

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 пошуку дозвілля»

Виконала студентка групи ІПЗ-22-4
ДЕМЧУК Олеся Володимирівна

Керівник роботи:
ЧИЖМОТРИЯ Олексій Володимирович

Рецензент:
МОРОЗОВ Андрій Васильович

Житомир – 2026

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
«27» лютого 2026 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **ДЕМЧУК Олеся Володимирівна**

Керівник роботи: **ЧИЖМОТРЯ Олексій Володимирович**

Тема роботи: **«Вебплатформа для пошуку дозвілля»,**

затверджена Наказом закладу вищої освіти від «27» лютого 2026 р., №92/с

Вихідні дані для роботи: **розробка вебплатформи для пошуку дозвілля, що забезпечує індивідуальний підбір для користувача, проходження маршрутного шляху, відстеження прогресу користувача, взаємодію у спільноті та проведення заходів у межах однієї платформи.**

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

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Чижмотря О.В.	01.03.2026р.	01.03.2026р.
2	Чижмотря О.В.	01.03.2026р.	01.03.2026р.
3	Чижмотря О.В.	01.03.2026р.	01.03.2026р.

РЕФЕРАТ

Випускна кваліфікаційна робота бакалавра складається з програмного продукту у вигляді вебплатформи для пошуку дозвілля та пояснювальної записки. Робота складається зі вступу, 3-х розділів, висновків, списку використаних джерел із 32 найменувань (3 сторінки), 1 додатку (2 сторінки), 29 таблиць та 54 ілюстрацій по тексту. Повний обсяг робіт становить 103 сторінки, з них 100 сторінок основного тексту.

Метою роботи є створення вебплатформи, де користувач може підібрати за алгоритмом, вручну або випадково захоплення для подальшого розвитку, отримати покроковий маршрут для його освоєння та спостерігати за своїми успіхами на одній платформі, також є можливість стати практиком та заповнювати потрібну інформацію, запланувати події, додавати нові вузли в маршрутний шлях, публікувати пости у закріпленому за ним дозвіллі.

У ході роботи було сформовано перелік ключових завдань, проведено аналіз тематичних аналогів та виявлено їх функціональні обмеження та переваги для подальшого вдосконалення платформи. Також було описано взаємодію з системою користувач в залежності від його ролі.

В результаті сформовано список вимог до вебплатформи, що забезпечує каталог дозвілля, алгоритм вибору, формування відповідного маршруту для засвоєння певного захоплення, прогресу користувача та взаємодію спільноти між собою.

КЛЮЧОВІ СЛОВА: ВЕБПЛАТФОРМА, ДОЗВІЛЛЯ, КАТАЛОГ, ПУБЛІКАЦІЇ, СПИСОК РЕКОРДІВ, ІГРОФІКАЦІЯ, АДМІНІСТРУВАННЯ, ПОКРОКОВИЙ МАРШРУТ, СПІЛЬНОТА, ПОДІЇ.

					ІПЗ.КР.Б – 121 – 26 – ПЗ							
Змн.	Арк.	№ докум.	Підпис	Дата								
Розроб.		Демчук О.В.			«Вебплатформа для пошуку дозвілля»				Літ.		Арк.	Аркушів
Керівник		Чижмостря О.В.									3	103
Рецензент		Морозов А.В.			Пояснювальна записка				Житомирська політехніка, група ІПЗ-22-4			
Зав. каф.		Вакалюк Т.А.										

ABSTRACT

The bachelor's qualification thesis consists of a software product in the form of a web platform for discovering leisure activities and hobbies, as well as an explanatory report. The thesis includes an introduction, three chapters, conclusions, a list of references containing 32 sources (3 pages), one appendix (2 pages), 29 tables, and 54 figures throughout the text. The total volume of the thesis is 103 pages, including 100 pages of the main text.

The purpose of the thesis is to develop a web platform where users can select a hobby for further personal development through an algorithm-based recommendation system, manual selection, or random choice, receive a step-by-step roadmap for mastering the chosen hobby, and track their progress within a single platform. The platform also enables users to become practitioners, contribute relevant information, organize events, add new nodes to learning roadmaps, and publish posts related to their assigned hobbies.

During the development process, a list of key objectives was defined, thematic analogs were analyzed, and their functional advantages and limitations were identified to improve the proposed platform. In addition, user interaction with the system was described according to different user roles.

As a result, a set of requirements for the web platform was formulated. The platform provides a hobby catalog, a recommendation and selection algorithm, generation of personalized learning roadmaps for mastering specific hobbies, user progress tracking, and community interaction features.

KEYWORDS: WEB PLATFORM, LEISURE ACTIVITIES, CATALOG, PUBLICATIONS, LEADERBOARD, GAMIFICATION, ADMINISTRATION, STEP-BY-STEP ROADMAP, COMMUNITY, EVENTS.

		Демчук О.В.			ІІЗ.КР.Б – 121 – 26 – ІІЗ	Арк.
		Чижмоторя О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		4

ЗМІСТ

РЕФЕРАТ	3
ABSTRACT	4
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ	6
ВСТУП	8
РОЗДІЛ 1. АНАЛІЗ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ...	10
1.1 Загальна характеристика предметної галузі	10
1.2 Аналіз аналогів програмного продукту	13
1.3 Вибір архітектури системи	18
1.4 Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення	21
Висновки до першого розділу	24
РОЗДІЛ 2. ПРОЄКТУВАННЯ ВЕБПЛАТФОРМИ ДЛЯ ПОШУКУ	
ДОЗВІЛЛЯ	26
2.1 Визначення варіантів використання та об'єктно-орієнтованої структури системи	26
2.2 Розробка бази даних системи	36
2.3 Проєктування та реалізація алгоритмів роботи системи	48
2.4 Реалізація основних функціональних модулів системи	58
Висновки до другого розділу	64
РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ВЕБПЛАТФОРМОЮ	
ДЛЯ ПОШУКУ ДОЗВІЛЛЯ	66
3.1 Порядок розгортання та налаштування системи	66
3.2 Опис інтерфейсу вебплатформи	69
3.3 Тестування програмного комплексу	89
Висновки до третього розділу	94
ВИСНОВКИ	96
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	98
ДОДАТКИ	101
Додаток А	102

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

API (Application Programming Interface) – інтерфейс прикладного програмування, набір визначень, протоколів та інструментів для взаємодії між клієнтською та серверною частинами вебплатформи.

CSS (Cascading Style Sheets) – каскадні таблиці стилів, що використовуються для опису зовнішнього вигляду та оформлення інтерфейсу вебзастосунку.

CI/CD (Continuous Integration / Continuous Delivery) – неперервна інтеграція та неперервна доставка, практика автоматизованого складання, тестування та розгортання програмного забезпечення.

CRUD (Create, Read, Update, Delete) – чотири базові операції роботи з даними: створення, читання, оновлення та видалення.

HTTP / HTTPS (Hypertext Transfer Protocol / Secure) – протокол передачі гіпертексту (захищений), що забезпечує зв'язок та безпечний обмін даними у форматі JSON між клієнтом і сервером.

HTML (Hypertext Markup Language) – мова розмітки гіпертексту, що використовується для структурування та відображення контенту вебсторінок.

JSON (JavaScript Object Notation) – текстовий формат обміну даними, який використовується для передачі інформації між REST API сервера та клієнтською частиною.

JSONB (JSON Binary) – бінарний формат зберігання JSON-документів у PostgreSQL, що забезпечує ефективне індексування та пошук по структурованих полях.

JWT (JSON Web Token) – компактний засіб безпечної передачі тверджень між двома сторонами, що використовується для автентифікації та авторизації користувачів на платформі.

MSW (Mock Service Worker) – інструмент перехоплення HTTP-запитів на рівні Service Worker, що використовується для імітації API під час тестування клієнтської частини застосунку.

ORM (Object-Relational Mapping) – технологія програмування, яка зв'язує бази даних з концепціями об'єктно-орієнтованих мов програмування.

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижомотря О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		6

OWASP (Open Web Application Security Project) – відкритий проєкт з безпеки вебзастосунків, що надає рекомендації та стандарти захисту.

REST (Representational State Transfer) – архітектурний стиль взаємодії компонентів розподіленого застосунку за допомогою HTTP-запитів.

RFC (Request for Comments) – стандарт, що публікується організацією IETF.

SQL (Structured Query Language) – мова структурованих запитів, що використовується для керування реляційними базами даних та маніпулювання даними.

SPA (Single Page Application) – односторінковий застосунок, архітектурний підхід до побудови вебзастосунків, за якого початкова сторінка завантажується один раз, а подальша навігація здійснюється без перезавантаження браузера.

SSD (Solid-State Drive) – твердотільний накопичувач, тип швидкого енергонезалежного комп'ютерного сховища, необхідний для розгортання серверної частини системи.

UML (Unified Modeling Language) – уніфікована мова моделювання, що використовується для графічного опису архітектури, варіантів використання та алгоритмів системи.

URL (Uniform Resource Locator) – єдиний вказівник ресурсу, адреса вебсторінки або медіафайлу в мережі Інтернет.

UUID (Universally Unique Identifier) – універсальний унікальний ідентифікатор довжиною 128 біт, що використовується як первинний ключ для сутностей у базі даних PostgreSQL.

XP (Experience Points) – бали досвіду, віртуальні одиниці, що нараховуються користувачеві за виконання вузлів маршруту та використовуються для гейміфікації (ігрофікації) і побудови таблиці лідерів.

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмолтя О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		7

ВСТУП

Актуальність теми полягає в тому, що на сьогоднішній день інтернет пропонує величезну кількість матеріалів про різні види дозвілля, однак процес вибору та початку нового заняття залишається складним і неструктурованим. Користувач змушений самотійно шукати інформацію на різноманітних платформах, що призводить до інформаційного перевантаження та ускладнює прийняття рішення. Наявні вебсервіси, як правило, орієнтовані на окремі задачі: одні пропонують навчальні курси, інші зосереджені на пошуку подій або соціальній взаємодії. Жоден із них не забезпечує комплексного підходу, який би охоплював увесь шлях користувача від вибору заняття до його поетапного освоєння з відстеженням власного прогресу. Відсутність такого рішення знижує мотивацію та підвищує ймовірність того, що людина залишить нове захоплення вже на початковому етапі. У зв'язку з цим розробка вебплатформи, яка об'єднує підбір дозвілля, структурований маршрут його освоєння та інструменти взаємодії між користувачами в межах єдиного середовища, є актуальною та практично значущою задачею.

Метою роботи є розробка вебплатформи, що забезпечує персоналізований підбір дозвілля на основі інтересів користувача, поетапне проходження структурованого маршруту освоєння обраного заняття, збереження прогресу та взаємодію між учасниками платформи.

Для досягнення поставленої мети необхідно вирішити ряд завдань. У межах роботи планується дослідити предметну область пошуку та освоєння нових занять, проаналізувати наявні програмні рішення та визначити їх функціональні обмеження. На підставі отриманих результатів передбачається сформулювати вимоги до системи, обґрунтувати архітектурні рішення та спроєктувати структуру бази даних. Практична частина включає реалізацію гібридного алгоритму рекомендацій, модуля покрокового маршруту з відстеженням прогресу та системи ролей користувачів, а також проведення тестування розроблених функціональних модулів.

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ІЗ	Арк.
		Чижомотря О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		8

Об’єктом дослідження є процес пошуку та освоєння нових захоплень користувачем у цифровому середовищі.

Предметом дослідження є методи, моделі та програмні засоби, що застосовуються при створенні вебплатформи для персоналізованого підбору дозвілля та організації його поетапного освоєння з можливістю відстеження результатів.

Методи дослідження. У роботі використано методи системного аналізу при вивченні предметної області та формуванні вимог, об’єктно-орієнтованого проєктування при побудові архітектури та моделюванні структури даних, а також інструменти візуального моделювання на основі стандарту UML для опису сценаріїв взаємодії користувачів із системою та алгоритмів її роботи.

Практичне значення отриманих результатів. Розроблена вебплатформа являє собою повноцінний застосунок із чотирма рівнями доступу, що включають гостя, зареєстрованого користувача, практика та адміністратора. Система поєднує каталог захоплень із підбором через гібридний алгоритм рекомендацій, покроковий маршрут освоєння з нарахуванням досвіду та системою досягнень, стрічку публікацій із модерацією контенту практиками, систему подій та інструменти відстеження прогресу у вигляді теплової карти активності. Результати роботи можуть бути використані як готове рішення для людей, що прагнуть знайти нове заняття та займатися ним системно.

Апробація. За темою кваліфікаційної роботи було опубліковано 2 тези конференцій [1, 2].

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоторя О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		9

РОЗДІЛ 1. АНАЛІЗ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

1.1 Загальна характеристика предметної галузі

Сфера організації дозвілля в умовах розвитку цифрових технологій зазнала суттєвих змін. Сьогодні користувач має доступ до великої кількості інформації про різні види занять та способи проведення вільного часу, зокрема навчальні ресурси, тематичні спільноти, платформи з подіями або окремі інформаційні сайти. Попри таке різноманіття, процес пошуку та освоєння дозвілля залишається недостатньо структурованим.

Користувач змушений самостійно працювати з великою кількістю джерел, окремо шукаючи ідеї, матеріали для навчання та можливості для спілкування з іншими людьми. Окремі етапи цього процесу не пов'язані між собою і не формують єдиного цілісного маршруту, що ускладнює досягнення реального результату та знижує загальну ефективність. Цифрове середовище, попри свої можливості, не лише не спрощує цей процес, а подекуди навпаки ускладнює його через надмір пропозицій без чіткого орієнтира.

Найбільш складним залишається початковий етап вибору виду дозвілля. У більшості випадків людина не має чітко сформованих інтересів і стикається з надлишком інформації, яка не допомагає прийняти обґрунтоване рішення. Пошуковий запит у мережі повертає десятки різних варіантів без урахування індивідуальних особливостей, рівня підготовки чи наявного часу конкретного користувача. Навіть після того як вибір зроблено, подальші дії залишаються невизначеними, оскільки немає чіткої послідовності кроків для освоєння обраного заняття. Відсутність структурованого підходу на цьому етапі є однією з головних причин того, що більшість людей припиняють спроби вже на початку.

Проблема вибору ускладнюється ще й тим, що більшість доступних ресурсів орієнтовані на вже визначену аудиторію. Навчальні платформи, як правило, пропонують курси для тих, хто вже знає, чим хоче займатися.

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоштря О.В.				10
Змн.	Арк.	№ докум.	Підпис	Дата		

Тематичні спільноти об'єднують людей, які вже мають досвід у певному напрямі. Таким чином, людина на самому початку шляху, яка ще не визначилась із вибором, залишається без належної підтримки.

Після вибору заняття користувач стикається з наступною проблемою, яка полягає у відсутності інструментів для організації власного навчального процесу. Матеріали розкидані по різних джерелах, немає чіткого порядку їх опрацювання, а прогрес ніде не фіксується. Людина не розуміє, на якому етапі вона знаходиться і що робити далі, що негативно впливає на мотивацію продовжувати. Без чіткого орієнтира та відчуття руху вперед інтерес до нового заняття швидко згасає, і людина повертається до звичного розпорядку, так і не досягнувши жодного помітного результату.

Важливо також враховувати психологічний аспект початку нового заняття. Відчуття невизначеності, брак зрозумілого плану та неможливість оцінити власний прогрес є суттєвими бар'єрами, які заважають людині рухатися далі. Теорія самодетермінації Десі та Раяна [3] встановлює, що внутрішня мотивація людини суттєво зростає за умови задоволення трьох базових психологічних потреб: автономії, відчуття компетентності та належності до спільноти. Бандура [4] доповнює це поняттям самоефективності, яка безпосередньо впливає на наполегливість у навчанні та готовність долати труднощі. Структурований маршрут із видимим прогресом задовольняє обидва ці чинники та знижує ймовірність того, що людина кине нову справу на початковому етапі.

Важливу роль у підтримці інтересу до нового заняття відіграє спільнота. Можливість ділитися результатами, отримувати зворотний зв'язок та спостерігати за досвідом інших учасників суттєво підвищує залученість і бажання розвиватися далі. Проте більшість наявних платформ або взагалі не мають соціальної складової, або реалізують її окремо від навчального процесу, що не дозволяє повноцінно поєднати ці два аспекти. У результаті соціальна взаємодія і навчання існують на різних платформах і не утворюють єдиного середовища, яке підтримувало б користувача комплексно.

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижомотря О.В.				11
Змн.	Арк.	№ докум.	Підпис	Дата		

Окремої уваги заслуговує питання верифікації досвідчених учасників. На багатьох платформах будь-який користувач може публікувати контент без перевірки його якості, що знижує довіру до матеріалів і ускладнює вибір надійних джерел. Для новачка, який тільки починає освоювати нове заняття, це є суттєвою проблемою, оскільки він ще не має достатнього досвіду, щоб самостійно оцінити якість інформації. Наявність механізму підтвердження статусу практика дозволяє вирізняти перевірених учасників і формувати більш якісне середовище для навчання. Такий підхід підвищує довіру до контенту на платформі та забезпечує користувачам доступ до перевіреної інформації від людей із реальним досвідом у певному напрямі.

Окремим аспектом предметної галузі є організація подій, пов'язаних із конкретними видами дозвілля. Онлайн та офлайн заходи є важливим інструментом для розвитку спільноти та обміну досвідом між учасниками. Проте більшість платформ або не мають власного інструменту для організації подій, або реалізують його окремо від навчального контенту. Інтеграція системи подій у єдине середовище разом із каталогом дозвілля та маршрутом освоєння дозволяє створити більш цілісний досвід для користувача.

Таким чином, предметна галузь характеризується наявністю значної кількості розрізнених інформаційних ресурсів, які не інтегровані між собою. Відсутність єдиного середовища, що охоплює вибір, структуроване освоєння дозвілля та взаємодію між учасниками, створює суттєві труднощі для користувача на кожному етапі цього процесу.

У зв'язку з цим виникає необхідність у розробці вебплатформи, яка об'єднає всі ці складові в межах єдиної системи, забезпечить персоналізований підбір дозвілля на основі індивідуальних інтересів користувача, надасть структурований маршрут його освоєння з можливістю відстеження прогресу та підтримає взаємодію між учасниками на всіх етапах роботи з платформою.

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоштра О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		12

1.2 Аналіз аналогів програмного продукту

У сфері пошуку та освоєння нових захоплень існує ряд вебсервісів, що частково вирішують окремі задачі підбору дозвілля або взаємодії між користувачами. З метою визначення обмежень наявних рішень та обґрунтування доцільності розробки власної платформи було проведено аналіз найбільш релевантних аналогів, а саме HobbyCue, Hobbynes та HobbyStack.

HobbyCue (рис. 1.1) є вебплатформою, що орієнтована на пошук занять і подій за інтересами. Основною метою сервісу є допомога користувачам у знаходженні нових захоплень у своєму регіоні та встановленні контактів із людьми зі схожими інтересами. Платформа надає можливість перегляду подій, приєднання до груп і спілкування в межах спільноти. Інтерфейс сервісу побудований за принципом соціальної мережі, де користувачі можуть створювати публікації, підписуватись на тематичні групи та переглядати активність інших учасників.

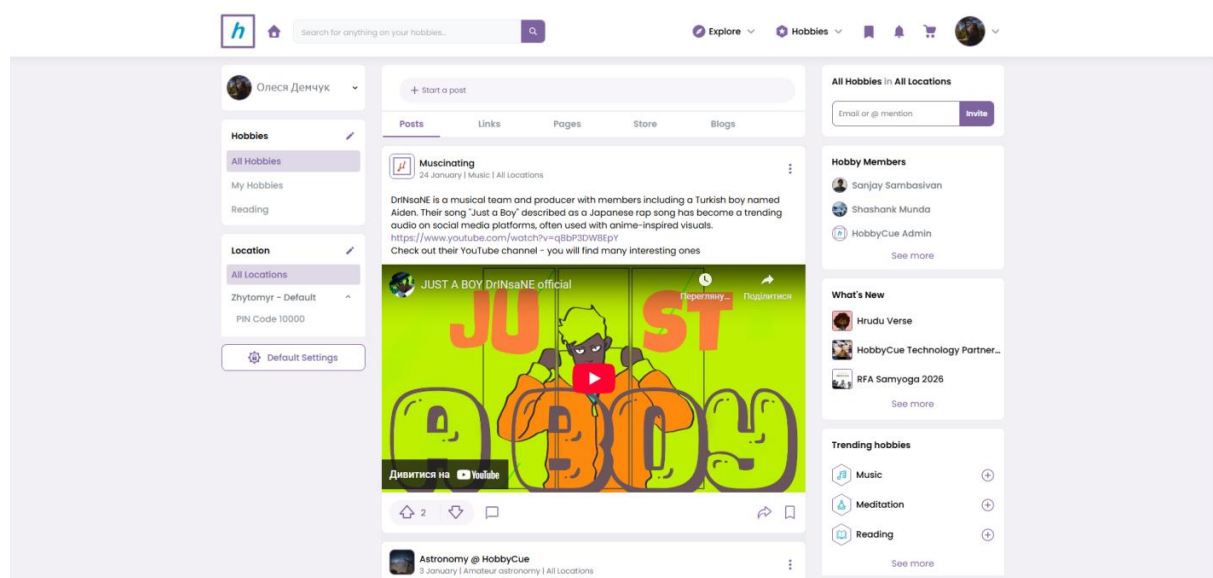


Рис. 1.1. Інтерфейс HobbyCue

До переваг HobbyCue належить простота використання та акцент на соціальній взаємодії між учасниками. Платформа дає змогу швидко знайти захід або групу за інтересами та встановити контакт із людьми, які

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижевотря О.В.				13
Змн.	Арк.	№ докум.	Підпис	Дата		

захоплюються тим самим. Наявність географічної прив'язки дозволяє шукати події поблизу місця проживання користувача, що робить сервіс зручним для пошуку офлайн-заходів.

Водночас платформа має суттєві обмеження з точки зору підтримки користувача на етапі вибору та освоєння дозвілля. Відсутній будь-який механізм персоналізованого підбору заняття відповідно до інтересів та можливостей конкретної людини. Після знаходження групи чи події користувач не отримує жодного структурованого маршруту подальших дій. Платформа не передбачає відстеження прогресу, системи мотивації або інструментів для поетапного освоєння обраного напрямку, що робить її інструментом переважно для пошуку соціальних контактів і подій, а не повноцінним середовищем для розвитку. Модерація контенту на платформі також відсутня, тому якість публікацій залежить виключно від самих учасників.

Hobbynes (рис. 1.2) позиціонується як сервіс для пошуку нових захоплень і взаємодії між людьми зі спільними інтересами. Платформа дозволяє переглядати дозвілля за категоріями, знаходити однодумців і долучатися до тематичних груп. Структура сервісу побудована навколо спільнот, де кожен вид дозвілля має власну сторінку з учасниками та можливістю обговорення.

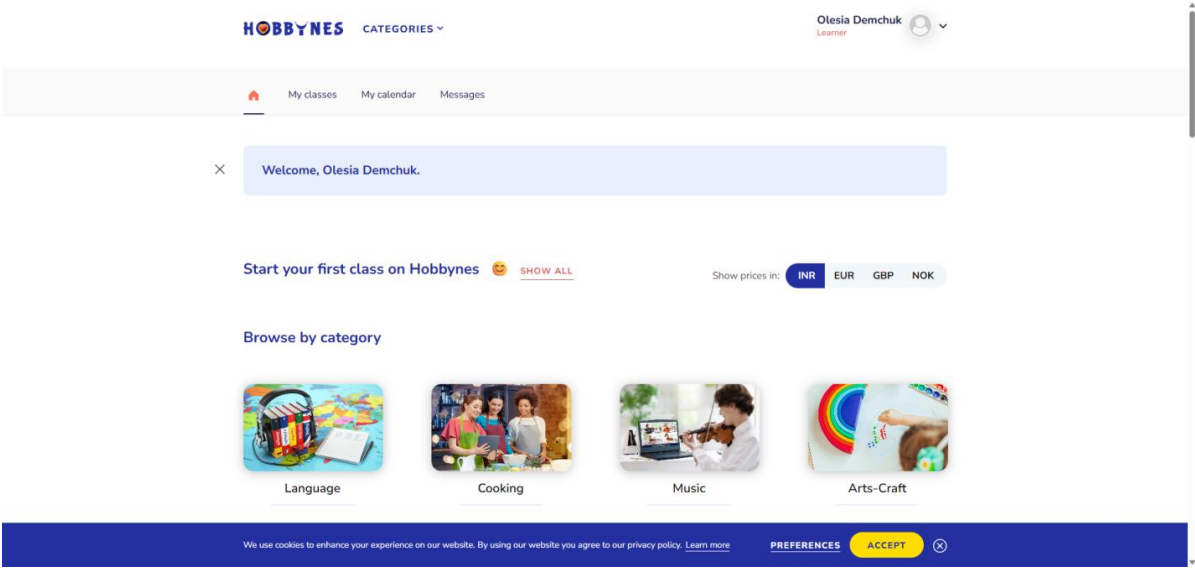


Рис. 1.2. Інтерфейс Hobbynes

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижомотря О.В.				14
Змн.	Арк.	№ докум.	Підпис	Дата		

Серед переваг Hobbynes варто відзначити орієнтацію на формування спільнот навколо конкретних занять та зручний перегляд каталогу за тематичними категоріями. На відміну від інших аналогів, платформа має систему верифікованих практиків, які підтверджують свій досвід у певному напрямі, а також часткову модерацію контенту. Користувач може отримати загальне уявлення про певне захоплення та знайти людей, які вже ним займаються. Сервіс також частково реалізує можливість публікацій користувачів та анонси подій у межах окремих спільнот.

Разом із тим Hobbynes не містить механізму підбору дозвілля відповідно до особистих інтересів користувача. Покроковий маршрут освоєння та відстеження прогресу реалізовані лише частково і не утворюють єдиної структурованої системи, яка б послідовно вела користувача від вибору заняття до його реального освоєння.

HobbyStack (рис. 1.3) є безкоштовним вебсервісом для пошуку занять, що містить структуровану базу захоплень із детальними описами, практичними рекомендаціями та тестом для підбору дозвілля. Платформа вирізняється серед аналогів наявністю алгоритму підбору, що враховує особисті характеристики користувача, бюджет, доступний час та формат занять. Кожне захоплення в каталозі має детальну сторінку з описом, орієнтовною вартістю старту та рекомендаціями для початківців.

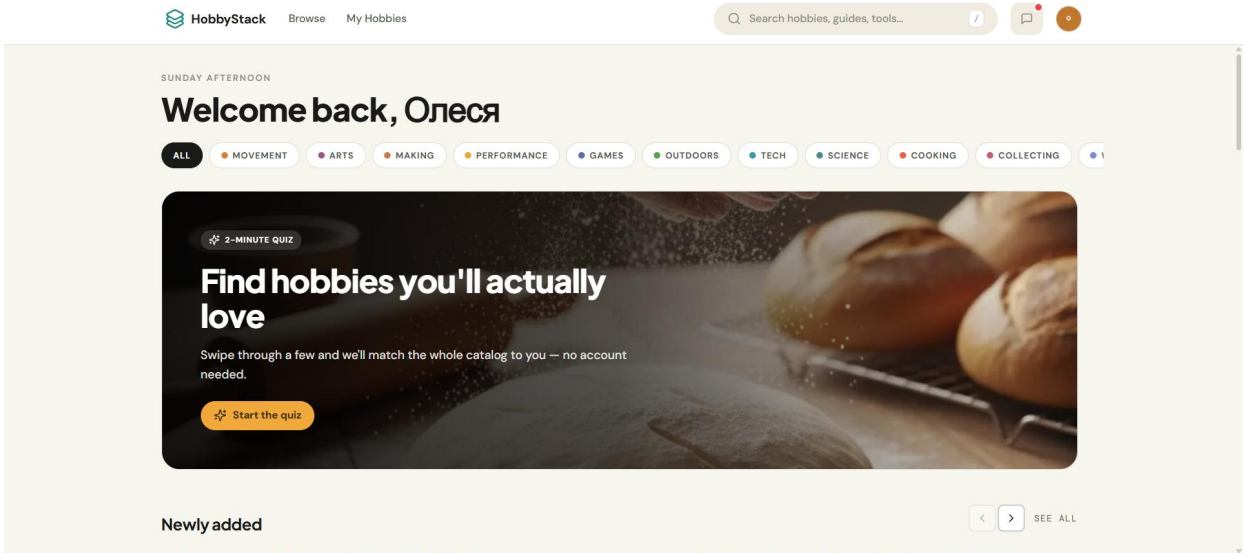


Рис. 1.3. Інтерфейс HobbyStack

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ІЗ	Арк.
		Чижмоштра О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		15

До переваг HobbyStack належить якісно структурований каталог і наявність механізму підбору, що дає змогу отримати персоналізовані рекомендації без реєстрації. Сервіс орієнтований на людей, які тільки починають шукати нове заняття і не мають чіткого уявлення про те, що їм підійде. Базові рекомендації для початківців на сторінках захоплень можна розглядати як часткову реалізацію навчального маршруту, а елементи відстеження активності присутні у вигляді збереження результатів тесту.

Проте HobbyStack завершує свою роботу фактично на етапі рекомендації. Після того як користувач отримав підбірку захоплень, платформа не пропонує інструментів для подальшої роботи з обраним напрямом. Спільнота реалізована лише частково у вигляді сторінок захоплень без повноцінної взаємодії між учасниками. Публікації користувачів, модерація контенту та верифікація практиків на платформі відсутні, що обмежує її застосування як повноцінного середовища для розвитку.

На основі проведеного аналізу було сформовано порівняльну таблицю основних характеристик розглянутих програмних платформ (табл. 1.1).

