами звернень. Такий підхід забезпечує цілісність інформації та запобігає виконанню несанкціонованих дій.

Одним із головних бізнес-процесів вебплатформи є створення нового звернення користувачем. Саме цей процес забезпечує реалізацію основної функції системи та є початком життєвого циклу кожного звернення.

Основним алгоритмом роботи вебплатформи є процес створення нового звернення користувачем ([Рис. 2.4](#)). Саме цей алгоритм забезпечує реєстрацію проблеми в системі та ініціює її подальше опрацювання. Процес складається з декількох послідовних етапів, кожен із яких виконує окрему перевірку та забезпечує коректність збереження інформації.

На початку роботи користувач заповнює форму створення звернення, вказуючи категорію, заголовок, опис проблеми, адресу її виникнення, а також за необхідності додає фотографії, що підтверджують наявність проблеми. Після отримання інформації система виконує перевірку правильності введених даних. Контролюється заповнення обов'язкових полів, коректність координат, категорії

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		52

звернення та опису проблеми. Якщо виявлено помилки або відсутні необхідні дані, система повідомляє користувача про необхідність їх виправлення.

Після успішного проходження перевірки зверненню автоматично присвоюється початковий статус «Нове». Далі формується новий запис у базі даних, до якого зберігаються відомості про автора звернення, категорію, опис проблеми, координати місця розташування, дату створення та початковий статус. Після успішного збереження інформації користувачу відображається повідомлення про успішне створення звернення.

Результатом виконання даного алгоритму є поява нового звернення в системі та початок його подальшого життєвого циклу.



Рисунок 2.4 – Діаграма процесу створення звернення

Другим важливим бізнес-процесом є опрацювання звернення представником органу місцевого самоврядування ([Рис. 2.5](#)). Після створення

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		53

нового звернення воно автоматично потрапляє до списку звернень, доступних відповідальним працівникам. Основним завданням представника є аналіз отриманої інформації та прийняття рішення щодо можливості подальшого опрацювання звернення.

Під час перегляду звернення відповідальна особа отримує доступ до його повного опису, категорії, місця розташування, фотографій та інформації про автора. На основі отриманих даних виконується аналіз звернення та приймається рішення щодо початку його опрацювання або відхилення.

Якщо звернення відповідає встановленим вимогам, його статус змінюється на «В роботі», після чого інформація оновлюється в базі даних. Користувач автоматично отримує повідомлення про початок розгляду його звернення.

Якщо звернення не може бути виконане або містить некоректну інформацію, його статус змінюється на «Відхилено». Після оновлення запису система також повідомляє автора звернення про результат розгляду.



Рисунок 2.5 – Діаграма процесу опрацювання звернення представником органу влади

Окрім процесу створення та опрацювання звернень, вебплатформа підтримує механізм контролю їх життєвого циклу. Кожне звернення проходить

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				54
Змн.	Арк.	№ докум.	Підпис	Дата		

декілька послідовних станів, які відображають етап його опрацювання та визначають доступні дії користувачів системи ([Рис. 2.6](#)).

Початковим станом є «Нове», який присвоюється зверненню відразу після його успішного створення. На цьому етапі інформація вже збережена в базі даних, однак ще не була розглянута відповідальними працівниками.

Після перевірки звернення представником органу місцевого самоврядування воно переходить до стану «В роботі», що означає початок його опрацювання. На цьому етапі виконуються необхідні заходи для усунення виявленої проблеми або перевірки зазначеної інформації.

Після завершення всіх необхідних робіт звернення отримує статус «Завершено», який свідчить про повне вирішення проблеми. Якщо під час розгляду встановлено, що звернення не може бути виконане або не відповідає встановленим вимогам, воно переводиться до стану «Відхилено».



Рисунок 2.6 – Діаграма життєвого циклу звернення

Таким чином, у процесі проєктування було розроблено алгоритми функціонування вебплатформи CityVoice, які визначають порядок взаємодії користувачів із системою та забезпечують коректне виконання основних бізнес-

процесів. Побудовані діаграми діяльності дозволили формалізувати процес створення звернення, його опрацювання представниками органів місцевого самоврядування та керування зміною статусів протягом усього життєвого циклу.

Запропоновані алгоритми враховують перевірку коректності введених даних, контроль прав доступу користувачів, підтримання цілісності інформації та допустимі переходи між станами звернень. Завдяки цьому забезпечується стабільна робота вебплатформи, прозорість взаємодії між громадянами та органами місцевого самоврядування, а також можливість подальшого розширення функціональних можливостей системи.

Висновки до другого розділу

У другому розділі було виконано проєктування вебплатформи CityVoice, визначено її архітектуру, структуру бази даних та алгоритми функціонування основних бізнес-процесів. Під час проєктування сформовано загальну структуру програмної системи, визначено взаємодію між її компонентами та розроблено модель даних, що забезпечує ефективне зберігання й обробку інформації про звернення користувачів.

Було спроектовано реляційну базу даних, яка містить таблиці для зберігання інформації про користувачів, звернення, категорії, статуси, коментарі, голосування та інших службових даних. Використання первинних і зовнішніх ключів, а також механізму міграцій Laravel забезпечує цілісність даних, контроль структури бази даних і можливість її подальшого розширення.

Окрему увагу приділено проєктуванню алгоритмів функціонування вебплатформи. За допомогою UML-діаграм діяльності та діаграми станів було формалізовано процес створення звернення, його опрацювання представниками органів місцевого самоврядування та життєвий цикл звернення від моменту створення до завершення або відхилення. Використання статусної моделі дозволяє забезпечити прозорість роботи системи, контроль виконання звернень і підтримання актуальності інформації.

Отримані результати проєктування створюють основу для програмної реалізації вебплатформи CityVoice та забезпечують можливість її подальшого

		Чмут Д.К.			ПІЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				56
Змн.	Арк.	№ докум.	Підпис	Дата		

розвитку. Розроблені архітектурні рішення відповідають поставленим вимогам, сприяють масштабованості програмного забезпечення та забезпечують надійне функціонування системи в умовах реальної експлуатації.

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		57

РОЗДІЛ 3 ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ ІЗ СИСТЕМОЮ

3.1 Порядок встановлення та налаштування параметрів системи

Вебплатформа CityVoice реалізована на основі сучасної клієнт-серверної архітектури із використанням фреймворку Laravel, що забезпечує централізовану обробку запитів користувачів, взаємодію з базою даних та формування вебінтерфейсу. Усі функціональні компоненти системи працюють у межах одного програмного застосунку, що дозволяє спростити процес адміністрування, супроводу та подальшого розвитку програмного забезпечення.

На відміну від мікросервісної архітектури, де окремі функціональні модулі розгортаються як незалежні сервіси, вебплатформа CityVoice реалізована за принципом монолітного вебзастосунку. Такий підхід забезпечує високу продуктивність, зменшує складність взаємодії між окремими компонентами та спрощує процес розгортання системи. Усі модулі працюють у межах одного середовища виконання Laravel та використовують спільну реляційну базу даних.

Функціональні можливості вебплатформи логічно поділено на декілька основних модулів:

- модуль реєстрації та автентифікації користувачів;
- модуль керування зверненнями громадян;
- модуль категорій звернень;
- модуль коментарів та голосування;
- модуль модерації звернень;
- модуль адміністрування користувачів;
- модуль формування статистики та PDF-звітів;
- модуль електронних сповіщень.

Кожен із зазначених модулів містить власні контролери, моделі, представлення та бізнес-логіку, що забезпечує низький рівень зв'язності між окремими компонентами системи та значно спрощує подальший супровід програмного забезпечення. Незважаючи на логічний поділ, усі модулі працюють із єдиною реляційною базою даних, що забезпечує цілісність інформації та ефективну взаємодію між окремими функціональними підсистемами.

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				58
Змн.	Арк.	№ докум.	Підпис	Дата		

У межах архітектури вебплатформи сервер застосунку виконує обробку HTTP-запитів, реалізує бізнес-логіку та забезпечує взаємодію із сервером бази даних. Для роботи із файлами користувачів використовується файлове сховище Laravel Storage, у якому зберігаються фотографії звернень та інші завантажені матеріали. Надсилання електронних повідомлень реалізовано із використанням поштового механізму Laravel Mail, а під час розробки для перевірки електронних листів застосовується Mailpit [9].

Взаємодія користувачів із вебплатформою здійснюється через вебінтерфейс за допомогою мережі Інтернет. Доступ до системи можливий із персональних комп'ютерів, ноутбуків, планшетів та мобільних пристроїв без необхідності встановлення додаткового програмного забезпечення. Для роботи із платформою достатньо сучасного веббраузера, який підтримує технології HTML [10], CSS [11] та JavaScript [12, 13]

Використання централізованої архітектури забезпечує низку переваг під час експлуатації системи. Зокрема, адміністратор може виконувати оновлення програмного забезпечення без необхідності координувати роботу декількох окремих сервісів. Крім того, значно спрощується процес резервного копіювання бази даних, моніторингу роботи системи та відновлення працездатності у випадку виникнення помилок. Реалізація нових функціональних можливостей також потребує менших витрат часу, оскільки всі компоненти вебплатформи функціонують як єдиний програмний застосунок.

Для кращого розуміння фізичного розміщення основних компонентів та взаємодії між ними було побудовано діаграму розгортання вебплатформи CityVoice ([Рис. 3.1](#)).

На діаграмі розгортання представлено основні компоненти вебплатформи CityVoice та взаємодію між ними. Користувачі працюють із системою через веббраузер, використовуючи мережу Інтернет або локальне мережеве з'єднання. Усі HTTP-запити надходять до вебсервера, який виконує застосунок Laravel та забезпечує їх подальшу обробку.

