Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

РОЗРОБКА ВЕБЗАСТОСУНКУ ДЛЯ КООРДИНАЦІЇ ВОЛОНТЕРСЬКОЇ ДІЯЛЬНОСТІ ТА ПІДТРИМКИ ВНУТРІШНЬО ПЕРЕМІЩЕНИХ ОСІБ
(тема)

Виконав:
здобувач 4 року навчання,
групи ІТІНФ-22-2
Деговцов В. О.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник проф. Шафроненко А. Ю.
(посада, прізвище, ініціали)

Допускається до захисту

Завідувач кафедри інформатики _____ Кобилін О. А.
(підпис) (прізвище, ініціали)

2026 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту

Кафедра Інформатики

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2026 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Деговцову Всеволоду Олеговичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка вебзастосунку для координації волонтерської діяльності та підтримки внутрішньо переміщених осіб

затверджена наказом університету від 26 травня 2026 року № 435Ст
2. Термін подання здобувачем роботи до екзаменаційної комісії 26 червня 2026 р.
3. Вихідні дані до роботи науково-методична та науково-технічна література, матеріали конференцій, дані інтернет-мережі, фреймворк ASP.NET Core Web API, бібліотеки jQuery.

4. Перелік питань, що потрібно опрацювати в роботі
1. Проаналізувати наявні аналоги програмних рішень на ринку.
2. Дослідити сучасні технології розробки вебзастосунків.
3. Обрати оптимальний технологічний стек для реалізації проєкту.
4. Розробити вебзастосунок для координації волонтерської діяльності та підтримки переміщених осіб.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) вступ, актуальність проблеми розробки застосунку, постановка задачі, програмна реалізація, висновки.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	13.04.2026	
2	Аналіз завдання, підбір літератури	14.04.26-16.04.26	
3	Аналіз літератури з проблеми, що вирішується	17.04.26-19.04.26	
4	Аналіз існуючих методів розпізнавання	20.04.26-22.04.26	
5	Формування кроків вирішення проблеми	23.04.26-05.05.26	
6	Програмна реалізація	26.04.26-20.05.26	
7	Аналіз отриманих результатів	21.05.26-04.06.26	
8	Оформлення пояснювальної записки	05.06.26-09.06.26	
9	Перевірка на нормоконтроль	10.06.26-19.06.26	
10	Перевірка на плагіат	11.06.26-30.06.26	
11	Рецензування	20.06.26-30.06.26	
12	Підготовка презентації та доповіді	20.06.26-30.06.26	
13	Занесення роботи в електронний архів	30.06.26-09.07.26	
14	Попередній захист кваліфікаційної роботи	30.06.26-09.07.26	

Дата видачі завдання 13 квітня 2026 р.

Здобувач _____
(підпис)

Керівник роботи _____ проф. Шафроненко А. Ю.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 77 с., 4 табл., 22 рис., 39 джерел.

ВЕБЗАСТОСУНОК, СУБД, ASP.NET, ANGULAR, C#, MICROSOFT AZURE.

Об’єктом роботи є створення вебзастосунку для координації волонтерської діяльності та підтримки внутрішньо переміщених осіб.

Метою роботи є розробка адаптивного вебзастосунка для координації волонтерської діяльності, що включає автоматизовану систему обробки звернень та централізовану базу даних потреб внутрішньо переміщених осіб.

Для розробки архітектури вебзастосунку було обрано сучасний технологічний стек. Серверна частина реалізована на базі фреймворку ASP.NET Core Web API з використанням мови C#. Клієнтська частина побудована на фреймворку Angular (TypeScript) із застосуванням мов розмітки HTML5, стилізації CSS3 та бібліотеки jQuery. Робота з даними забезпечується СУБД PostgreSQL, а розгортання та хостинг системи передбачені в хмарному середовищі Microsoft Azure.

Практичне значення роботи полягає у вдосконаленні механізмів отримання допомоги через онлайн-інструменти. Розроблений вебзастосунок є масштабованим і придатним для використання у міжнародних гуманітарних проєктах.

ANGULAR, ASP.NET, C#, DBMS, MICROSOFT AZURE, WEB APPLICATION.

The objective of the work is to create a web application to coordinate volunteer activities and support internally displaced persons.

The goal of the work is to develop an adaptive web application for coordinating volunteer activities, which includes an automated request processing system and a centralized database of the needs of internally displaced persons.

A modern technology stack was chosen to develop the web application's architecture. The server-side implementation is based on the ASP.NET Core Web API framework and written in C#. The client-side is built on the Angular (TypeScript) framework using HTML5 markup, CSS3, and the jQuery library. Data is managed by the PostgreSQL DBMS, and the system is deployed and hosted on Microsoft Azure.

The practical significance of the work lies in improving the mechanisms for receiving assistance through online tools. The developed application is scalable and suitable for use in international humanitarian projects.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ВПО – внутрішньо переміщені особи

ІТ – інформаційні технології

КСЗІ – комплексна система захист інформації

ООН – Організація Об’єднаних Націй

СУБД – системи управління базами даних

ЦНАП – центр надання адміністративних послуг

ЮНІСЕФ – Дитячий фонд Організації Об’єднаних Націй

MVC – Model-View-Controller (архітектурний шаблон «Модель-Вид-Контролер»)

UNHCR (УВКБ ООН) – Управління Верховного комісара ООН у справах біженців

VMS – Volunteer Management Software (управління волонтерами)

W3C – World Wide Web Consortium (Консорціум Всесвітньої павутини)

ВСТУП

У сучасному світі інформаційні технології відіграють ключову роль у вирішенні соціальних, гуманітарних та організаційних завдань. Цифровізація суспільства сприяє створенню нових інструментів взаємодії між громадянами, державними установами, благодійними організаціями та волонтерськими спільнотами. Особливої актуальності такі рішення набувають у кризових ситуаціях, коли від швидкості обробки інформації та координації дій залежить добробут і безпека великої кількості людей.

Повномасштабне вторгнення Російської Федерації на територію України спричинило масштабну гуманітарну кризу, наслідки якої торкнулися практично всіх регіонів країни. Мільйони громадян були змушені залишити свої домівки через бойові дії, руйнування житлової інфраструктури та загрозу життю. Значна частина населення набула статусу внутрішньо переміщених осіб і потребує постійної підтримки у вигляді житла, продуктів харчування, медичної допомоги, одягу, психологічної підтримки та інших необхідних ресурсів. За цих умов волонтерський рух став одним із найважливіших елементів підтримки населення, забезпечуючи оперативне реагування на нагальні потреби постраждалих.

З початку повномасштабної війни в Україні сформувалася велика кількість волонтерських організацій, благодійних фондів і громадських ініціатив. Незважаючи на високий рівень залученості суспільства до волонтерської діяльності, процеси взаємодії між волонтерами та особами, які потребують допомоги, досі супроводжуються значною кількістю організаційних труднощів. Більшість звернень про допомогу поширюється через соціальні мережі, месенджери та тематичні групи, що призводить до розпорошення інформації й ускладнює її систематизацію. Внаслідок цього частина запитів залишається непоміченою або обробляється із суттєвими затримками.

Додатковою проблемою є відсутність єдиного інформаційного середовища, яке б забезпечувало централізований облік потреб переселенців, контроль за виконанням заявок та ефективний розподіл наявних ресурсів. У багатьох випадках гуманітарна допомога надходить нерівномірно: окремі категорії громадян отримують її в достатньому обсязі, тоді як інші залишаються без належної підтримки через недостатню поінформованість волонтерів про їхні потреби. Також існує проблема пошуку найближчих пунктів видачі допомоги, гуманітарних штабів і волонтерських центрів, особливо для людей, які перебувають у новому для себе населеному пункті.

Сучасні вебтехнології дозволяють створювати ефективні цифрові платформи, здатні автоматизувати значну частину процесів взаємодії між волонтерами та внутрішньо переміщеними особами. Використання централізованого вебзастосунку дає змогу спростити подання заявок на отримання допомоги, забезпечити оперативне опрацювання звернень, автоматизувати пошук відповідальних волонтерів, вести облік наданої допомоги та формувати аналітичну звітність щодо діяльності системи. Крім того, інтеграція картографічних сервісів дозволяє користувачам швидко знаходити найближчі пункти допомоги та отримувати актуальну інформацію про їхнє місцезнаходження.