Таблиця 1.1

Порівняльна таблиця основних характеристик аналогів

Характеристика	HobbyCue	Hobbynes	HobbyStack
Каталог захоплень	Реалізовано	Реалізовано	Реалізовано
Пошук та фільтрація	Реалізовано	Реалізовано	Реалізовано
Спільнота та взаємодія	Реалізовано	Реалізовано	Частково
Система подій	Реалізовано	Частково	Не реалізовано
Підбір за інтересами	Не реалізовано	Не реалізовано	Реалізовано
Покроковий маршрут	Не реалізовано	Частково	Частково

Відстеження прогресу	Не реалізовано	Частково	Частково
Публікації користувачів	Реалізовано	Частково	Не реалізовано
Модерація контенту	Не реалізовано	Частково	Не реалізовано
Верифіковані практики	Не реалізовано	Реалізовано	Не реалізовано

Аналіз даних показує, що кожна з розглянутих платформ вирішує лише окремий аспект взаємодії користувача із захопленням. HobbyCue має найрозвиненішу соціальну складову та повноцінну систему подій, проте повністю позбавлений інструментів персоналізованого підбору дозвілля, структурованого маршруту освоєння та модерації контенту. Hobbynes вирізняється серед аналогів наявністю верифікованих практиків і частковою модерацією, однак покроковий маршрут та відстеження прогресу реалізовані недостатньо і не формують єдиної системи супроводу користувача. HobbyStack є єдиним із аналогів, що має повноцінний алгоритм підбору за інтересами, але після отримання рекомендації платформа фактично завершує свою роботу, не пропонуючи ані спільноти, ані маршруту освоєння, ані публікацій.

Застосування таких рішень на практиці змушує користувача одночасно користуватися кількома різними сервісами, щоб закрити всі свої потреби. Підбір дозвілля, його освоєння та спілкування з іншими учасниками реалізуються на різних платформах, що ускладнює організацію процесу та не дає цілісного розуміння власного прогресу. Відсутність верифікованих практиків у більшості сервісів додатково знижує якість інформаційного середовища, оскільки контент формується без перевірки достовірності.

Проведений аналіз підтверджує доцільність розробки платформи, яка об'єднує персоналізований підбір дозвілля, структурований покроковий маршрут його освоєння, відстеження прогресу, систему верифікованих

практиків із модерацією контенту та інструменти взаємодії між учасниками спільноти в межах єдиного середовища. Саме на усунення виявлених обмежень наявних аналогів і спрямована розробка запропонованої програмної системи.

1.3 Вибір архітектури системи

Вибір архітектури є одним із ключових етапів проєктування вебплатформи, оскільки саме вона визначає спосіб організації компонентів системи, принципи їх взаємодії та можливості подальшого розвитку. Для вебзастосунку, що працює з користувацькими даними, алгоритмами підбору дозвілля, покроковим маршрутом і соціальною взаємодією, архітектурні рішення безпосередньо впливають на продуктивність, зручність супроводу та масштабованість. Тому перед прийняттям остаточного рішення було розглянуто основні підходи, що застосовуються у сучасних вебсистемах.

Монолітна архітектура передбачає реалізацію всього застосунку як єдиного цілого, де інтерфейс, бізнес-логіка та робота з даними об'єднані в межах одного програмного продукту. Такий підхід є найпростішим з точки зору розробки та розгортання, оскільки всі компоненти знаходяться в одному середовищі і не потребують складної інфраструктури для взаємодії між собою. Моноліт добре підходить для проєктів із чітко визначеним і порівняно стабільним набором функцій, а також на початкових етапах розробки, коли важлива швидкість реалізації. Проте з часом, у міру зростання функціоналу, компоненти починають сильно залежати один від одного. Це ускладнює внесення змін, збільшує ризик помилок при оновленнях і знижує зручність тестування окремих частин системи.

Мікросервісна архітектура будується на протилежному принципі. Система розбивається на набір невеликих незалежних сервісів, кожен із яких відповідає за окрему частину функціоналу та може розгортатися, оновлюватися і масштабуватися окремо від інших [8]. Такий підхід

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижомотря О.В.				18
Змн.	Арк.	№ докум.	Підпис	Дата		

забезпечує високу гнучкість і дозволяє ефективно розподіляти навантаження між компонентами системи. Однак мікросервісна архітектура потребує значно складнішої інфраструктури, включаючи інструменти для оркестрації сервісів, моніторингу та управління міжсервісною взаємодією. Для проєкту середнього масштабу, що розробляється однією особою або невеликою командою, такий підхід є надмірно складним і недоцільним на початковому етапі.

З урахуванням функціональних вимог платформи та масштабу проєкту було обрано архітектуру модульного моноліту. Цей підхід поєднує простоту класичного моноліту з логічним поділом системи на внутрішні модулі, кожен із яких має чітко визначену відповідальність і взаємодіє з іншими через визначені інтерфейси. Така організація зменшує зв'язаність між компонентами, спрощує тестування та підтримку коду, а за потреби дозволяє поступово виносити окремі модулі в самостійні сервіси без повної перебудови системи.

У межах розробленої платформи серверна частина організована за принципом Routes - Controller - Service [7], де кожен рівень має чітко визначену роль. Маршрути відповідають за обробку HTTP-запитів і передачу їх до відповідного контролера. Контролер координує виконання запиту та формує відповідь. Сервісний рівень містить бізнес-логіку системи та взаємодіє з базою даних через ORM. Такий поділ забезпечує структурованість коду та спрощує його супровід у міру зростання функціоналу.

Система реалізована в межах клієнт-серверної моделі. Клієнтська частина являє собою односторінковий застосунок на базі React [9], що забезпечує швидку навігацію між сторінками без перезавантаження та динамічне оновлення інтерфейсу. Вона відповідає за відображення даних і обробку дій користувача, зокрема перегляд каталогу дозвілля, проходження тесту, взаємодію з покроковим маршрутом та публікацію матеріалів. Для керування станом застосунку використовується бібліотека Jotai [11], яка

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ІЗ	Арк.
		Чижмоторя О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		19

забезпечує зберігання інформації про поточного користувача, його токен автентифікації та інші глобальні дані між компонентами.

Серверна частина реалізована на базі Node.js [13] із використанням фреймворку Express [14] і надає REST API [15] для взаємодії з клієнтом. Вона відповідає за всю бізнес-логіку системи, включаючи автентифікацію та авторизацію користувачів, роботу гібридного алгоритму рекомендацій, управління покроковим маршрутом і відстеження прогресу, обробку публікацій і модерацію контенту, систему подій та адміністративні функції. Взаємодія між клієнтською та серверною частинами відбувається через захищений HTTPS-протокол у форматі JSON із використанням JWT-токенів [16] для автентифікації.

Серверна частина поділена на функціональні модулі відповідно до основних предметних областей системи. Модуль автентифікації відповідає за реєстрацію, вхід та верифікацію користувачів. Модуль каталогу забезпечує роботу з дозволями, категоріями, пошуком і фільтрацією. Модуль тестув реалізує алгоритм підбору дозволів на основі відповідей користувача. Модуль покрокового маршруту керує вузлами, ресурсами та відстеженням прогресу. Модуль публікацій забезпечує роботу зі стрічкою, лайками, коментарями та модерацією контенту. Модуль подій відповідає за організацію та реєстрацію учасників. Модуль користувачів забезпечує управління профілем, системою досвіду та заявками на розширення прав. Такий поділ дозволяє незалежно розробляти та тестувати кожен частину системи без впливу на інші модулі.

Зберігання даних реалізовано на базі реляційної бази даних PostgreSQL [17], яка забезпечує цілісність інформації та підтримку складних зв'язків між сутностями. Для взаємодії з базою даних використовується ORM-бібліотека Prisma [18], що дозволяє працювати з даними через типізовані запити та зменшує ймовірність помилок. Завантаження зображень для аватарів і публікацій користувачів реалізоване через інтеграцію з хмарним сервісом

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ІЗ	Арк.
		Чижмоторя О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		20

Cloudinary[19], що знімає навантаження зі серверної частини та забезпечує надійне зберігання медіафайлів.

Обрана архітектура забезпечує оптимальний баланс між простотою реалізації та можливістю подальшого розвитку платформи. Модульна організація серверної частини дозволяє розширювати функціонал системи без порушення наявної структури, а клієнт-серверна модель із чітким поділом відповідальності спрощує супровід і тестування коду на всіх етапах розробки.

1.4 Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення

Для реалізації вебплатформи пошуку дозвілля було обрано сучасний стек веб-розробки, який дозволяє поєднати гнучкість інтерфейсу, стабільність серверної частини та можливість подальшого розширення системи без суттєвих змін її структури. Вибір кожного інструменту обґрунтований його відповідністю технічним вимогам платформи та практичною доцільністю в контексті проєкту.

Клієнтська частина реалізована з використанням бібліотеки React у поєднанні зі збирачем Vite [12] що забезпечує швидкий запуск середовища розробки та ефективне формування кінцевого застосунку. React дає змогу будувати інтерфейс як набір незалежних компонентів що дозволяє розділити функціональність на логічні частини, зокрема каталог захоплень, профіль користувача, сторінку покрокового маршруту, стрічку публікацій та сторінку подій. Компонентний підхід спрощує тестування та повторне використання елементів інтерфейсу. Порівняно з Vue.js та Angular, React має більшу гнучкість у виборі архітектурних підходів і не нав'язує жорсткої структури проєкту. Для організації навігації використовується React Router [10] що забезпечує плавний перехід між сторінками без їх перезавантаження у межах односторінкового застосунку.

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоштря О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		21

Стан застосунку керується за допомогою бібліотеки Jotai що реалізує модель керування станом де підписуються лише ті компоненти які безпосередньо використовують конкретні дані. Це дозволяє зберігати та синхронізувати інформацію про поточного користувача, його JWT-токен та роль у системі. На відміну від Redux, Jotai не потребує шаблонного коду і легко інтегрується з React-хуками. Токен зберігається в localStorage і автоматично додається до кожного запиту через Axios-інтерсептор [21] що забезпечує прозору автентифікацію на рівні клієнтської частини.

Інтерфейс оформлюється із використанням CSS-модулів та сучасних підходів до адаптивної верстки що забезпечують ізоляцію стилів і запобігають їх конфліктам між компонентами. Адаптивний дизайн реалізовано з використанням медіа-запитів що дозволяє коректно відображати систему на різних пристроях. Для зберігання зображень інтегровано хмарний сервіс Cloudinary [19] який забезпечує надійне зберігання медіафайлів без навантаження на власний сервер.

Серверна частина побудована на платформі Node.js із використанням фреймворку Express що забезпечує асинхронну обробку запитів і дозволяє ефективно обробляти велику кількість одночасних підключень. Express є мінімалістичним фреймворком що гнучко організовує маршрутизацію і middleware. Сервер відповідає за реалізацію основної бізнес-логіки системи, зокрема роботу гібридного алгоритму рекомендацій, управління покроковим маршрутом, відстеження прогресу та модерацію контенту. Обробка помилок реалізована на рівні сервісного шару де контролери перехоплюють їх і повертають клієнту у стандартизованому вигляді.

Для роботи з базою даних використовується ORM-бібліотека Prisma [18] що спрощує взаємодію з реляційною структурою та дозволяє уникнути прямого написання SQL-запитів. Prisma забезпечує типізацію запитів, автоматичну генерацію клієнта та міграцій на основі схеми даних. Як система керування базами даних обрано PostgreSQL [17] яка забезпечує надійне зберігання інформації, підтримує складні зв'язки між сутностями,

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоторя О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		22

масиви рядків для зберігання тегів і JSONB-поля для збереження відповідей тесту. У базі зберігаються дані про користувачів та їх ролі, каталог захоплень, структура маршруту з вузлами та ресурсами, публікації, події, підписки та прогрес проходження.

Для забезпечення безпеки доступу використовується механізм автентифікації на основі JWT відповідно до стандарту RFC 7519. Токен містить ідентифікатор користувача, його електронну пошту та роль і має обмежений термін дії. Система підтримує три ролі: звичайний користувач, практик та адміністратор. Перевірка токена та перевірка ролі реалізовані як два окремих middleware що застосовуються незалежно до різних маршрутів. Паролі зберігаються у хешованому вигляді з використанням бібліотеки bcryptjs відповідно до рекомендацій OWASP [20]. Для відновлення пароля реалізовано механізм одноразових токенів з обмеженим терміном дії що надсилаються на електронну пошту через nodemailer.

У процесі розробки застосовується система контролю версій Git із репозиторієм на університетській інсталяції GitLab. Гілкова стратегія базується на GitFlow де гілка main містить стабільний виробничий код а гілка develop використовується для поточної розробки. Якість коду підтримується через ESLint [22] і Prettier що забезпечують єдиний стиль оформлення. Для тестування серверної частини використовується Vitest у поєднанні з Supertest а для клієнтської Vitest з Testing Library та MSW [31]. Навантажувальне тестування виконується через k6 [32].

Обраний стек технологій забезпечує узгоджену роботу всіх компонентів системи та створює надійну основу для подальшого масштабування платформи. Усі обрані інструменти є відкритими, активно підтримуються спільнотою та мають якісну документацію.

Для розгортання системи необхідне середовище що відповідає певним технічним вимогам. Апаратна частина має включати щонайменше 8 ГБ оперативної пам'яті з рекомендованим обсягом 16 ГБ та 4-ядерний процесор де рекомендованим є 8-ядерний варіант. Дискоче сховище повинно

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоторя О.В.				23
Змн.	Арк.	№ докум.	Підпис	Дата		

базуватися на SSD-накопичувачі з обсягом вільного простору від 50 ГБ. З програмної сторони підійде будь-яка операційна система Windows, Linux або macOS за умови підтримки Node.js та PostgreSQL. Для локального розгортання необхідна встановлена платформа Docker Desktop. Для коректної роботи системи необхідне стабільне інтернет-з'єднання з підтримкою захищеного HTTPS-протоколу.

Обраний набір технологій дозволяє реалізувати весь необхідний функціонал платформи та забезпечує зручну підтримку коду в процесі розробки. Кожен інструмент вирішує конкретну задачу в межах системи а їх поєднання створює стабільну основу для подальшого розширення платформи.

Висновки до першого розділу

У першому розділі було розглянуто предметну галузь організації дозвілля в цифровому середовищі та визначено основні труднощі, з якими стикається користувач на етапі вибору і освоєння нового заняття. Виявилося, що велика кількість доступних ресурсів не спрощує цей процес, а навпаки ускладнює прийняття рішення через інформаційне перевантаження. Відсутність структурованого маршруту після вибору заняття, неможливість відстежувати власний прогрес та брак якісного середовища для взаємодії з іншими учасниками є головними причинами того, що більшість людей залишають нове захоплення вже на початковому етапі.

У межах розділу також було проаналізовано наявні програмні рішення у цій сфері. Розглянуті платформи показали, що кожна з них вирішує лише окрему частину задачі. Одні зосереджені на соціальній взаємодії та пошуку подій, інші пропонують підбір дозвілля, але не супроводжують користувача далі. Через це жодне з наявних рішень не забезпечує одночасно персоналізованого підбору, структурованого маршруту освоєння, відстеження прогресу та модерованого середовища для взаємодії в межах єдиної системи.

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ІЗ	Арк.
		Чижмоштря О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		24

На основі аналізу архітектурних підходів обрано модульну монолітну архітектуру як найбільш доцільну для реалізації платформи такого масштабу. Визначено технологічний стек та сформульовано вимоги до апаратного забезпечення.

Проведений аналіз показав, що об'єднання підбору дозвілля, покрокового маршруту його освоєння та інструментів взаємодії між учасниками в межах єдиного середовища дозволяє усунути основні обмеження наявних рішень. Саме це формує основу для подальшого проєктування та реалізації вебплатформи.

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоотря О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		25

РОЗДІЛ 2. ПРОЄКТУВАННЯ ВЕБПЛАТФОРМИ ДЛЯ ПОШУКУ ДОЗВІЛЛЯ

2.1 Визначення варіантів використання та об'єктно-орієнтованої структури системи

На етапі проєктування вебплатформи для пошуку дозвілля було визначено основні сценарії взаємодії користувачів із системою та сформовано рольову модель доступу. В основі проєктування лежить аналіз реальних потреб користувача, що дозволяє сформувати функціональний базис для подальшої розробки бази даних та API.

Платформа повинна надавати єдине середовище, де користувач може знайти заняття за своїми інтересами, пройти структурований маршрут його освоєння та взаємодіяти з іншими учасниками. Підбір дозвілля здійснюється через алгоритм рекомендацій на основі відповідей користувача. Система підтримує мотивацію через нарахування досвіду та відображення прогресу. Контент платформи формується за участю верифікованих практиків із перевіркою якості матеріалів. Розмежування прав доступу між ролями забезпечує контроль над функціоналом та цілісність системи.

У системі визначено чотири ролі: неавторизований користувач, авторизований користувач, практик та адміністратор. Для систематизованого опису функціональних можливостей платформи варіанти використання деталізовано у вигляді структурованих таблиць відповідно до методу прецедентів, описаного Рамбо та ін. [5] у стандарті UML як основного інструменту фіксації сценаріїв взаємодії між акторами та системою. Ларман [6] конкретизує форму опису: кожен прецедент містить назву, перелік акторів та короткий опис сценарію. Першу групу сценаріїв, пов'язаних з автентифікацією та управлінням обліковим записом, наведено в табл. 2.1.

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоштря О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		26

Таблиця 2.1

Опис прецедентів автентифікації та управління профілем

Назва	Актори	Опис
Зареєструватися	Неавторизований користувач	Створення облікового запису з введенням імені, email та пароля
Увійти до системи	Неавторизований користувач	Автентифікація за email та паролем з отриманням JWT-токена
Переглянути профіль	Авторизований користувач	Перегляд персональних даних, накопиченого досвіду та підписок
Редагувати профіль	Авторизований користувач	Зміна імені, аватара та персональних налаштувань

Після автентифікації користувач отримує доступ до каталогу захоплень та механізму підбору дозвілля. Сценарії взаємодії з каталогом та тестом наведено в табл. 2.2.

Таблиця 2.2

Опис прецедентів роботи з каталогом та підбором дозвілля

Назва	Актори	Опис
Переглянути каталог	Усі користувачі	Перегляд списку захоплень із фільтрацією за категоріями та пошуком
Переглянути захоплення	Усі користувачі	Ознайомлення з описом та матеріалами конкретного заняття
Пройти тест	Усі користувачі	Проходження тесту для отримання персоналізованих рекомендацій на основі гібридного алгоритму

продовж. табл. 2.2

Підписатися на захоплення	Авторизований користувач	Встановлення зв'язку між користувачем та обраним заняттям з отриманням доступу до маршруту
Додати до обраного	Авторизований користувач	Збереження заняття до списку вподобань

Після підписки користувач отримує доступ до покрокового маршруту та інструментів відстеження власного розвитку. Відповідні сценарії наведено в табл. 2.3.

Таблиця 2.3

Опис прецедентів проходження маршруту та відстеження прогресу

Назва	Актори	Опис
Переглянути маршрут	Авторизований користувач	Перегляд структури вузлів та послідовності їх проходження
Виконати вузол	Авторизований користувач	Відмітка вузла як виконаного з нарахуванням досвіду
Переглянути ресурси	Авторизований користувач	Перегляд навчальних матеріалів прив'язаних до конкретного вузла
Переглянути прогрес	Авторизований користувач	Відображення рівня завершення маршруту у вигляді відсотків і теплової карти
Переглянути таблицю лідерів	Авторизований користувач	Перегляд рейтингу учасників за накопиченим досвідом у межах заняття

Окрему групу становлять сценарії соціальної взаємодії між учасниками платформи через публікації та події. Їх наведено в табл. 2.4.

Таблиця 2.4

Опис прецедентів роботи з публікаціями та подіями

Назва	Актори	Опис
Створити публікацію	Авторизований користувач	Додавання матеріалу до стрічки із зображенням та текстом
Переглянути стрічку	Авторизований користувач	Перегляд опублікованих матеріалів учасників заняття
Вподобати публікацію	Авторизований користувач	Вираження позитивної реакції на публікацію іншого учасника
Переглянути події	Усі користувачі	Перегляд доступних онлайн та офлайн заходів
Зареєструватися на подію	Авторизований користувач	Підтвердження участі в обраному заході
Подати заявку на практика	Авторизований користувач	Надсилання запиту на отримання розширених прав із обґрунтуванням

Практик отримує розширені права для керування контентом свого заняття після підтвердження заявки адміністратором. Сценарії взаємодії практика з платформою наведено в табл. 2.5.

Таблиця 2.5

Опис прецедентів практика

Назва	Актори	Опис
-------	--------	------

продовж. табл. 2.5

Додати вузол	Практик	Створення нового етапу маршруту з описом та ресурсами
Редагувати вузол	Практик	Зміна змісту власного вузла в маршруті
Видалити вузол	Практик	Видалення власного вузла з маршруту заняття
Додати ресурс	Практик	Прив'язка навчального матеріалу до конкретного вузла
Модерувати публікації	Практик	Зміна статусу публікацій у стрічці свого заняття
Створити подію	Практик	Додавання нового заходу із зазначенням формату та дати проведення

Адміністратор має повний доступ до всіх функціональних можливостей системи через окрему панель керування. Перелік адміністративних прецедентів наведено в табл. 2.6.

Таблиця 2.6

Опис прецедентів адміністратора

Назва	Актори	Опис
Створити захоплення	Адміністратор	Додавання нового заняття до каталогу з категорією та описом
Редагувати захоплення	Адміністратор	Зміна інформації та матеріалів заняття

Керувати користувачами	Адміністратор	Перегляд списку користувачів, зміна ролей та видалення
Розглянути заявку	Адміністратор	Підтвердження або відхилення заявки на статус практика з призначенням заняття
Переглянути статистику	Адміністратор	Перегляд показників платформи, топ-5 захоплень та користувачів
Модерувати контент	Адміністратор	Приховування або видалення публікацій і подій платформи

З огляду на визначені варіанти використання формується об'єктно-орієнтована структура системи, що включає такі основні сутності як користувач, роль, захоплення, категорія, маршрут, вузол, ресурс, прогрес, підписка, подія, публікація та заявка. Взаємозв'язки між цими сутностями відображають логіку роботи платформи та забезпечують реалізацію визначених функціональних вимог. Діаграму варіантів використання системи наведено на рис. 2.1, де відображено всі ролі та їх взаємодію з функціональними модулями платформи.

Діаграма охоплює чотири ролі: неавторизований користувач, авторизований користувач, практик та адміністратор. Кожна роль має чітко визначений набір доступних сценаріїв. Неавторизований користувач взаємодіє лише з публічною частиною платформи. Авторизований користувач отримує доступ до підписки, маршруту, публікацій та подій. Практик розширює цей набір функціями керування контентом свого

захоплення. Адміністратор має повний доступ до всіх можливостей системи через окрему панель керування.

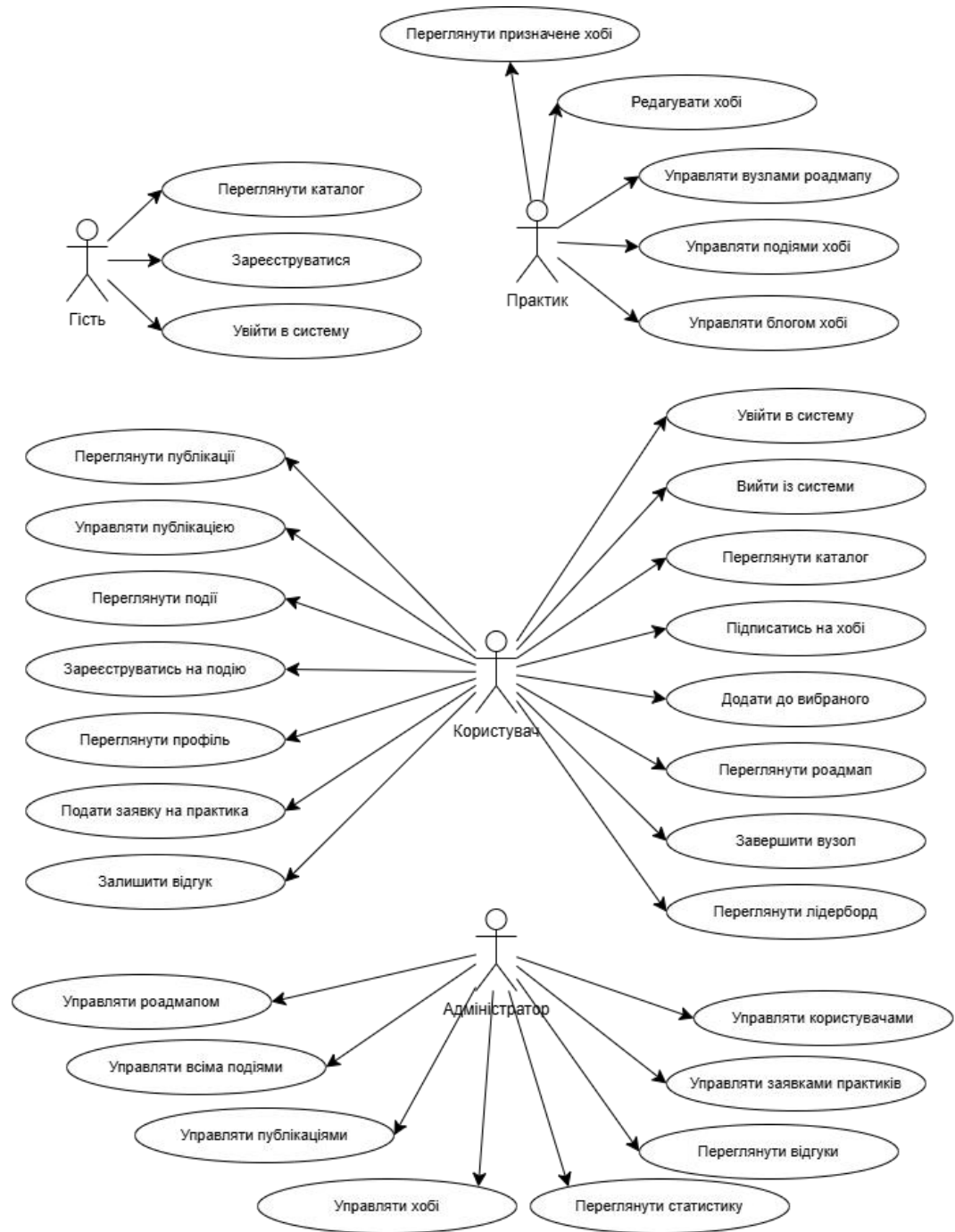


Рис. 2.1. Варіанти використання системи

Після визначення варіантів використання та побудови таблиць прецедентів доцільно описати кожен функціональний модуль системи окремо, щоб сформувати цілісне уявлення про її структуру та принципи роботи.

Модуль автентифікації та управління профілем забезпечує базову взаємодію користувача з платформою на всіх етапах роботи з нею. Неавторизований користувач може створити обліковий запис, вказавши ім'я, електронну пошту та пароль, або виконати вхід до системи з уже наявними даними. Після успішної автентифікації сервер генерує JWT-токен, який зберігається на клієнтській частині та автоматично додається до кожного наступного запиту для підтвердження особи користувача. У межах авторизованої сесії користувач може переглядати та редагувати власний профіль, змінювати аватар і персональні дані, відстежувати список підписок, накопичений досвід і теплову карту власної активності за останні тижні. Профіль також відображає загальну кількість виконаних вузлів і публікацій, що дає змогу оцінити власний внесок у розвиток спільноти.

Модуль каталогу захоплень відповідає за відображення доступних занять, їх структурування за категоріями та забезпечення зручного пошуку і фільтрації. Будь-який відвідувач платформи може переглядати каталог і знайомитися з описами захоплень без необхідності реєстрації. Кожне заняття в каталозі має окрему сторінку з детальним описом, переліком категорій та тегів, інформацією про кількість учасників і загальний рівень активності спільноти. Пошук дозволяє знаходити заняття за ключовими словами, а фільтрація за категоріями спрощує перегляд у випадку, коли користувач має загальне уявлення про напрям, який його цікавить. Адміністратор здійснює створення та редагування захоплень, формує їх базову структуру, призначає категорії та теги, що впливають на роботу алгоритму рекомендацій, і контролює відповідність контенту загальній концепції платформи.

Модуль тесту реалізує гібридний алгоритм підбору дозвілля на основі відповідей користувача. Тест складається з питань, відповіді на які дозволяють системі визначити інтереси, доступний час та формат занять, що підходить конкретній людині. Після завершення тесту система обчислює оцінку для кожного заняття в каталозі за формулою, що враховує відповідність тегів відповідям користувача та показник популярності заняття.

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоторя О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		33

Результати зберігаються в базі даних і доступні користувачеві в будь-який момент через профіль. За підсумками тесту формується персоналізована рекомендація трьох найбільш відповідних захоплень із можливістю переходу до детального перегляду або підписки на кожне з них. Проходження тесту доступне як для авторизованих, так і для неавторизованих користувачів, однак збереження результатів вимагає автентифікації.

Модуль покрокового маршруту забезпечує структуроване освоєння обраного заняття. Маршрут складається з послідовних вузлів, кожен із яких відповідає певному етапу розвитку у межах захоплення. Вузли відображаються у вигляді візуальної схеми, де користувач бачить як уже виконані етапи, так і ті, що ще попереду. Авторизований користувач переглядає зміст вузла, опрацьовує прив'язані матеріали та відмічає вузол як завершений, що автоматично нараховує йому досвід і оновлює загальний прогрес. Вузли, створені адміністратором, утворюють основну послідовність маршруту і відображаються у вигляді ланцюжка. Вузли практика виділяються окремо з зазначенням імені автора, що дозволяє користувачеві розуміти походження кожного матеріалу. Адміністратор має повний доступ до редагування будь-якого вузла та може керувати всією структурою маршруту незалежно від його джерела.

Модуль управління вузлами та ресурсами дозволяє формувати окремі етапи проходження заняття та наповнювати їх навчальним контентом. Кожен вузол містить заголовок, текстовий опис і може мати прив'язані ресурси у вигляді зовнішніх посилань або відеоматеріалів. Практик створює власні вузли в межах призначеного йому заняття, додає до них матеріали та може редагувати або видаляти лише власний контент. При видаленні вузла система автоматично очищує пов'язані ресурси та записи прогресу, а події, що були прив'язані до цього вузла, зберігаються з обнуленим посиланням на нього. Такий підхід забезпечує цілісність даних навіть у разі змін у структурі маршруту.