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		59

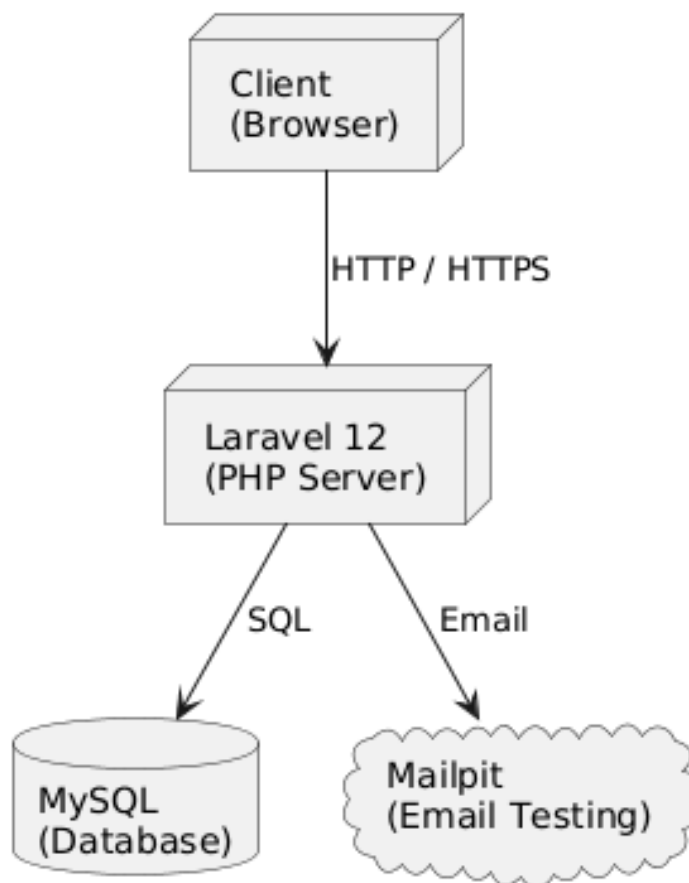


Рис. 3.1 – Діаграма розгортання вебплатформи CityVoice

Сервер застосунку реалізує бізнес-логіку вебплатформи, забезпечує автентифікацію користувачів, обробку звернень, взаємодію з базою даних, формування PDF-звітів та надсилання електронних повідомлень. Для зберігання інформації про користувачів, звернення, категорії, статуси, коментарі, голосування та інші службові дані застосовується система керування базами даних MySQL [4].

Завантажені користувачами фотографії звернень та інші мультимедійні файли зберігаються у файловому сховищі Laravel Storage. Надсилання повідомлень електронною поштою реалізовано за допомогою механізму Laravel Mail, а під час розробки перевірка роботи електронних листів здійснюється із використанням сервісу Mailpit [9].

Для забезпечення стабільної роботи серверної частини вебплатформи необхідно використовувати відповідне програмне та апаратне забезпечення, мінімальні вимоги до якого наведено в (Табл. 3.1).

		Чмут Д.К.			ІПЗ.КР.Б – 121 – 26 – ІЗ	Арк.
		Єфремов. Ю.М.				60
Змн.	Арк.	№ докум.	Підпис	Дата		

Мінімальні вимоги

Компонент	Мінімальні вимоги
Операційна система	Apache 2.4+ або Nginx
Вебсервер	Apache або Nginx
Система керування базами даних	MySQL
Інтерпретатор PHP	PHP 8.2 або новіша
Оперативна пам'ять	від 4 ГБ
Процесор	2 ядра
Вільне місце на диску	від 10 ГБ

Після підготовки середовища виконання необхідно встановити всі програмні залежності вебплатформи, виконати початкове налаштування параметрів застосунку та створити структуру бази даних. Завдяки використанню менеджерів пакетів Composer [8] і npm [20] (Node.js [19]) більшість необхідних компонентів встановлюється автоматично, що значно спрощує процес розгортання системи.

Початкове налаштування вебплатформи включає створення конфігураційного файлу середовища, генерацію службового ключа застосунку, налаштування параметрів підключення до бази даних, виконання міграцій та компіляцію клієнтських ресурсів [27]. Послідовність основних етапів встановлення та запуску вебплатформи наведено в (Табл. 3.2).

Після виконання наведених етапів вебплатформа готова до експлуатації. Доступ до системи здійснюється через веббраузер за адресою локального або віддаленого сервера. Після запуску користувачі можуть проходити реєстрацію та авторизацію, підтверджувати номер телефону, створювати звернення, переглядати їхній поточний стан, брати участь у голосуванні, залишати коментарі та отримувати сповіщення про зміни, що відбуваються у процесі їх опрацювання.

Використання сучасного програмного забезпечення, механізму міграцій Laravel, автоматичного керування залежностями за допомогою Composer [8] і npm, а також засобів компіляції клієнтських ресурсів забезпечує швидке

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				61
Змн.	Арк.	№ докум.	Підпис	Дата		

розгортання вебплатформи, стабільну роботу системи та можливість її подальшого розвитку без суттєвих змін архітектури.

Таблиця 3.2

Послідовність встановлення та запуску вебплатформи

№	Етап	Опис	Команда
1	Отримання проєкту	Клонування або відкриття вихідного коду проєкту	git clone <repository>
2	Встановлення PHP-залежностей	Завантаження бібліотек Laravel	composer install
3	Встановлення JavaScript-залежностей	Завантаження пакетів Node.js	npm install
4	Створення конфігурації	Створення файлу налаштувань середовища	copy .env.example .env
5	Генерація ключа	Генерація службового ключа застосунку	php artisan key:generate
6	Налаштування БД	Заповнення параметрів підключення до MySQL у файлі .env	Редагування файлу .env
7	Створення структури БД	Виконання міграцій Laravel	php artisan migrate
8	Створення файлового сховища	Створення символічного посилання для файлів	php artisan storage:link
9	Компіляція ресурсів	Збірка CSS та JavaScript-файлів	npm run build
10	Запуск вебплатформи	Запуск локального вебсервера Laravel	php artisan serve

3.2 Структура інтерфейсу платформи

Після реалізації серверної частини вебплатформи було розроблено користувацький інтерфейс, який забезпечує взаємодію громадян, модераторів та адміністраторів із системою. Під час проєктування інтерфейсу основна увага приділялася простоті використання, зрозумілій навігації, адаптивності та швидкому доступу до основних функціональних можливостей вебплатформи.

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				62
Змн.	Арк.	№ докум.	Підпис	Дата		

Інтерфейс реалізовано із використанням шаблонізатора Blade, HTML [10], CSS [11], Bootstrap [7] та JavaScript [12, 13]. Такий підхід забезпечує коректне відображення сторінок на персональних комп'ютерах, планшетах і мобільних пристроях, а також дозволяє швидко реагувати на дії користувача без необхідності перезавантаження окремих елементів сторінки.

Першою сторінкою, яку бачить користувач після відкриття вебплатформи, є головна сторінка CityVoice (Рис. 3.2). Вона містить основну інформацію про призначення системи, перелік її функціональних можливостей та навігаційне меню для переходу до основних розділів вебплатформи.

На головній сторінці розміщено інформаційний блок із коротким описом призначення системи, а також кнопки швидкого переходу до створення нового звернення, перегляду звернень та авторизації користувача. Верхнє меню забезпечує швидкий доступ до всіх основних розділів вебплатформи незалежно від поточної сторінки.

Головна сторінка побудована відповідно до принципів адаптивного дизайну, що забезпечує її коректне відображення на пристроях із різною роздільною здатністю екрана. Використання Bootstrap [7] дозволяє автоматично змінювати розташування окремих елементів інтерфейсу залежно від ширини вікна браузера, що підвищує зручність використання вебплатформи.

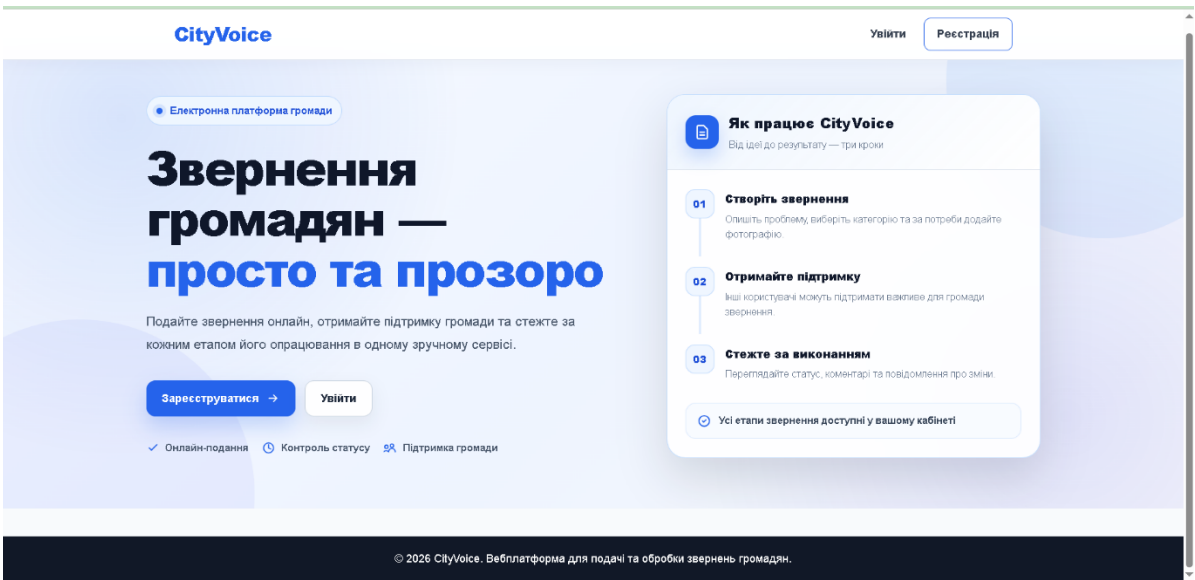


Рис. 3.2 – Головна сторінка

Після відкриття головної сторінки користувач має можливість створити новий обліковий запис або авторизуватися у системі. Реєстрація є обов'язковим етапом для отримання доступу до функцій створення звернень, коментування, голосування та перегляду особистої інформації. Форма реєстрації вебплатформи CityVoice наведена на (Рис. 3.3).