Актуальність роботи зумовлена необхідністю підвищення ефективності взаємодії між волонтерами та внутрішньо переміщеними особами шляхом створення сучасного вебзастосунку, який забезпечує централізоване управління заявками на допомогу, автоматизацію процесів координації волонтерської діяльності та зручний доступ користувачів до необхідних сервісів. Запропоноване рішення спрямоване на зменшення адміністративного навантаження на волонтерів, скорочення часу обробки запитів і підвищення доступності гуманітарної допомоги для населення.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Стан та динаміка внутрішнього переміщення в Україні як чинник соціальної дестабілізації

Масштаби внутрішнього переміщення в Україні внаслідок збройної агресії досягли критичних показників, що вимагають не лише оперативних гуманітарних заходів, а й стратегічного переосмислення методів координації ресурсів. Станом на листопад 2025 року загальна кількість офіційно зареєстрованих внутрішньо переміщених осіб (ВПО) становить 4,6 мільйона. Ця цифра відображає лише частину реальної картини, оскільки значна кількість людей уникає офіційної реєстрації з різних причин, зокрема через недовіру до бюрократичних процедур або відсутність надії на реальну підтримку. Демографічний зріз цієї групи населення демонструє глибокий гендерний та віковий дисбаланс: серед ВПО налічується 2,7 мільйона жінок та 1,8 мільйона чоловіків, що свідчить про високе навантаження на систему соціального забезпечення в контексті підтримки материнства та дитинства. Зокрема, 837,6 тисячі дітей віком до 18 років потребують спеціальних умов для адаптації, навчання та психологічної реабілітації.

Аналіз географічного розподілу виявляє нерівномірність навантаження на інфраструктуру приймаючих регіонів. Найвищі показники концентрації ВПО зафіксовані у Дніпропетровській та Харківській областях і в місті Києві [1, 2]. Ситуація в столиці є показовою для розуміння темпів урбанізації переміщених осіб: якщо у 2022 році в Києві було зареєстровано 170 тисяч ВПО, то до кінця 2025 року ця кількість зросла майже втричі – до 433 тисяч осіб.

Такий стрімкий приріст населення створює надлишковий тиск на ринок оренди житла, систему охорони здоров’я та транспортну мережу, що вимагає автоматизованих систем для швидкого розподілу навантаження й координації волонтерських зусиль (табл. 1.1).

Таблиця 1.1 – Динаміка реєстрації ВПО у Києві (2022-2025 роки)

Рік	**Кількість ВПО (тис. осіб)**	**Приріст (%) порівняно з попереднім роком**
2022	170	–
2023	370	+117,6%
2024	420	+13,5%
2025	433	+3,1%

Зниження темпів приросту у 2025 році не свідчить про стабілізацію ситуації, а радше вказує на досягнення межі абсорбційної спроможності міста. Більшість нових переселенців стикаються з проблемою відсутності доступного житла та низьким рівнем соціальної підтримки на початкових етапах переміщення. Це зумовлює актуальність розробки цифрових платформ, які б інтегрували дані про наявність гуманітарної допомоги та можливості працевлаштування в режимі реального часу.

1.2 Соціально-економічний профіль ВПО та бар’єри економічної інтеграції

Економічна самостійність ВПО є ключовим фактором їхньої успішної інтеграції, проте дані за 2024-2025 роки демонструють значні перешкоди на цьому шляху. Серед 4,7 мільйонів переміщених осіб понад 2 мільйони перебувають у працездатному віці. Однак лише 128 тисяч осіб офіційно потребували працевлаштування через центри зайнятості, а допомогу з безробіття отримували лише 26,5 тисячі осіб. Такий розрив між кількістю працездатних осіб та офіційно зареєстрованих безробітних свідчить про високу частку тіньової зайнятості та низьку ефективність наявних державних механізмів працевлаштування цієї категорії громадян.

Важливим чинником зміни поведінки ВПО стали законодавчі зміни умов виплати допомоги на проживання, запроваджені у 2024 році. Вони передбачають обов'язкове працевлаштування або пошук роботи для продовження отримання виплат, що спричинило різке зростання кількості звернень до центрів зайнятості: з 4 тисяч щомісяця у 2023 році до 15 тисяч у березні 2024 року. Попри це, 34% ВПО називають можливість заробітку своєю найгострішою потребою, що є значно вищим показником порівняно з місцевим населенням (24%) (табл. 1.2).

Таблиця 1.2 – Аналіз критичних потреб домогосподарств ВПО

Тип потреби	**Частка домогосподарств (%)**	**Примітки щодо динаміки**
Енергетичне обладнання (генератори, павербанки)	59%	Зростає в періоди дефіциту електроенергії
Медикаменти та медичні послуги	40%	Критично для 50% малозабезпечених сімей
Можливості для стабільного заробітку	34%	Ключовий фактор запобігання поверненню в небезпечні зони
Доступне житло	34%	Основна стаття витрат у великих містах
Професійне навчання та перекваліфікація	28%	Бажана форма допомоги замість прямих виплат

Фінансова криза серед ВПО поглиблюється: 50% домогосподарств із доходом нижче 7064 грн змушені скорочувати витрати на охорону здоров'я. Це вказує на необхідність розширення функціоналу волонтерських платформ у напрямку медичного волонтерства та координації доступу до безоплатних

медичних послуг. Відсутність стабільного доходу та висока вартість оренди житла є основними детермінантами, що змушують людей повертатися до місць постійного проживання навіть попри високі ризики для безпеки. Це створює замкнене коло соціальної вразливості, яке цифрова система координації має допомогти розірвати шляхом надання швидкого доступу до вакансій і соціального житла.

1.3 Огляд та порівняльний аналіз існуючих цифрових платформ підтримки

Ринок IT-рішень для підтримки ВПО та волонтерства в Україні характеризується поєднанням масштабних державних проєктів і гнучких волонтерських ініціатив. Держава зосередила основні ресурси на платформі «Дія» [3, 4], яка стала головним інструментом взаємодії людей з державними органами. Цифрова грамотність українців демонструє стрімке зростання: 81% громадян перебувають в онлайні щодня, а 48% користуються послугами через застосунок або портал «Дія».

Зокрема, послуга отримання статусу ВПО та оформлення грошової допомоги в «Дії» є найбільш затребуваною, оскільки дозволяє подати одну заяву на всю сім’ю, зокрема на дітей, які народилися після переміщення.

Проте «Дія» має певні функціональні обмеження. Послуга онлайн доступна лише для тих, хто раніше не мав статусу ВПО; у разі повторного переміщення або потреби в оновленні даних користувачі часто змушені звертатися до ЦНАПів офлайн.

Крім того, державні сервіси орієнтовані на фіксовані виплати та довідки, тоді як оперативна гуманітарна допомога (продукти, ліки, одяг) залишається прерогативою волонтерських платформ (табл. 1.3).

Таблиця 1.3 – Порівняльна характеристика волонтерських та державних сервісів

Платформа	**Модель верифікації**	**Час обробки запиту**	**Основний функціонал**
Дія	Державні реєстри (ID-картка, закордонний паспорт)	Миттєво (автоматично)	Реєстрація статусу, грошові виплати
Паляниця.Інфо	Модерація менеджерами (ручна перевірка)	До 12 годин	Пошук допомоги за 15 категоріями
єДопомога	Технічна перевірка Мінсоцполітики	До 14 днів	Грошова допомога від міжнародних фондів
Dopomagai.org	Кол-центр (верифікація власників житла)	Залежить від завантаження	Пошук безкоштовного житла для переселенців

Платформа «Паляниця.Інфо» [5], розроблена Українською волонтерською службою у партнерстві з SoftServe, наразі є найбільш структурованим агрегатором гуманітарних ініціатив. Вона містить дані про понад 660 організацій, що надають допомогу переселенцям по всій Україні. Система фільтрації дозволяє шукати допомогу за локаціями (місто, село, область) та категоріями (евакуація, їжа, ліки, допомога дітям). Проте відсутність інтегрованого модуля реального часу для відстеження наявності ресурсів у конкретних пунктах видачі знижує ефективність планування логістики для самих ВПО.

Платформа «єДопомога» [6] забезпечує місток між постраждалими та міжнародними донорами (ООН, ЮНІСЕФ тощо), проте процес верифікації

заявок на цій платформі є тривалим і часто непрозорим для кінцевого користувача, що створює інформаційний вакуум під час очікування. Це підкреслює необхідність створення системи з високим рівнем фідбеку та можливістю відстежувати статус заявки на кожному етапі.