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижомотря О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		34

Модуль відстеження прогресу фіксує результати проходження маршруту та відображає рівень завершення заняття у відсотках від загальної кількості вузлів. Система зберігає окремий запис прогресу для кожного виконаного вузла із прив'язкою до конкретного користувача. Це дозволяє точно відстежувати, які саме етапи вже пройдено, а які залишаються. Окрім загального відсотка завершення, платформа відображає динаміку активності користувача через теплову карту, яка показує дні, в які він виконував вузли. Такий підхід дає змогу бачити не лише поточний результат, а й регулярність занять, що є важливим чинником підтримки мотивації. У межах кожного заняття також функціонує таблиця лідерів, де відображається рейтинг учасників за накопиченим досвідом.

Модуль публікацій забезпечує роботу стрічки матеріалів у межах кожного заняття. Авторизований користувач може публікувати матеріали з текстом і зображенням, переглядати публікації інших учасників та виражати позитивну реакцію через лайки. Нова публікація після створення отримує статус очікування і не відображається у загальній стрічці до проходження модерації. Практик переглядає публікації зі статусом очікування та приймає рішення щодо їх публікації або повернення у приватний режим. Адміністратор бачить усі публікації незалежно від статусу та може приховувати або видаляти будь-який контент. Такий підхід забезпечує якість матеріалів у стрічці та запобігає поширенню недостовірної інформації.

Модуль подій забезпечує організацію онлайн та офлайн заходів у межах конкретного заняття. Практик створює події із зазначенням назви, опису, формату та дати проведення. Для онлайн-подій вказується посилання для підключення, для офлайн-заходів додається місце проведення. Мінімальний термін між моментом створення події та датою її проведення складає 24 години, що перевіряється як на рівні клієнтської частини, так і на сервері. Авторизований користувач може переглядати доступні заходи, фільтрувати їх за форматом або датою та реєструватися на ті, що відповідають його інтересам.

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоторя О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		35

Модуль заявок забезпечує обробку запитів на отримання статусу практика. Авторизований користувач заповнює форму заявки з обґрунтуванням свого досвіду у певному напрямі та вказує заняття, до якого хоче долучитися як практик. Заявка зберігається в системі зі статусом очікування. Адміністратор переглядає список поданих заявок, може ознайомитися з деталями кожної та прийняти рішення щодо підтвердження або відхилення. У разі схвалення роль користувача автоматично змінюється на практика, а до його облікового запису прив'язується відповідне заняття.

Адміністративний модуль забезпечує централізоване керування платформою через окрему панель із вкладками. Вкладка керування користувачами дозволяє переглядати повний список зареєстрованих учасників, змінювати їх ролі та видаляти облікові записи з каскадним очищенням пов'язаних даних. Вкладка заявок відображає всі запити на отримання статусу практика з можливістю їх розгляду та обробки. Вкладка статистики надає зведені показники платформи, зокрема загальну кількість користувачів, захоплень і публікацій, а також топ-5 найпопулярніших занять і найактивніших учасників. Наявність адміністративної панелі забезпечує повний контроль над структурною цілісністю системи та дотриманням правил взаємодії в межах платформи.

2.2 Розробка бази даних системи

Від правильно спроектованої моделі даних безпосередньо залежить стабільність роботи платформи та можливість її подальшого розширення. Елмасрі та Наватхе [23] визначають нормалізацію та забезпечення референційної цілісності як ключові принципи побудови реляційних баз даних, що запобігають надлишковості та аномаліям при оновленні записів. Дейт [24] додатково підкреслює важливість декларативного опису обмежень на рівні схеми, що гарантує коректність даних незалежно від логіки застосунку. Головне завдання полягало в тому, щоб відобразити всі сутності

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ІЗ	Арк.
		Чижмоторя О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		36

предметної галузі та зв'язки між ними в межах реляційної структури, яка забезпечує цілісність даних і підтримує реалізацію повного функціоналу системи.

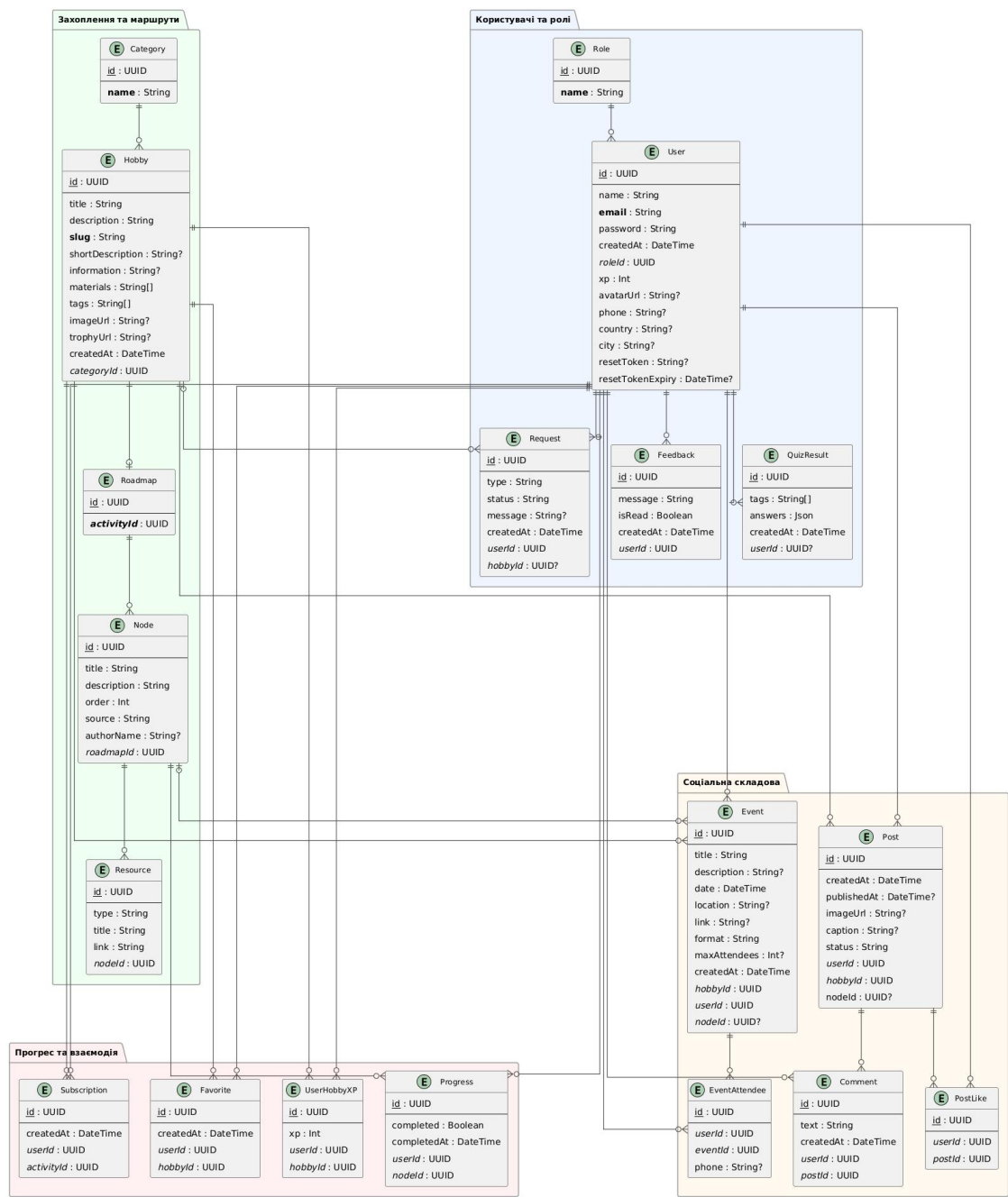


Рис. 2.2. Структура таблиць бази даних системи

База даних побудована навколо кількох ключових груп сутностей. Перша група стосується користувачів та їх ролей у системі. Друга описує захоплення, їх категорії та маршрути освоєння. Третя охоплює соціальну складову платформи, зокрема публікації, коментарі та події. Четверта фіксує

прогрес користувачів і результати їх взаємодії з контентом. Такий поділ дозволяє чітко окреслити зони відповідальності між таблицями та уникнути надлишкового дублювання даних. Загальну структуру таблиць та зв'язки між ними наведено на рис. 2.2.

Першу групу утворюють таблиці що описують учасників системи та їх права доступу. Центральною є таблиця User, до якої прив'язані майже всі інші сутності платформи. Кожен користувач має роль що визначається через таблицю Role і впливає на доступний функціонал. Таблиця QuizResult зберігає результати підбору дозвілля і пов'язана з конкретним користувачем. Через таблицю Request відбувається процес отримання розширених прав практика, а таблиця Feedback забезпечує зворотний зв'язок між користувачами та адміністратором.

Таблиця User є основною сутністю системи і зберігає всі дані про зареєстрованих учасників. Крім ідентифікаційних полів вона містить поле xp для загального накопиченого досвіду, дані про місце проживання та поля для механізму скидання пароля. Структуру таблиці наведено в табл. 2.7.

Таблиця 2.7

Структура таблиці User

Поле	Тип	Опис
id	UUID	Унікальний ідентифікатор користувача
name	String	Ім'я користувача
email	String	Електронна пошта для автентифікації
password	String	Зашифрований пароль
createdAt	DateTime	Дата реєстрації
roleId	UUID	Ідентифікатор ролі користувача
xp	Int	Загальний накопичений досвід
avatarUrl	String?	Посилання на аватар
phone	String?	Номер телефону
country	String?	Країна проживання

city	String?	Місто проживання
resetToken	String?	Токен скидання пароля
resetTokenExpiry	DateTime?	Час дії токена скидання пароля

Таблиця Role є довідниковою і містить перелік можливих ролей системи. У поточній реалізації визначено три ролі: звичайний користувач, практик та адміністратор. Зв'язок між User і Role дозволяє гнучко змінювати права доступу без змін у структурі інших таблиць. Структуру таблиці наведено в табл. 2.8.

Таблиця 2.8

Структура таблиці Role

Поле	Тип	Опис
id	UUID	Унікальний ідентифікатор ролі
name	String	Назва ролі в системі

Таблиця QuizResult зберігає результати проходження тесту. Відповіді фіксуються у форматі JSON, а масив тегів формується на їх основі і використовується алгоритмом рекомендацій. Поле userId є необов'язковим оскільки тест доступний і без авторизації. Структуру таблиці наведено в табл. 2.9.

Таблиця 2.9

Структура таблиці QuizResult

Поле	Тип	Опис
id	UUID	Унікальний ідентифікатор результату
tags	String[]	Масив тегів за результатами тесту
answers	Json	Відповіді користувача у форматі JSON
createdAt	DateTime	Дата проходження тесту
userId	UUID?	Ідентифікатор користувача

Таблиця Request забезпечує обробку заявок на отримання статусу практика. Поле hobbyId є необов'язковим на момент подання і заповнюється після схвалення заявки адміністратором. Структуру таблиці наведено в табл. 2.10.

Таблиця 2.10

Структура таблиці Request

Поле	Тип	Опис
id	UUID	Унікальний ідентифікатор заявки
type	String	Тип заявки
status	String	Статус розгляду заявки
message	String?	Повідомлення від користувача
createdAt	DateTime	Дата подання заявки
userId	UUID	Ідентифікатор користувача
hobbyId	UUID?	Ідентифікатор призначеного захоплення

Таблиця Feedback зберігає повідомлення користувачів до адміністратора. Поле isRead дозволяє відстежувати які повідомлення вже переглянуто через адміністративну панель. Структуру таблиці наведено в табл. 2.11.

Таблиця 2.11

Структура таблиці Feedback

Поле	Тип	Опис
id	UUID	Унікальний ідентифікатор відгуку
message	String	Текст відгуку
isRead	Boolean	Статус прочитання адміністратором
createdAt	DateTime	Дата створення відгуку
userId	UUID	Ідентифікатор автора відгуку

Другу групу утворюють таблиці що описують захоплення та їх структуру. Центральною тут є таблиця Hobby, яка пов'язана з Category через

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмогтя О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		40

зовнішній ключ. Кожне захоплення має один маршрут у таблиці Roadmap, який складається з послідовних вузлів у таблиці Node. До кожного вузла можуть бути прив'язані навчальні матеріали через таблицю Resource. Такий ланцюжок утворює повну структуру покрокового освоєння заняття: кожне захоплення має один маршрут, маршрут складається з вузлів, а кожен вузол може містити навчальні матеріали.

Таблиця Hobby зберігає всю інформацію про захоплення на платформі. Поле slug забезпечує унікальні URL-адреси, поле tags містить масив тегів для алгоритму рекомендацій, а поля materials і information формують контент сторінки заняття. Структуру таблиці наведено в табл. 2.12.

Таблиця 2.12

Структура таблиці Hobby

Поле	Тип	Опис
id	UUID	Унікальний ідентифікатор захоплення
title	String	Назва захоплення
description	String	Повний опис захоплення
slug	String	URL-ідентифікатор для навігації
shortDescription	String?	Короткий опис для каталогу
information	String?	Інформаційний блок сторінки
materials	String[]	Масив навчальних матеріалів
tags	String[]	Теги для алгоритму рекомендацій
imageUrl	String?	Посилання на зображення
trophyUrl	String?	Посилання на зображення трофею
createdAt	DateTime	Дата створення
categoryId	UUID	Ідентифікатор категорії захоплення

Таблиця Category є довідниковою і дозволяє групувати захоплення за тематичними напрямками в каталозі. Структуру таблиці наведено в табл. 2.13.

Таблиця 2.13

Структура таблиці Category

Поле	Тип	Опис
id	UUID	Унікальний ідентифікатор категорії
name	String	Назва категорії

Таблиця Roadmap реалізує відношення один до одного між захопленням і його маршрутом. Поле activityId є одночасно зовнішнім і унікальним ключем що гарантує прив'язку лише одного маршруту до кожного захоплення. Структуру таблиці наведено в табл. 2.14.

Таблиця 2.14

Структура таблиці Roadmap

Поле	Тип	Опис
id	UUID	Унікальний ідентифікатор маршруту
activityId	UUID	Ідентифікатор пов'язаного захоплення

Таблиця Node описує окремі етапи маршруту. Поле order визначає послідовність відображення вузлів. Поле source розрізняє вузли адміністратора від вузлів практика, а поле authorName зберігає ім'я практика для відображення у візуальній схемі маршруту. Структуру таблиці наведено в табл. 2.15.

Таблиця 2.15

Структура таблиці Node

Поле	Тип	Опис
id	UUID	Унікальний ідентифікатор вузла
title	String	Заголовок вузла
description	String	Опис змісту вузла
order	Int	Порядковий номер у маршруті
source	String	Джерело створення вузла

authorName	String?	Ім'я автора для вузлів практика
roadmapId	UUID	Ідентифікатор маршруту вузла

Таблиця Resource зберігає навчальні матеріали прив'язані до конкретного вузла. Один вузол може мати довільну кількість ресурсів що забезпечує гнучкість наповнення контентом. Поле type визначає спосіб відображення ресурсу в інтерфейсі. Структуру таблиці наведено в табл. 2.16.

Таблиця 2.16

Структура таблиці Resource

Поле	Тип	Опис
id	UUID	Унікальний ідентифікатор ресурсу
type	String	Тип ресурсу
title	String	Назва ресурсу
link	String	URL ресурсу
nodeId	UUID	Ідентифікатор вузла ресурсу

Третю групу утворюють таблиці соціальної складової платформи. Публікації користувачів зберігаються у таблиці Post і проходять модерацію практиком перед відображенням у стрічці. Реакції на публікації фіксуються через таблицю PostLike, а коментарі через таблицю Comment. Події в межах захоплення організовуються через таблицю Event, а реєстрація учасників на них зберігається у таблиці EventAttendee. Всі ці таблиці пов'язані як з User так і з Hobby що дозволяє відображати соціальну активність у контексті конкретного заняття.

Таблиця Post зберігає публікації учасників. Поле status визначає стан публікації в процесі модерації і набуває значень private, pending або published. Структуру таблиці наведено в табл. 2.17.

Таблиця 2.17

Структура таблиці Post

Поле	Тип	Опис
id	UUID	Унікальний ідентифікатор публікації
createdAt	DateTime	Дата створення
publishedAt	DateTime?	Дата публікації
imageUrl	String?	Посилання на зображення
caption	String?	Текст публікації
status	String	Статус публікації в процесі модерації
userId	UUID	Ідентифікатор автора публікації
hobbyId	UUID	Ідентифікатор захоплення публікації
nodeId	UUID?	Ідентифікатор пов'язаного вузла

Таблиця PostLike забезпечують взаємодію між учасниками в межах стрічки. PostLike фіксує реакцію користувача на публікацію. Структуру таблиці наведено в табл. 2.18.

Таблиця 2.18

Структура таблиці PostLike

Поле	Тип	Опис
id	UUID	Унікальний ідентифікатор лайку
userId	UUID	Ідентифікатор користувача
postId	UUID	Ідентифікатор публікації

Comment зберігає текстові відповіді. Структуру таблиці наведено в табл. 2.19.

Таблиця 2.19

Структура таблиці Comment

Поле	Тип	Опис
id	UUID	Унікальний ідентифікатор коментаря

text	String	Текст коментаря
createdAt	DateTime	Дата створення коментаря
userId	UUID	Ідентифікатор автора коментаря
postId	UUID	Ідентифікатор публікації

Таблиця Event зберігає інформацію про заходи в межах захоплення. Поле format визначає тип заходу: для онлайн-подій заповнюється link, для офлайн-заходів відповідно location. Поле nodeId є необов'язковим і при видаленні вузла обнулюється без видалення події. Структуру таблиці наведено в табл. 2.20.

Таблиця 2.20

Структура таблиці Event

Поле	Тип	Опис
id	UUID	Унікальний ідентифікатор події
title	String	Назва події
description	String?	Опис події
date	DateTime	Дата проведення
location	String?	Місце проведення для офлайн-подій
link	String?	Посилання для онлайн-подій
format	String	Формат проведення події
maxAttendees	Int?	Максимальна кількість учасників
createdAt	DateTime	Дата створення події
hobbyId	UUID	Ідентифікатор захоплення події
userId	UUID	Ідентифікатор організатора події
nodeId	UUID?	Ідентифікатор пов'язаного вузла

Таблиця EventAttendee фіксує реєстрацію учасників на події. Поле phone є необов'язковим і може використовуватися організатором для

контакту з учасниками офлайн-заходів. Структуру таблиці наведено в табл. 2.21.

Таблиця 2.21

Структура таблиці EventAttendee

Поле	Тип	Опис
id	UUID	Унікальний ідентифікатор запису
userId	UUID	Ідентифікатор учасника події
eventId	UUID	Ідентифікатор події
phone	String?	Номер телефону учасника

Четверту групу утворюють таблиці що фіксують результати взаємодії користувача з контентом платформи. Таблиця Progress відстежує виконання вузлів маршруту і є основою для відображення прогресу. Підписки на захоплення зберігаються у таблиці Subscription і визначають до якого контенту користувач має доступ. Таблиця Favorite дозволяє зберігати заняття що зацікавили користувача, а таблиця UserHobbyXP забезпечує окремий облік досвіду для кожного захоплення і використовується для формування таблиці лідерів.

Таблиця Progress створює окремий запис для кожної пари користувач-вузол що дозволяє точно фіксувати індивідуальний прогрес. Поле completedAt використовується для побудови теплової карти активності в профілі користувача. Структуру таблиці наведено в табл. 2.22.

Таблиця 2.22

Структура таблиці Progress

Поле	Тип	Опис
id	UUID	Унікальний ідентифікатор запису
completed	Boolean	Статус виконання вузла
completedAt	DateTime	Дата виконання вузла
userId	UUID	Ідентифікатор користувача

nodeId	UUID	Ідентифікатор виконаного вузла
--------	------	--------------------------------

Таблиця Subscription встановлює зв'язок між користувачем і захопленням. Наявність запису в цій таблиці відкриває доступ до маршруту та соціальних функцій заняття. Структуру таблиці наведено в табл. 2.23.

Таблиця 2.23

Структура таблиці Subscription

Поле	Тип	Опис
id	UUID	Унікальний ідентифікатор підписки
createdAt	DateTime	Дата підписки
userId	UUID	Ідентифікатор користувача
activityId	UUID	Ідентифікатор захоплення

Таблиця Favorite зберігає захоплення додані користувачем до списку вподобань. Вона встановлює простий зв'язок між User і Hobby без додаткових полів. Структуру таблиці наведено в табл. 2.24.

Таблиця 2.24

Структура таблиці Favorite

Поле	Тип	Опис
id	UUID	Унікальний ідентифікатор запису
createdAt	DateTime	Дата додавання до вподобань
userId	UUID	Ідентифікатор користувача
hobbyId	UUID	Ідентифікатор захоплення

Таблиця UserHobbyXP зберігає досвід користувача в межах конкретного захоплення окремо від загального поля xp. Це дозволяє будувати незалежний рейтинг учасників для кожного заняття. Структуру таблиці наведено в табл. 2.25.

Структура таблиці UserHobbyXP

Поле	Тип	Опис
id	UUID	Унікальний ідентифікатор запису
xp	Int	Накопичений досвід у межах захоплення
userId	UUID	Ідентифікатор користувача
hobbyId	UUID	Ідентифікатор захоплення

Запропонована структура бази даних забезпечує цілісність інформації та логічну узгодженість зв'язків між сутностями. Реляційна модель дозволяє масштабувати систему та розширювати її можливості без порушення базових принципів організації даних, а чіткий розподіл відповідальності між таблицями спрощує підтримку та супровід системи в процесі подальшого розвитку.

2.3 Проєктування та реалізація алгоритмів роботи системи

Для опису алгоритмів функціонування платформи використовуються діаграми активності, що відображають послідовність дій користувача та системи в межах основних сценаріїв роботи. Такі діаграми дозволяють формалізувати логіку бізнес-процесів, визначити точки прийняття рішень та забезпечити узгодженість між клієнтською та серверною частинами застосунку. Кожен алгоритм описує конкретний сценарій взаємодії користувача з платформою від початкової дії до отримання результату. У межах підрозділу розглянуто шість ключових алгоритмів що реалізують основний функціонал системи.

Першим розглядається алгоритм підбору дозвілля на основі тесту, метою якого є визначення найбільш відповідних захоплень відповідно до інтересів користувача. Алгоритм доступний як для авторизованих так і для неавторизованих користувачів, однак збереження результатів вимагає

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоштря О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		48

автентифікації. Передумовою виконання сценарію є відкрита сторінка тесту та готовність користувача до проходження тесту.

Процес розпочинається з ініціації проходження тесту. Система завантажує із бази даних набір питань і послідовно відображає їх користувачеві. Кожна відповідь фіксується на клієнтській частині та прив'язана до певного набору тегів що відповідають конкретним напрямам дозвілля. Після того як користувач відповів на всі питання, клієнтська частина надсилає зібраний масив відповідей на сервер для обробки.

Серверна частина виконує обчислення оцінки для кожного захоплення в каталозі за гібридним алгоритмом рекомендацій. Річчі та ін. [25] обґрунтовують перевагу гібридних підходів над чисто контентними або колаборативними: їх поєднання нівелює проблему холодного старту та підвищує точність рекомендацій. Су та Хошгофтаар [26] встановлюють, що підхід на основі схожості між користувачами є ефективним за наявності достатньої кількості спільних взаємодій у системі. Оцінка розраховується за формулою де гібридна оцінка дорівнює сумі добутку коефіцієнта 0.6 на показник відповідності тегів та добутку коефіцієнта 0.4 на показник колаборативної популярності. Показник відповідності тегів відображає зважену кількість збігів між тегами конкретного захоплення та тегами отриманими з відповідей користувача, нормалізовану до діапазону від нуля до ста. Показник колаборативної популярності визначається як частка схожих за інтересами користувачів платформи що підписані на конкретне захоплення, де схожість встановлюється за наявністю щонайменше трьох спільних тегів із поточним результатом тесту; значення також нормалізується до діапазону від нуля до ста. Якщо кількість схожих користувачів у базі менша за десять, колаборативна складова не застосовується і підсумкова оцінка визначається виключно показником відповідності тегів.

Після обчислення оцінок система сортує захоплення за спаданням та повертає три найвищих результати. Якщо користувач авторизований результати зберігаються в таблиці QuizResult разом із масивом тегів та

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ІЗ	Арк.
		Чижмоштра О.В.				49
Змн.	Арк.	№ докум.	Підпис	Дата		

вихідними відповідями у форматі JSON. Це дозволяє у майбутньому відстежувати зміни в інтересах користувача та порівнювати результати різних проходжень тесту. Результатом виконання алгоритму є відображення персоналізованої рекомендації трьох захоплень з можливістю переходу до детального перегляду кожного або підписки на обране. Діаграму активності цього алгоритму наведено на рис. 2.3.

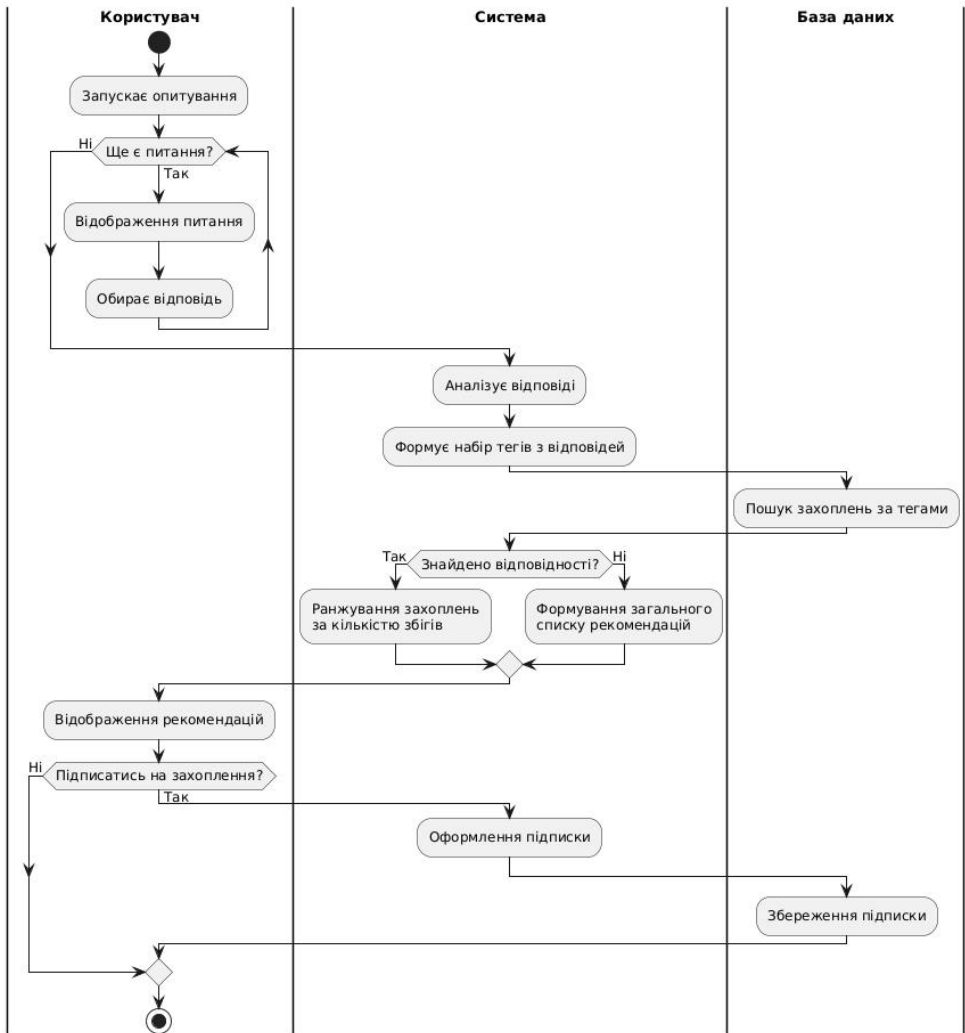


Рис. 2.3. Діаграма активності алгоритму підбору дозвілля

Наступним розглядається алгоритм підписки на захоплення, метою якого є встановлення зв'язку між користувачем та обраним захопленням з подальшим отриманням доступу до маршруту та механізму відстеження прогресу. Передумовою є авторизація користувача та відкрита сторінка

конкретного захоплення. Підписка є обов'язковою умовою для доступу до покрокового маршруту, стрічки публікацій та подій у межах заняття.

Після переходу на сторінку захоплення авторизований користувач ініціює дію підписки натисканням відповідної кнопки. Клієнтська частина надсилає запит на сервер із ідентифікатором захоплення. Сервер перевіряє чи не існує вже активного запису в таблиці Subscription для даної пари користувач-захоплення. Якщо підписка вже існує сервер повертає повідомлення про це і інтерфейс відображає відповідний стан кнопки. В іншому випадку система створює новий запис у таблиці Subscription із зазначенням дати підписки. Після успішного збереження клієнтська частина оновлює інтерфейс і відображає користувачеві доступ до вкладок маршруту, публікацій та подій. Результатом виконання алгоритму є можливість розпочати проходження вузлів і відстежувати власний прогрес у межах обраного захоплення. Діаграму активності наведено на рис. 2.4.