Під час реєстрації користувач заповнює персональні дані, зокрема прізвище, ім'я, адресу електронної пошти, номер телефону та пароль. Для забезпечення безпеки пароль необхідно повторно підтвердити, що дозволяє мінімізувати кількість помилок під час створення нового облікового запису.

Після надсилання форми система виконує перевірку правильності введених даних. Контролюється коректність адреси електронної пошти, унікальність облікового запису, формат номера телефону, а також відповідність пароля встановленим вимогам безпеки. У випадку виявлення помилок користувач отримує повідомлення із зазначенням полів, які потребують виправлення.

Після успішної реєстрації створюється новий обліковий запис користувача, який надалі використовується для автентифікації у вебплатформі та взаємодії з усіма її функціональними можливостями.

СТВОРЕННЯ ОБЛІКОВОГО ЗАПИСУ

Реєстрація в CityVoice

Вкажіть справжні персональні дані. Повне ПІБ, електронна пошта та номер телефону повинні бути унікальними.

Прізвище: Ім'я:

По батькові:

Електронна пошта: Номер телефону:

Пароль: Підтвердження пароля:

Зареєструватися →

Уже маєте обліковий запис? [Увійти](#)

Рис.3.3 – Форма реєстрації користувача

		Чмут Д.К.			ПІЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		64

Після успішної реєстрації користувачеві необхідно підтвердити номер мобільного телефону. Такий механізм дозволяє підвищити рівень безпеки системи, підтвердити достовірність контактних даних та запобігти створенню фіктивних облікових записів. Інтерфейс підтвердження номера телефону наведено на (Рис. 3.4).

Для підтвердження номера користувач вводить одноразовий код, отриманий у повідомленні. Після перевірки правильності введеного коду система активує обліковий запис і відкриває доступ до повного функціоналу вебплатформи.

Використання двоетапної процедури реєстрації забезпечує додатковий рівень захисту персональних даних та дозволяє підвищити достовірність інформації про користувачів системи.

Рис. 3.4 – Підтвердження номера телефону

Після завершення реєстрації користувач може виконати вхід до вебплатформи за допомогою форми авторизації, зображеної на (Рис. 3.5). Для входу необхідно ввести адресу електронної пошти та пароль, що були використані під час створення облікового запису.

		Чмут Д.К.			ПІЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		65

Після надсилання форми система виконує перевірку введених облікових даних. Якщо інформація є коректною, користувач успішно проходить автентифікацію та отримує доступ до функціональних можливостей вебплатформи відповідно до своєї ролі. У разі введення неправильного логіна або пароля система повідомляє користувача про помилку та пропонує повторити спробу входу.

Форма авторизації також містить посилання для відновлення пароля, що дозволяє швидко відновити доступ до облікового запису у випадку його втрати.

Рис. 3.5 – Форма авторизації користувача

У випадку втрати пароля користувач може скористатися функцією його відновлення. Для цього у вебплатформі передбачено окрему форму, інтерфейс якої наведено на (Рис. 3.6, а).

Для відновлення доступу до облікового запису користувач вводить адресу електронної пошти, яка була використана під час реєстрації. Після надсилання запиту система перевіряє наявність відповідного облікового запису та формує спеціальне посилання для зміни пароля, про що користувач отримує відповідне повідомлення (Рис. 3.6, б).

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				66
Змн.	Арк.	№ докум.	Підпис	Дата		

Реалізація даного механізму дозволяє безпечно відновити доступ до облікового запису без втручання адміністратора, а також забезпечує захист персональних даних користувачів шляхом використання одноразових посилань із обмеженим терміном дії.

ВІДНОВЛЕННЯ ДОСТУПУ

Забули пароль?

Вкажіть електронну пошту, пов'язану з вашим обліковим записом. Ми надішлемо посилання для створення нового пароля.

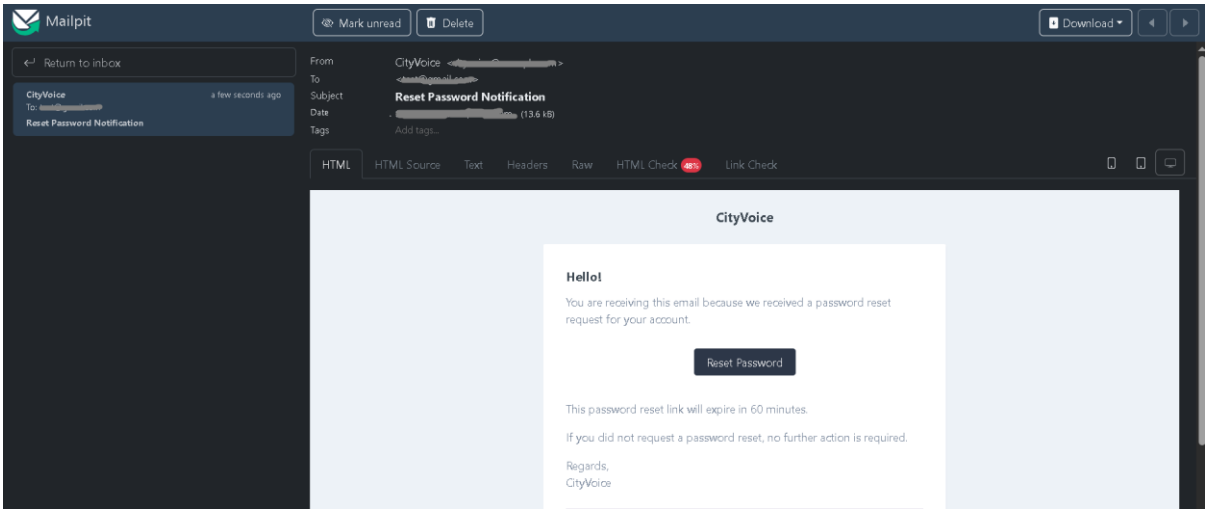
Електронна пошта

hame@example.com

Надіслати посилання →

Згадали пароль? [Повернутися до входу](#)

а)



б)

Рисунок 3.6 – Відновлення пароля:
а – форма відновлення пароля;
б – електронний лист для відновлення пароля.

Після переходу за посиланням із листа користувач потрапляє на сторінку створення нового пароля ([Рис. 3.7](#)).

Рисунок 3.7 – Форма створення нового пароля

Після успішного створення нового пароля користувач може повторно виконати авторизацію та отримати доступ до всіх функціональних можливостей вебплатформи. Основною функцією системи є створення, перегляд та опрацювання звернень громадян, тому після входу користувач переходить до сторінки зі списком звернень, інтерфейс якої наведено на ([Рис. 3.8](#)).

На сторінці відображається перелік звернень, створених користувачами вебплатформи. Для кожного звернення виводиться коротка інформація, зокрема назва, категорія, автор, кількість коментарів, дата створення та поточний статус. Крім цього, користувач має можливість перейти до детального перегляду будь-якого звернення.

Для швидкого пошуку необхідної інформації реалізовано механізм фільтрації за назвою, категорією та статусом звернення. Використання таких інструментів дозволяє значно спростити навігацію навіть за великої кількості опублікованих звернень.

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				68
Змн.	Арк.	№ докум.	Підпис	Дата		

У верхній частині сторінки розміщено кнопку створення нового звернення, що забезпечує швидкий перехід до відповідної форми без необхідності переходу між декількома сторінками.

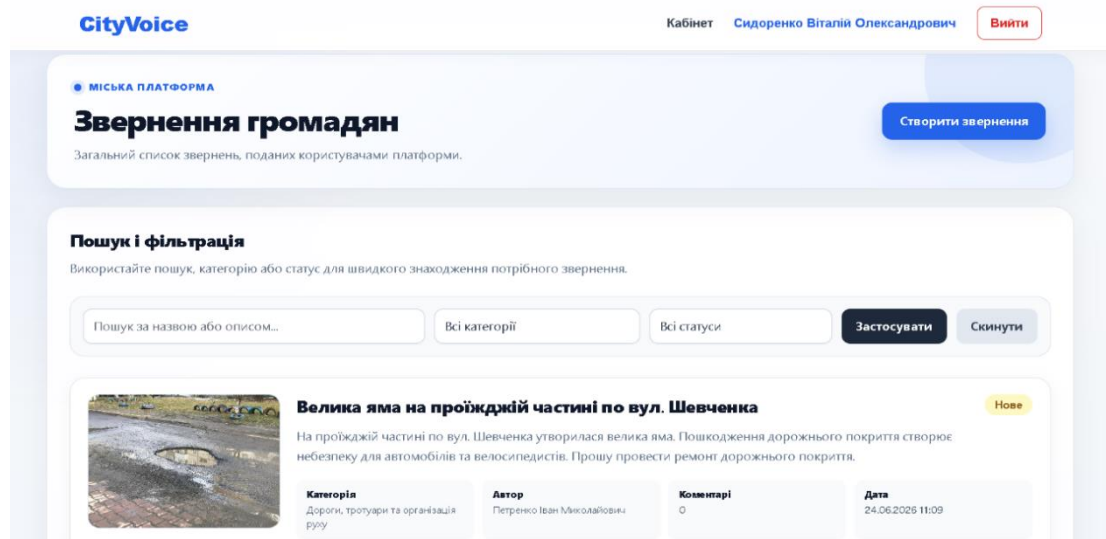


Рис. 3.8 – Список звернень вебплатформи

Для створення нового звернення громадянину необхідно заповнити спеціальну форму, інтерфейс якої наведено на (Рис. 3.9).