Міжнародний досвід менеджменту біженців та ВПО акцентує увагу на переході від моделі «надання послуг» до моделі «партнерства та лідерства переміщених осіб» (Refugee Leadership) [7]. Уряди Канади, Великої Британії та Австралії активно використовують програми волонтерського менторства та спонсорства, за яких місцеві жителі беруть на себе відповідальність за інтеграцію однієї сім'ї ВПО [8]. Такі програми демонструють вищу ефективність у довгостроковій перспективі, ніж масові гуманітарні центри, оскільки вони забезпечують емоційну підтримку, соціальні зв'язки та прямий доступ до місцевого ринку праці [8].

Ключовим елементом успішних волонтерських програм є наявність професійного координатора та спеціалізованого програмного забезпечення для управління волонтерами (Volunteer Management Software, VMS). Такі системи забезпечують ефективну організацію роботи волонтерів на всіх етапах їхньої діяльності. Зокрема, вони автоматизують процес рекрутингу та онбордингу [9] шляхом створення брендованих реєстраційних форм, збору необхідних даних і перевірки документів кандидатів. Крім того, VMS спрощують планування та логістику, дозволяючи формувати графіки роботи, призначати завдання відповідно до навичок волонтерів і їхнього місцезнаходження, а також автоматично заповнювати вільні зміни [10]. Важливою функцією таких систем є ведення звітності та оцінювання результативності волонтерської діяльності. Вони забезпечують облік відпрацьованих годин, формування статистичних звітів та аналіз внеску волонтерів, що дає змогу оцінювати соціальний і економічний ефект реалізованих програм та сприяє залученню грантового фінансування.

Досвід Організації Об'єднаних Націй з питань біженців (UNHCR) свідчить про важливість створення консультативних рад, які дозволяють

залучати експертизу самих переміщених осіб для розробки IT-інструментів. Прикладом успішної технічної реалізації є Refugee-Led Innovation Fund, який фінансує цифрові рішення, створені безпосередньо постраждалими спільнотами [11]. В українському контексті це може означати інтеграцію ВПО як модераторів і верифікаторів заявок у вебзастосунку.

Проте волонтерські програми також мають ризики. Брак підтримки та навчання волонтерів може призвести до надання неякісних послуг або навіть заподіяння шкоди вразливим категоріям населення. Це вимагає впровадження у вебзастосунок модулів для навчання (LMS-компоненти) та обов'язкової психологічної перевірки волонтерів перед наданням доступу до конфіденційних даних ВПО.

1.4 Технологічні тренди та архітектурне проектування вебзастосунків

Вибір архітектури для волонтерської платформи у 2025 році визначається потребою у високій швидкості розробки на старті та можливості безболісного масштабування в майбутньому. Традиційне протистояння монолітної та мікросервісної архітектур залишається актуальним, проте з'являються нові гібридні підходи.

Для гуманітарних проєктів, де бюджет часто обмежений, а терміни реалізації критичні, рекомендованою стратегією є «швидкий старт на моноліті» з подальшим винесенням найбільш завантажених частин (наприклад, модуля обробки зображень або геоінформаційної системи) у мікросервіси [12]. Це дозволяє уникнути технічного боргу на ранніх етапах і забезпечити стабільну роботу під час пікових навантажень.

Щодо технологічного стека, у 2025 році спостерігається відхід від надмірно складних клієнтських фреймворків на користь гіпермедіаорієнтованих рішень. Поєднання Go (Golang) з HTMX дозволяє

створювати динамічні інтерфейси без необхідності розробки складного API та керування станом на стороні клієнта [13]. Це значно підвищує продуктивність розробки та знижує навантаження на пристрої користувачів, що критично для ВПО, які можуть мати старі моделі смартфонів або обмежений трафік.

Іншим ефективним рішенням для швидкого прототипування є Ruby on Rails. Фреймворк слідує принципу «Convention over Configuration», що дозволяє розробнику зосередитися на бізнес-логіці допомоги, а не на конфігурації бази даних чи маршрутизації. Для систем, що потребують оновлення даних у реальному часі (наприклад, чати допомоги), Cloud Firestore є оптимальним вибором як документна база даних, що забезпечує автоматичну синхронізацію без написання складного коду WebSocket [14].

1.4.1 Правові аспекти обробки персональних даних та кібербезпека

Розробка вебзастосунку для ВПО нерозривно пов’язана з обробкою надчутливих персональних даних, що включає інформацію про поточне місце перебування, стан здоров’я, склад сім’ї та історію переміщення. Порушення конфіденційності таких даних може загрожувати безпеці користувачів, особливо в умовах активних бойових дій.

Законодавство України та міжнародні стандарти встановлюють жорсткі вимоги до таких систем [15]. Обробка повинна здійснюватися на засадах законності, обмеження мети та мінімізації даних. Це означає, що застосунок не має права збирати більше інформації, ніж потрібно для конкретного виду допомоги.

Ключові принципи захисту даних у системі:

– інформована згода. Користувач повинен чітко розуміти, хто, як і для чого використовує його дані. Згода має бути добровільною та може бути відкликана в будь-який момент [16];

– обмеження зберігання. Дані ВПО мають видалятися автоматично після завершення терміну їх актуальності або після того, як потреба у допомозі була задоволена [15];

– технічні заходи захисту. Відповідно до стандартів, вебресурси соціального захисту повинні мати комплексну систему захисту інформації (КСЗІ) [17]. Це передбачає захист від несанкціонованого доступу на рівні вебсервера та бази даних, а також регулярний аудит безпеки [16];

– право на забуття. Система повинна надавати користувачеві інструменти для самостійного видалення профілю або подання вимоги про припинення обробки даних, яка має бути розглянута протягом 10 днів [18].

Важливим аспектом є також верифікація волонтерів. Оскільки вони отримують доступ до адрес ВПО, система повинна вимагати авторизацію через «Дія.Підпис» або BankID для підтвердження особи волонтера перед наданням йому доступу до бази заявок [19]. Це мінімізує ризики шахрайства та використання платформи з ворожою метою.

Проблема координації волонтерства часто полягає у неефективному розподілі обмежених ресурсів. Традиційні черги не враховують індивідуальні потреби та пріоритети. Наукова новизна даної роботи полягає у розробці та впровадженні інтелектуальних алгоритмів матчингу (matching) між потребами ВПО та можливостями волонтерів.

1.4.2 Використання методів машинного навчання

Для підвищення ефективності платформи пропонується застосування алгоритмів колаборативної фільтрації. Побудова моделі взаємодії дозволяє системі пропонувати ВПО найбільш релевантну допомогу на основі їхнього попереднього досвіду та профілів користувачів із схожими потребами. Наприклад, якщо сім’я з маленькими дітьми часто звертається по дитяче

харчування, система може автоматично надсилати повідомлення про надходження підгузків до найближчого волонтерського штабу.

Іншим напрямком є оптимізація маршрутів для волонтерів-кур'єрів. Використання графів та алгоритмів пошуку найкоротшого шляху з урахуванням комендантської години та безпекової ситуації дозволяє скоротити час доставки гуманітарних вантажів. Практична цінність таких рішень підтверджується досвідом розробки систем збору коштів, де впровадження аналітичних інструментів (табл. 1.4) дає змогу оптимізувати волонтерські кампанії та підвищити прозорість використання ресурсів.

Таблиця 1.4 – Методи машинного навчання

Метод	**Очікуваний ефект**	**Технічна реалізація**
Рекомендаційні системи	Збільшення швидкості задоволення потреб на 20-30%	Python (Scikit-learn), матриці корисності
Геолокаційний матчинг	Зменшення транспортних витрат волонтерів	PostGIS, Leaflet API
Аналіз тональності відгуків	Виявлення недобросовісних волонтерів та фондів	NLP (Natural Language Processing)
Автоматична верифікація	Скорочення часу очікування з 14 днів до декількох годин	Інтеграція з реєстрами через API

Ефективність функціонування волонтерських організацій значною мірою залежить не лише від якісної координації роботи та наявності необхідних ресурсів, а й від здатності залучати та утримувати активних учасників протягом тривалого часу. Оскільки волонтерська діяльність переважно ґрунтується на добровільній участі, важливим завданням є створення системи мотивації, яка підтримуватиме зацікавленість волонтерів та сприятиме їхній подальшій активності. Дослідження показують, що одним із найбільш ефективних інструментів підвищення залученості користувачів є впровадження елементів гейміфікації.

Гейміфікація передбачає використання механік, характерних для комп’ютерних ігор, у неігрових середовищах. У контексті волонтерської діяльності це можуть бути цифрові бейджі за виконані завдання, система рейтингів, накопичення балів активності, відображення особистих досягнень, а також формування сертифікатів про участь у волонтерських проєктах. Такі елементи дозволяють користувачам бачити результати власної діяльності, відчувати особистий внесок у спільну справу та отримувати додаткове моральне заохочення.