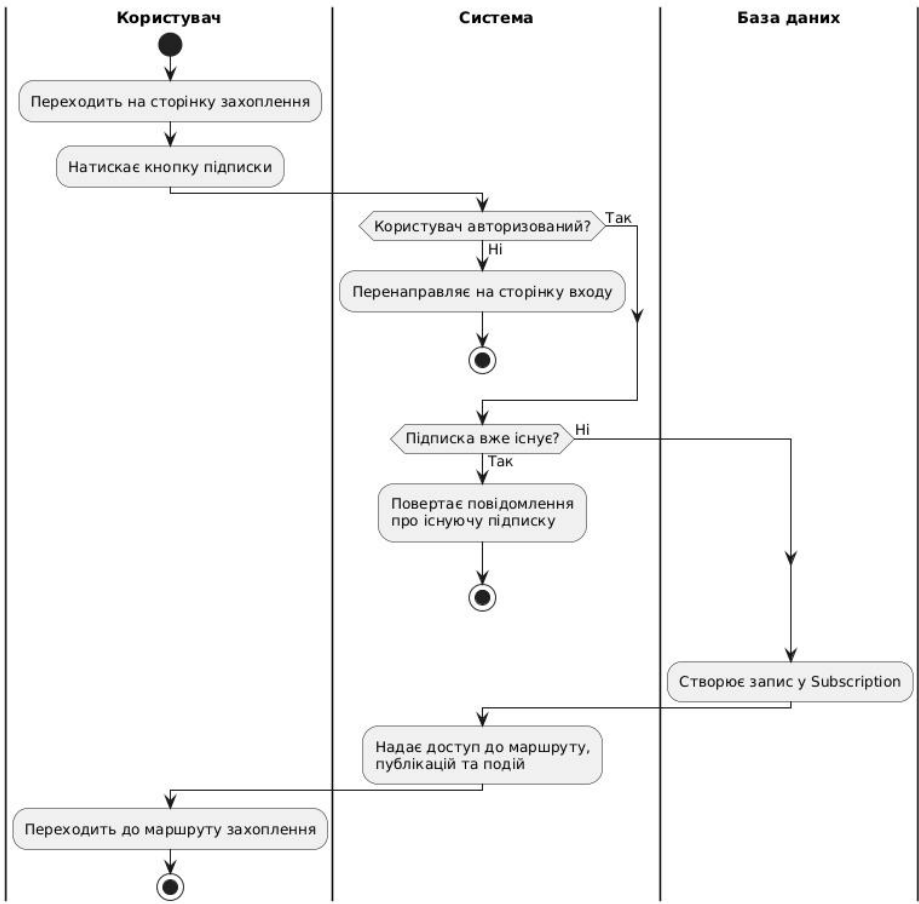


Рис. 2.4. Діаграма активності алгоритму підписки на захоплення

Алгоритм проходження маршруту та нарахування досвіду забезпечує фіксацію прогресу користувача в межах обраного захоплення та є центральним з точки зору мотиваційної механіки платформи. Детердінг та ін. [27] визначають гейміфікацію як застосування ігрових елементів, тобто балів, індикаторів прогресу та таблиць лідерів, у неігрових контекстах із метою підвищення залученості. Зіхерманн і Каннінгем [28] додатково обґрунтовують, що система нарахування очок у поєднанні з видимим рейтингом є одним із найдієвіших інструментів утримання користувача.

Користувач відкриває маршрут де бачить візуальну схему вузлів із позначенням виконаних, поточних та заблокованих етапів. Він обирає доступний вузол для опрацювання та переглядає його зміст: опис, прив'язані ресурси у вигляді посилань або відеоматеріалів. Після ознайомлення з матеріалами користувач ініціює дію відмітки вузла як виконаного. Клієнтська частина надсилає запит на сервер із ідентифікатором вузла.

Сервер перевіряє наявність активної підписки та відповідного запису в таблиці Progress для даної пари користувач-вузол. Якщо запис відсутній система створює новий із полем completed встановленим у значення true та фіксує точний час виконання у полі completedAt. Якщо запис вже існує операція не виконується повторно. Після збереження прогресу система перераховує загальний відсоток завершення маршруту як відношення кількості виконаних вузлів до загальної кількості вузлів у маршруті.

Паралельно система нараховує досвід за виконаний вузол. Базова кількість становить 10 балів за сам факт виконання. Додатково нараховується 5 балів якщо до вузла прикріплено зображення та 3 бали якщо у користувача є опублікована публікація в межах цього захоплення. Нараховані бали додаються до загального поля xp у таблиці User а також оновлюють відповідний запис у таблиці UserHobbyXP для коректного відображення в таблиці лідерів конкретного захоплення. Інтерфейс оновлюється і відображає актуальний стан прогресу, оновлений показник досвіду та теплову карту активності в профілі користувача. Діаграму активності наведено на рис. 2.5.

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ІЗ	Арк.
		Чижомотря О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		52

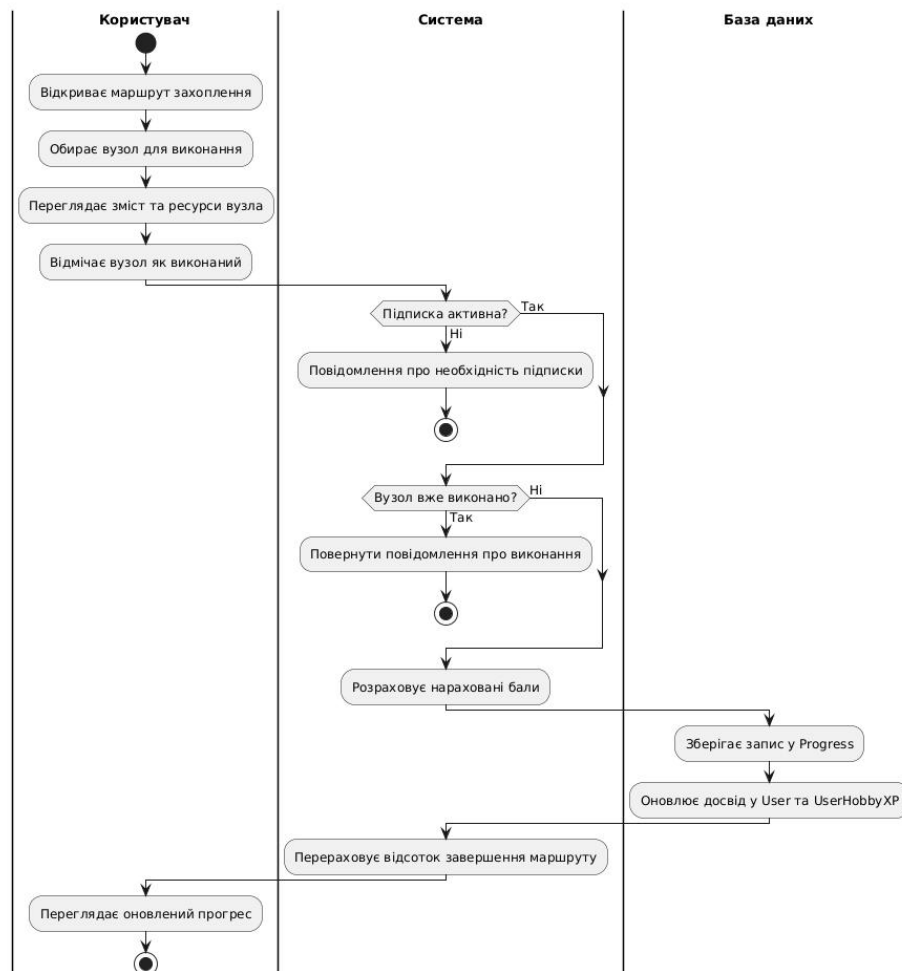


Рис. 2.5. Діаграма активності алгоритму проходження маршруту

Алгоритм модерації публікацій забезпечує якість контенту в стрічці захоплення через обов'язкову перевірку матеріалів практиком перед їх відображенням іншим учасникам. Передумовою є наявність у системі призначеного практика для конкретного захоплення та авторизованого користувача який хоче опублікувати матеріал.

Авторизований користувач переходить до вкладки публікацій захоплення та створює новий матеріал із текстом та зображенням. Після збереження публікація автоматично отримує статус pending і не відображається в загальній стрічці для інших учасників. Автор бачить свою публікацію зі статусом очікування у власному профілі.

Практик призначений до цього захоплення бачить у своїй панелі всі публікації зі статусом pending. Він переглядає кожну з них та приймає

рішення. У разі схвалення статус публікації змінюється на published, фіксується дата публікації у полі publishedAt і матеріал стає доступним у загальній стрічці для всіх підписників захоплення. У разі відхилення статус повертається до private і публікація стає видимою лише автору без можливості повторного подання без редагування. Адміністратор має розширені права і може у будь-який момент приховати будь-яку опубліковану публікацію змінивши її статус назад на private або повністю видалити її з бази даних. Діаграму активності наведено на рис. 2.6.

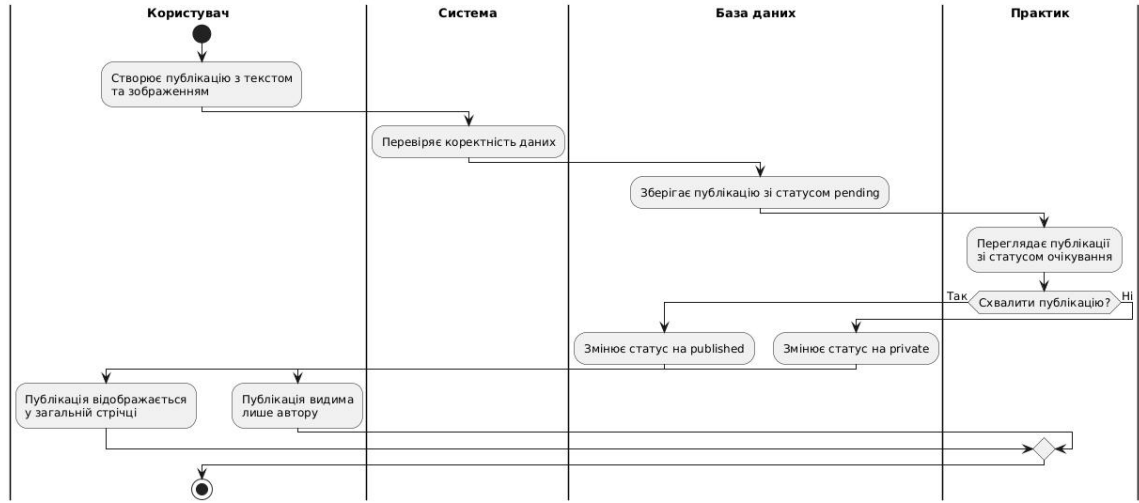


Рис. 2.6. Діаграма активності алгоритму модерації публікацій

Алгоритм подання та розгляду заявки на статус практика реалізує процес надання авторизованому користувачеві розширених прав для керування контентом конкретного захоплення. Передумовою є авторизація користувача та відсутність у нього поточної ролі практика або адміністратора.

Авторизований користувач переходить до розділу подання заявок у своєму профілі та заповнює форму із зазначенням захоплення до якого він хоче долучитися та обґрунтуванням свого досвіду у цьому напрямі. Після натискання кнопки відправлення система перевіряє заповненість обов'язкових полів та зберігає запис у таблиці Request зі статусом pending. Користувач отримує повідомлення про успішне подання заявки.

Адміністратор бачить нову заявку у відповідній вкладці адміністративної панелі разом із деталями: ім'ям користувача, обраним захопленням та текстом обґрунтування. Після ознайомлення він приймає рішення. У разі схвалення система виконує кілька дій одночасно: роль користувача змінюється на practitioner, у записі заявки встановлюється ідентифікатор захоплення у полі hobbyId та статус змінюється на approved. Після наступного входу до системи користувач отримує JWT-токен із оновленою роллю та бачить вітальне повідомлення з поясненням своїх нових можливостей через модальне вікно. У разі відхилення статус заявки змінюється на rejected і роль користувача залишається незмінною. Діаграму активності наведено на рис. 2.7.

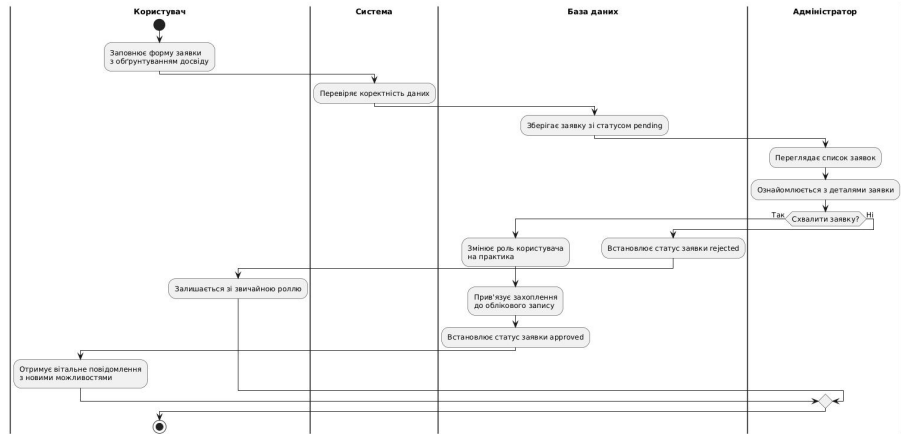


Рис. 2.7. Діаграма активності алгоритму подання заявки на статус практика

Завершальним розглядається алгоритм створення події практиком, метою якого є організація тематичного заходу в межах конкретного захоплення з можливістю реєстрації учасників. Передумовою є наявність у користувача ролі практика та призначеного захоплення.

Практик переходить до вкладки подій свого захоплення та натискає кнопку створення нової події. Система відображає форму де практик заповнює назву, опис, формат заходу та дату проведення. Залежно від обраного формату форма динамічно відображає додаткові поля: для онлайн-подій це посилання для підключення, для офлайн-заходів це місце проведення. Також може бути вказана максимальна кількість учасників.

Після заповнення форми та натискання кнопки збереження клієнтська частина виконує базову валідацію і надсилає дані на сервер. Сервер перевіряє чи дата проведення відстоїть від поточного моменту не менше ніж на 24 години. Якщо перевірку пройдено система зберігає запис у таблиці Event і подія стає доступною для перегляду всіма користувачами платформи. Якщо дата не відповідає вимогам сервер повертає помилку валідації і форма залишається відкритою для виправлення. Авторизований користувач може переглянути деталі події та зареєструватися на неї після чого система створює запис у таблиці EventAttendee із обов'язковим полем phone для контакту. Діаграму активності наведено на рис. 2.8.

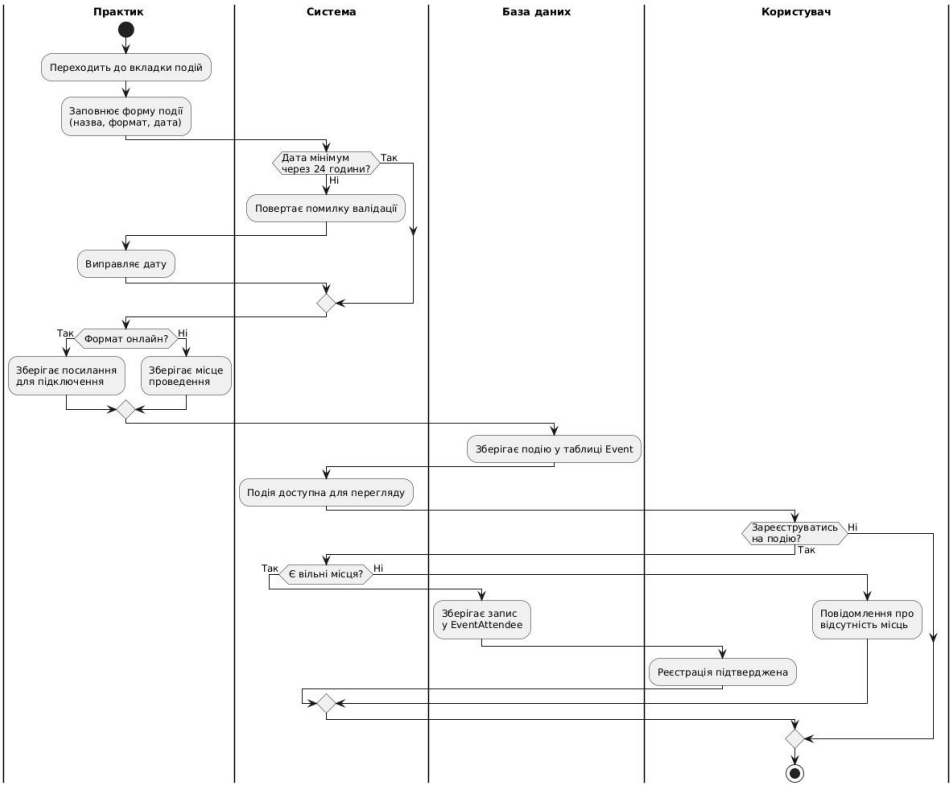


Рис. 2.8. Діаграма активності алгоритму створення події

Розглянуті алгоритми охоплюють основні сценарії взаємодії користувачів із платформою та відображають логіку роботи ключових функціональних модулів системи. Кожен із них реалізований з урахуванням рольової моделі доступу та забезпечує коректну обробку як успішних сценаріїв так і виняткових ситуацій. Сукупність описаних алгоритмів формує

цілісну картину функціонування платформи та створює основу для подальшої реалізації та тестування програмного комплексу.

Загальну структуру сервісів серверної та клієнтської частини відображено на діаграмі класів наведеній на рис. 2.9. Серверна частина організована навколо восьми основних сервісів: AuthService, UsersService, QuizService, ProgressService, EventsService, PostsService, RoadmapService та HobbiesService. Кожен сервіс відповідає за окремий домен системи та взаємодіє з базою даних через єдиний екземпляр PrismaClient що реалізує патерн Singleton. Доступ до захищених маршрутів контролюється через ланцюжок AuthMiddleware та RoleMiddleware що реалізує патерн Chain of Responsibility. Сортювання захоплень у каталозі реалізовано через патерн Strategy де кожен із трьох варіантів є окремою реалізацією інтерфейсу ISortStrategy. На клієнтській частині взаємодія з API централізована через єдиний AxiosInstance що виступає фасадом для всіх модулів запитів, а глобальний стан користувача та токена керується через атоми Jotai.

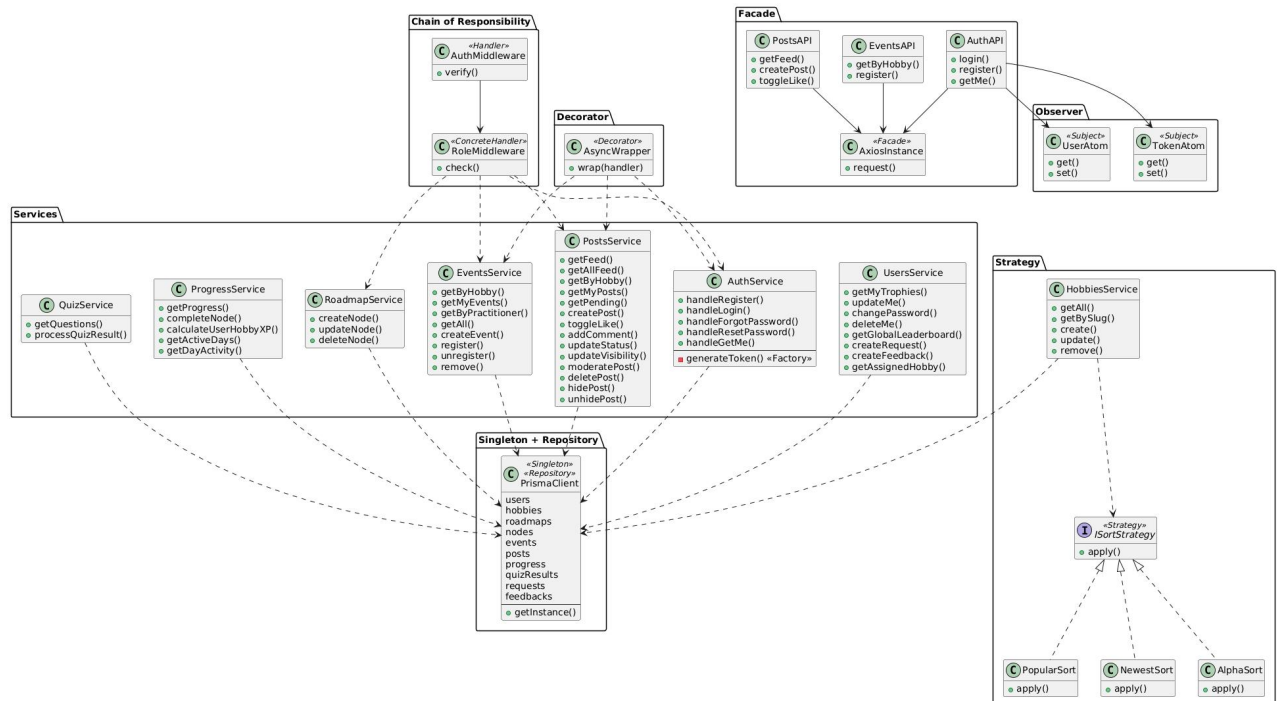


Рис. 2.9. Діаграма класів основних сервісів системи

Розглянуті алгоритми охоплюють основні сценарії взаємодії користувачів із платформою та відображають логіку роботи ключових функціональних модулів системи. Кожен із них реалізований з урахуванням рольової моделі доступу та забезпечує коректну обробку як успішних сценаріїв так і виняткових ситуацій. Сукупність описаних алгоритмів формує цілісну картину функціонування платформи та створює основу для подальшої реалізації та тестування програмного комплексу.

2.4 Реалізація основних функціональних модулів системи

Одним із перших функціональних модулів з якими взаємодіє користувач є модуль підбору дозвілля. Саме через нього платформа формує перше персоналізоване враження і пропонує захоплення що відповідають індивідуальним інтересам. Поряд із ним реалізовано модуль відстеження прогресу що супроводжує користувача впродовж усього освоєння захоплення та модуль подій що забезпечує соціальну взаємодію між учасниками. Для кожного з цих модулів наведено фрагменти коду що демонструють найбільш значущі архітектурні рішення та пояснюють логіку їх роботи. Повний лістинг вихідного коду наведено у додатках до роботи.

Серверна частина побудована за принципом розподілу відповідальності між трьома рівнями. Маршрути визначають HTTP-ендпоінти та прив'язують до них middleware для перевірки автентифікації і ролей. Контролери отримують параметри запиту, викликають відповідний сервіс та формують відповідь клієнту. Сервіси містять всю бізнес-логіку та взаємодіють із базою даних через Prisma. Такий поділ дозволяє змінювати логіку обробки даних без впливу на транспортний рівень і навпаки. Обробка помилок реалізована через викидання об'єктів із полями status та message які контролери перехоплюють і повертають клієнту у стандартизованому вигляді.

Клієнтська частина побудована на React із використанням Vite як інструменту збірки. Для керування глобальним станом використовується

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижомотря О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		58

Jotai де зберігаються дані поточного користувача та JWT-токен. Токен зберігається в localStorage і автоматично додається до кожного запиту через Axios-інтерсептор. Компонент MainLayout при кожному завантаженні сторінки викликає ендпоінт getMe що оновлює дані користувача включаючи поточну роль та ідентифікатор призначеного захоплення для практика. Це забезпечує актуальність даних після зміни ролі без необхідності повторного входу.

Першим детально розглядається модуль підбору дозвілля реалізований у сервісі quiz. Він є центральним з точки зору персоналізації платформи і реалізує гібридний алгоритм рекомендацій [25; 26] що поєднує два підходи: підбір на основі тегів та колаборативну фільтрацію на основі поведінки схожих користувачів. Алгоритм працює у п'ять послідовних кроків.

На першому кроці система збирає зважені теги з відповідей користувача. Тест складається з 15 питань що мають різну числову вагу від одного до трьох. Питання з вищою вагою розташовані ближче до кінця тесту і дозволяють уточнити інтереси користувача після загальних питань на початку. При обробці відповідей теги з обраних варіантів накопичуються у словнику де значення кожного тегу збільшується на вагу питання де він зустрівся.

```
for (const [questionIdStr, optionId] of Object.entries(answers)) {
  const questionId = parseInt(questionIdStr, 10)
  const question = QUESTIONS.find((q) => q.id === questionId)
  if (!question) continue

  const option = question.options.find((o) => o.id === optionId)
  if (!option) continue

  for (const tag of option.tags) {
    weightedTags[tag] = (weightedTags[tag] || 0) + question.weight
  }
}
```

Це дозволяє враховувати не лише сам факт вибору тегу а й наскільки важливим з точки зору алгоритму було питання де він зустрівся.

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ІЗ	Арк.
		Чижмоштра О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		59

На другому кроці для кожного дозвілля в каталозі обчислюється показник відповідності тегів. Система проходить по тегах кожного захоплення та підсумовує ваги тих що збіглися з тегами відповідей користувача. Отримане значення нормалізується відносно максимального серед усіх захоплень що приводить усі оцінки до діапазону від нуля до ста.

```
const tagScores = activities.map((activity) => {
  const rawScore = activity.tags.reduce(
    (sum, tag) => sum + (weightedTags[tag] || 0), 0
  )
  return { activity, rawScore }
})

const maxRaw = Math.max(...tagScores.map((s) => s.rawScore), 1)

const normalized = tagScores.map(({ activity, rawScore }) => ({
  activity,
  tagScore: Math.round((rawScore / maxRaw) * 100),
})))
```

Третій крок реалізує колаборативну частину алгоритму. Система шукає у базі даних попередні результати тестів інших користувачів що мали щонайменше три спільних теги з поточним результатом. Якщо таких користувачів знайдено десять і більше система вважає що даних достатньо для обчислення показника популярності. В іншому випадку колаборативна частина не використовується і алгоритм залишається суто теговим. Такий підхід дозволяє системі коректно функціонувати навіть на початку роботи платформи коли база користувачів ще невелика.

```
const trulySimilar = candidateResults.filter((result) => {
  const matches = result.tags.filter((t) => tagList.includes(t)).length
  return matches >= 3
})

const hasSufficientData = similarUserIds.length >= 10
```

На четвертому кроці якщо даних достатньо система отримує підписки схожих користувачів та обчислює для кожного захоплення частку схожих

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоторя О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		60

користувачів що підписані на нього. Це і є показник популярності нормалізований до діапазону від нуля до ста. Він відображає реальну поведінку людей зі схожими інтересами і доповнює суто тегову оцінку соціальним сигналом.

П'ятим кроком обчислюється фінальна гібридна оцінка. Якщо даних для колаборативної частини достатньо оцінка розраховується як зважена сума двох показників де коефіцієнт відповідності тегів становить 0.6 а коефіцієнт популярності 0.4. Якщо даних недостатньо використовується лише показник тегів.

```
const score = hasSufficientData
  ? Math.round(0.6 * tagScore + 0.4 * popularityScore)
  : tagScore;
```

Результатом є три захоплення з найвищою оцінкою які зберігаються разом із масивом тегів та вихідними відповідями у таблиці QuizResult. Поле algorithm у відповіді вказує який режим було використано що корисно для аналітики та розуміння поведінки системи в різних умовах. Алгоритм автоматично переходить від тегового до гібридного режиму в міру накопичення даних без будь-яких змін у коді.

Другим розглядається модуль нарахування досвіду реалізований у сервісі progress. Він є основою мотиваційної механіки платформи і впливає як на загальний профіль користувача так і на таблицю лідерів конкретного захоплення.

При відмітці вузла як виконаного функція completeNode спочатку перевіряє існування вузла та відсутність повторного виконання. Потім вона використовує операцію upsert що дозволяє або створити новий запис прогресу або оновити наявний. Це захищає від дублювання навіть якщо клієнт надішле кілька однакових запитів одночасно.

```
export const completeNode = async (userId, nodeId, { hobbyId, imageUrl, status } = {}) => {
```

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижомотря О.В.				61
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    await prisma.progress.upsert({
      where: { userId_nodeId: { userId, nodeId } },
      update: { completed: true },
      create: { userId, nodeId, completed: true },
    })

    let xpEarned = 10
    if (imageUrl) xpEarned += 5
    if (status === 'pending' || status === 'published') xpEarned += 3

    await prisma.user.update({
      where: { id: userId },
      data: { xp: { increment: xpEarned } },
    })

    const totalXP = hobbyId ? await calculateUserHobbyXP(userId, hobbyId) : xpEarned
    return { success: true, xpEarned, totalXP }
  }

```

Нарахування досвіду відбувається за трьома критеріями одночасно. Базові 10 балів нараховуються за виконання вузла. Додаткові 5 балів нараховуються якщо до вузла прикріплено зображення що стимулює користувачів документувати свій прогрес. Ще 3 бали нараховуються якщо публікація пов'язана з виконанням має статус pending або published що заохочує ділитися результатами зі спільнотою.

Функція calculateUserHobbyXP перераховує досвід користувача в межах конкретного захоплення з нуля на основі поточного стану бази даних. Вона враховує всі виконані вузли маршруту незалежно від їх джерела тобто вузли практика зараховуються так само як і вузли адміністратора. Такий підхід гарантує що таблиця лідерів завжди відображає актуальні дані навіть якщо структура маршруту змінилась після того як користувач вже виконав частину вузлів.

Окрім основного прогресу модуль підтримує теплову карту активності. Функція getActiveDays повертає список унікальних дат коли користувач виконував вузли за вказаний період. Функція getDayActivity повертає деталі активності за конкретну дату включаючи назви вузлів та захоплень. Ці дані використовуються на сторінці профілю для відображення регулярності занять у вигляді теплової карти.

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ІЗ	Арк.
		Чижмоторя О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		62

Третім розглядається модуль подій реалізований у сервісі events. Він демонструє цікаві рішення пов'язані з рольовою моделлю доступу та багаторівневою валідацією даних.

Отримання подій реалізовано через чотири окремих методи залежно від ролі користувача. Метод getByHobby повертає події конкретного захоплення і доступний для всіх відвідувачів. Метод getMyEvents повертає події лише тих захоплень на які підписаний поточний користувач що реалізує персоналізовану стрічку подій. Метод getByPractitioner визначає призначене захоплення практика через таблицю Request та повертає події лише цього захоплення. Метод getAll доступний виключно адміністратору і повертає всі події платформи без будь-якої фільтрації. У кожному режимі до результату додається обчислюване поле isRegistered що показує чи поточний користувач вже зареєстрований на кожну конкретну подію.