Форма містить поля для введення назви звернення, детального опису проблеми, вибору відповідної категорії, а також можливість додавання фотографій і відеоматеріалів, які підтверджують факт існування проблеми. Завантаження мультимедійних матеріалів дозволяє більш детально проілюструвати ситуацію та спрощує її подальший розгляд модераторами.

Після натискання кнопки створення звернення система виконує перевірку коректності введених даних. Якщо всі обов'язкові поля заповнено правильно, інформація зберігається у базі даних, а зверненню автоматично присвоюється початковий статус «Нове».

Структурована форма введення забезпечує повноту інформації про проблему та дозволяє мінімізувати кількість помилок під час створення звернення.

Інформація про звернення

Поля, позначені зірочкою, є обов'язковими для заповнення.

Назва *

Назва має коротко передавати суть звернення.

Опис *

Детально опишіть проблему, місце та важливі обставини

Категорія *

Оберіть категорію

Фото до звернення

Обрати файл

Файл не обрано

Доступні формати: JPG, PNG або WEBP. Максимальний розмір — 2 МБ.

Відеофіксація

Обрати відео

Файл не обрано

Доступні формати: MP4 або WEBM. Максимальний розмір — 50 МБ. Для найкращої сумісності рекомендовано MP4.

Створити звернення

Скасувати

Рис.3.9 – Форма створення нового звернення

Після створення звернення користувач може переглянути його детальну інформацію. Відповідна сторінка наведена на ([Рис. 3.10](#)).

На сторінці відображаються назва звернення, його категорія, автор, дата створення, кількість голосів, поточний статус, детальний опис проблеми та додані фотографії. Такий підхід дозволяє отримати повну інформацію про звернення без необхідності переходу між різними сторінками системи.

Нижче основної інформації розташовано блок коментарів, у якому користувачі можуть обговорювати проблему, залишати власні повідомлення та

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				70
Змн.	Арк.	№ докум.	Підпис	Дата		

відповідати на коментарі інших учасників. Реалізований механізм підтримує вкладені відповіді, що забезпечує зручне ведення дискусій.

Також на сторінці реалізовано механізм підтримки звернення шляхом голосування. Це дозволяє визначити найбільш актуальні проблеми громади та сформувати пріоритетність їх подальшого розгляду.

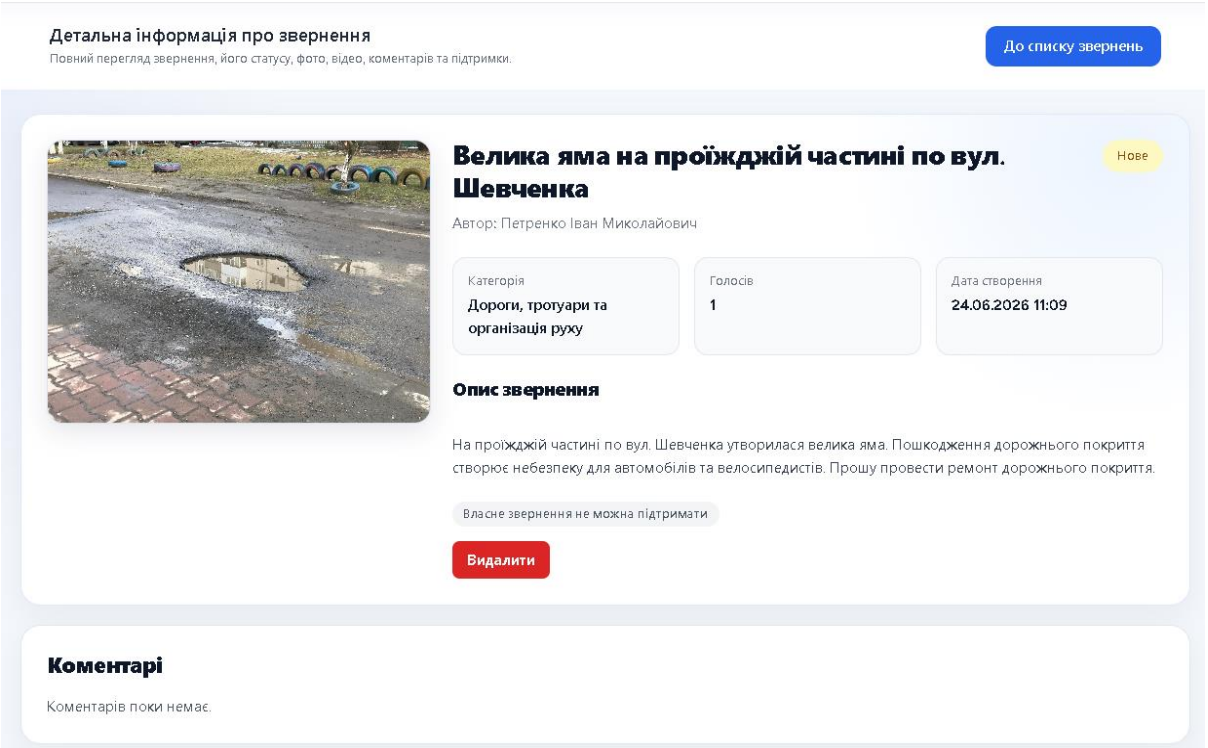


Рис. 3.10 – Сторінка перегляду звернення

Таким чином, сторінка перегляду звернення забезпечує користувачу доступ до повної інформації про проблему, дозволяє відстежувати хід її опрацювання, брати участь в обговоренні та підтримувати звернення шляхом голосування. Це сприяє відкритості процесу взаємодії між громадянами та органами місцевого самоврядування, а також підвищує рівень громадської активності.

Після авторизації кожний зареєстрований користувач отримує доступ до особистого кабінету, який містить персональну інформацію, список власних звернень та повідомлення про зміни їхнього стану. Інтерфейс особистого кабінету наведено на (Рис. 3.11).

Особистий кабінет відображає основні персональні дані користувача, зокрема повне ім'я, службовий номер, адресу електронної пошти, номер

телефону, роль у системі та стан підтвердження номера телефону. Крім цього, користувач може перейти до редагування профілю та оновлення власної інформації.

Окремий блок сторінки призначений для відображення сповіщень про зміну статусів звернень та появу нових коментарів. Це дозволяє користувачу оперативно отримувати інформацію про перебіг розгляду створених ним звернень без необхідності постійно перевіряти їх вручну.

Нижче розташовано список звернень, створених користувачем. Для кожного запису відображаються назва, категорія, кількість коментарів, дата створення та поточний статус. Також передбачено можливість перегляду детальної інформації або видалення звернення до моменту його остаточного опрацювання.

Таким чином, особистий кабінет забезпечує централізований доступ до персональної інформації користувача, створених звернень та повідомлень системи, що значно підвищує зручність використання вебплатформи.

Огляд

Налаштування профілю

Інформація про обліковий запис

Ваші основні персональні дані та стан підтвердження телефону.

Редагувати профіль

ПОВНЕ ПІБ

Петренко Іван Миколайович

НОМЕР КОРИСТУВАЧА

CV-22

РОЛЬ

Користувач

ЕЛЕКТРОННА ПОШТА

ivan.petrenko@example.com

НОМЕР ТЕЛЕФОНУ

+380671111111

ПІДТВЕРДЖЕННЯ ТЕЛЕФОНУ

Підтверджено

Привіт, Петренко Іван Миколайович!

У цьому кабінеті ви можете переглядати власні звернення, їхній поточний статус та видаляти створені записи.

Створити звернення

Сповіщення

Повідомлення про зміну статусу ваших звернень та коментарів.

Показати

Мої звернення

Список звернень, які були створені вами у системі.



Велика яма на проїжджій частині по вул. Шевченка

На проїжджій частині по вул. Шевченка утворилася велика яма. Пошкодження дорожнього покриття створює небезпеку для автомобілів та велосипедистів. Прошу провести ремонт дорожнього п...

Категорія: Дороги, тротуари та організація руху

Коментарів: 2

Дата: 24.06.2026 11:09

Детальніше

Видалити

Нове

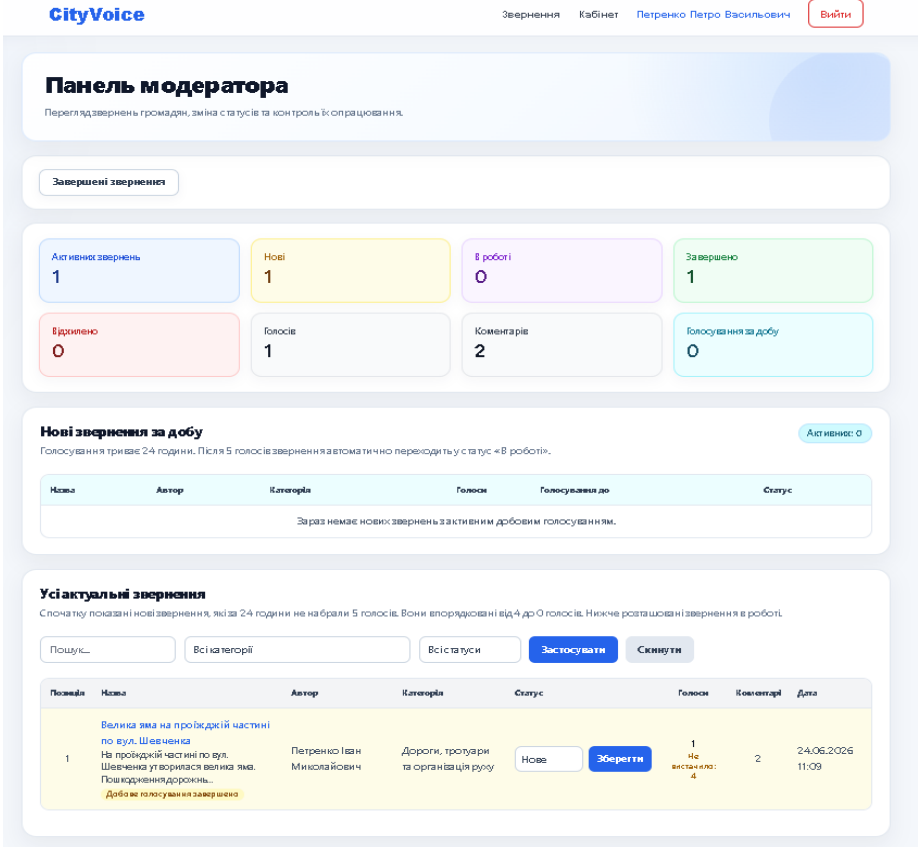
Рис.3.11 – Особистий кабінет користувача

Після авторизації модератор отримує доступ до спеціалізованої панелі керування зверненнями, яка забезпечує контроль процесу їх опрацювання. Інтерфейс панелі модератора наведено на (Рис. 3.12, а).