Важливим аспектом є те, що система досягнень сприяє формуванню довгострокової мотивації та відчуття належності до волонтерської спільноти. Волонтери, які бачать визнання своїх зусиль, частіше продовжують брати участь у діяльності організації та залучають до неї нових учасників. Крім того, цифрові сертифікати та історія виконаних завдань можуть використовуватися як підтвердження громадської активності під час працевлаштування, вступу до освітніх закладів або участі в грантових програмах.

З економічної точки зору утримання досвідчених волонтерів є значно ефективнішим, ніж постійний пошук та навчання нових учасників. Нові волонтери потребують часу для ознайомлення з правилами роботи, проходження інструктажу та адаптації до внутрішніх процесів організації. Натомість досвідчені учасники вже володіють необхідними навичками, розуміють специфіку виконання завдань та можуть самостійно координувати окремі напрями діяльності.

Саме тому впровадження мотиваційних механізмів і засобів гейміфікації є важливою складовою сучасних систем управління волонтерською діяльністю. Крім того, гейміфікація стимулює розвиток здорової конкуренції між учасниками, заохочує до виконання більшої кількості завдань і підвищує відповідальність за їх виконання.

1.5 Постановка задачі

В умовах зростання кількості внутрішньо переміщених осіб та підвищеного навантаження на волонтерські організації виникає потреба у створенні інформаційної системи, здатної забезпечити ефективну взаємодію між волонтерами та особами, які потребують допомоги. Наявні способи координації волонтерської діяльності, що переважно базуються на використанні соціальних мереж і месенджерів, не забезпечують належного рівня централізації інформації, контролю за виконанням заявок та оперативного розподілу ресурсів.

Об’єктом роботи є створення вебзастосунку для координації волонтерської діяльності та підтримки внутрішньо переміщених осіб.

Метою роботи є розробка адаптивного вебзастосунка для координації волонтерської діяльності, що включає автоматизовану систему обробки звернень і централізовану базу даних потреб внутрішньо переміщених осіб.

Для досягнення поставленої мети необхідно виконати такі завдання:

– провести аналіз предметної області та існуючих рішень для організації волонтерської діяльності;

– визначити функціональні та нефункціональні вимоги до вебзастосунку;

– спроєктувати архітектуру системи та структуру бази даних;

– реалізувати механізми реєстрації та авторизації користувачів;

– розробити функціонал створення, обробки та контролю виконання заявок на допомогу;

– реалізувати інтеграцію картографічних сервісів для пошуку пунктів допомоги;

– провести тестування програмного продукту та оцінити ефективність його роботи.

2 ОГЛЯД МЕТОДІВ ДЛЯ РОЗРОБКИ ВЕБЗАСТОСУНКА

2.1 Середовище розробки PHPStorm

Редактор вихідного коду PhpStorm [20] є спеціалізованим середовищем для веброзробки та призначений для створення вебдодатків і програмних систем із застосуванням мов програмування PHP, HTML, JavaScript і CSS. Це середовище підтримує синхронізацію з процесом розгортання проєктів, а також надає можливість керувати ними через протокол FTP.

До основних можливостей PhpStorm належать автоматичне доповнення коду PHP, перевірка коректності програмного коду, підтримка різноманітних методів рефакторингу, а також швидка навігація між елементами проєкту. Крім того, середовище містить значну кількість плагінів для інтеграції з популярними бекенд- і фронтенд-фреймворками та надає інструменти для статичного аналізу коду.

Головною перевагою PhpStorm є висока продуктивність і широкий набір функціональних можливостей, орієнтованих на розробку програмного забезпечення мовою PHP. Середовище забезпечує підтримку синтаксису PHP, інструменти автоматичного доповнення коду, виявлення помилок і систему підказок, що сприяє написанню структурованого та ефективного програмного коду.

Також PhpStorm містить розширені засоби редагування, серед яких автоматичне форматування коду, швидкий перехід до визначень функцій і класів, пошук і заміна фрагментів коду, а також вбудований термінал. Наявність цих інструментів підвищує зручність роботи та пришвидшує процес розробки програмного забезпечення.

Додатково середовище має інтегровану підтримку систем контролю версій, зокрема Git, Subversion і Mercurial. Це дозволяє розробникам безпосередньо з інтерфейсу PhpStorm виконувати основні операції з репозиторіями, зокрема коміти, створення гілок, злиття змін та інші дії,

пов’язані з керуванням версіями програмного коду. На рисунку 2.1 наведено інтерфейс PHPStorm.

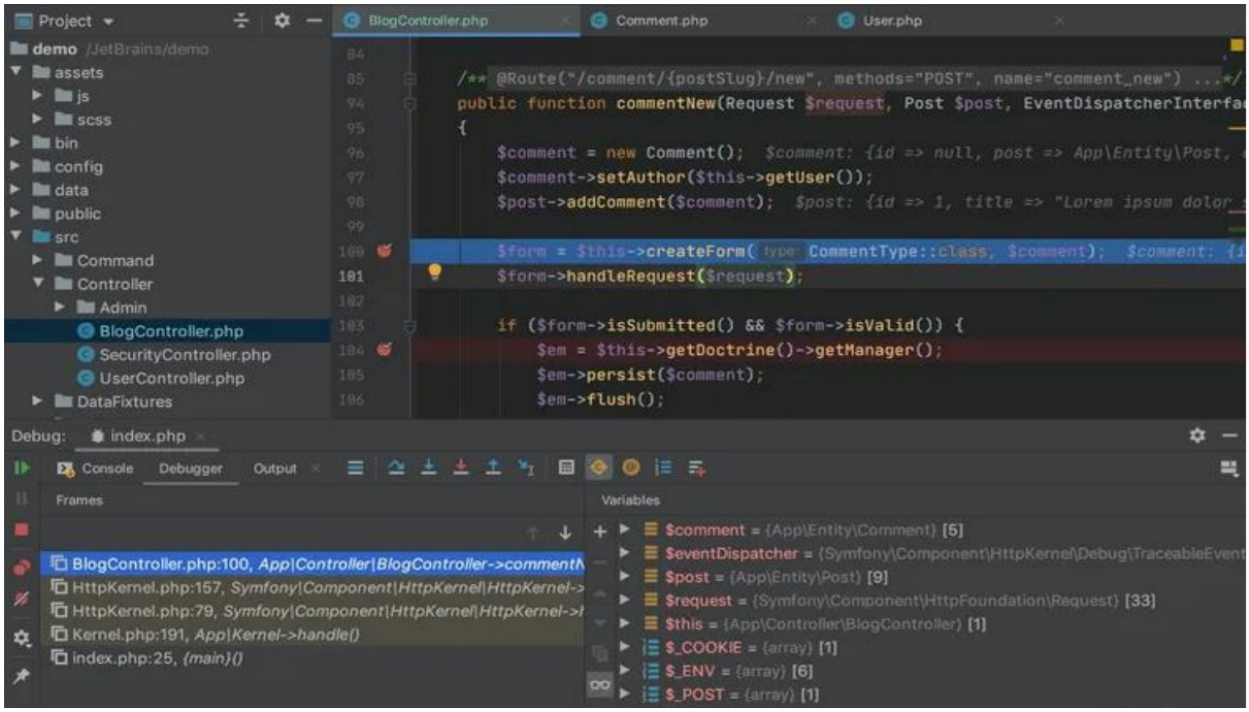

Рисунок 2.1 – Інтерфейс PHPStorm

2.2 Середовище розробки Visual Studio Code

Visual Studio Code – це сучасний кросплатформний редактор вихідного коду, розроблений компанією Microsoft для створення програмного забезпечення, вебдодатків і хмарних сервісів [21]. Дане середовище підтримує велику кількість мов програмування, серед яких JavaScript, TypeScript, Python, Java, C++, PHP та інші, що робить його універсальним інструментом для розробників різних напрямів.

Visual Studio Code характеризується високою продуктивністю, невеликим споживанням системних ресурсів і зручним інтерфейсом користувача. Редактор функціонує на операційних системах Windows, Linux та macOS, що забезпечує можливість кросплатформної розробки програмних продуктів. На рисунку 2.2 представлено інтерфейс Visual Studio Code.