```
export const register = async (userId, eventId, phone) => {
  const event = await prisma.event.findUnique({ where: { id: eventId } })
  if (!event) throw { status: 404, message: 'Event not found' }

  const existing = await prisma.eventAttendee.findUnique({
    where: { userId_eventId: { userId, eventId } },
  })
  if (existing) throw { status: 409, message: 'Already registered' }

  if (event.maxAttendees) {
    const count = await prisma.eventAttendee.count({ where: { eventId } })
    if (count >= event.maxAttendees) throw { status: 400, message: 'Event is full' }
  }

  return prisma.eventAttendee.create({ data: { userId, eventId, phone: phone || null } })
}
```

При реєстрації на подію система послідовно виконує три перевірки. Спочатку перевіряється існування події в базі даних. Потім перевіряється чи не зареєстрований вже цей користувач на дану подію що запобігає дублюванню записів. Нарешті якщо для події встановлено обмеження на кількість учасників система підраховує поточну кількість і блокує реєстрацію якщо місця закінчились. Кожна перевірка повертає специфічний HTTP-код

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижевська О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		63

помилки що дозволяє клієнтській частині відображати відповідне повідомлення користувачеві.

При створенні події реалізовано два рівні валідації дати. Клієнтська частина перевіряє дату ще до відправки запиту що забезпечує миттєвий зворотний зв'язок. Серверна частина повторно перевіряє що дата проведення відстоїть від поточного моменту не менше ніж на 24 години що захищає від обходу клієнтської валідації. Поля link та location заповнюються вибірково залежно від формату заходу: для онлайн-подій зберігається лише посилання, для офлайн-заходів лише місце проведення що забезпечує чистоту даних у базі.

Висновки до другого розділу

У другому розділі здійснено проектування структури та архітектури вебплатформи для пошуку дозвілля. На основі визначених функціональних і нефункціональних вимог сформовано рольову модель доступу що охоплює чотири ролі: неавторизований користувач, авторизований користувач, практик та адміністратор. Для кожної ролі визначено перелік доступних сценаріїв взаємодії із системою що деталізовано у вигляді таблиць прецедентів та діаграми варіантів використання.

Спроектовано реляційну базу даних що складається з 19 таблиць організованих у чотири логічні групи. Перша група описує користувачів та їх ролі, друга охоплює захоплення, категорії та маршрути освоєння, третя стосується соціальної складової платформи, четверта фіксує прогрес та взаємодію користувачів із контентом. Структура бази даних відображає всі зв'язки між сутностями та забезпечує цілісність даних на рівні реляційної моделі.

Описано алгоритми роботи шести ключових сценаріїв системи: підбір дозвілля на основі тесту з гібридним алгоритмом рекомендацій, підписка на захоплення, проходження маршруту та нарахування досвіду, модерація

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ІЗ	Арк.
		Чижмоторя О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		64

публікацій, подання заявки на статус практика та створення події. Для кожного алгоритму побудовано діаграму активності що відображає послідовність дій користувача та системних компонентів.

Реалізовано три ключові функціональні модулі платформи. Модуль підбору дозвілля реалізує гібридний алгоритм що поєднує підбір на основі зважених тегів із колаборативною фільтрацією на основі поведінки схожих користувачів і автоматично адаптується залежно від обсягу накопичених даних. Модуль прогресу забезпечує відстеження виконання вузлів маршруту, нарахування досвіду за кількома критеріями та формування теплової карти активності користувача. Модуль подій реалізує чотири режими отримання даних залежно від ролі користувача та дворівневу валідацію при створенні заходу.

Отримані результати формують цілісну технічну основу платформи та створюють передумови для подальшого тестування системи і підготовки до розгортання.

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмотря О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		65

РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ВЕБПЛАТФОРМОЮ ДЛЯ ПОШУКУ ДОЗВІЛЛЯ

3.1 Порядок розгортання та налаштування системи

Розгортання платформи реалізовано з використанням сучасних підходів до контейнеризації та автоматизованої доставки коду. Застосунок повністю контейнеризований за допомогою Docker [29], де кожен компонент системи розгортається в окремому контейнері. Такий підхід забезпечує ізоляцію середовищ, відтворюваність розгортання та незалежність від конфігурації хостового середовища. Контейнеризація також спрощує горизонтальне масштабування окремих компонентів та дозволяє запускати повний стек системи як локально, так і в хмарному середовищі без зміни коду застосунку.

Dockerfile серверної частини реалізує однофазову збірку на основі образу node:20-alpine. Під час збірки встановлюються залежності та генерується Prisma-клієнт. При запуску контейнера автоматично застосовуються міграції бази даних командою prisma migrate deploy, після чого стартує Node.js сервер. Використання легковагового alpine-образу дозволяє суттєво зменшити розмір фінального контейнера порівняно зі стандартним node-образом, що прискорює завантаження при розгортанні. Автоматичне застосування міграцій при старті гарантує, що схема бази даних завжди відповідає поточній версії коду без ручного втручання оператора.

Dockerfile клієнтської частини реалізує двофазову збірку. На першій фазі виконується компіляція React-застосунку засобами Vite, де URL серверної частини передається як аргумент збірки. На другій фазі статичні файли копіюються в образ nginx:alpine. Такий підхід дозволяє відокремити середовище збірки від середовища виконання, що значно зменшує розмір фінального образу та виключає з нього залежності, потрібні лише під час компіляції. Entrypoint-скрипт при старті динамічно переписує конфігурацію

nginx відповідно до порту, наданого платформою, що забезпечує сумісність з хмарними середовищами, де порт призначається динамічно. nginx також налаштовано для коректної обробки SPA-маршрутизації — усі запити до невідомих шляхів перенаправляються на index.html.

Для локальної розробки створено файл docker-compose.yml, який запускає всі три сервіси з перевіркою готовності бази даних через healthcheck перед стартом серверної частини.

Визначення залежностей між сервісами через depends_on з умовою service_healthy усуває помилки підключення, що виникають при паралельному старті контейнерів. Завдяки цьому розробник може відтворити повне виробниче середовище однією командою docker compose up без необхідності ручного налаштування будь-якого компонента. Структуру сервісів наведено в табл. 3.1.

Таблиця 3.1

Сервіси платформи

Сервіс	Тип	Опис
PostgreSQL	Managed Database	Реляційна база даних
Backend	Web Service	Express API з healthcheck ендпоінтом
Frontend	Web Service	nginx зі статикою та SPA-роутингом

Система неперервної інтеграції та доставки (CI/CD) реалізована на базі GitLab CI/CD [30]. Конфігурація pipeline складається з чотирьох стадій: test, build, deploy та load-test. Стадія тестування запускається автоматично при кожному push у гілки develop та main, а також при створенні Merge Request. Паралельне виконання задач тестування серверної та клієнтської частин скорочує загальний час проходження pipeline порівняно з послідовним запуском. Збір покриття коду під час тестування серверної частини дозволяє відстежувати метрику охопленості тестами в часі та виявляти недостатньо протестовані модулі. Діаграму конфігурації pipeline наведено на рис. 3.1.

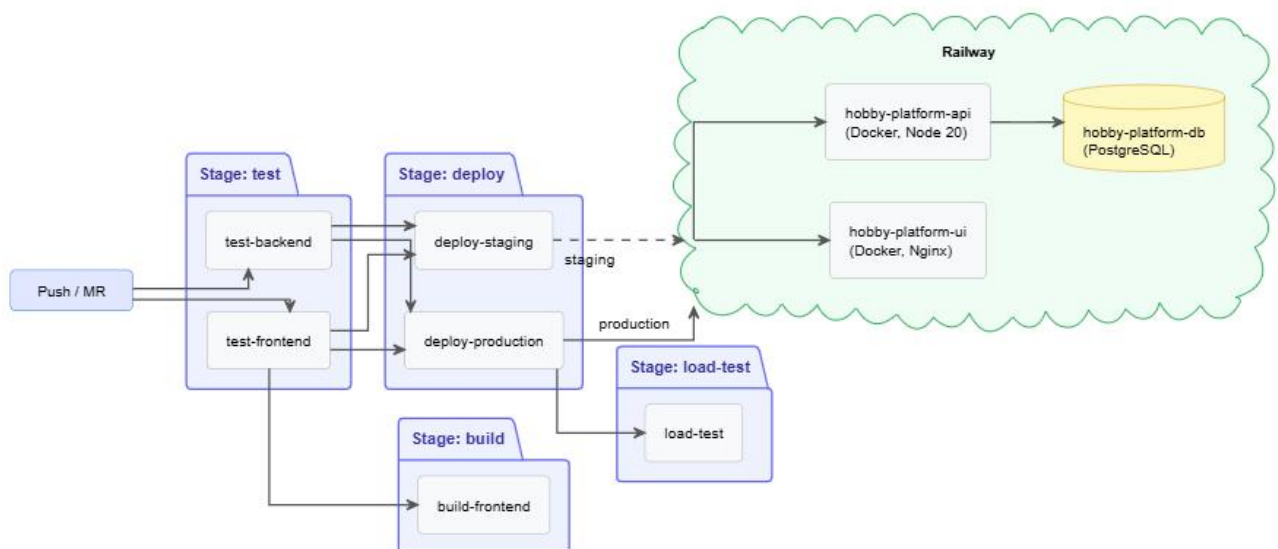


Рис. 3.1. Діаграма конфігурації CI/CD pipeline

Стадія розгортання реалізує безперервну доставку за стратегією GitFlow: при push у гілку develop код автоматично потрапляє до середовища staging після проходження тестів, а при push у гілку main, розгортається у виробниче середовище. Наявність окремого staging-середовища дозволяє перевіряти функціонал перед виходом у production та знижує ризик появи регресій у робочому застосунку. Механізм розгортання базується на дзеркалюванні репозиторію: GitLab CI після успішних тестів надсилає поточний коміт у відповідну гілку дзеркального репозиторію на GitHub, який підключено до хмарної платформи Railway. Railway виявляє зміну і автоматично ініціює новий деплой, що скорочує час між злиттям коду і його появою у виробничому середовищі та усуває потребу в сторонніх інструментах для інтеграції між GitLab і хмарною платформою. Навантажувальне тестування через кб запускається вручну лише в основній гілці, що запобігає надмірному навантаженню на середовище при рутинних оновленнях.

Виробниче середовище розгорнуто на платформі Railway, де проєкт складається з трьох сервісів у двох середовищах: staging та production. Конфігурація кожного сервісу описана у файлах railway.json в директоріях серверної та клієнтської частин, що дозволяє зберігати налаштування

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоштра О.В.				68
Змн.	Арк.	№ докум.	Підпис	Дата		

розгортання безпосередньо в репозиторії та підтримувати їх у актуальному стані разом із кодом. Усі чутливі змінні середовища, зокрема JWT-секрет, ключі Cloudinary та облікові дані поштового сервісу, передаються через механізм змінних платформи і ніколи не зберігаються у репозиторії, що відповідає вимогам інформаційної безпеки та методології Twelve-Factor App. Використання окремих середовищ дозволяє підключати різні облікові записи зовнішніх сервісів для staging і production, що запобігає потраплянню тестових даних до виробничого середовища. Діаграму розгортання системи наведено на рис. 3.2.