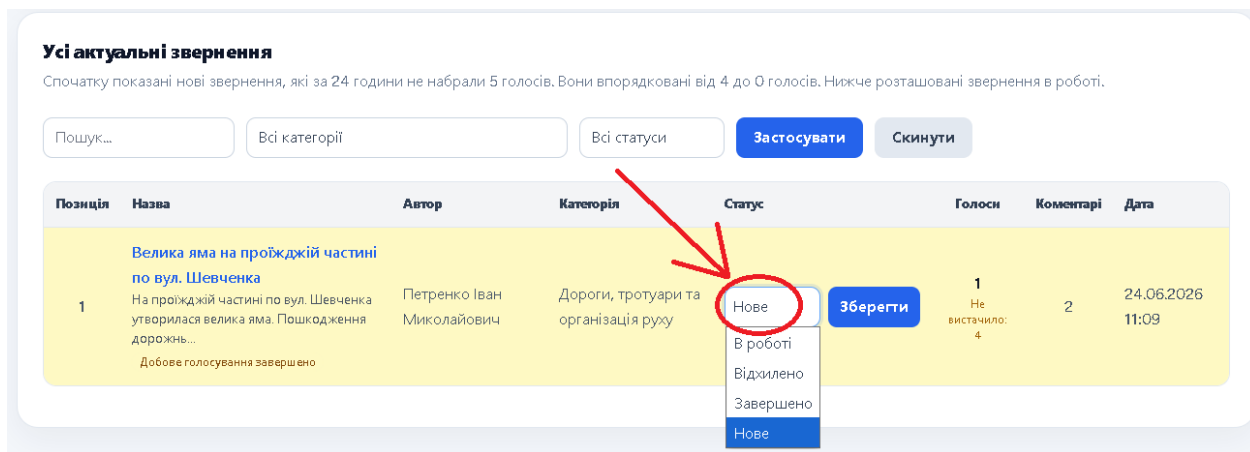
На головній сторінці відображаються статистичні показники роботи вебплатформи, серед яких кількість активних, нових, завершених та відхилених звернень, загальна кількість голосів і коментарів. Крім цього, формується перелік нових звернень, що перебувають у періоді громадського голосування.

Модератор може переглядати інформацію про кожне звернення, використовувати пошук та фільтрацію, а також змінювати його поточний статус. Для цього реалізовано спеціальний список вибору, приклад використання якого наведено на (Рис. 3.12, б).

Залежно від результатів розгляду звернення модератор може перевести його у статус «В роботі», «Завершено» або «Відхилено». Після збереження змін інформація автоматично оновлюється у базі даних, а користувач отримує відповідне повідомлення про зміну статусу.



(а)



(б)

Рис. 3.12 – Панель модератора вебплатформи:

а – головна сторінка панелі модератора;

б – зміна статусу звернення.

Окрім функцій модерації звернень, у вебплатформі реалізовано окрему панель адміністратора, яка призначена для керування обліковими записами користувачів та контролю їхніх ролей у системі. Інтерфейс сторінки керування користувачами наведено на (Рис. 3.13).

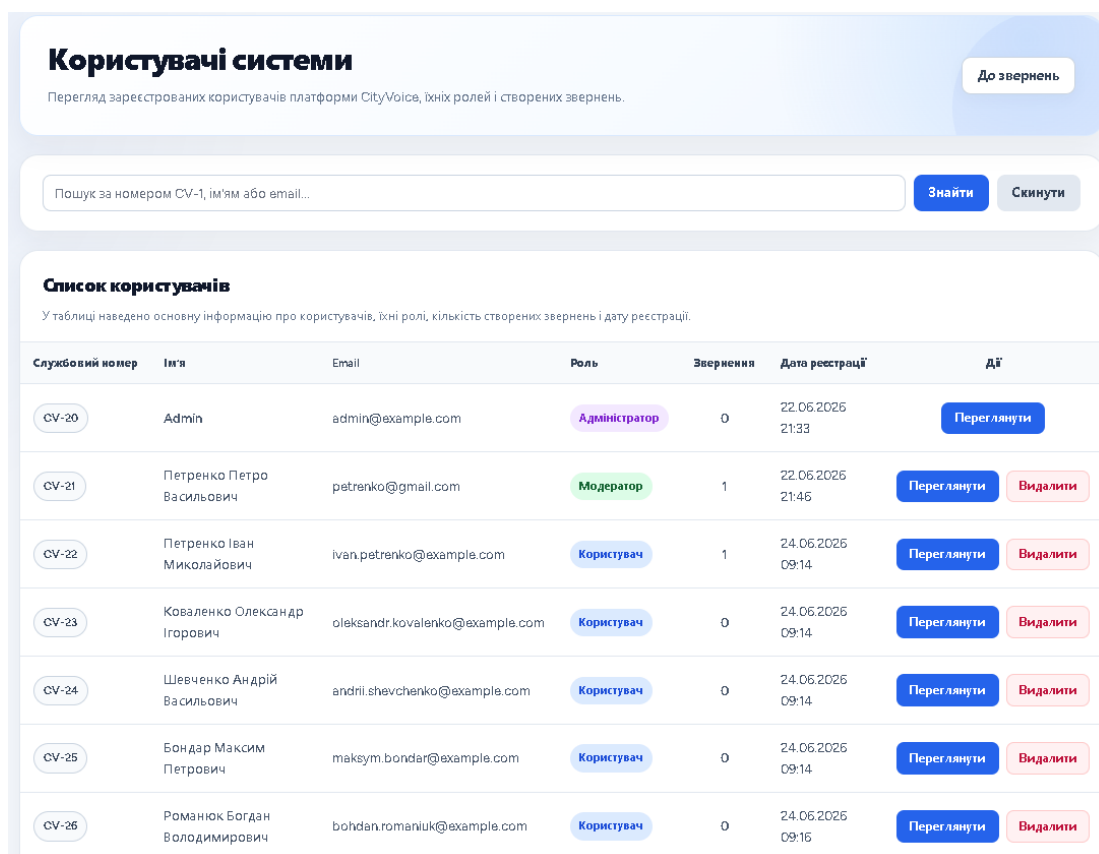


Рис.3.13 – Список користувачів

На сторінці відображається список усіх зареєстрованих користувачів із зазначенням службового номера, повного імені, адреси електронної пошти, ролі, кількості створених звернень та дати реєстрації. Для швидкого пошуку реалізовано поле пошуку за службовим номером, ім'ям або електронною поштою.

Для кожного користувача адміністратор може переглянути детальну інформацію або виконати видалення облікового запису. Це забезпечує централізоване керування користувачами та дозволяє підтримувати актуальність інформації у вебплатформі.

Після вибору користувача адміністратор переходить до сторінки перегляду його облікового запису, інтерфейс якої наведено на (Рис. 3.14).

CV-22

Петренко Іван Миколайович
ivan.petrenko@example.com

Назад до користувачів

Службовий номер
CV-22

Поточна роль
Користувач

Користувач

Модератор

Користувач

Зберегти

Кількість звернень
1

Дата реєстрації
24.06.2026

Видалення облікового запису
Обліковий запис, звернення, коментарі, голоси та інші пов'язані дані користувача буде видалено без можливості відновлення.

Видалити користувача

Звернення користувача
Список звернень, створених цим користувачем.

Назва	Категорія	Статус	Голоси	Коментарі	Дата	Дії
Велика яма на проїжджій частині по вул. Шевченка На проїжджій частині по вул. Шевченка утворилася велика яма. Пошкодження дорожнь...	Дороги, тротуари та організація руху	Нове	1	2	24.06.2026 11:09	Відкрити

Рис. 3.14 – Керування обліковим записом користувача

На сторінці відображається повна інформація про користувача, зокрема службовий номер, дату реєстрації, поточну роль у системі та кількість створених звернень. Адміністратор може змінити роль користувача, обравши необхідне значення зі списку, після чого зміни зберігаються у базі даних.

Нижче розташовано перелік звернень, створених обраним користувачем, що дозволяє швидко переглянути історію його активності. Крім цього, реалізовано механізм безповоротного видалення облікового запису разом із пов'язаними зверненнями, коментарями та голосами, що забезпечує цілісність даних вебплатформи.

Після завершення опрацювання звернення воно автоматично переноситься до архіву завершених звернень. Для перегляду таких записів у вебплатформі реалізовано окрему сторінку, інтерфейс якої наведено на (Рис. 3.15).