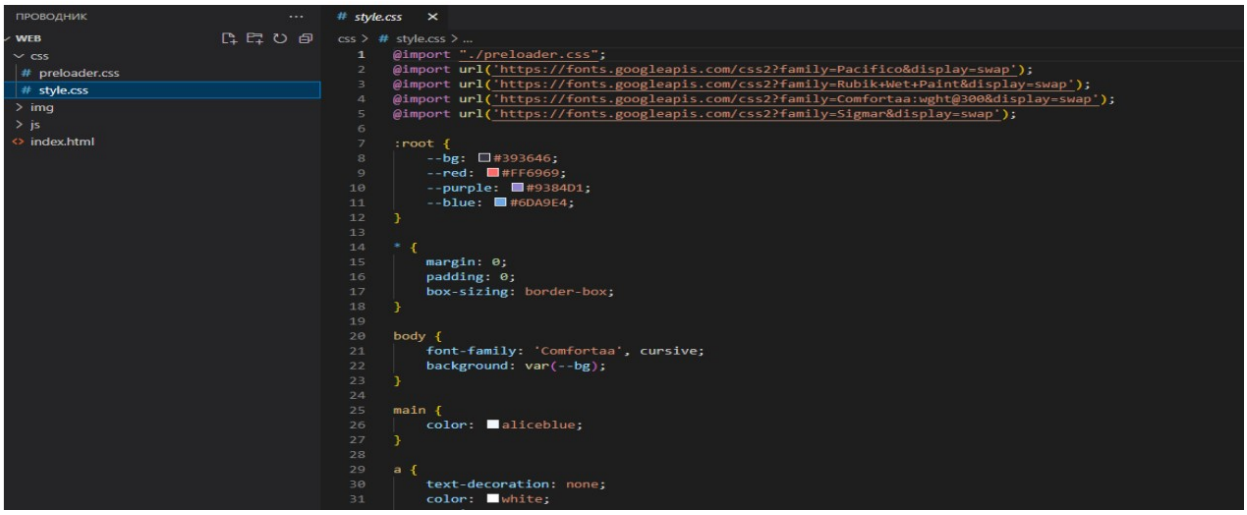

Рисунок 2.2. – Інтерфейс Visual Studio Code

Однією з ключових переваг середовища є підтримка великої кількості розширень, які дозволяють адаптувати редактор до потреб конкретного проєкту. За допомогою спеціальних плагінів користувач може додавати підтримку нових мов програмування, фреймворків, систем контролю версій, засобів тестування та інших інструментів розробки.

Редактор містить вбудований відладчик, що дає змогу аналізувати виконання програмного коду, знаходити та виправляти помилки без використання стороннього програмного забезпечення. Також Visual Studio Code підтримує інтеграцію із системою контролю версій Git, що забезпечує ефективну командну роботу над програмними проєктами та спрощує керування змінами у вихідному коді.

Середовище надає широкий набір функцій для редагування коду, зокрема автоматичне доповнення конструкцій, підсвічування синтаксису, форматування коду, контекстні підказки, швидку навігацію між файлами та елементами програми, а також інструменти рефакторингу. Наявність цих можливостей підвищує швидкість розробки та покращує якість програмного забезпечення.

Крім того, Visual Studio Code підтримує роботу з вебтехнологіями та сучасними фреймворками, такими як React, Angular, Vue.js та Node.js, що

робить його однією з найбільш популярних середовищ для створення сучасних вебдодатків і серверних рішень.

2.3 Платформа ASP.NET WEB API

ASP.NET Web API – це програмна платформа, розроблена компанією Microsoft, яка призначена для створення HTTP-сервісів і вебінтерфейсів прикладного програмування (API) [22]. Ця технологія входить до складу платформи ASP.NET та використовується для розробки серверної частини вебдодатків, мобільних застосунків і хмарних сервісів.

ASP.NET Web API забезпечує можливість обміну даними між клієнтською та серверною частинами програмного забезпечення через протокол HTTP. Платформа підтримує передачу даних у форматах JSON і XML, що дозволяє інтегрувати різні програмні системи незалежно від мови програмування або типу клієнтського застосунку.

Однією з основних переваг ASP.NET Web API є підтримка архітектурного стилю REST, який забезпечує просту та ефективну взаємодію між компонентами системи. Використання REST-підходу сприяє підвищенню масштабованості, продуктивності та зручності інтеграції вебсервісів із зовнішніми системами.

Платформа підтримує механізми маршрутизації запитів, автентифікації та авторизації користувачів, обробки винятків, модульного тестування та роботи з базами даних. Крім того, ASP.NET Web API інтегрується з технологією Entity Framework, що спрощує реалізацію доступу до даних і взаємодію з реляційними базами даних.

Середовище також забезпечує високу продуктивність, гнучкість налаштувань та можливість створення багаторівневих програмних систем. Завдяки підтримці сучасних технологій і засобів безпеки ASP.NET Web API широко використовується для розробки веборієнтованих інформаційних

систем, мобільних сервісів та корпоративних застосунків. На рисунку 2.3 зображено шаблони API.

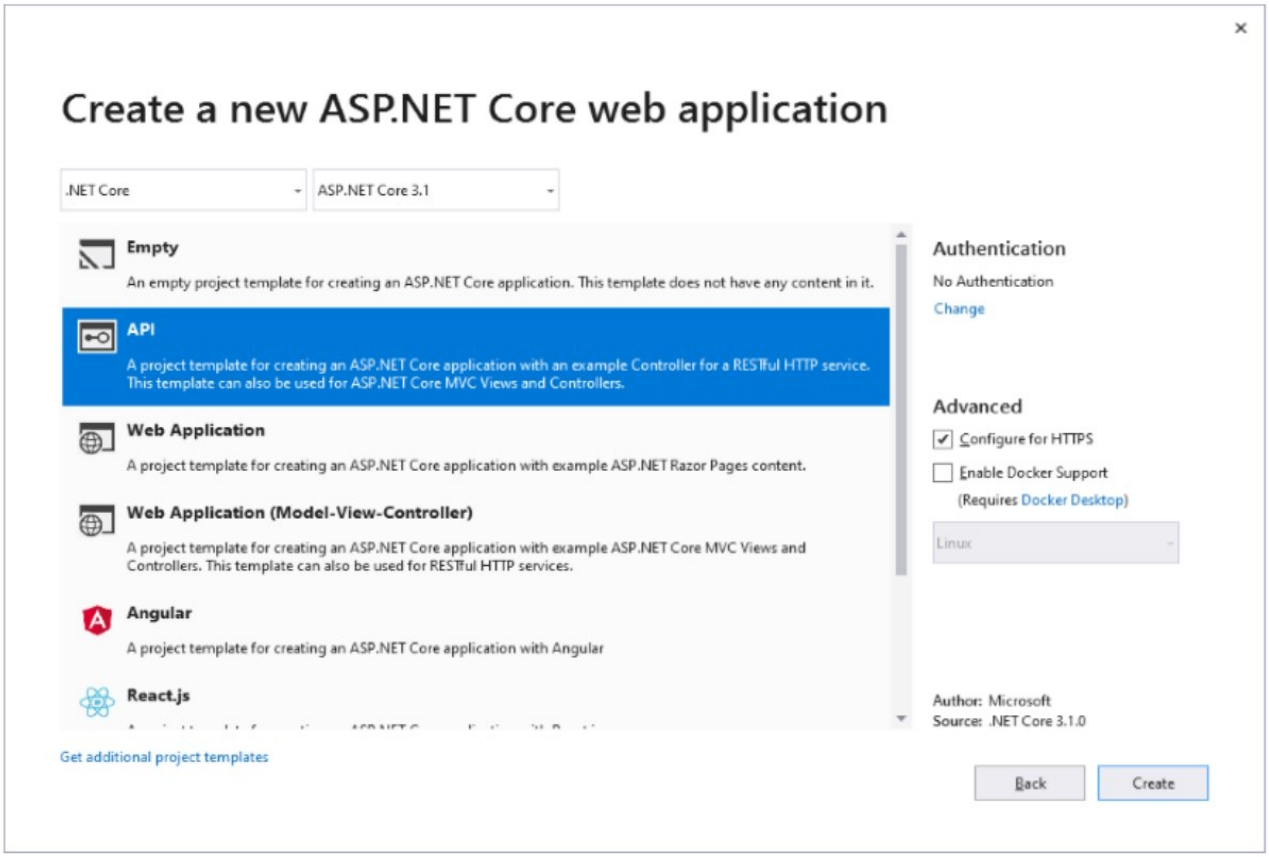

Рисунок 2.3 – Проєкт ASP.NET Core серед шаблонів API

Web API – це інтерфейс прикладного програмування, який забезпечує взаємодію між різними програмними системами через мережу Інтернет за допомогою протоколу HTTP. Ця технологія дозволяє клієнтським застосункам обмінюватися даними із сервером, надсилати запити та отримувати відповіді у стандартизованих форматах, зокрема JSON або XML.