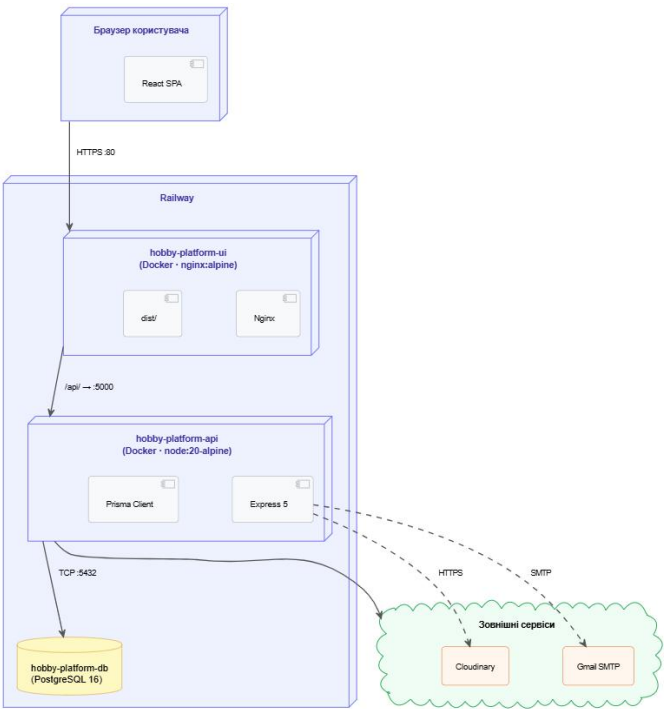


Рис. 3.2. Діаграма розгортання системи

3.2 Опис інтерфейсу вебплатформи

Інтерфейс платформи побудований з урахуванням рольової моделі доступу та орієнтований на зручність користувача на кожному етапі взаємодії з системою. Залежно від ролі користувач бачить різний набір функціональних можливостей: неавторизований відвідувач має доступ лише до публічної частини, авторизований користувач отримує повний функціонал для освоєння захоплень, практик керує контентом свого заняття через окрему

панель, а адміністратор має доступ до централізованої панелі керування платформою. Вітальна сторінка платформи є першим екраном з яким стикається неавторизований відвідувач. Як показано на рис. 3.3, вона пропонує три способи розпочати роботу з платформою: пройти тест для персоналізованого підбору дозвілля, отримати випадкове захоплення або перейти одразу до каталогу. У правому верхньому куті розташована кнопка входу до системи.

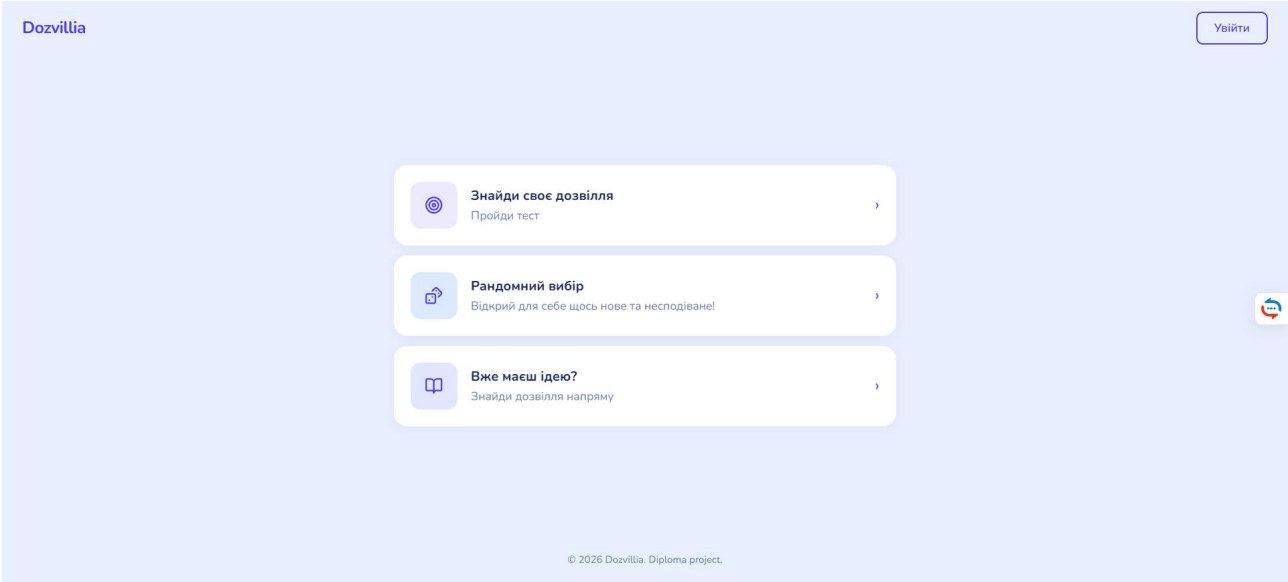


Рис. 3.3 Вітальна сторінка платформи

При натисканні на першу картку відкривається сторінка тесту. Вона відображає питання послідовно одне за одним із прогрес-баром у верхній частині що показує поточний номер питання із загальних п'ятнадцяти. Як наведено на рис. 3.4, кожне питання містить чотири варіанти відповіді розташовані у вигляді сітки та позначені літерами від А до D. Після вибору відповіді система автоматично переходить до наступного питання. У нижній частині сторінки передбачена кнопка повернення до попереднього питання.

#1

Де ти найчастіше любиш проводити вільний час?

A Вдома, у затишку

B На природі, на свіжому повітрі

C У спортзалі або студії

D Біля води: річка, море, озеро

← Назад

Рис. 3.4. Сторінка тесту підбору дозвілля

Після завершення тесту система відображає сторінку результатів із трьома рекомендованими захопленнями. Як показано на рис. 3.5, кожне захоплення представлено карткою з порядковим номером, круговою діаграмою відповідності у відсотках та категорією. Для переходу до детальної сторінки захоплення передбачена кнопка "Детальніше", а також можливість пройти тест повторно.

Твоє ідеальне дозвілля ✨

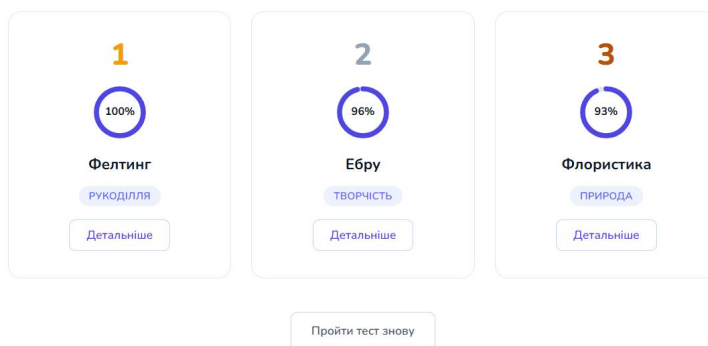


Рис. 3.5 Сторінка результатів тесту

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоторя О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		71

При переході за посиланням реєстрації відкривається форма створення облікового запису наведена на рис. 3.6. Вона містить поля для введення імені, електронної пошти, пароля та його підтвердження з вимогою мінімум шести символів.

The screenshot shows a registration form titled "Створи акаунт" (Create Account) with the subtitle "Приєднуйся до Dozwillia". The form contains the following fields:

- Ім'я** (Name): A text input field with the placeholder "Іван Іваненко".
- Email**: A text input field with the placeholder "you@example.com".
- Пароль** (Password): A password input field with a strength indicator "мінімум 6 символів" (minimum 6 symbols) and a visibility toggle.
- Підтвердження пароля** (Password Confirmation): A password input field with a visibility toggle.

 Below the fields is a blue "Зареєструватись" (Register) button. At the bottom of the form, there is a link: "Вже є акаунт? Увійти" (Already have an account? Log in). The background is a light blue gradient with the "Dozwillia" logo in the top left and a "Увійти" (Log in) button in the top right.

Рис. 3.6 Форма реєстрації

Для отримання доступу до повного функціоналу платформи користувач натискає кнопку входу у правому верхньому куті та потрапляє на сторінку авторизації. Як наведено на рис. 3.7, форма містить поля для введення електронної пошти та пароля з можливістю відновлення пароля та переходу до реєстрації.

The screenshot shows a login form titled "Ласкаво просимо" (Welcome) with the subtitle "Увійди щоб продовжити" (Log in to continue). The form contains the following fields:

- Email**: A text input field with the placeholder "you@example.com".
- Пароль** (Password): A password input field with a strength indicator and a "Забути пароль?" (Forgot password?) link.

 Below the fields is a blue "Увійти" (Log in) button. At the bottom of the form, there is a link: "Ще немає акаунту? Зареєструватись" (Don't have an account? Register). The background is a light blue gradient with the "Dozwillia" logo in the top left and a "Реєстрація" (Registration) button in the top right. A footer at the bottom reads "© 2026 Dozwillia. Diploma project."

Рис. 3.7 Сторінка входу до системи

Після успішної авторизації користувач потрапляє до каталогу захоплень наведеного на рис. 3.8 де може обрати заняття що його цікавить. Сторінка містить пошуковий рядок, фільтрацію за категоріями у вигляді тегів та сортування. Захоплення відображаються у вигляді карток із зображенням, назвою та категорією.

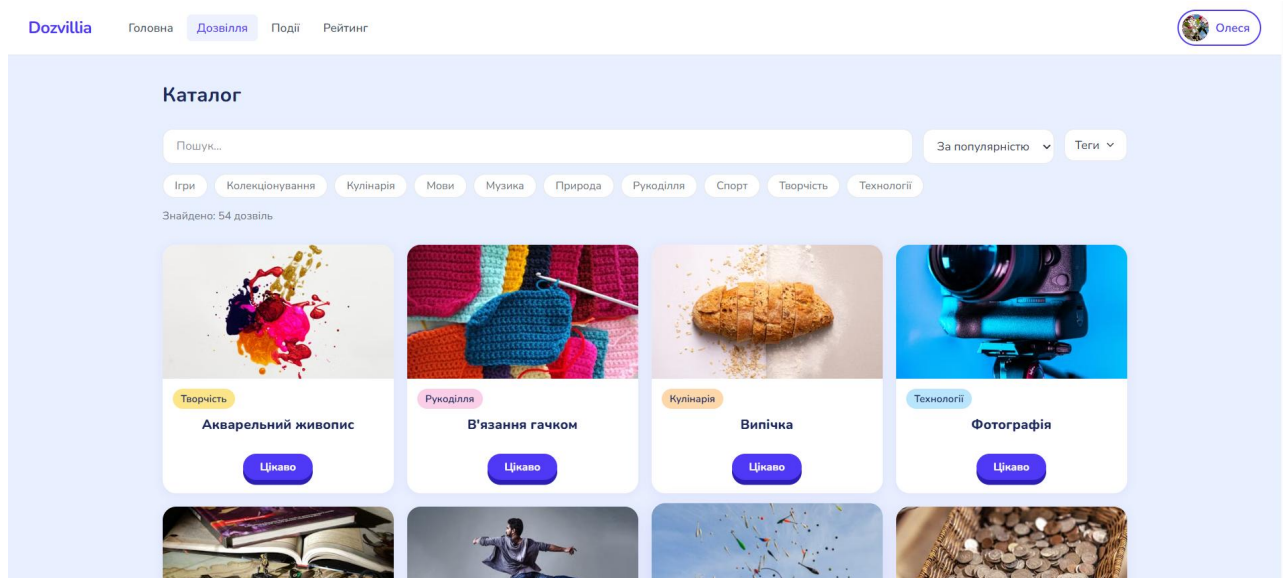


Рис. 3.8. Сторінка каталогу захоплень

При натисканні на картку захоплення у каталозі користувач переходить на його детальну сторінку наведену на рис. 3.9. У верхній частині відображається зображення, назва, теги, кількість підписників, вподобань та загальний досвід спільноти. Нижче розташовані вкладки Інформація, Блог, Шлях, Події та Рекорд. Вкладка Інформація містить детальний опис захоплення.

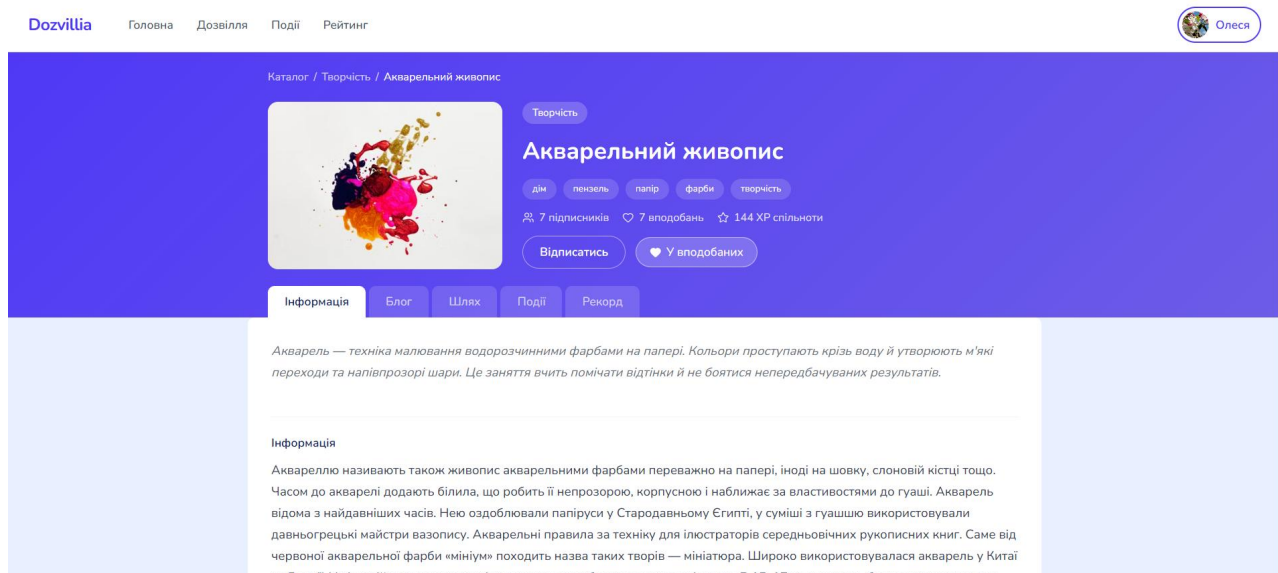


Рис. 3.9. Сторінка захоплення, вкладка Інформація

При переході на вкладку Блог наведену на рис. 3.10 відображається стрічка публікацій учасників цього конкретного захоплення з можливістю вподобати та коментувати матеріали.

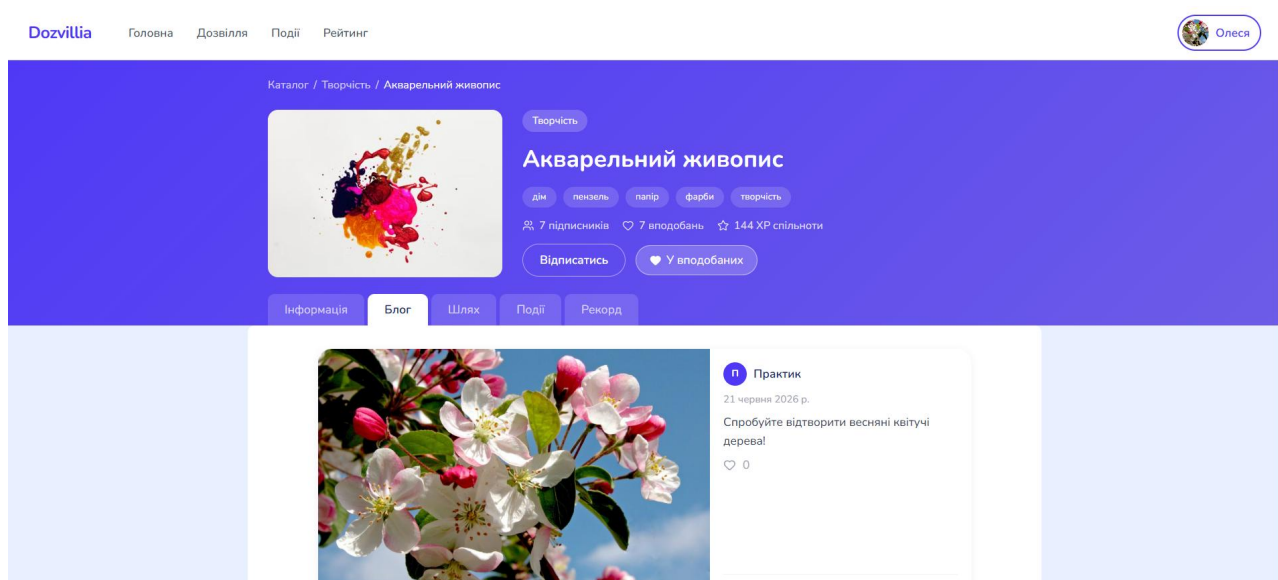


Рис. 3.10. Сторінка захоплення, вкладка Блог

Вкладка Шлях наведена на рис. 3.11 відображає візуальну схему покрокового маршруту освоєння захоплення. Вузли адміністратора розташовані у вигляді синього ланцюжка зі змінним відступом, а вузли практика відображаються окремим зеленим рядком із підписом імені автора.

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижевська О.В.				74
Змн.	Арк.	№ докум.	Підпис	Дата		

Виконані вузли позначаються і над ними відображається кількість нарахованого досвіду.

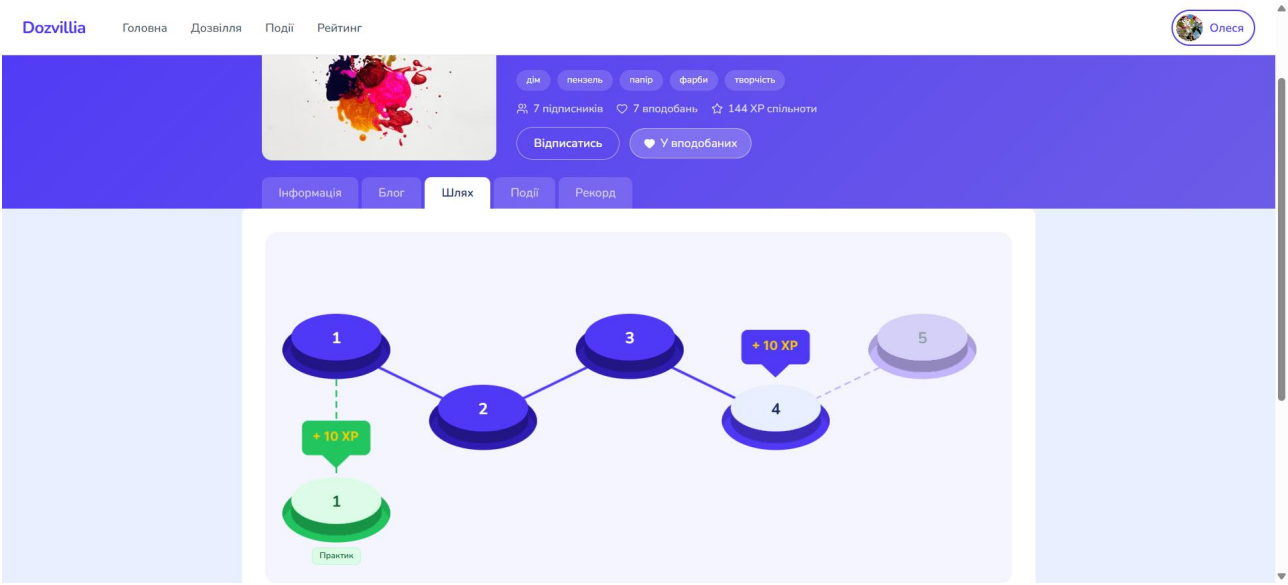


Рис. 3.11. Сторінка захоплення, вкладка Шлях

При натисканні на вузол відкривається його детальний вміст наведений на рис. 3.12 з описом, прив'язаними ресурсами у вигляді посилань та можливістю завантажити фото виконаного завдання. Після виконання система нараховує досвід і відображає повідомлення з кількістю отриманих балів. Також передбачена можливість опублікувати результат у блог захоплення.

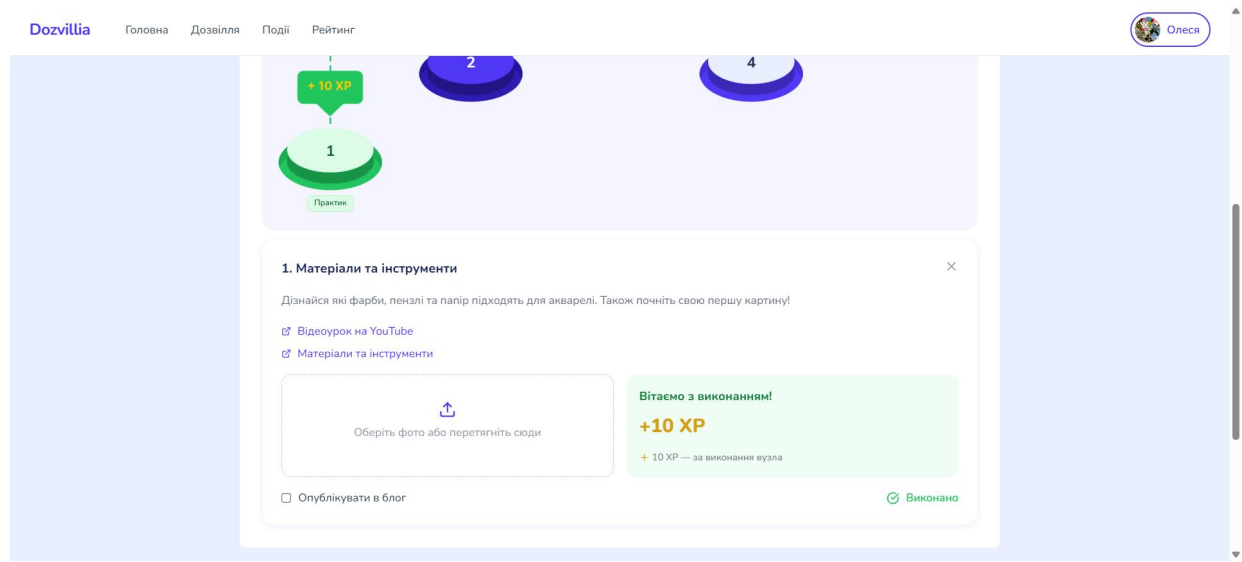


Рис. 3.12. Вміст вузла маршруту з нарахуванням досвіду

Вкладка Події наведена на рис. 3.13 відображає календар та список найближчих заходів пов'язаних із конкретним захопленням з можливістю реєстрації на кожен із них.

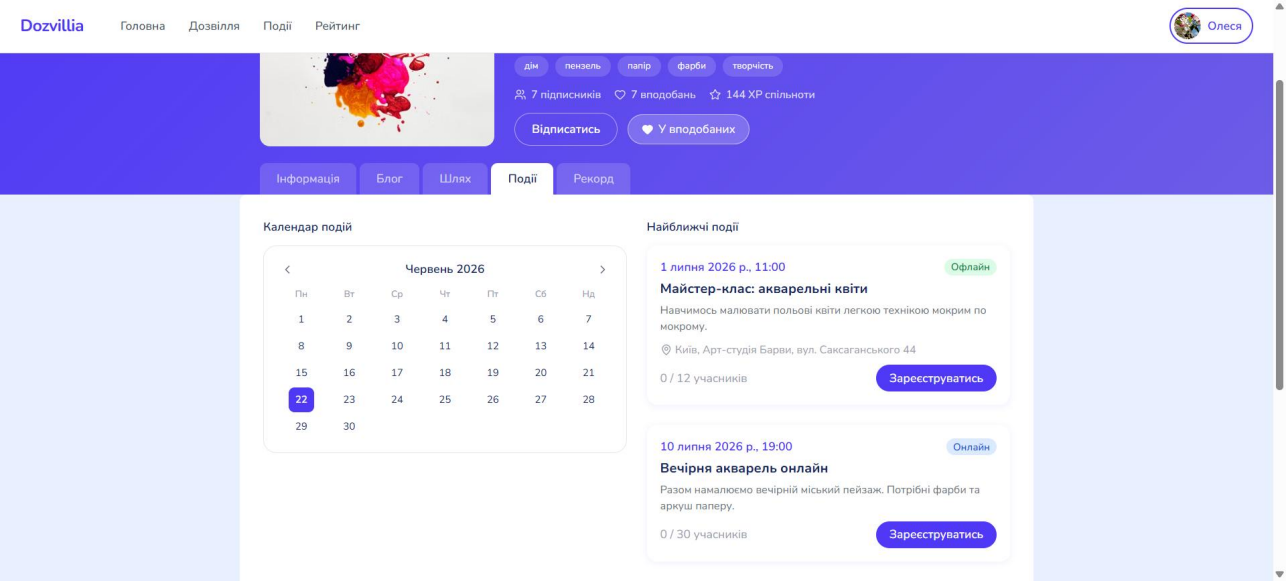


Рис. 3.13. Сторінка захоплення, вкладка Події

При переході на вкладку Рекорд наведену на рис. 3.14 користувач бачить таблицю лідерів конкретного захоплення де відображається рейтинг учасників за накопиченим досвідом саме в цьому дозвіллі. Поточний користувач виділяється рамкою у списку.

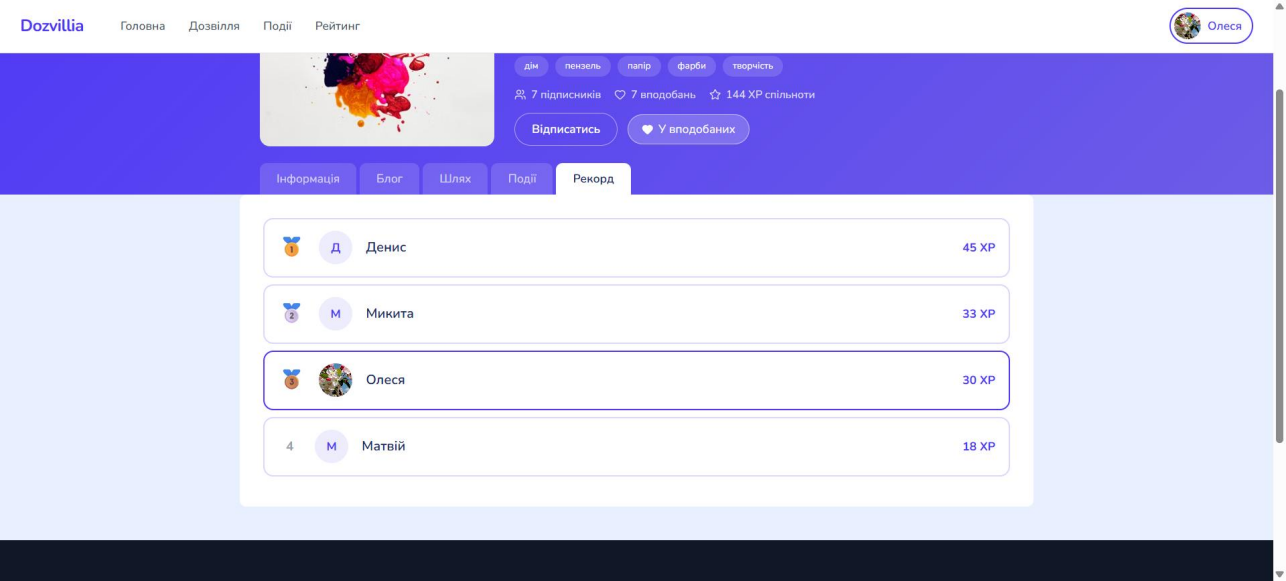


Рис. 3.14. Сторінка захоплення, вкладка Рекорд

Після того як користувач підписується на захоплення, у навігаційному розділі Головна з'являється стрічка публікацій від учасників тих захоплень на які він підписаний. Як наведено на рис. 3.15, кожна публікація містить зображення, ім'я автора, назву захоплення, дату, текст та можливість залишити лайк або коментар. Якщо користувач ще не підписався на жодне захоплення стрічка залишається порожньою.

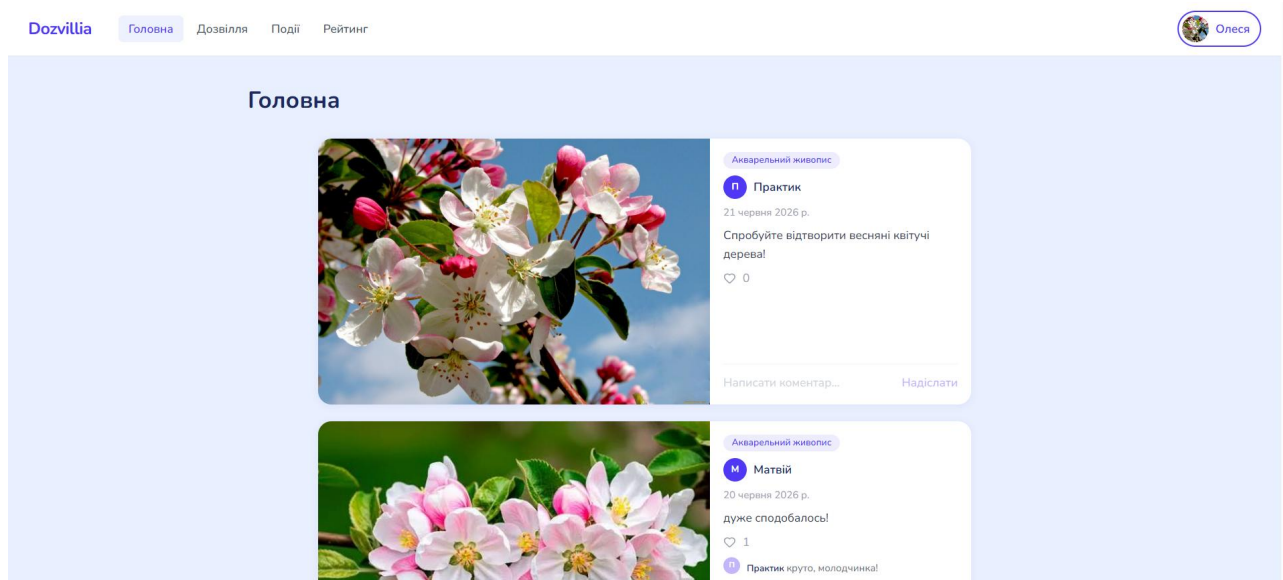


Рис. 3.15. Головна сторінка з стрічкою публікацій

При переході до розділу Події відкривається сторінка наведена на рис. 3.16 яка містить календар подій зліва та список найближчих заходів справа. Кожна подія відображає дату, час, формат онлайн або офлайн, назву захоплення, опис, кількість учасників та кнопку реєстрації.

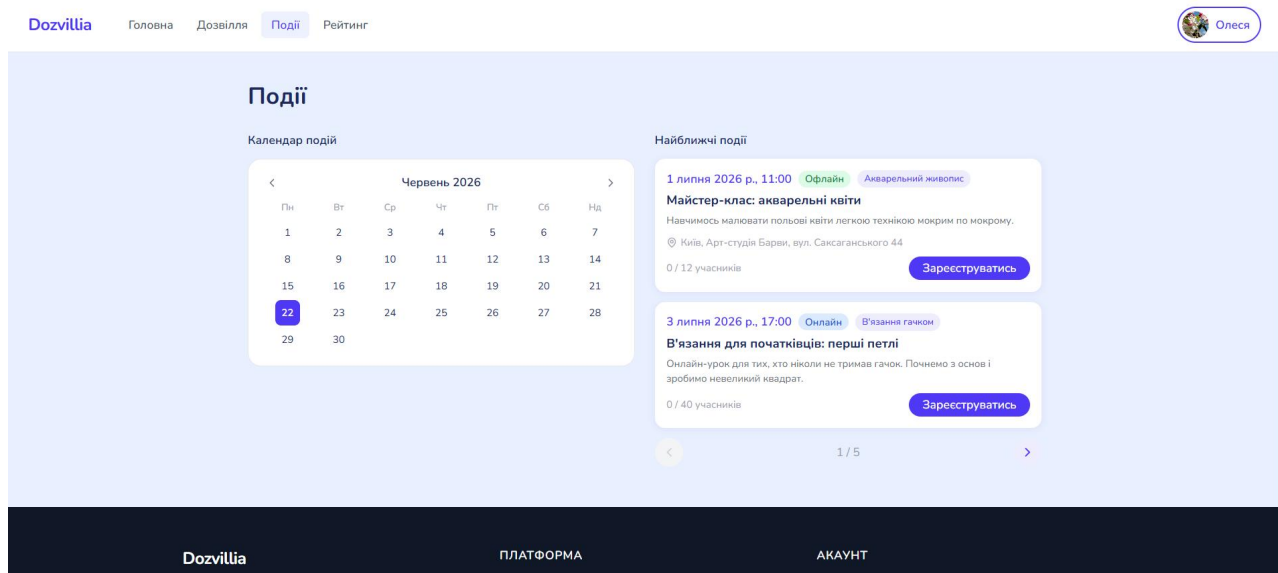


Рис. 3.16. Сторінка подій

При натисканні кнопки реєстрації відкривається модальне вікно наведене на рис. 3.17 з формою що містить поля для введення прізвища та імені і номера телефону для контакту з організатором.

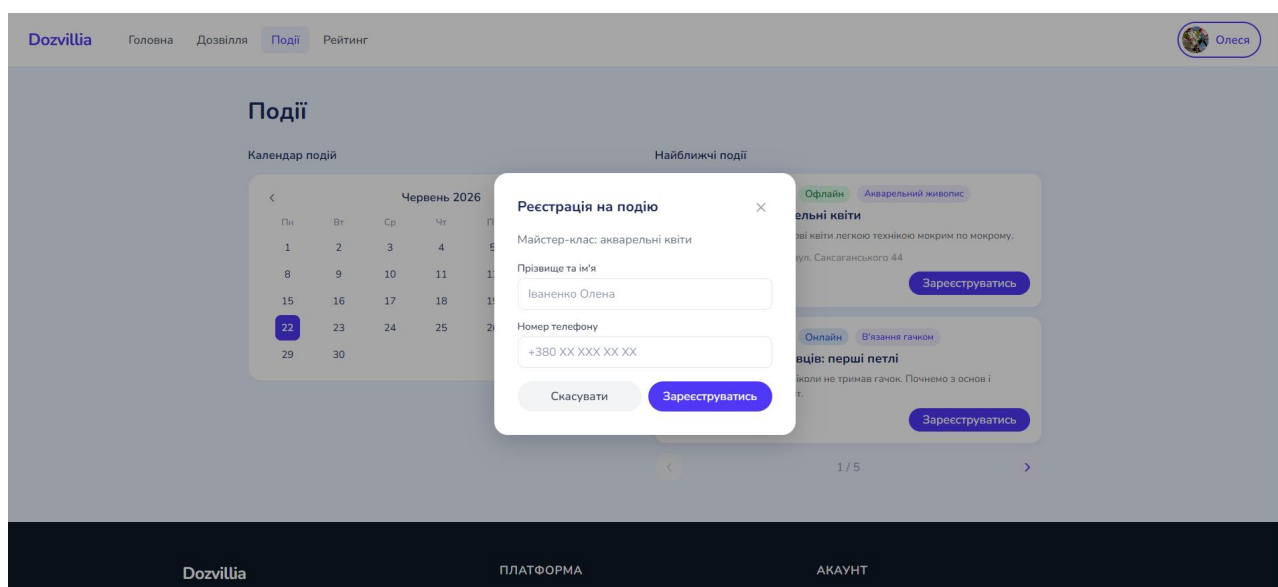


Рис. 3.17. Модальне вікно реєстрації на подію

Розділ Рейтинг наведений на рис. 3.18 відображає загальну таблицю лідерів платформи з переліком учасників відсортованих за накопиченим досвідом. Поточний користувач виділяється у списку кольором що дозволяє одразу бачити свою позицію серед інших.

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижевська О.В.				78
Змн.	Арк.	№ докум.	Підпис	Дата		

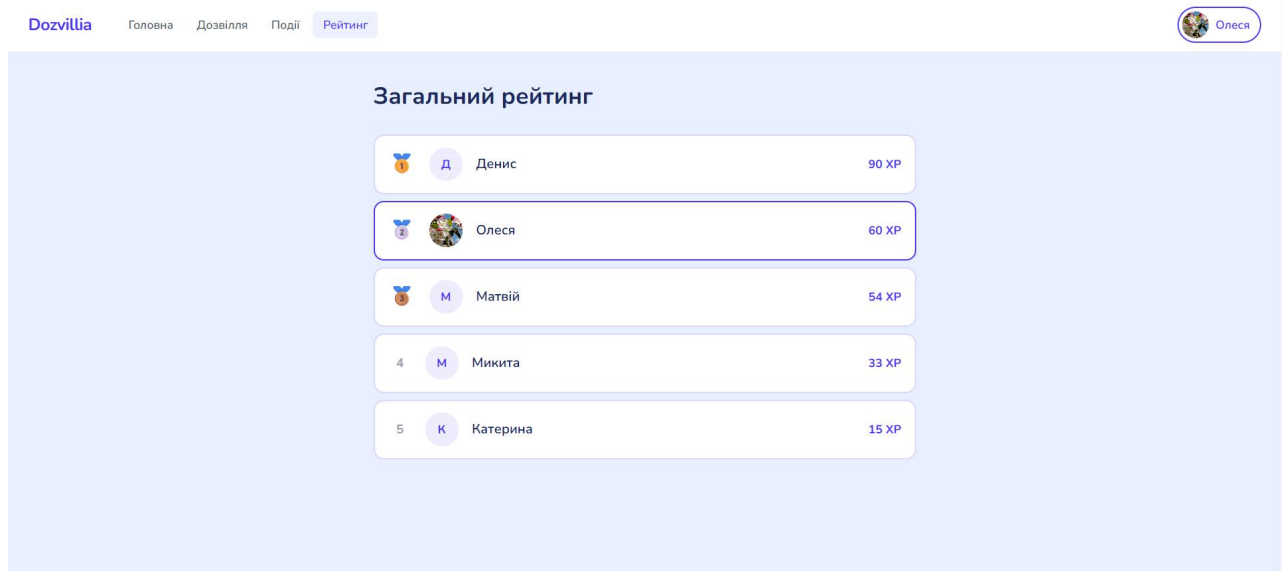


Рис. 3.18. Сторінка загального рейтингу

При натисканні на своє ім'я у правому верхньому куті користувач переходить до сторінки профілю наведеної на рис. 3.19. Вона містить три блоки: список підписок із прогрес-баром для кожного захоплення, колекцію отриманих трофеїв та календар активності що відображає дні коли користувач виконував вузли.

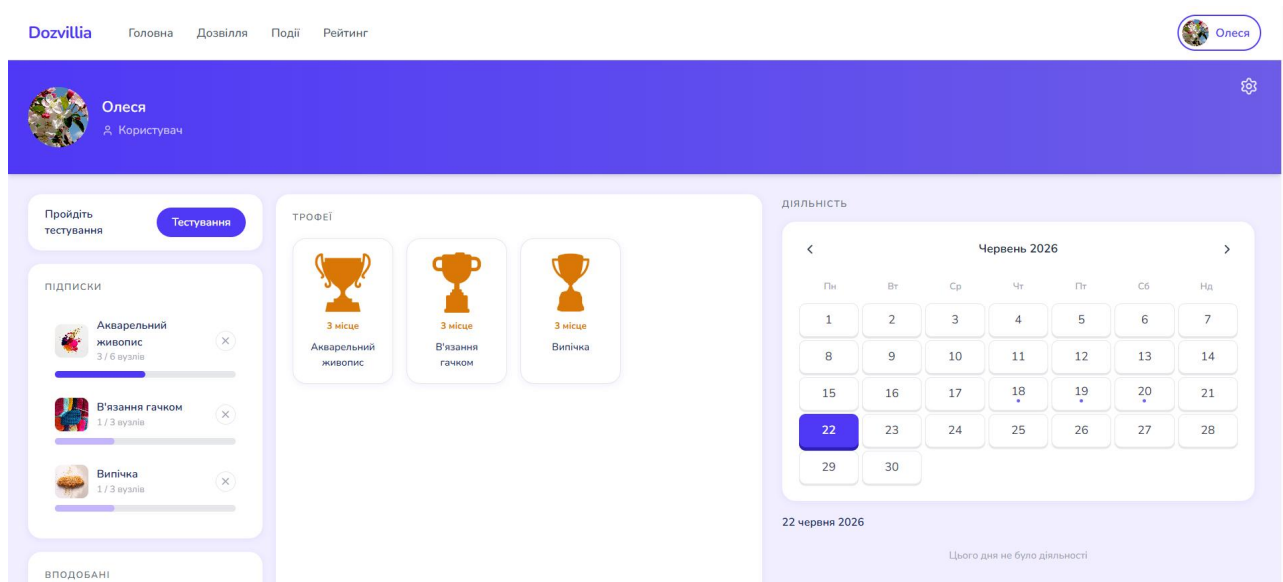


Рис. 3.19. Сторінка профілю користувача

При натисканні на іконку налаштувань у правому верхньому куті профілю відкривається модальне вікно наведене на рис. 3.20. Воно містить

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоторя О.В.				79
Змн.	Арк.	№ докум.	Підпис	Дата		

кілька вкладок: профіль та акаунт для редагування персональних даних і аватара, безпека для зміни пароля, заявка на практика для подання запиту на розширені права, відгук для надсилання повідомлення адміністратору та вихід з системи.