На сторінці відображається перелік усіх завершених звернень із зазначенням автора, категорії, кількості голосів, кількості коментарів та дати завершення опрацювання. Для зручності користувача реалізовано пошук за назвою, описом або автором звернення, а також фільтрацію за категорією і часовим проміжком.

Окремий інформаційний блок відображає загальну кількість завершених звернень, що дозволяє швидко оцінити результати роботи системи за вибраний період. Такий підхід забезпечує зручний доступ до архівних даних та спрощує їх подальший аналіз.

Завершені звернення

Архів звернень, опрацювання яких було завершено.

До активних звернень

Місячний PDF-звіт

Оберіть місяць і рік, за які потрібно сформувати звіт.

Місяць

Рік

Червень

2026

Сформувати PDF

Усього завершених

1

Назва, опис або автор...

Всі категорії

Дата від

Дата до

Застосувати

Скинути

Назва	Автор	Категорія	Голоси	Коментарі	Завершено	Дії
1	Петренко Петро	Будівництво та	0	0	24.06.2026 07:49	Відкрити
2	Васильович	архітектура				

Рис. 3.15 – Сторінка завершених звернень

Однією з додаткових можливостей вебплатформи є автоматичне формування звітів про завершені звернення. Для цього на сторінці архіву реалізовано спеціальний блок вибору місяця та року, після чого користувач може сформувати звіт у форматі PDF. Приклад сформованого документа наведено на [\(Рис. 3.16\)](#).

Під час формування звіту система автоматично отримує інформацію із бази даних та створює документ, який містить перелік завершених звернень за вибраний період, їх категорії, авторів, дати завершення, кількість голосів і коментарів. Сформований документ може бути використаний для подальшого аналізу діяльності органів місцевого самоврядування або архівного зберігання.

Автоматизація процесу створення звітів значно скорочує час підготовки необхідної документації, усуває необхідність ручного опрацювання інформації та мінімізує ймовірність виникнення помилок під час формування статистичних матеріалів.

CityVoice

Звіт про завершені звернення

Місячний звіт за Червень 2026

Звітний період Червень 2026	Завершено звернень 1	Дата формування [REDACTED]
---------------------------------------	--------------------------------	-------------------------------

Підсумок за категоріями

Категорія	Кількість звернень
Будівництво та архітектура	1

Перелік завершених звернень

№	Назва звернення	Автор	Категорія	Голоси	Коментарі	Дата завершення
1	1 2	Петренко Петро Васильович	Будівництво та архітектура	0	0	[REDACTED]

Звіт автоматично сформовано інформаційною системою CityVoice.

Рис. 3.16 – Формування PDF-звіту про завершені звернення

3.3 Тестування програмної системи

Після завершення розробки вебплатформи CityVoice було проведено комплексне тестування всіх реалізованих функціональних можливостей. Основною метою тестування була перевірка коректності роботи системи,

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				77
Змн.	Арк.	№ докум.	Підпис	Дата		

стабільності взаємодії між її компонентами та відповідності реалізованого функціоналу поставленим вимогам.

Під час тестування перевірялися процеси реєстрації користувачів, підтвердження номера телефону, авторизації, відновлення пароля, створення звернень, завантаження фотографій та відеофайлів, голосування за звернення, додавання коментарів, зміни статусів звернень модератором, формування PDF-звітів, а також робота адміністративної панелі. Крім перевірки користувацького інтерфейсу, окрема увага приділялася правильності взаємодії застосунку із базою даних та обробці помилкових ситуацій.

Для підтвердження працездатності серверної частини було використано автоматизоване тестування, реалізоване засобами фреймворку Laravel. Запуск тестів здійснювався безпосередньо у середовищі виконання проєкту, що дозволило перевірити основні функціональні модулі вебплатформи, правильність виконання бізнес-логіки та коректність роботи окремих компонентів системи (Рис. 3.17).

```

Администратор: C:\Windows\System32\cmd.exe
D:\Diploma\cityvoice>docker compose exec app php artisan test

PASS Tests\Unit\ExampleTest
  that true is true 0.06s

PASS Tests\Feature\Auth\AuthenticationTest
  login screen can be rendered 4.71s
  users can authenticate using the login screen 0.53s
  users can not authenticate with invalid password 0.33s
  users can logout 0.11s

PASS Tests\Feature\Auth\EmailVerificationTest
  email verification screen can be rendered 0.08s
  email can be verified 0.13s
  email is not verified with invalid hash 0.11s

PASS Tests\Feature\Auth>PasswordConfirmationTest
  confirm password screen can be rendered 0.09s
  password can be confirmed 0.06s
  password is not confirmed with invalid password 0.28s

PASS Tests\Feature\Auth>PasswordResetTest
  reset password link screen can be rendered 0.10s
  reset password link can be requested 0.29s
  reset password screen can be rendered 0.12s
  password can be reset with valid token 0.12s

PASS Tests\Feature\Auth>PasswordUpdateTest
  password can be updated 0.07s
  correct password must be provided to update password 0.07s

PASS Tests\Feature\Auth\RegistrationTest
  registration screen can be rendered 0.12s
  new users can register 0.08s

PASS Tests\Feature\ExampleTest
  the application returns a successful response 0.07s

PASS Tests\Feature\ProfileTest
  profile page is displayed 0.21s
  profile information can be updated 0.18s
  email verification status is unchanged when the email address is unchanged 0.07s
  user can delete their account 0.06s
  correct password must be provided to delete account 0.07s

Tests: 25 passed (58 assertions)
Duration: 8.89s
  
```

Рис. 3.17 – Запуск вебплатформи CityVoice у середовищі Docker[5, 6]

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				78
Змн.	Арк.	№ докум.	Підпис	Дата		

У результаті виконання автоматизованого тестування всі перевірки були успішно пройдені, що підтверджує працездатність реалізованого програмного забезпечення та відсутність критичних помилок у його функціонуванні.

Під час тестування також було перевірено механізм автоматичного інформування користувачів про зміни, що відбуваються із їхніми зверненнями. Після оновлення статусу звернення модератором система автоматично формує повідомлення, яке відображається в особистому кабінеті користувача ([Рис. 3.18](#)).

Перевірка підтвердила, що повідомлення коректно створюються, зберігаються у базі даних та відображаються у відповідному розділі вебплатформи. Це дозволяє користувачам оперативно отримувати інформацію про перебіг розгляду звернення без необхідності постійно перевіряти його сторінку.

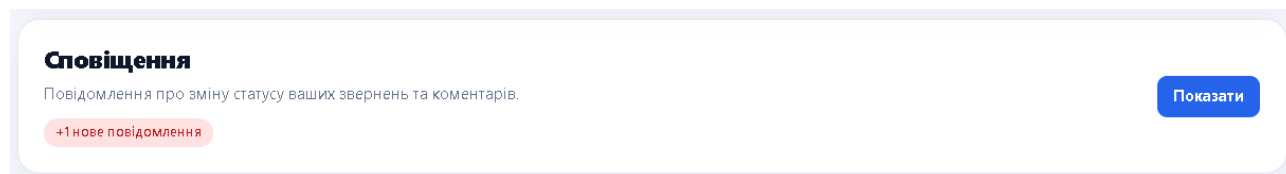


Рис. 3.18 – Повідомлення про зміну статусу звернення

Наступним етапом було протестовано систему коментування звернень. Під час перевірки було створено коментарі та відповіді до них, після чого перевірено правильність їх збереження, відображення та впорядкування ([Рис. 3.19](#)).

Результати тестування підтвердили коректну роботу механізму коментування. Коментарі та відповіді успішно зберігаються у базі даних, відображаються у правильній послідовності та містять інформацію про автора і час створення. Реалізований функціонал забезпечує ефективне обговорення звернень між користувачами вебплатформи.

За результатами проведеного тестування встановлено, що всі реалізовані функціональні можливості вебплатформи CityVoice працюють коректно та відповідають визначеним функціональним вимогам. Під час перевірки не було виявлено критичних помилок, які могли б вплинути на працездатність системи або призвести до втрати даних.

		Чмут Д.К.			ПІЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				79
Змн.	Арк.	№ докум.	Підпис	Дата		

Коментарі

Напишіть коментар...

Додати коментар

Шевченко Андрій Васильович

24.06.2026 21:21

Відповісти

Ця проблема вже пів року триває

Вподобано Лайків: 1

Романюк Богдан Володимирович

24.06.2026 21:22

Згоден

Лайків: 0

Шевченко Андрій Васильович

Видалити

Рис. 3.19 – Коментарі та відповіді користувачів

Висновки до третього розділу

У третьому розділі було розглянуто процес розгортання, налаштування та практичного використання розробленої вебплатформи CityVoice. Визначено мінімальні програмні вимоги до середовища виконання, описано порядок встановлення необхідних компонентів та запуску системи. Наведена діаграма розгортання демонструє взаємодію основних компонентів вебплатформи, зокрема клієнтського інтерфейсу, сервера застосунку Laravel, бази даних MySQL, файлового сховища та сервісу електронної пошти.

Також було представлено результати роботи основних функціональних модулів вебплатформи, серед яких реєстрація користувачів, підтвердження номера телефону, авторизація, відновлення пароля, створення та перегляд звернень, система голосування, коментування, модерація звернень, керування користувачами та формування PDF-звітів.

Окрему увагу приділено тестуванню вебплатформи. Проведене автоматизоване та функціональне тестування підтвердило коректність роботи реалізованих модулів, правильність взаємодії між компонентами системи та відсутність критичних помилок під час виконання основних сценаріїв використання.

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				80
Змн.	Арк.	№ докум.	Підпис	Дата		

ВИСНОВКИ

У кваліфікаційній роботі було розроблено вебплатформу CityVoice, призначену для подання, обговорення та опрацювання звернень громадян щодо проблем міської інфраструктури. Під час виконання роботи було проведено аналіз предметної області, визначено основні вимоги до програмного забезпечення та виконано порівняльний аналіз існуючих аналогів, що дозволило сформулювати функціональні вимоги до майбутньої інформаційної системи.