Web API широко використовується для створення вебдодатків, мобільних застосунків, хмарних сервісів та інтеграції різних інформаційних систем. Основною функцією Web API є забезпечення взаємодії між серверною та клієнтською частинами програмного забезпечення незалежно від мови програмування чи платформи, на яких вони реалізовані.

Однією з найважливіших переваг технології Web API є підтримка архітектурного стилю REST (Representational State Transfer), який забезпечує стандартизований підхід до організації взаємодії між клієнтською та серверною частинами програмної системи. Використання REST дає змогу здійснювати обмін даними через HTTP-протокол із застосуванням стандартних методів запитів, таких як GET, POST, PUT і DELETE. Такий підхід сприяє створенню масштабованих, гнучких та легко підтримуваних програмних рішень, а також значно спрощує інтеграцію вебзастосунків із зовнішніми сервісами та сторонніми інформаційними системами.

Завдяки Web API клієнтські застосунки можуть отримувати доступ до серверних ресурсів незалежно від використовуваної платформи або мови програмування. Дані, як правило, передаються у форматі JSON або XML, що забезпечує їхню універсальність і сумісність із різними типами програмного забезпечення. Це дозволяє використовувати один і той самий програмний інтерфейс для взаємодії з вебзастосунками, мобільними застосунками, настільними програмами та іншими інформаційними системами.

Важливою характеристикою Web API є наявність вбудованих механізмів забезпечення безпеки та контролю доступу. Технологія підтримує різні способи автентифікації та авторизації користувачів, що дозволяє обмежувати доступ до ресурсів відповідно до встановлених ролей і прав. Крім того, Web API забезпечує ефективну маршрутизацію запитів до відповідних контролерів і методів, що спрощує структурування програмного коду та підвищує його підтримуваність. Для забезпечення стабільної роботи системи також реалізуються механізми обробки винятків і помилок, які дозволяють своєчасно виявляти та усувати проблеми під час виконання запитів.

Застосування Web API є невід’ємною складовою сучасної веброзробки, оскільки ця технологія забезпечує ефективний обмін даними між компонентами програмної системи, сприяє реалізації клієнт-серверної архітектури та створює умови для подальшого масштабування програмного

забезпечення. Завдяки своїй універсальності, простоті інтеграції та високій продуктивності Web API широко використовується під час розробки корпоративних інформаційних систем, вебпорталів, мобільних застосунків та хмарних сервісів. Архітектуру Web API наведено на рисунку 2.4.

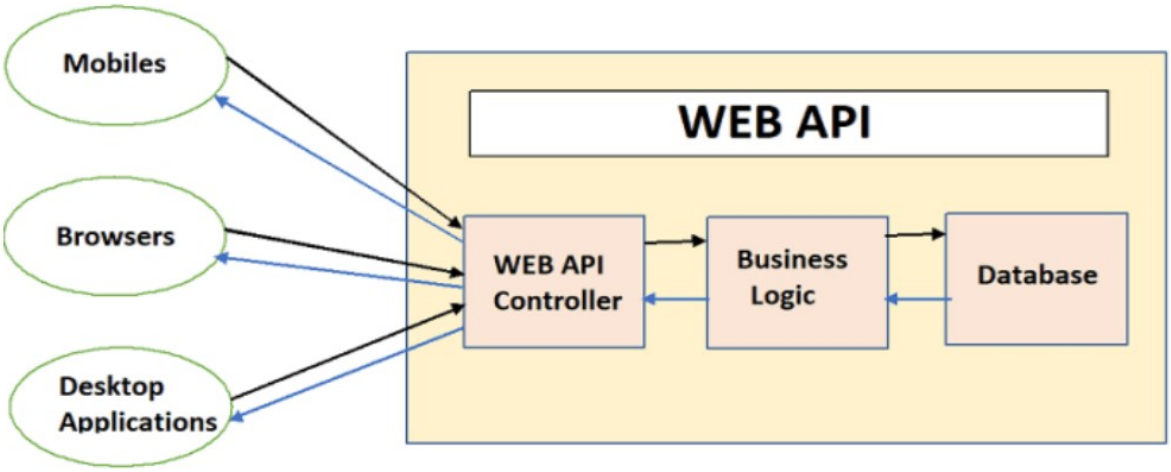

Рисунок 2.4 – Архітектура додатку з використанням WEB API

2.4 Графічний редактор Figma

Figma – сучасний хмарний графічний редактор, призначений для проєктування користувацьких інтерфейсів, створення інтерактивних прототипів та організації спільної роботи над програмними проєктами. Завдяки своїй функціональності та доступності через веббраузер цей інструмент набув широкого поширення серед дизайнерів, розробників і менеджерів проєктів [23]. Figma дозволяє створювати макети вебсайтів, мобільних застосунків, інформаційних систем та інших цифрових продуктів, забезпечуючи повний цикл розробки інтерфейсу від початкової концепції до фінального дизайну. Однією з основних переваг Figma є можливість колективної роботи в режимі реального часу. Кілька учасників команди можуть одночасно працювати над одним проєктом, переглядати зміни,

залишати коментарі та обговорювати окремі елементи інтерфейсу. Такий підхід значно підвищує ефективність командної роботи та скорочує час узгодження дизайнерських рішень.

Інструмент надає широкий набір засобів для створення графічних компонентів, побудови адаптивних макетів і розробки інтерактивних прототипів. Використання компонентів і стилів дозволяє забезпечити єдність дизайну в межах проєкту та спрощує подальше внесення змін. Крім того, Figma підтримує створення дизайн-систем, що є особливо важливим під час розробки великих програмних продуктів із великою кількістю інтерфейсних елементів.

Важливою функцією Figma є можливість створення інтерактивних прототипів, які дозволяють моделювати поведінку майбутнього вебзастосунку ще до початку програмної реалізації, що дає змогу перевірити зручність навігації, оцінити користувацький досвід та своєчасно виявити потенційні недоліки інтерфейсу. Завдяки цьому процес розробки стає більш передбачуваним, а ризик виникнення помилок на етапі реалізації суттєво зменшується. Інтерфейс графічного редактора Figma наведено на рисунку 2.5.

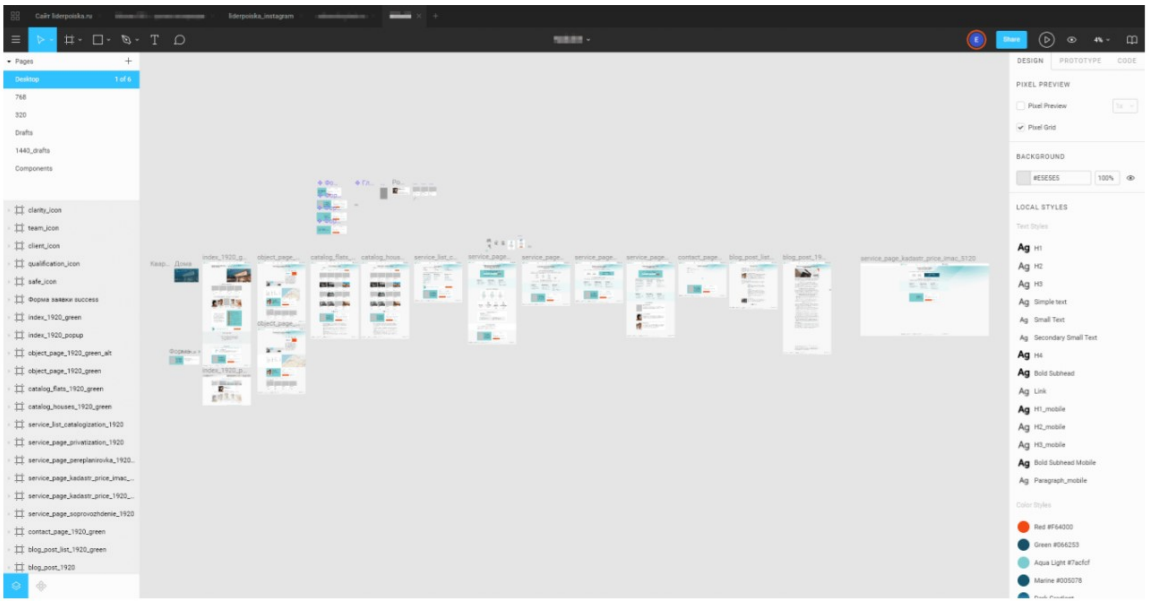

Рисунок 2.5 – Інтерфейс графічного редактора Figma

Однією з основних переваг Figma є можливість колективної роботи в режимі реального часу. Користувачі можуть одночасно редагувати макети, залишати коментарі та вносити зміни до проєкту, що значно спрощує взаємодію між дизайнерами, розробниками та іншими учасниками команди.