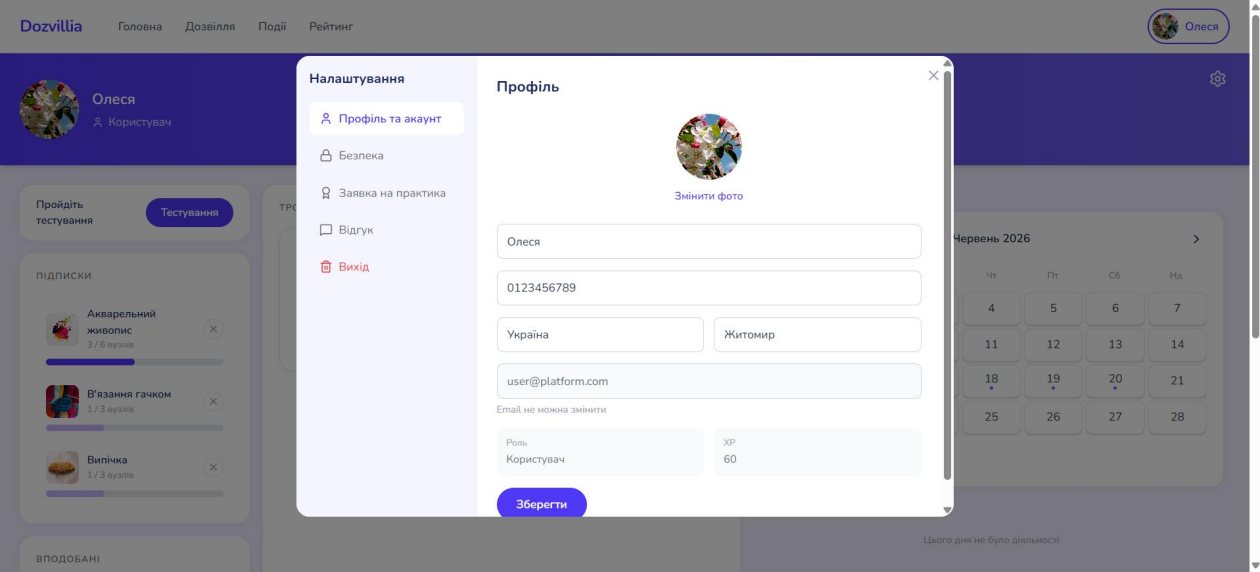


Рис. 3.20. Модальне вікно налаштувань профілю

Після отримання статусу практика користувач отримує доступ до додаткової вкладки Моє дозвілля у своєму профілі наведеної на рис. 3.21. Вона відображає призначене захоплення з чотирма підвкладками: Загальне, Шлях, Блог та Події. Вкладка Загальне містить повний опис захоплення та кнопку редагування.

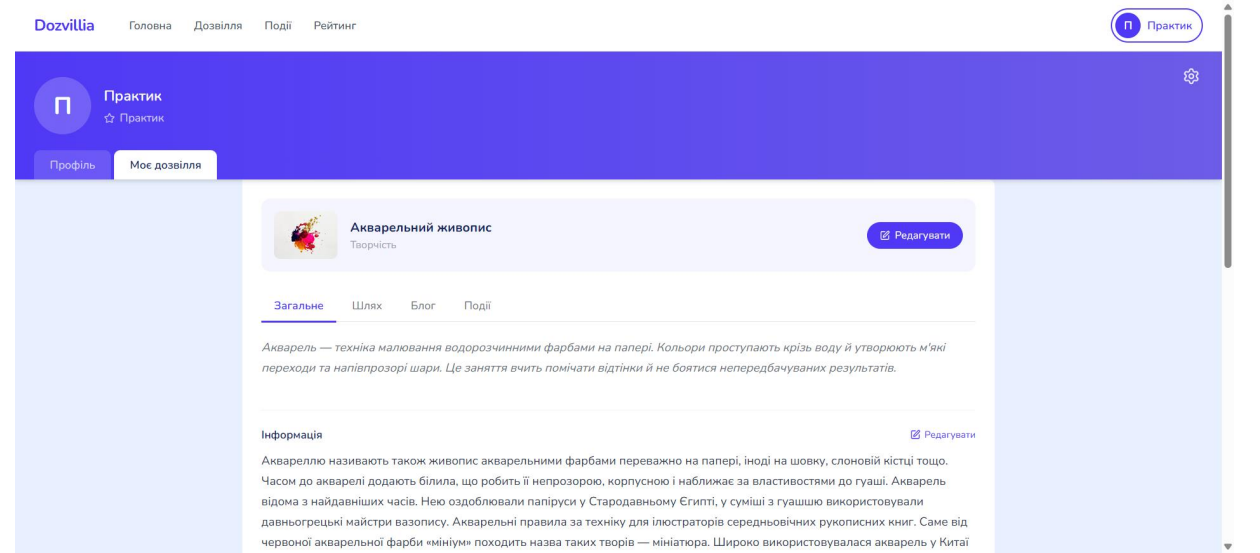


Рис. 3.21. Панель практика, вкладка Загальне

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоторя О.В.				80
Змн.	Арк.	№ докум.	Підпис	Дата		

При натисканні кнопки Редагувати відкривається модальне вікно наведене на рис. 3.22, що дозволяє змінити назву дозвілля, категорію, короткий опис, повний опис, теги та зображення.

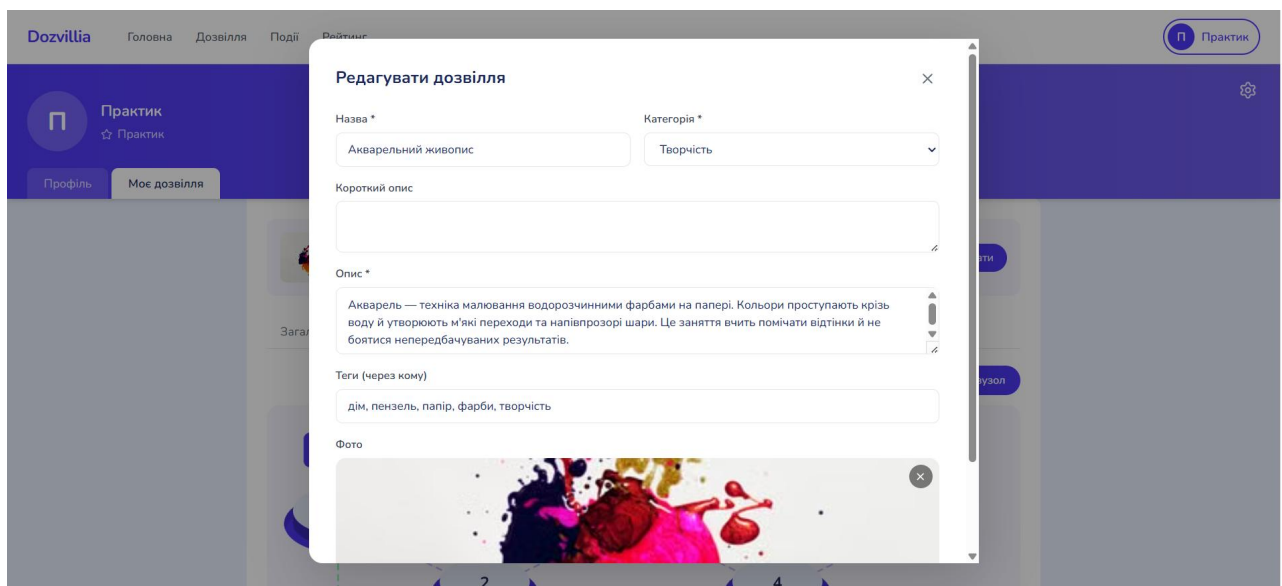


Рис. 3.22. Модальне вікно редагування дозвілля

При переході на вкладку Шлях практик бачить маршрут свого захоплення з кнопкою додавання нового вузла. При натисканні на неї відкривається модальне вікно наведене на рис. 3.23 де заповнюються назва вузла, порядковий номер, опис та ресурси у вигляді назви і посилання.

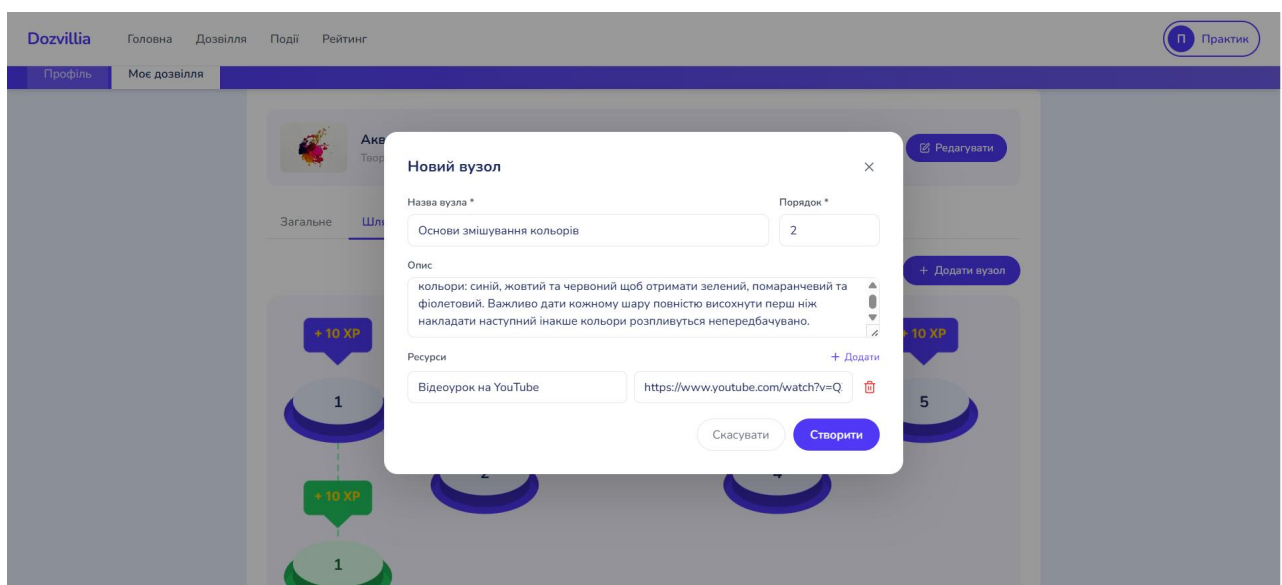


Рис. 3.23. Модальне вікно створення нового вузла

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоторя О.В.				81
Змн.	Арк.	№ докум.	Підпис	Дата		

Після збереження вузол з'являється у маршруті захоплення. Як наведено на рис. 3.24, вузли створені практиком відображаються окремим зеленим рядком під основним ланцюжком адміністратора з підписом Практик під кожним вузлом. Над виконаними вузлами відображається кількість нарахованого досвіду. При натисканні на вузол нижче схеми розгортається його зміст з описом та ресурсами, а також кнопки редагування і видалення доступні лише для вузлів створених практиком.



Рис. 3.24. Вкладка Шлях з вузлами практика

При переході на вкладку Блог практик може створювати нові публікації для стрічки захоплення. Як наведено на рис. 3.25, форма публікації містить поле для завантаження зображення та текстове поле з лічильником символів. Після натискання кнопки Опублікувати публікація одразу з'являється у стрічці захоплення оскільки практик має право публікувати без проходження модерації.

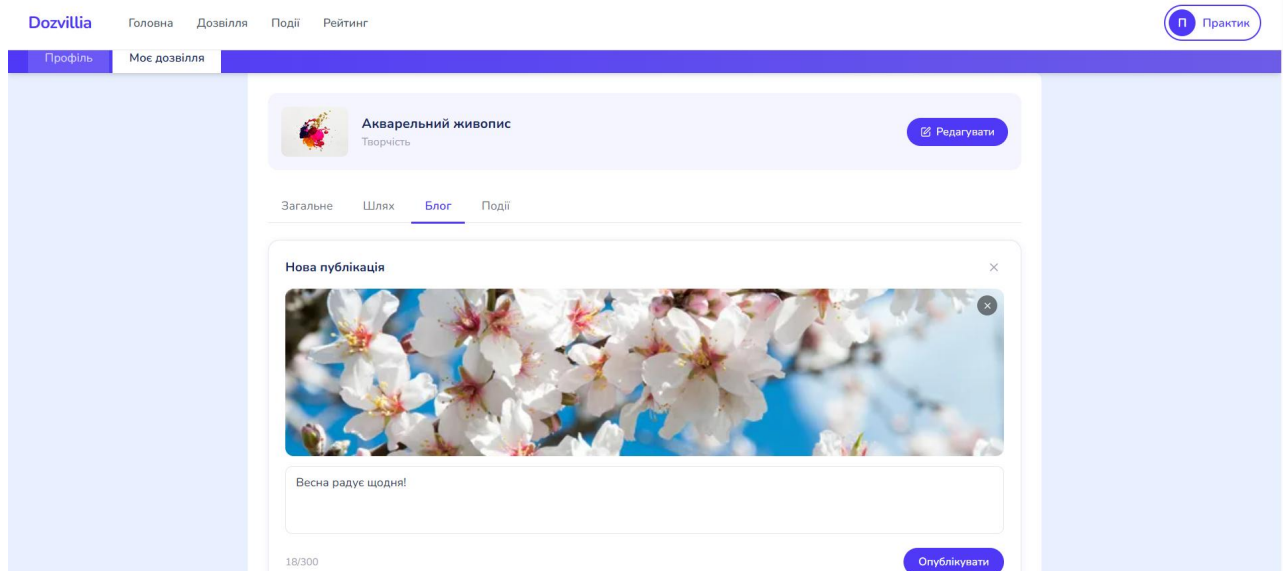


Рис. 3.25. Форма створення публікації у панелі практика

При переході на вкладку Події практик може створювати нові заходи для свого захоплення. При натисканні кнопки Додати подію відкривається модальне вікно наведене на рис. 3.26 де заповнюється назва події, опис, дата і час, формат онлайн або офлайн, посилання для підключення та максимальна кількість учасників. Система перевіряє що дата проведення відстоїть від поточного моменту не менше ніж на 24 години і у разі порушення цієї умови відображає повідомлення про помилку.

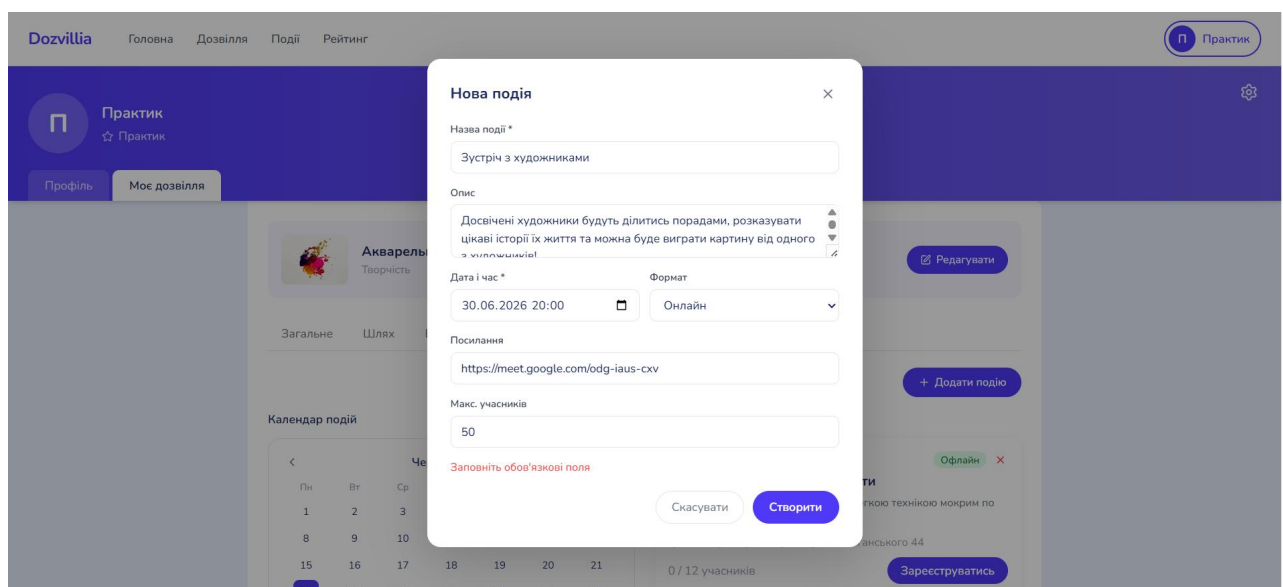


Рис. 3.26. Модальне вікно створення нової події

Адміністратор має доступ до розширеного функціоналу платформи через додаткові вкладки у своєму профілі. Профіль адміністратора містить вкладки Користувачі, Заявки, Відгуки та Статистика на додаток до стандартного Профілю.

Вкладка Користувачі наведена на рис. 3.27 відображає повний список зареєстрованих учасників платформи з їх іменами, електронними адресами, поточними ролями та датами реєстрації. Для кожного користувача доступна зміна ролі через випадаючий список та кнопка видалення облікового запису. Також передбачений пошук за іменем або електронною адресою.

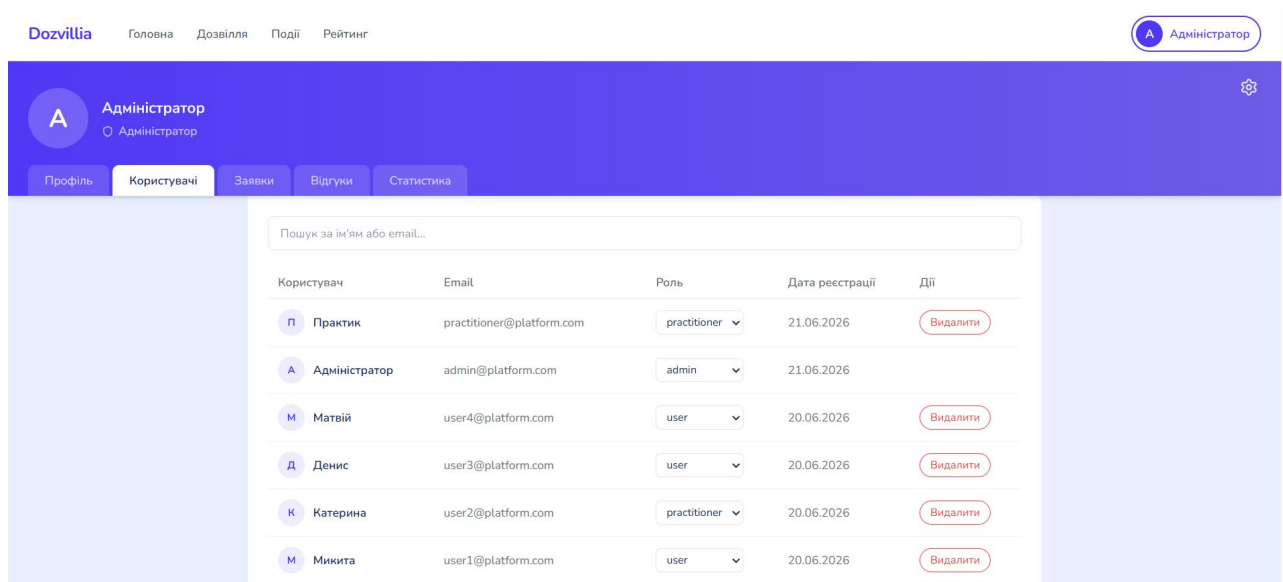


Рис. 3.27. Адміністративна панель, вкладка Користувачі

При переході на вкладку Заявки наведену на рис. 3.28 адміністратор бачить список усіх поданих заявок на отримання статусу практика з фільтрацією за статусом: всі, очікують, підтверджені та відхилені. Кожна заявка містить ім'я користувача, назву захоплення, текст обґрунтування та дату подання. Для заявок зі статусом очікування доступні кнопки підтвердження та відхилення.

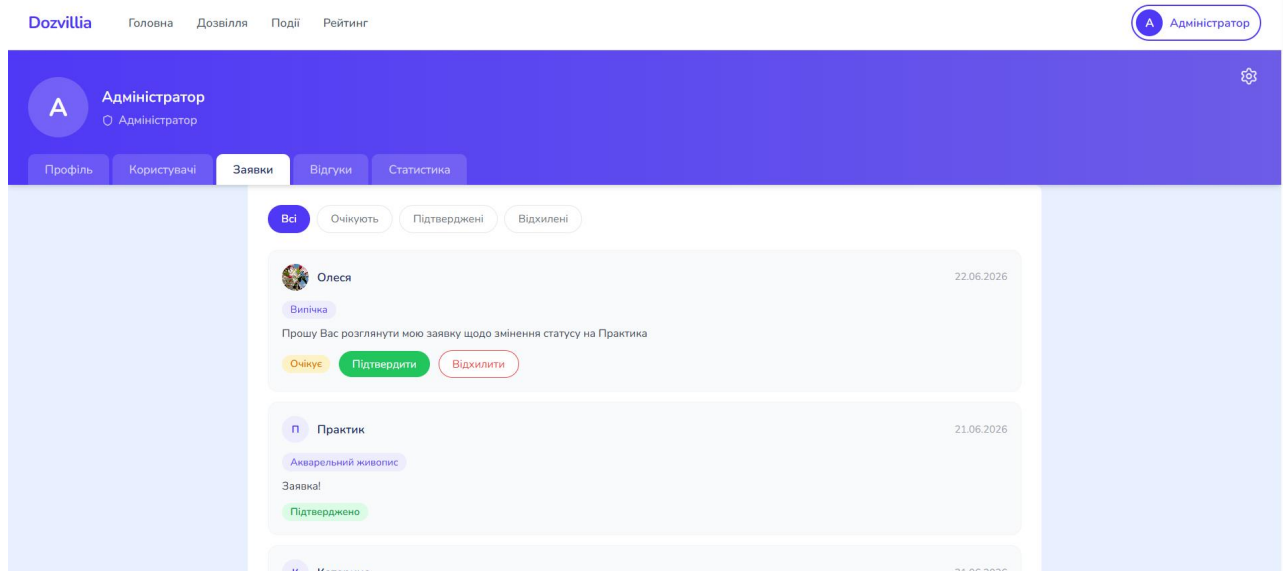


Рис. 3.28. Адміністративна панель, вкладка Заявки

Вкладка Відгуки наведена на рис. 3.29 відображає повідомлення надіслані користувачами через форму зворотного зв'язку. Кожен відгук містить ім'я автора, текст повідомлення та дату. Передбачений пошук за іменем або текстом відгуку.

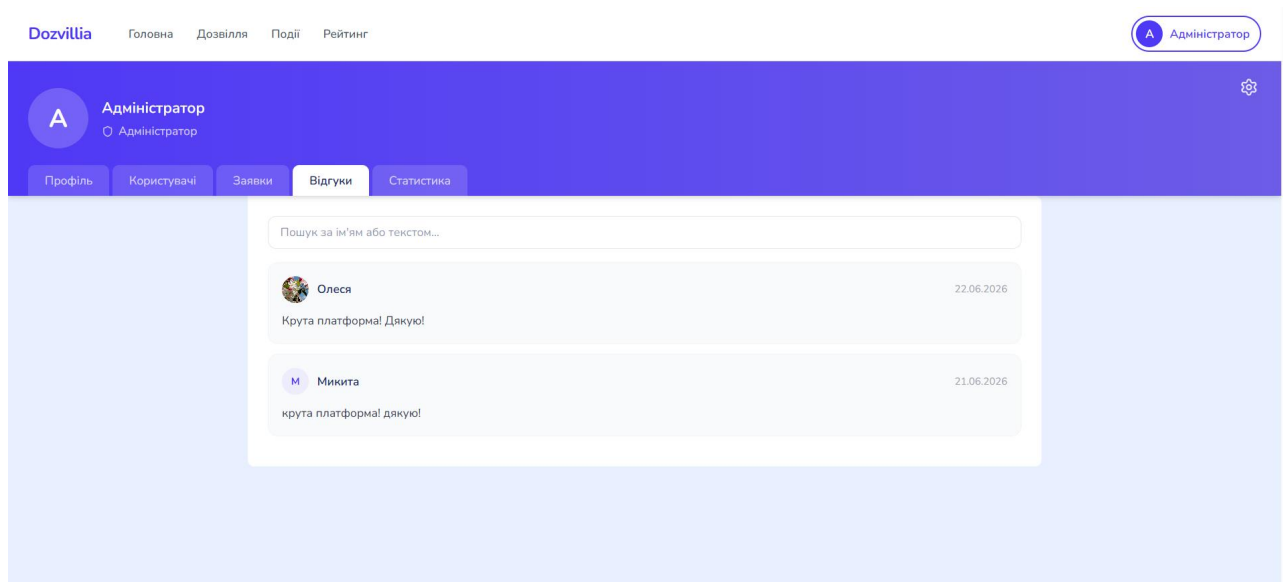


Рис. 3.29. Адміністративна панель, вкладка Відгуки

Вкладка Статистика наведена на рис. 3.30 надає зведені показники платформи у вигляді чотирьох карток із загальною кількістю користувачів,

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоштря О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		85

видів дозвілля, публікацій та подій. Нижче відображаються розбивки за роллю користувачів, статусом публікацій та локацією учасників.

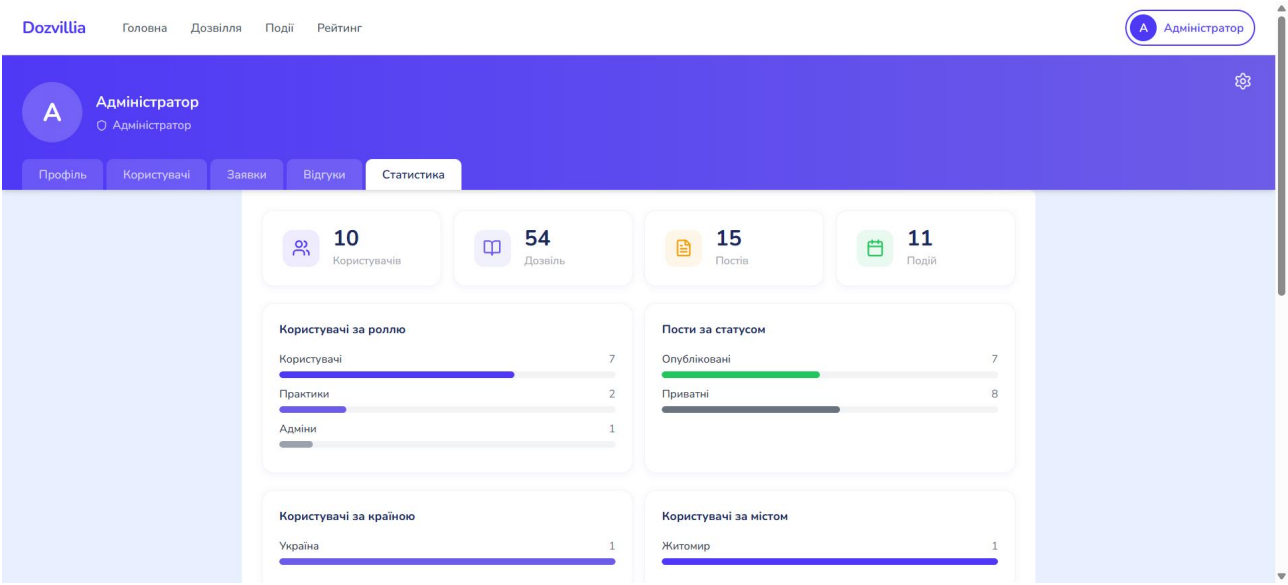


Рис. 3.30. Адміністративна панель, вкладка Статистика

Адміністратор також має розширені можливості при роботі зі стрічкою публікацій. Як наведено на рис. 3.31, на головній сторінці поряд із кожною публікацією відображаються кнопки приховування та видалення що дозволяють модерувати контент безпосередньо зі стрічки без переходу до окремого розділу.

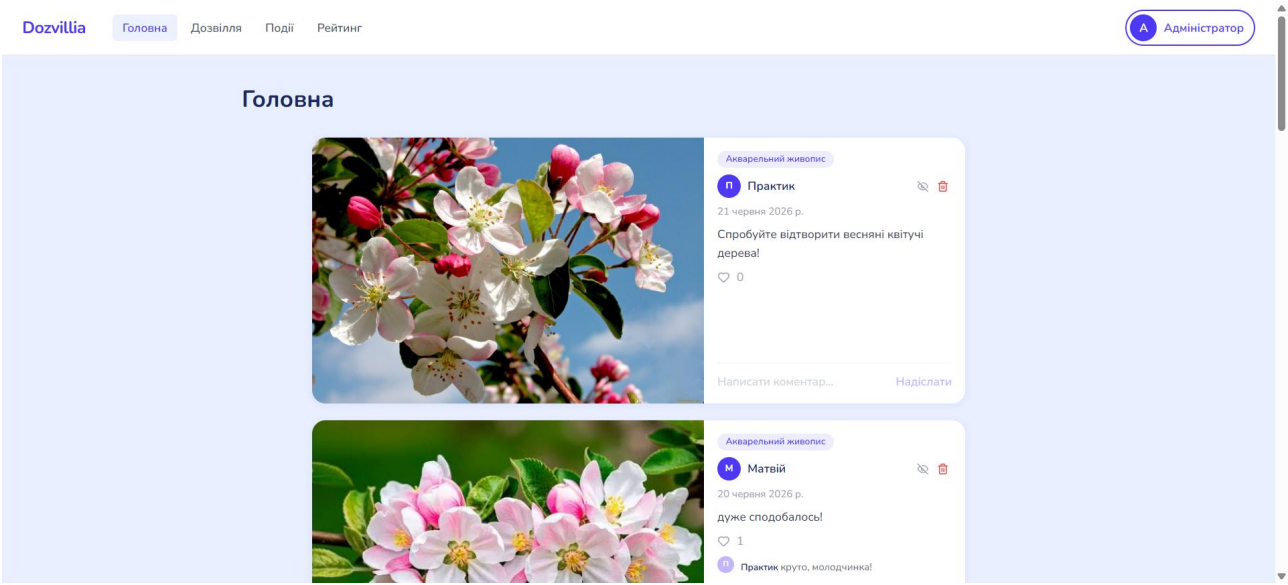


Рис. 3.31. Головна сторінка з інструментами модерації адміністратора

При переході до каталогу адміністратор бачить кнопку Додати дозвілля у верхньому правому куті. При натисканні на неї відкривається модальне вікно наведене на рис. 3.32 де заповнюються назва, категорія, короткий опис, повний опис, теги та завантажуються зображення.

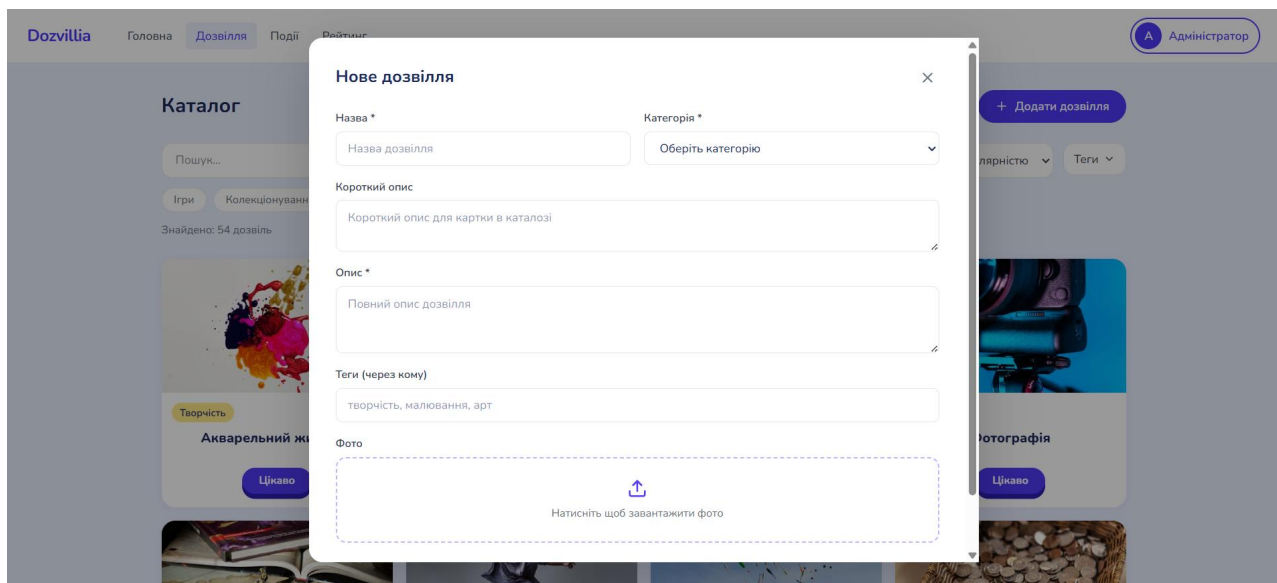


Рис. 3.32. Модальне вікно створення нового дозвілля

На сторінці конкретного дозвілля адміністратор бачить додаткові кнопки Редагувати та Видалити поряд із назвою як наведено на рис. 3.33.

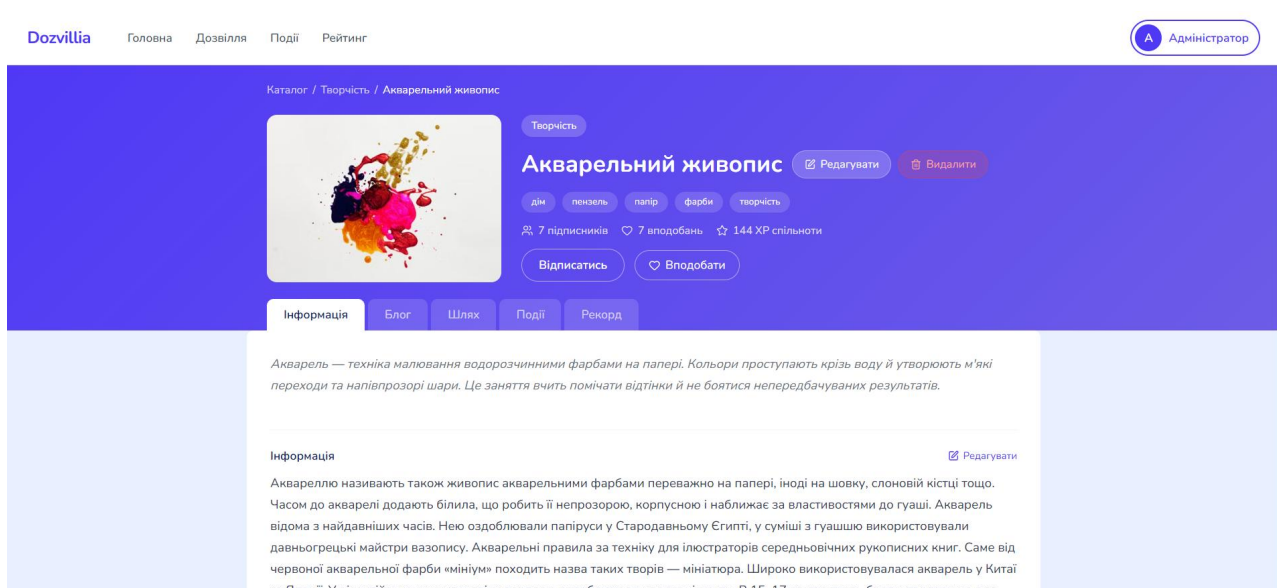


Рис. 3.33. Сторінка дозвілля з кнопками керування для адміністратора

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоторя О.В.				87
Змн.	Арк.	№ докум.	Підпис	Дата		

Вкладка Блог для адміністратора наведена на рис. 3.34 відображає всі публікації включаючи ті що очікують на модерацію з кнопками приховування та видалення для кожної з них.

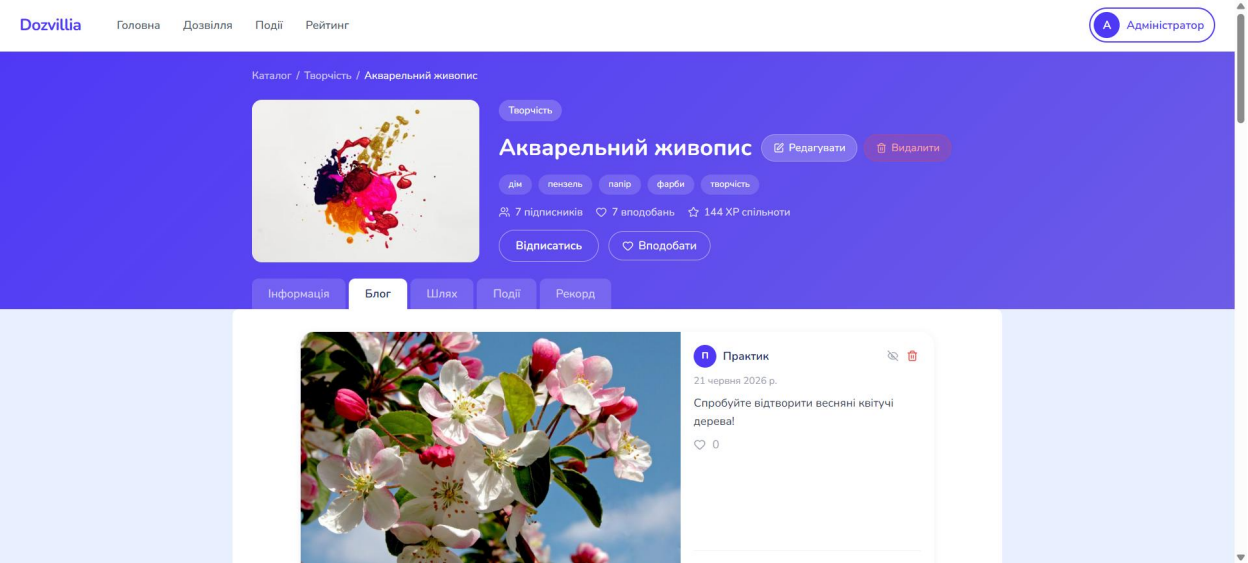


Рис. 3.34. Вкладка Блог з інструментами модерації адміністратора

На вкладці Шлях наведеній на рис. 3.35 адміністратор має доступ до всіх вузлів маршруту незалежно від їх джерела та може додавати нові через кнопку Додати вузол.

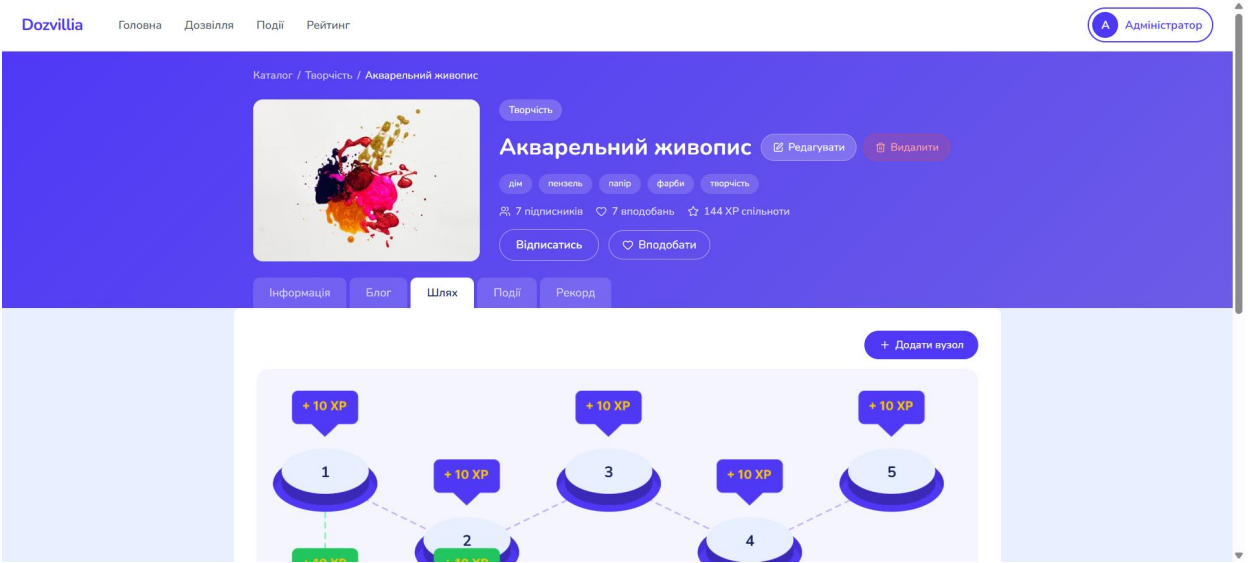


Рис. 3.35. Вкладка Шлях для адміністратора

При натисканні відкривається модальне вікно наведене на рис. 3.36 де заповнюється назва вузла, порядковий номер, опис та ресурси.

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоторя О.В.				88
Змн.	Арк.	№ докум.	Підпис	Дата		

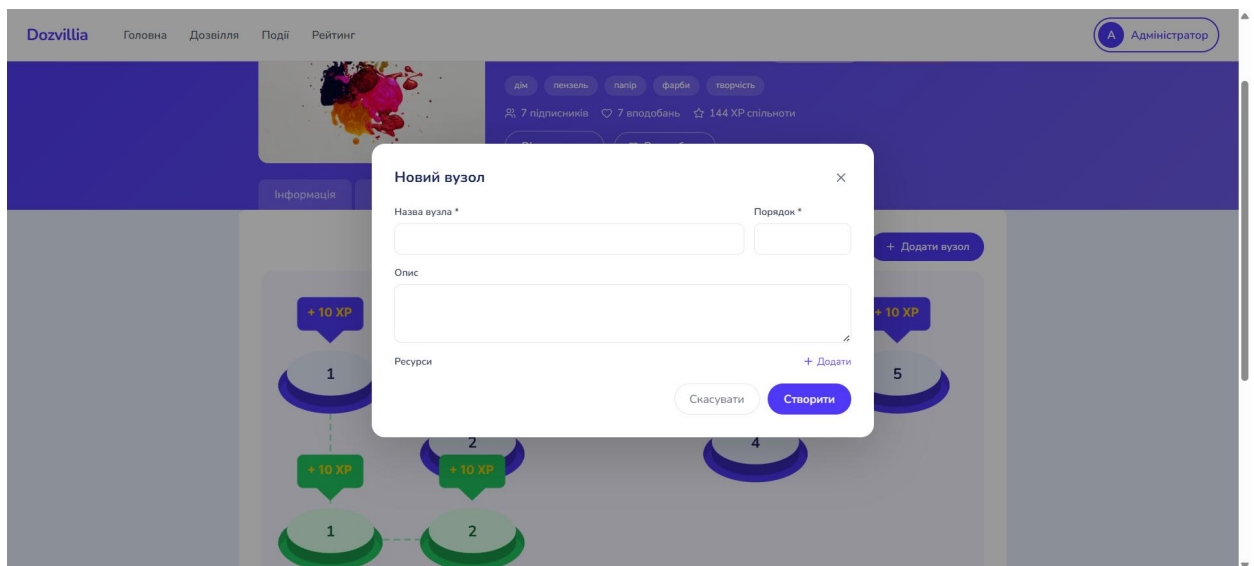


Рис. 3.36. Модальне вікно створення вузла адміністратором

На вкладці Події адміністратор може створювати нові заходи через модальне вікно наведене на рис. 3.37 яке містить ті самі поля що і у практика та додатково відображає кнопки видалення поряд із наявними подіями.

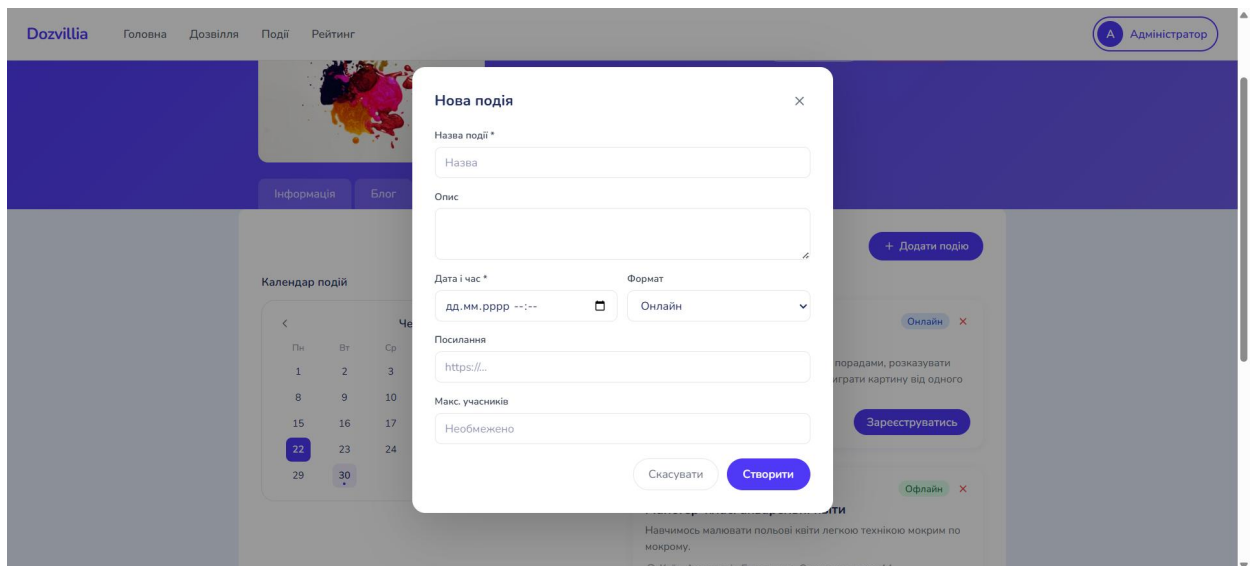


Рис. 3.37. Модальне вікно створення події адміністратором

3.3 Тестування програмного комплексу

Тестування є обов'язковим етапом розробки програмного комплексу що дозволяє перевірити коректність реалізованого функціоналу та впевнитися у відповідності системи визначеним вимогам [31]. У межах проєкту

реалізовано багаторівневу стратегію тестування що охоплює серверну та клієнтську частини платформи, а також навантажувальне тестування.

Для запуску тестів на серверній частині використовується Vitest у поєднанні з бібліотекою Supertest що дозволяє виконувати інтеграційні тести HTTP-маршрутів без запуску реального сервера. На клієнтській частині використовується Vitest з Testing Library та MSW [31] для перехоплення HTTP-запитів. Навантажувальне тестування реалізовано через інструмент k6 від Grafana [32]. Загальну структуру тестової інфраструктури наведено в табл. 3.2.

Таблиця 3.2

Стек тестової інфраструктури

Шар	Інструменти
Backend runner	Vitest + Supertest
Frontend runner	Vitest + Testing Library + MSW
Load tests	k6 (Grafana)
CI	GitLab CI
Coverage	v8 provider

Серверна частина покрита чотирнадцятьма тест-файлами що охоплюють усі основні домени системи. Стратегія мокінгу побудована на повному заміщенні Prisma через механізм vi.mock() що дозволяє запускати тести без підключення до реальної бази даних. Перелік доменів та відповідного функціоналу що тестується наведено в табл. 3.3.

Таблиця 3.3

Покриття серверної частини тестами

Домен	Що тестується
auth	Реєстрація, вхід, скидання пароля, отримання профілю, JWT
hobbies	CRUD операції, пагінація, пошук, каскадне видалення, перевірка ролей
posts	Створення, модерація, нарахування досвіду при публікації, лайки, коментарі

progress	Завершення вузла з нарахуванням досвіду за різними критеріями
roadmap	Авторство адміністратора та практика, каскадне видалення
events	Валідація мінімального терміну 24 години, CRUD операції
subscriptions	Підписка, відписка
quiz	Обробка 15 питань, алгоритм підбору за тегами
admin	Схвалення заявок із підвищенням ролі, статистика платформи
middleware	Перевірка JWT токена, рольова модель доступу

Тестування серверних маршрутів виконується через Supertest що дозволяє надсилати реальні HTTP-запити до Express-застосунку та перевіряти відповіді. Для кожного домену тестуються як успішні сценарії так і обробка помилкових ситуацій: відсутність токена, недостатні права доступу, некоректні дані та спроби повторного виконання вже виконаних операцій. Скріншот результатів виконання серверних тестів наведено на рис. 3.38.