У процесі проєктування було розроблено структуру бази даних, визначено взаємозв'язки між основними сутностями та побудовано UML-діаграми[15], які описують архітектуру системи, алгоритми її роботи та життєвий цикл звернень. Це дозволило сформувати цілісну модель функціонування вебплатформи та забезпечити правильну реалізацію бізнес-логіки.

Практична частина роботи полягала у створенні вебплатформи з використанням фреймворку Laravel системи керування базами даних MySQL та сучасних вебтехнологій HTML CSS і JavaScript. Реалізовано систему реєстрації та авторизації користувачів, підтвердження номера телефону, відновлення пароля, створення звернень із можливістю додавання фотографій і відеофайлів, голосування за звернення, коментування, систему сповіщень, модерацію звернень, керування користувачами та формування PDF-звітів.

Проведене тестування підтвердило працездатність усіх основних функціональних можливостей вебплатформи. Отримані результати засвідчили коректність реалізації бізнес-логіки, стабільність роботи програмного забезпечення та правильність взаємодії між його компонентами.

Розроблена вебплатформа CityVoice може бути використана органами місцевого самоврядування або територіальними громадами для автоматизації процесу подання та опрацювання звернень громадян, забезпечення прозорості взаємодії між мешканцями та відповідальними особами, а також підвищення ефективності контролю за вирішенням актуальних проблем міської інфраструктури.

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		81

Перспективами подальшого розвитку вебплатформи є інтеграція з картографічними сервісами для відображення місця виникнення проблем, реалізація push-повідомлень, мобільного застосунку, підтримка електронного цифрового підпису, інтеграція із державними інформаційними системами та впровадження аналітичних інструментів для оцінювання ефективності роботи органів місцевого самоврядування.

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		82

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Закон України «Про звернення громадян» від 02.10.1996 №393/96-ВР [Електронний ресурс]. — 1996. — Режим доступу: <https://zakon.rada.gov.ua/laws/show/393/96-вр> (дата звернення: 17.05.2026).
2. Laravel documentation [Електронний ресурс]. — 2026. — Режим доступу: <https://laravel.com/docs> (дата звернення: 20.06.2026).
3. PHP documentation [Електронний ресурс]. — 2026. — Режим доступу: <https://www.php.net/docs.php> (дата звернення: 20.06.2026).
4. MySQL 8.0 Reference Manual [Електронний ресурс]. — 2026. — Режим доступу: <https://dev.mysql.com/doc/> (дата звернення: 20.06.2026).
5. Docker documentation [Електронний ресурс]. — 2026. — Режим доступу: <https://docs.docker.com/> (дата звернення: 10.05.2026).
6. Docker Compose documentation [Електронний ресурс]. — 2026. — Режим доступу: <https://docs.docker.com/compose/> (дата звернення: 10.05.2026).
7. Bootstrap documentation [Електронний ресурс]. — 2023. — Режим доступу: <https://getbootstrap.com/docs/5.3/> (дата звернення: 16.06.2026).
8. Composer documentation [Електронний ресурс]. — 2026. — Режим доступу: <https://getcomposer.org/doc/> (дата звернення: 28.06.2026).
9. Mailpit documentation [Електронний ресурс]. — 2026. — Режим доступу: <https://mailpit.axllent.org/docs/> (дата звернення: 28.06.2026).
10. WHATWG. HTML Living Standard [Електронний ресурс]. — 2026. — Режим доступу: <https://html.spec.whatwg.org/> (дата звернення: 28.06.2026).
11. W3C. CSS Cascading Style Sheets [Електронний ресурс]. — 2026. — Режим доступу: <https://www.w3.org/Style/CSS/> (дата звернення: 28.06.2026).
12. Mozilla Developer Network. JavaScript Guide [Електронний ресурс]. — 2026. — Режим доступу: <https://developer.mozilla.org/docs/Web/JavaScript> (дата звернення: 26.05.2026).
13. Mozilla Developer Network. MDN Web Docs [Електронний ресурс]. — 2026. — Режим доступу: <https://developer.mozilla.org/> (дата звернення: 24.05.2026).

		Чмут Д.К.			ПІЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				83
Змн.	Арк.	№ докум.	Підпис	Дата		

14. PlantUML documentation [Електронний ресурс]. — 2026. — Режим доступу: <https://plantuml.com/> (дата звернення: 23.06.2026).
15. Object Management Group. UML Resource Page [Електронний ресурс]. — 2017. — Режим доступу: <https://www.omg.org/spec/UML/> (дата звернення: 17.06.2026).
16. Git documentation [Електронний ресурс]. — 2026. — Режим доступу: <https://git-scm.com/doc> (дата звернення: 20.06.2026).
17. GitHub Docs [Електронний ресурс]. — 2026. — Режим доступу: <https://docs.github.com/en> (дата звернення: 20.06.2026).
18. Visual Studio Code documentation [Електронний ресурс]. — 2026. — Режим доступу: <https://code.visualstudio.com/docs> (дата звернення: 18.04.2026).
19. Node.js documentation [Електронний ресурс]. — 2026. — Режим доступу: <https://nodejs.org/docs/latest/api/> (дата звернення: 20.04.2026).
20. Npm documentation [Електронний ресурс]. — 2026. — Режим доступу: <https://docs.npmjs.com/> (дата звернення: 28.06.2026).
21. MySQL. MySQL 8.0 Reference Manual: Stored Objects [Електронний ресурс]. — 2026. — Режим доступу: <https://dev.mysql.com/doc/refman/8.0/en/stored-objects.html> (дата звернення: 20.06.2026).
22. MySQL. MySQL 8.0 Reference Manual: Using Foreign Keys [Електронний ресурс]. — 2026. — Режим доступу: <https://dev.mysql.com/doc/refman/8.0/en/example-foreign-keys.html> (дата звернення: 20.06.2026).
23. Atlassian. Software Development Life Cycle (SDLC) [Електронний ресурс]. — 2026. — Режим доступу: <https://www.atlassian.com/agile/software-development> (дата звернення: 15.06.2026).
24. Mozilla Developer Network. MVC [Електронний ресурс]. — 2026. — Режим доступу: <https://developer.mozilla.org/en-US/docs/Glossary/MVC> (дата звернення: 16.06.2026).

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				84
Змн.	Арк.	№ докум.	Підпис	Дата		

25. Microsoft. Monolithic application architecture [Електронний ресурс]. — 2025. — Режим доступу: <https://learn.microsoft.com/en-us/dotnet/architecture/modern-web-apps-azure/common-web-application-architectures> (дата звернення: 18.06.2026).
26. Refactoring.Guru. Design Patterns [Електронний ресурс]. — 2026. — Режим доступу: <https://refactoring.guru/design-patterns> (дата звернення: 10.06.2026).
27. Vite. Vite documentation [Електронний ресурс]. — 2026. — Режим доступу: <https://vitejs.dev/guide/> (дата звернення: 17.06.2026).
28. Lucidchart. Entity-Relationship Diagram (ERD) Tutorial [Електронний ресурс]. — 2026. — Режим доступу: <https://www.lucidchart.com/pages/er-diagrams> (дата звернення: 19.06.2026).
29. Refactoring.Guru. SOLID Principles [Електронний ресурс]. — 2026. — Режим доступу: <https://refactoring.guru/refactoring/what-is-refactoring> (дата звернення: 10.06.2026).
30. Laravel. Service Providers — Laravel documentation [Електронний ресурс]. — 2026. — Режим доступу: <https://laravel.com/docs/providers> (дата звернення: 13.06.2026).

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				85
Змн.	Арк.	№ докум.	Підпис	Дата		

ДОДАТКИ

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		86

web.php

```

<?php

use App\Http\Controllers\AdminController;
use App\Http\Controllers\AppealController;
use App\Http\Controllers\CommentController;
use App\Http\Controllers\CommentLikeController;
use App\Http\Controllers\HomeController;
use App\Http\Controllers\ProfileController;
use App\Http\Controllers\VoteController;
use Illuminate\Support\Facades\Route;

Route::get('/', [HomeController::class, 'index'])->name('home');

Route::get('/dashboard', [AppealController::class, 'dashboard'])
    ->middleware(['auth', 'verified', 'phone.verified'])
    ->name('dashboard');

Route::middleware(['auth', 'phone.verified'])->group(function () {
    Route::get('/appeals', [AppealController::class, 'index'])
        ->name('appeals.index');

    Route::get('/appeals/create', [AppealController::class, 'create'])
        ->name('appeals.create');

    Route::post('/appeals', [AppealController::class, 'store'])
        ->name('appeals.store');

    Route::delete('/appeals/{appeal}', [AppealController::class, 'destroy'])
        ->name('appeals.destroy');

    Route::get('/appeals/{appeal}/video', [AppealController::class, 'video'])
        ->name('appeals.video');

    Route::get('/appeals/{appeal}', [AppealController::class, 'show'])
        ->name('appeals.show');

    Route::post('/appeals/{appeal}/comments', [CommentController::class,
'store'])
        ->name('comments.store');

    Route::delete('/comments/{comment}', [CommentController::class, 'destroy'])
        ->name('comments.destroy');

    Route::post('/comments/{comment}/like', [CommentLikeController::class,
'toggle'])
        ->name('comments.likes.toggle');

    Route::post('/appeals/{appeal}/vote', [VoteController::class, 'toggle'])
        ->name('votes.toggle');

    Route::post('/notifications/mark-as-read', function () {
        auth()->user()->unreadNotifications->markAsRead();

        return response()->json([
            'success' => true,
        ]);
    })->name('notifications.markAsRead');

    Route::get('/profile', [ProfileController::class, 'edit'])
        ->name('profile.edit');