Графічний редактор функціонує у веббраузері та не потребує складного встановлення програмного забезпечення, що забезпечує доступ до проєктів із різних пристроїв і операційних систем. Крім того, Figma підтримує роботу з компонентами, стилями, бібліотеками елементів інтерфейсу та системами дизайну, що дозволяє підвищити ефективність створення та підтримки великих проєктів.

Середовище містить інструменти для створення інтерактивних прототипів, анімацій переходів між екранами та моделювання поведінки користувацького інтерфейсу. Це дозволяє оцінювати зручність використання майбутнього програмного продукту ще на етапі проєктування.

Також Figma забезпечує інтеграцію з іншими сервісами та платформами розробки, що спрощує передачу дизайнерських макетів у процес програмної реалізації. Завдяки широкому набору функцій, зручності використання та підтримці командної роботи Figma є одним із найпоширеніших інструментів UI/UX-дизайну.

2.5 Мова програмування та розмітки гіпертексту HTML

HTML – це стандартизована мова розмітки гіпертекстових документів, яка використовується для створення вебсторінок і вебдодатків, доступних через мережу Інтернет. Ця мова дозволяє описувати структуру вебдокументів за допомогою спеціальних тегів, що визначають розташування тексту, зображень, таблиць, форм, мультимедійних елементів та інших компонентів сторінки [24]. Браузер інтерпретує HTML-код і відображає його у вигляді зручного для користувача вебінтерфейсу.

Мова HTML була розроблена на початку 1990-х років британським ученим Tim Berners-Lee як основа для функціонування Всесвітньої павутини. Перші версії HTML забезпечували базові можливості для створення гіпертекстових документів і навігації між ними за допомогою гіперпосилань. У подальшому мова постійно вдосконалювалася, а її стандартизацією займався World Wide Web Consortium (W3C), який продовжує розвиток HTML та інших вебтехнологій.

Для створення сучасних вебзастосунків HTML зазвичай використовується разом із каскадними таблицями стилів CSS та мовою програмування JavaScript. У такому поєднанні HTML відповідає за структуру вебсторінки, CSS – за її зовнішній вигляд і стилізацію, а JavaScript – за реалізацію інтерактивних функцій та динамічної поведінки елементів інтерфейсу.

Основою HTML є правила стандарту SGML, який визначає принципи створення мов розмітки. На базі HTML та XML була створена мова XHTML, яка поєднує можливості HTML із суворішими правилами XML. Використання XHTML дозволяє застосовувати стандартні засоби обробки XML-документів, що спрощує автоматизовану обробку та підвищує структурованість вебдокументів. Рекомендації щодо використання XHTML були офіційно представлені консорціумом W3C у 2000 та 2001 роках для версій XHTML 1.0 і XHTML 1.1 відповідно.

HTML є текстовим форматом представлення вебдокументів, що забезпечує простоту створення, редагування та інтерпретації вебдокументів. Однією з основних переваг мови є її платформонезалежність, оскільки HTML-документи можуть коректно відображатися на різних операційних системах і в різних браузерах. Крім того, текстовий формат забезпечує невеликий розмір файлів, зручність редагування та високу швидкість обробки вебсторінок.

Завдяки простоті використання, універсальності та широкій підтримці браузерами HTML є базовою технологією сучасної веброзробки та використовується для створення більшості вебресурсів у мережі Інтернет.

HTML є мовою розмітки гіпертекстових документів, яка використовується для створення структури вебсторінок і визначення їхнього змісту. За допомогою HTML можна формувати різні типи елементів вебдокумента, зокрема текстові блоки з певними параметрами форматування, списки, таблиці, зображення, гіперпосилання та інші компоненти інтерфейсу.

Для визначення таких елементів застосовуються спеціальні команди розмітки, які називаються тегами. Браузер інтерпретує HTML-документ, обробляючи ці теги, і відображає результат у вигляді вебсторінки. Кожен елемент зазвичай складається з відкриваючого та закриваючого тегів, між якими розміщується відповідний вміст. Наприклад, тег абзацу використовується для початку текстового блоку, а – для його завершення, що дозволяє чітко структурувати вміст сторінки.

Під час створення вебсторінок розробник використовує набір тегів та атрибутів, які розпізнаються будь-яким сучасним браузером і забезпечують коректне відображення контенту. Атрибути дозволяють задавати додаткові параметри елементів, розширюючи їхню функціональність і вплив на вигляд або поведінку сторінки.

Сучасним стандартом мови є HTML5, який використовується для розробки більшості вебресурсів в Інтернеті. HTML5 розширює можливості попередніх версій, зокрема забезпечує вбудовану підтримку мультимедійного контенту, такого як аудіо та відео, без необхідності використовувати додаткові плагіни.

HTML5 є основою сучасної веброзробки, оскільки саме ця мова розмітки визначає структуру та логічну організацію вебсторінок. За допомогою HTML5 створюються основні елементи інтерфейсу, такі як заголовки, текстові блоки, таблиці, форми, списки, зображення та мультимедійний контент. Водночас візуальне оформлення сторінок

реалізується за допомогою каскадних таблиць стилів CSS, а динамічна поведінка та інтерактивність забезпечуються мовою програмування JavaScript. Для обробки даних і реалізації бізнес-логіки можуть використовуватися різні серверні технології та фреймворки, зокрема PHP, ASP.NET Core, Node.js та інші. Таким чином, HTML5 виступає фундаментом, на якому будується повноцінна архітектура сучасних вебзастосунків.

Однією з головних переваг HTML5 є його підтримка сучасних вебстандартів і мультимедійних можливостей без необхідності використання сторонніх плагінів. Специфікація HTML5 містить нові семантичні елементи, які покращують структурування контенту та сприяють підвищенню доступності вебресурсів для користувачів і пошукових систем. Крім того, мова підтримує вбудоване відтворення аудіо- та відеоматеріалів, роботу з графікою за допомогою елемента Canvas, геолокацію, локальне зберігання даних та інші можливості, необхідні для створення сучасних інтерактивних вебсервісів. Завдяки своїй універсальності, сумісності з усіма сучасними браузерами та підтримці міжнародних стандартів HTML5 став базовою технологією для розробки вебсайтів і вебзастосунків різного призначення. Використання HTML5 забезпечує високу сумісність вебресурсів, зручність подальшої підтримки та можливість інтеграції з широким спектром сучасних вебтехнологій.

2.6 Мова програмування C#

Під час розробки вебзастосунку як основну мову програмування було обрано C# [25], однак її використання саме по собі є недостатнім для реалізації повноцінного програмного продукту. Для створення сучасного вебзастосунку також необхідно визначити відповідну платформу та фреймворк, які забезпечують необхідну архітектуру, інструменти й

середовище виконання.

Сучасною базовою платформою для розробки на C# є .NET Core, яка є модульною, кросплатформною та відкритою програмною платформою. Вона підтримує роботу на операційних системах Windows, Linux та macOS, що дозволяє створювати універсальні застосунки, які не залежать від конкретного середовища виконання.

Для реалізації вебзастосунків було обрано ASP.NET Core – сучасний відкритий кросплатформний фреймворк, розроблений компанією Microsoft за участі спільноти розробників. Даний фреймворк відзначається високою продуктивністю та ефективністю порівняно з попередніми версіями ASP.NET. Він працює поверх платформи .NET Core і дозволяє розгортати вебзастосунки на різних операційних системах, включаючи Windows, Linux та macOS.

Для розгортання вебзастосунків, створених на платформі ASP.NET Core, можуть використовуватися різні серверні рішення залежно від вимог до продуктивності та середовища експлуатації. Найпоширенішими варіантами є вебсервер Internet Information Services (IIS) та вбудований сервер Kestrel. IIS забезпечує надійну інтеграцію з операційною системою Windows, підтримує механізми автентифікації, балансування навантаження та централізованого адміністрування. Водночас Kestrel є сучасним високопродуктивним кросплатформним вебсервером, оптимізованим для роботи із застосунками ASP.NET Core та здатним ефективно обробляти велику кількість одночасних HTTP-запитів. Використання Kestrel дозволяє розгортати вебзастосунки не лише в середовищі Windows, а й на серверах під керуванням Linux або macOS, що значно розширює можливості використання платформи.