```

RUN v3.2.6 D:/ztu/4 курс/Диплом/hobby-platform/backend

✓ src/__tests__/roadmap.service.test.js (6 tests) 48ms
✓ src/__tests__/hobbies.service.test.js (8 tests) 53ms
✓ src/__tests__/posts.service.test.js (14 tests) 58ms
✓ src/__tests__/users.service.test.js (8 tests) 46ms
✓ src/__tests__/admin.service.test.js (4 tests) 45ms
✓ src/__tests__/progress.service.test.js (7 tests) 48ms
✓ src/__tests__/quiz.service.test.js (5 tests) 46ms
✓ src/__tests__/events.service.test.js (7 tests) 33ms
✓ src/__tests__/subscriptions.service.test.js (4 tests) 36ms
✓ src/__tests__/favorites.service.test.js (4 tests) 36ms
✓ src/__tests__/auth.service.test.js (8 tests) 39ms
✓ src/__tests__/auth.routes.test.js (6 tests) 316ms
✓ src/__tests__/hobbies.routes.test.js (7 tests) 355ms
✓ src/__tests__/middleware.test.js (5 tests) 296ms

Test Files 14 passed (14)
Tests 93 passed (93)
Start at 18:14:54
Duration 7.54s (transform 2.27s, setup 0ms, collect 17.94s, tests 1.45s, environment 14ms, prepare 11.39s)

```

Рис. 3.38. Результати виконання серверних тестів

Клієнтська частина покрита чотирма тест-файлами що перевіряють найбільш критичні з точки зору безпеки та функціональності компоненти.

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоторя О.В.				91
Змн.	Арк.	№ докум.	Підпис	Дата		

Тест `axios.test.js` перевіряє коректність додавання Bearer токена до запитів та очищення стану при отриманні відповіді з кодом 401. Тест `PrivateRoute.test.jsx` перевіряє що неавторизований користувач перенаправляється на сторінку входу. Тест `LoginPage.test.jsx` перевіряє роботу форми входу та відображення помилки при невдалій спробі авторизації. Тест `utils/roadmap.test.js` перевіряє функцію `stateOf` що визначає стан вузла маршруту як виконаний, поточний або заблокований та перевіряє розблокування всіх вузлів для адміністратора. Скріншот результатів виконання клієнтських тестів наведено на рис. 3.39.

```

RUN v4.1.9 D:/ztu/4 курс/Диплом/hobby-platform/frontend
✓ src/utils/roadmap.test.js (5 tests) 45ms
✓ src/components/PrivateRoute.test.jsx (2 tests) 205ms
Not implemented: navigation to another Document
✓ src/api/axios.test.js (3 tests) 171ms
Not implemented: navigation to another Document
✓ src/pages/LoginPage.test.jsx (3 tests) 1313ms
  ✓ renders login button 332ms
  ✓ shows error message on failed login 841ms

Test Files  4 passed (4)
Tests      13 passed (13)
Start at   18:18:39
Duration   13.66s (transform 1.33s, setup 8.99s, import 4.38s, tests 1.73s, environment 27.69s)
  
```

Рис. 3.39. Результати виконання клієнтських тестів

Навантажувальне тестування реалізовано через три сценарії k6 [32]. Перший сценарій `catalog` перевіряє публічний перегляд каталогу при навантаженні від 10 до 50 віртуальних користувачів протягом двох хвилин із порогом часу відповіді `p95` менше 5000 мс та `p99` менше 10000 мс (рис. 3.40).

```

TOTAL RESULTS

checks_total.....: 7710    63.951162/s
checks_succeeded...: 100.00% 7710 out of 7710
checks_failed.....: 0.00%   0 out of 7710

✓ hobbies list 200
✓ hobby detail 200
✓ categories 200

HTTP
http_req_duration.....: avg=17.92ms min=0s med=9.71ms max=240.57ms p(90)=39.74ms p(95)=65.93ms
{ expected_response:true }...: avg=17.92ms min=0s med=9.71ms max=240.57ms p(90)=39.74ms p(95)=65.93ms
http_req_failed.....: 0.00%   0 out of 7710
http_reqs.....: 7710    63.951162/s

EXECUTION
iteration_duration.....: avg=1.05s   min=1s med=1.03s max=1.37s p(90)=1.13s p(95)=1.2s
iterations.....: 2570    21.317054/s
vus.....: 1 min=1 max=50
vus_max.....: 50 min=50 max=50

NETWORK
data_received.....: 37 MB 309 kB/s
data_sent.....: 694 kB 5.8 kB/s

```

Рис. 3.40. Результати навантажувального тестування load-tests/catalog.k6.js

Другий сценарій auth тестує вхід до системи та отримання профілю з порогом p95 менше 5000 мс (рис. 3.41).

```

TOTAL RESULTS

checks_total.....: 1077    8.858109/s
checks_succeeded...: 100.00% 1077 out of 1077
checks_failed.....: 0.00%   0 out of 1077

✓ login 200
✓ has token
✓ GET /me 200

HTTP
http_req_duration.....: avg=3.56s min=515.4µs med=2.54s max=12.36s p(90)=8.76s p(95)=9.94s
{ expected_response:true }...: avg=3.56s min=515.4µs med=2.54s max=12.36s p(90)=8.76s p(95)=9.94s
http_req_failed.....: 0.00%   0 out of 719
http_reqs.....: 719    5.913631/s

EXECUTION
iteration_duration.....: avg=8.12s min=1.22s med=7.59s max=17.5s p(90)=14.74s p(95)=15.53s
iterations.....: 359    2.952703/s
vus.....: 3 min=0 max=50
vus_max.....: 50 min=50 max=50

NETWORK
data_received.....: 566 kB 4.7 kB/s
data_sent.....: 199 kB 1.6 kB/s

```

Рис. 3.41. Результати навантажувального тестування load-tests/auth.k6.js

Третій сценарій progress перевіряє операції підписки та відстеження прогресу при реальному навантаженні (рис. 3.42).

```

TOTAL RESULTS

checks_total.....: 4392    36.162426/s
checks_succeeded...: 100.00% 4392 out of 4392
checks_failed.....: 0.00%   0 out of 4392

✓ subscriptions 200
✓ progress 200

HTTP
http_req_duration.....: avg=117.4ms min=531µs med=56.95ms max=906.33ms p(90)=303.8ms p(95)=389.69ms
{ expected_response:true }...: avg=117.4ms min=531µs med=56.95ms max=906.33ms p(90)=303.8ms p(95)=389.69ms
http_req_failed.....: 0.00%   0 out of 4396
http_reqs.....: 4396    36.195361/s

EXECUTION
iteration_duration.....: avg=1.23s min=1.01s med=1.12s max=2.17s p(90)=1.63s p(95)=1.73s
iterations.....: 2196    18.081213/s
vus.....: 2 min=1 max=50
vus_max.....: 50 min=50 max=50

NETWORK
data_received.....: 5.2 MB 43 kB/s
data_sent.....: 1.7 MB 14 kB/s

```

Рис. 3.42. Результати навантажувального тестування load-tests/progress.k6.js

Усі тести інтегровано в CI pipeline що автоматично запускається при створенні merge request та при внесенні змін до гілок develop і main. На етапі тестування виконується встановлення залежностей, генерація Prisma клієнта та запуск тестів із збереженням результатів. Результати тестів відображаються безпосередньо в інтерфейсі GitLab що спрощує виявлення та усунення проблем.

Висновки до третього розділу

У третьому розділі описано порядок розгортання системи, інтерфейс вебплатформи та результати тестування програмного комплексу.

Розгортання реалізовано з використанням Docker де кожен компонент системи запускається в окремому контейнері. Для автоматизації доставки коду налаштовано CI/CD pipeline на базі GitLab CI/CD із чотирьох стадій що забезпечує автоматичний запуск тестів при кожній

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижевська О.В.				94
Змн.	Арк.	№ докум.	Підпис	Дата		

зміні коду. Виробниче середовище розгорнуто на платформі Railway з підтримкою двох середовищ staging та production.

Описано інтерфейс платформи для всіх чотирьох ролей. Неавторизований відвідувач має доступ до вітальної сторінки, тесту підбору дозвілля та каталогу. Авторизований користувач отримує повний функціонал для освоєння захоплень включаючи покроковий маршрут, стрічку публікацій та систему подій. Практик керує контентом свого дозвілля через окрему панель у профілі. Адміністратор має доступ до централізованої панелі керування з вкладками для управління користувачами, заявками, відгуками та статистикою.

Реалізовано багаторівневу стратегію тестування що охоплює чотирнадцять тест-файлів серверної частини з покриттям усіх основних доменів системи та чотири тест-файли клієнтської частини. Для перевірки продуктивності реалізовано три сценарії навантажувального тестування через k6 з визначеними порогами часу відповіді. Усі тести інтегровано в CI pipeline що запускається автоматично при кожній зміні коду.

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмогря О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		95

ВИСНОВКИ

У процесі виконання кваліфікаційної роботи було розроблено вебплатформу для пошуку та освоєння дозвілля що забезпечує комплексний підхід до персоналізованого підбору занять та організації їх поетапного освоєння в межах єдиного середовища. Реалізований програмний продукт усуває основні обмеження наявних рішень та надає користувачу повний цикл підтримки від першого вибору заняття до його системного освоєння з відстеженням власного прогресу та взаємодією зі спільнотою.

У першому розділі проведено аналіз предметної галузі та визначено основні проблеми з якими стикається користувач при виборі нового заняття. Встановлено що велика кількість розрізнених ресурсів не спрощує цей процес а навпаки ускладнює прийняття рішення через інформаційне перевантаження. Відсутність структурованого маршруту після вибору заняття, неможливість відстежувати власний прогрес та брак якісного середовища для взаємодії з іншими учасниками є головними причинами того що більшість людей залишають нове захоплення вже на початковому етапі. Дослідження наявних аналогів підтвердило що жодна з розглянутих платформ не забезпечує повного циклу підтримки від вибору дозвілля до його освоєння. На основі порівняльного аналізу обрано модульну монолітну архітектуру та визначено технологічний стек що включає React і Vite на клієнтській частині та Node.js, Express, PostgreSQL і Prisma на серверній.

У другому розділі спроектовано повну структуру системи. Реляційна база даних складається з 19 таблиць організованих у чотири логічні групи. Побудовано рольову модель доступу для чотирьох ролей та деталізовано варіанти використання системи у вигляді шести таблиць прецедентів і діаграми. Розроблено алгоритми шести ключових бізнес-процесів та побудовано відповідні діаграми активності. Реалізовано три ключові функціональні модулі платформи. Гібридний алгоритм рекомендацій поєднує підбір на основі зважених тегів із колаборативною фільтрацією де оцінка розраховується з коефіцієнтами 0.6 для відповідності тегів та 0.4 для

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоштря О.В.				96
Змн.	Арк.	№ докум.	Підпис	Дата		

показника популярності і автоматично переходить між режимами залежно від обсягу накопичених даних. Модуль прогресу забезпечує нарахування досвіду та формує теплову карту активності користувача. Модуль подій реалізує чотири режими отримання даних залежно від ролі та дворівневу валідацію при створенні заходу.

У третьому розділі налаштовано повну інфраструктуру розгортання та тестування системи. Застосунок контейнеризовано за допомогою Docker та налаштовано CI/CD pipeline на базі GitLab CI/CD що автоматично запускає тести при кожній зміні коду та розгортає систему на платформі Railway за стратегією GitFlow. Описано інтерфейс платформи для всіх чотирьох ролей користувачів з наведенням скріншотів основних сторінок та функціональних елементів. Реалізовано багаторівневу стратегію тестування що охоплює серверну та клієнтську частини платформи а також навантажувальне тестування через k6 з визначеними порогами часу відповіді.

Розроблена платформа реалізує повний цикл взаємодії користувача із захопленням в межах єдиного середовища та може використовуватися як готовий продукт для людей що прагнуть знайти нове заняття та займатися ним системно. Подальший розвиток платформи може включати розширення алгоритму рекомендацій з використанням моделей машинного навчання, додавання підтримки мобільних пристроїв та інтеграцію OAuth для авторизації через соціальні мережі.

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоторя О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		97

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Демчук О.В., Чижмотря О.В. Адаптивний алгоритм тест-підбору дозвілля. Тези XVI Міжнародної науково-технічної конференції «Інформаційно-комп'ютерні технології», м. Житомир, 02-03 квітня 2026 р. – Житомир: Житомирська політехніка, 2026. С. 79-80. URL: <https://conf.ztu.edu.ua/wp-content/uploads/2026/05/79.pdf>
2. Демчук О.В., Чижмотря О.В. Гейміфікація як інструмент утримання користувачів. Тези XVI Міжнародної науково-технічної конференції «Інформаційно-комп'ютерні технології», м. Житомир, 02-03 квітня 2026 р. – Житомир: Житомирська політехніка, 2026. С. 93-94. URL: <https://conf.ztu.edu.ua/wp-content/uploads/2026/05/93.pdf>
3. Deci E.L., Ryan R.M. Intrinsic Motivation and Self-Determination in Human Behavior. New York : Plenum Press, 1985. 371 p.
4. Bandura A. Self-Efficacy: The Exercise of Control. New York : W.H. Freeman, 1997. 604 p.
5. Rumbaugh J., Jacobson I., Booch G. The Unified Modeling Language Reference Manual. 2nd ed. Reading : Addison-Wesley, 2004. 721 p.
6. Larman C. Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design. 3rd ed. Upper Saddle River : Prentice Hall, 2004. 703 p.
7. Fowler M. Patterns of Enterprise Application Architecture. Boston : Addison-Wesley, 2002. 533 p.
8. Newman S. Building Microservices: Designing Fine-Grained Systems. 2nd ed. Sebastopol : O'Reilly Media, 2021. 615 p.
9. React Documentation. URL: <https://react.dev/> (дата звернення: 14.03.2026).
10. React Router Documentation. URL: <https://reactrouter.com/> (дата звернення: 14.03.2026).
11. Jotai Documentation. URL: <https://jotai.org/docs> (дата звернення: 17.03.2026).
12. Vite Documentation. URL: <https://vitejs.dev/> (дата звернення: 17.03.2026).

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмотря О.В.				98
Змн.	Арк.	№ докум.	Підпис	Дата		

13. Node.js Documentation. URL: <https://nodejs.org/en/docs/> (дата звернення: 20.03.2026).
14. Express.js Documentation. URL: <https://expressjs.com/> (дата звернення: 20.03.2026).
15. Fielding R.T. Architectural Styles and the Design of Network-based Software Architectures : PhD thesis / University of California. Irvine, 2000. 162 p.
16. Jones M., Bradley J., Sakimura N. JSON Web Token (JWT) : RFC 7519. IETF, 2015. URL: <https://tools.ietf.org/html/rfc7519> (дата звернення: 25.03.2026).
17. PostgreSQL Documentation. URL: <https://www.postgresql.org/docs/> (дата звернення: 01.04.2026).
18. Prisma Documentation. URL: <https://www.prisma.io/docs> (дата звернення: 01.04.2026).
19. Cloudinary Documentation. URL: <https://cloudinary.com/documentation> (дата звернення: 05.04.2026).
20. OWASP Password Storage Cheat Sheet. URL: https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html (дата звернення: 07.04.2026).
21. Axios Documentation. URL: <https://axios-http.com/docs/intro> (дата звернення: 10.04.2026).
22. ESLint Documentation. URL: <https://eslint.org/docs/latest/> (дата звернення: 10.04.2026).
23. Elmasri R., Navathe S. Fundamentals of Database Systems. 7th ed. Hoboken : Pearson, 2015. 1200 p.
24. Date C.J. An Introduction to Database Systems. 8th ed. Reading : Addison-Wesley, 2003. 1033 p.
25. Ricci F., Rokach L., Shapira B. Recommender Systems Handbook. 2nd ed. New York : Springer, 2015. 1003 p.
26. Su X., Khoshgoftaar T.M. A Survey of Collaborative Filtering Techniques // Advances in Artificial Intelligence. 2009. Article ID 421425. 19 p.

		Демчук О.В.			ІІЗ.КР.Б – 121 – 26 – ІІЗ	Арк.
		Чижомотря О.В.				99
Змн.	Арк.	№ докум.	Підпис	Дата		

27. Deterding S., Dixon D., Khaled R., Nacke L. From Game Design Elements to Gamefulness: Defining "Gamification" // Proceedings of the 15th International Academic MindTrek Conference. Tampere, 2011. P. 9–15.
28. Zichermann G., Cunningham C. Gamification by Design: Implementing Game Mechanics in Web and Mobile Apps. Sebastopol : O'Reilly Media, 2011. 208 p.
29. Docker Documentation. URL: <https://docs.docker.com/> (дата звернення: 15.04.2026).
30. Humble J., Farley D. Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation. Boston : Addison-Wesley, 2010. 512 p.
31. Myers G.J., Sandler C., Badgett T. The Art of Software Testing. 3rd ed. Hoboken : John Wiley & Sons, 2011. 240 p.
32. k6 Documentation. URL: <https://grafana.com/docs/k6/latest/> (дата звернення: 20.04.2026).

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоторя О.В.				100
Змн.	Арк.	№ докум.	Підпис	Дата		

ДОДАТКИ

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Чижмоторя О.В.				101
Змн.	Арк.	№ докум.	Підпис	Дата		

Лістинг CI/CD пайплайну

```

stages:
  - test
  - build
  - deploy
  - load-test

default:
  image: node:20-alpine

test-backend:
  stage: test
  script:
    - cd backend
    - npm ci
    - DATABASE_URL="postgresql://x:x@localhost/x" npx prisma generate
    - npm test -- --reporter=junit --outputFile=./test-results/results.xml
  coverage: '/Lines\s*:\s*(\d+\.\d+)%/'
  artifacts:
    when: always
    reports:
      junit: backend/test-results/results.xml
  rules:
    - if: $CI_PIPELINE_SOURCE == "merge_request_event"
    - if: $CI_COMMIT_BRANCH == "develop"
    - if: $CI_COMMIT_BRANCH == "main"

test-frontend:
  stage: test
  script:
    - cd frontend
    - npm ci
    - npm run lint
    - npm test -- --reporter=junit --outputFile=./test-results/results.xml
  artifacts:
    when: always
    reports:
      junit: frontend/test-results/results.xml
  rules:
    - if: $CI_PIPELINE_SOURCE == "merge_request_event"
    - if: $CI_COMMIT_BRANCH == "develop"
    - if: $CI_COMMIT_BRANCH == "main"

build-frontend:
  stage: build
  needs: [test-frontend]
  script:
    - cd frontend
    - npm ci
    - npm run build

```

		Демчук О.В.			ІПЗ.КР.Б – 121 – 26 – ІЗ	Арк.
		Чижмоторя О.В.				102
Змн.	Арк.	№ докум.	Підпис	Дата		

```

variables:
  VITE_API_URL: ""
artifacts:
  paths:
    - frontend/dist/
  expire_in: 1 hour
rules:
  - if: $CI_PIPELINE_SOURCE == "merge_request_event"
  - if: $CI_COMMIT_BRANCH == "develop"
  - if: $CI_COMMIT_BRANCH == "main"

```

```

deploy-staging:
  stage: deploy
  needs: [test-backend, test-frontend]
  image: alpine/git
  script:
    - git push https://oauth2:$GITHUB_TOKEN@github.com/DemchukOlesia/hobby-
      platform.git $CI_COMMIT_SHA:refs/heads/develop
  rules:
    - if: $CI_COMMIT_BRANCH == "develop"

```

```

deploy-production:
  stage: deploy
  needs: [test-backend, test-frontend]
  image: alpine/git
  script:
    - git push https://oauth2:$GITHUB_TOKEN@github.com/DemchukOlesia/hobby-
      platform.git $CI_COMMIT_SHA:refs/heads/main
  rules:
    - if: $CI_COMMIT_BRANCH == "main"

```

```

load-test:
  stage: load-test
  image: grafana/k6:latest
  script:
    - k6 run load-tests/catalog.k6.js
  rules:
    - if: $CI_COMMIT_BRANCH == "main"
      when: manual

```