```

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				87
Змн.	Арк.	№ докум.	Підпис	Дата		

```

Route::patch('/profile', [ProfileController::class, 'update'])
->name('profile.update');

Route::delete('/profile', [ProfileController::class, 'destroy'])
->name('profile.destroy');

Route::middleware('admin_or_moderator')
->prefix('admin')
->name('admin.')
->group(function () {
    Route::get('/appeals', [AdminController::class, 'appeals'])
        ->name('appeals');

    Route::get('/appeals/completed', [AdminController::class,
'completedAppeals'])
        ->name('appeals.completed');

    Route::get('/appeals/completed/report', [AdminController::class,
'completedAppealsReport'])
        ->name('appeals.completed.report');

    Route::patch('/appeals/{appeal}/status', [AdminController::class,
'updateStatus'])
        ->name('appeals.updateStatus');

    Route::get('/users', [AdminController::class, 'users'])
        ->name('users');

    Route::get('/users/{user}', [AdminController::class, 'showUser'])
        ->name('users.show');

    Route::delete('/users/{user}', [AdminController::class,
'destroyUser'])
        ->name('users.destroy');
});

Route::middleware('admin')
->prefix('admin')
->name('admin.')
->group(function () {
    Route::delete('/appeals/{appeal}', [AdminController::class,
'destroy'])
        ->name('appeals.destroy');

    Route::patch('/users/{user}/role', [AdminController::class,
'updateUserRole'])
        ->name('users.updateRole');
});
});

require __DIR__.'/auth.php';

```

docker-compose.yml

```

services:
  app:
    build:
      context: .
      dockerfile: Dockerfile
    container_name: cityvoice_app
    ports:
      - "80:8000"
    volumes:
      - ../var/www/html
    depends_on:
      - db
      - mailpit

  db:
    image: mysql:8.0
    container_name: cityvoice_db
    ports:
      - "3307:3306"
    environment:
      MYSQL_DATABASE: cityvoice_db
      MYSQL_ROOT_PASSWORD: root
    volumes:
      - db_data:/var/lib/mysql

  mailpit:
    image: axllent/mailpit:latest
    container_name: cityvoice_mailpit
    ports:
      - "1025:1025"
      - "8025:8025"

volumes:
  db_data:

```

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		89

AppealController.php

```

<?php

namespace App\Http\Controllers;

use App\Models\Appeal;
use App\Models\Category;
use App\Models>Status;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Storage;
use Illuminate\Support\Str;

class AppealController extends Controller
{
    public function dashboard()
    {
        $user = Auth::user();

        $appeals = Appeal::with([
            'category',
            'status',
            'comments.user',
            'votes',
        ])
            ->where('user_id', $user->id)
            ->latest()
            ->get();

        $notifications = $user
            ->notifications()
            ->latest()
            ->take(10)
            ->get();

        $unreadNotificationsCount = $user
            ->unreadNotifications()
            ->count();

        return view('dashboard', compact(
            'appeals',
            'notifications',
            'unreadNotificationsCount'
        ));
    }

    public function index(Request $request)
    {
        $query = Appeal::query()
            ->with([
                'category',
                'status',
                'user',
                'votes',
            ])
            ->withCount('comments')
            ->latest();

        if ($request->filled('search')) {
            $search = trim((string) $request->input('search'));
        }
    }
}

```

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				90
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        $query->where(function ($query) use ($search) {
            $query
                ->where('title', 'like', '%' . $search . '%')
                ->orWhere('description', 'like', '%' . $search . '%');
        });
    }

    if ($request->filled('category_id')) {
        $query->where('category_id', $request->integer('category_id'));
    }

    if ($request->filled('status_id')) {
        $query->where('status_id', $request->integer('status_id'));
    }

    $appeals = $query->paginate(10)->withQueryString();
    $categories = Category::orderBy('name')->get();
    $statuses = Status::orderBy('name')->get();

    return view('appeals.index', compact(
        'appeals',
        'categories',
        'statuses'
    ));
}

public function show(Appeal $appeal)
{
    $appeal->load([
        'category',
        'status',
        'user',
        'votes',
        'comments' => function ($query) {
            $query
                ->whereNull('parent_id')
                ->oldest()
                ->with([
                    'user',
                    'likes',
                    'replies' => function ($replyQuery) {
                        $replyQuery
                            ->oldest()
                            ->with([
                                'user',
                                'replyToUser',
                                'likes',
                            ]);
                    },
                ]);
        }
    ]);

    if (Auth::check()) {
        foreach (Auth::user()->unreadNotifications as $notification) {
            if (($notification->data['appeal_id'] ?? null) == $appeal->id) {
                $notification->markAsRead();
            }
        }
    }

    return view('appeals.show', compact('appeal'));
}

```

```

public function video(Appeal $appeal)
{
    if (!$appeal->video || !Storage::disk('public')->exists($appeal->video))
    {
        abort(404);
    }

    $path = Storage::disk('public')->path($appeal->video);
    $size = filesize($path);
    $start = 0;
    $end = $size - 1;
    $status = 200;

    $extension = strtolower(pathinfo($path, PATHINFO_EXTENSION));

    $mimeType = match ($extension) {
        'mp4' => 'video/mp4',
        'webm' => 'video/webm',
        'mov' => 'video/quicktime',
        default => 'application/octet-stream',
    };

    $headers = [
        'Content-Type' => $mimeType,
        'Accept-Ranges' => 'bytes',
        'Content-Disposition' => 'inline; filename="' . basename($path) .
        '"',
        'Cache-Control' => 'public, max-age=3600',
    ];

    $range = request()->header('Range');

    if ($range && preg_match('/bytes=(\d*)-(\d*)/', $range, $matches)) {
        $status = 206;

        if ($matches[1] !== '') {
            $start = (int) $matches[1];
        }

        if ($matches[2] !== '') {
            $end = (int) $matches[2];
        }

        if ($start > $end || $start >= $size) {
            return response('', 416, [
                'Content-Range' => 'bytes */' . $size,
            ]);
        }

        $end = min($end, $size - 1);

        $headers['Content-Range'] = "bytes {$start}-{$end}/{ $size}";
    }

    $length = $end - $start + 1;
    $headers['Content-Length'] = $length;

    return response()->stream(function () use ($path, $start, $end) {
        $stream = fopen($path, 'rb');

        if ($stream === false) {
            return;
        }
    });
}

```

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				92
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        fseek($stream, $start);

        $position = $start;
        $chunkSize = 1024 * 1024;

        while (!feof($stream) && $position <= $end) {
            $bytesToRead = min($chunkSize, $end - $position + 1);
            $buffer = fread($stream, $bytesToRead);

            if ($buffer === false) {
                break;
            }
            echo $buffer;
            flush();
            $position += strlen($buffer);
        }

        fclose($stream);
    }, $status, $headers);
}

public function create()
{
    if (Auth::user()?->isModerator()) {
        abort(403, 'Модератор не може створювати звернення.');
```

```

    }

    $categories = Category::orderBy('name')->get();

    $formToken = (string) Str::uuid();
    session(['appeal_form_token' => $formToken]);

    return view('appeals.create', compact('categories', 'formToken'));
}

public function store(Request $request)
{
    if (Auth::user()?->isModerator()) {
        abort(403, 'Модератор не може створювати звернення.');
```

```

    }

    $validated = $request->validate([
        'form_token' => ['required', 'string'],
        'title' => ['required', 'string', 'max:255'],
        'description' => ['required', 'string'],
        'category_id' => ['required', 'exists:categories,id'],
        'image' => [
            'nullable',
            'image',
            'mimes:jpg,jpeg,png,webp',
            'max:2048',
        ],
        'video' => [
            'nullable',
            'file',
            'mimes:mp4,webm',
            'max:51200',
        ],
    ], [
        'video.mimes' => 'Відео має бути у форматі MP4 або WEBM.',
        'video.max' => 'Розмір відео не повинен перевищувати 50 МБ.',
    ]);

    $sessionToken = session()->pull('appeal form token');
```

```

        if (!$sessionToken || !hash_equals($sessionToken,
$validated['form_token'])) {
            return redirect()
                ->route('appeals.index')
                ->with('error', 'Звернення вже було оброблено або форма
застаріла.');
```

```

        }

        $imagePath = $request->hasFile('image')
            ? $request->file('image')->store('appeals', 'public')
            : null;

        $videoPath = $request->hasFile('video')
            ? $request->file('video')->store('appeals/videos', 'public')
            : null;

        $newStatus = Status::where('name', 'Нове')->firstOrFail();

        Appeal::create([
            'title' => $validated['title'],
            'description' => $validated['description'],
            'user_id' => Auth::id(),
            'category_id' => $validated['category_id'],
            'status_id' => $newStatus->id,
            'image' => $imagePath,
            'video' => $videoPath,
        ]);

        return redirect()
            ->route('appeals.index')
            ->with('success', 'Звернення успішно створено.');
```

```

    }

    public function destroy(Appeal $appeal)
    {
        $appeal->load('status');

        if (!$appeal->isOwnedBy(Auth::user())) {
            abort(403);
        }

        if ($appeal->status->name === 'Завершено') {
            return redirect()
                ->back()
                ->with('error', 'Завершене звернення не можна видалити.');
```

```

        }

        if ($appeal->image) {
            Storage::disk('public')->delete($appeal->image);
        }

        if ($appeal->video) {
            Storage::disk('public')->delete($appeal->video);
        }

        $appeal->delete();

        return redirect()
            ->route('dashboard')
            ->with('success', 'Звернення видалено.');
```

```

    }
}

```

		Чмут Д.К.			ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Єфремов. Ю.М.				94
Змн.	Арк.	№ докум.	Підпис	Дата		