Однією з ключових переваг .NET Core є модульна архітектура, яка забезпечує високу гнучкість під час розробки програмного забезпечення. Замість включення всіх компонентів до складу платформи розробник має можливість підключати лише необхідні бібліотеки та залежності за допомогою пакетного менеджера NuGet. Такий підхід дозволяє зменшити

розмір застосунку, підвищити його продуктивність та спростити процес супроводження програмного коду. Крім того, модульна структура сприяє повторному використанню компонентів і прискорює розробку складних програмних систем.

Платформа також містить вбудований механізм ін’єкції залежностей (Dependency Injection), який реалізує принципи інверсії керування та забезпечує слабке зв’язування між компонентами системи. Завдяки цьому підвищується гнучкість архітектури застосунку, спрощується тестування окремих модулів і покращується підтримуваність програмного коду. Додатковою перевагою є наявність модульного конвеєра обробки HTTP-запитів (Middleware Pipeline), який дозволяє послідовно обробляти запити та відповіді за допомогою спеціалізованих компонентів. Це дає змогу легко реалізовувати такі функції, як автентифікація, авторизація, журналювання подій, обробка помилок, кешування та інші механізми, необхідні для побудови сучасних вебзастосунків.

Таким чином, використання ASP.NET Core та .NET Core забезпечує високу продуктивність, масштабованість, кросплатформність і гнучкість архітектури, що робить дану платформу ефективним рішенням для створення сучасних вебзастосунків різного рівня складності.Додатковою перевагою платформи .NET Core є її кросплатформність, що дозволяє розробляти, тестувати та розгортати вебзастосунки на різних операційних системах, зокрема Windows, Linux та macOS. Завдяки цьому забезпечується висока гнучкість під час вибору середовища розробки та серверної інфраструктури. Платформа також надає набір потужних інструментів командного рядка (CLI), які дозволяють виконувати компіляцію, тестування, запуск та розгортання застосунків без необхідності використання графічного середовища розробки. Це значно спрощує автоматизацію процесів розробки та впровадження програмного забезпечення, а також сприяє ефективному використанню сучасних підходів безперервної інтеграції та доставки програмних продуктів.

Проведений аналіз сучасних технологій та програмних засобів для створення вебзастосунків дозволив визначити оптимальний набір компонентів, необхідних для реалізації поставленої задачі. Для забезпечення роботи серверної частини було обрано платформу ASP.NET Core, яка функціонує в середовищі .NET Core та надає широкий спектр інструментів для розробки високопродуктивних і масштабованих вебсистем. Як вебсервер може використовуватися IIS або Kestrel. Сервер IIS забезпечує глибоку інтеграцію з операційною системою Windows та надає розширені можливості адміністрування і моніторингу, тоді як Kestrel є високопродуктивним кросплатформним вебсервером, оптимізованим для роботи із застосунками ASP.NET Core.

Для зберігання та обробки даних обрано систему керування базами даних Microsoft SQL Server, яка забезпечує високу надійність, продуктивність та підтримку складних механізмів роботи з даними. Використання даної СКБД дозволяє ефективно організувати зберігання інформації про користувачів, заявки, повідомлення та інші сутності вебзастосунку. Для адміністрування бази даних використовується SQL Server Management Studio, що надає зручний графічний інтерфейс для створення, налаштування, моніторингу та супроводження баз даних.

Таким чином, обраний технологічний стек поєднує сучасні засоби розробки, високу продуктивність, надійність та можливість масштабування, що повністю відповідає вимогам до створення вебзастосунку для координації взаємодії між волонтерами та внутрішньо переміщеними особами.

2.7 Фреймворк Angular

Angular є сучасною альтернативою AngularJS, створеною для розробки динамічних вебзастосунків із використанням мови TypeScript. AngularJS, розроблений у 2009 році на основі JavaScript, застосовується для позначення

версій Angular 1.x [26]. Нова версія Angular отримала вдосконалену модульну архітектуру, стала простішою у використанні та демонструє вищу продуктивність порівняно з AngularJS. Основною відмінністю між цими фреймворками є використання різних мов програмування: Angular побудований на TypeScript, який є розширенням ES6, тоді як AngularJS використовує JavaScript. Це впливає на архітектуру та принципи роботи обох технологій.

Однією з ключових особливостей Angular є компонентний підхід до побудови застосунків. Для організації структури використовуються компоненти, що формують ієрархію елементів інтерфейсу. У AngularJS, навпаки, основну роль відіграють директиви, які забезпечують повторне використання коду та створення окремих модулів. Хоча Angular також підтримує директиви, механізм їх реалізації суттєво відрізняється від AngularJS.

Архітектура Angular базується на компонентах, які поєднують структурні та атрибутивні директиви для зміни структури й поведінки DOM-елементів. Структурні директиви відповідають за зміну структури сторінки, а атрибутивні – за модифікацію зовнішнього вигляду елементів. Такий підхід дозволяє відокремити логіку застосунку від функціональних компонентів, що позитивно впливає на підтримку та масштабованість системи.

У свою чергу, AngularJS використовує архітектурний шаблон MVC (Model-View-Controller), де контролер відповідає за обробку даних, логіку роботи та взаємодію з користувачем. Модель забезпечує збереження та керування даними, а представлення відповідає за їх відображення. Такий підхід спрощує процес розробки та дозволяє скоротити кількість програмного коду завдяки механізму двостороннього зв’язування даних.

Angular також використовує ієрархічну систему ін’єкції залежностей, що сприяє підвищенню продуктивності та гнучкості застосунку. На відміну від нього, AngularJS більшою мірою спирається на директиви. Крім цього, механізм маршрутизації в Angular є сучаснішим: замість використання

@routeProvider.when, як в AngularJS, застосовується конфігурація @Route та робота з URL-адресами, що забезпечує ефективніше керування навігацією.

Angular має більш оптимізовану структуру, що дозволяє підвищити швидкість виконання операцій і загальну продуктивність застосунку. Попри те, що AngularJS підтримує двостороннє зв’язування даних, Angular демонструє вищу ефективність завдяки вдосконаленій архітектурі та оптимізації внутрішніх процесів. Для розробки застосунків AngularJS часто потрібно використовувати сторонні інструменти, зокрема IDE, наприклад WebStorm, для створення та налагодження програм. Angular, у свою чергу, надає власний інтерфейс командного рядка (CLI), який спрощує створення проєктів, тестування та автоматизацію процесів розробки. Завдяки більш структурованому підходу до організації коду Angular є більш придатним для створення масштабних і складних вебзастосунків порівняно з AngularJS.

2.8 Хмарні бази даних Microsoft Azure

Microsoft Azure – це хмарна платформа, яка надає широкий набір сервісів для зберігання, обробки та керування даними [27]. Хмарні бази даних Microsoft Azure дозволяють створювати масштабовані, надійні та високопродуктивні інформаційні системи без необхідності використовувати власну фізичну серверну інфраструктуру. Платформа підтримує як реляційні, так і нереляційні бази даних, що дає змогу обирати оптимальний варіант залежно від потреб проєкту.

Одним із найпоширеніших рішень є Azure SQL Database – керована реляційна база даних, побудована на основі Microsoft SQL Server. Вона забезпечує автоматичне резервне копіювання, масштабування, оновлення та високий рівень безпеки. Azure SQL Database підтримує роботу з SQL-запитами, транзакціями та механізмами контролю доступу, що робить її

придатною для створення корпоративних вебзастосунків і систем обробки даних.

Для роботи з великими обсягами неструктурованої інформації [28-30] платформа Azure пропонує Azure Cosmos DB – глобально розподілену NoSQL-базу даних. Вона підтримує декілька моделей даних, зокрема документи, графи, ключ-значення та колонки, а також забезпечує низькі затримки доступу до даних у різних регіонах світу, що робить Cosmos DB ефективним рішенням для високонавантажених вебсистем і мобільних застосунків.

Платформа Microsoft Azure також підтримує інтеграцію з різними мовами програмування, фреймворками та сервісами аналітики. Завдяки цьому розробники можуть створювати сучасні масштабовані системи з використанням хмарних технологій.

Додатковими перевагами Azure є автоматичне масштабування ресурсів, високий рівень відмовостійкості, можливість резервного копіювання та централізоване адміністрування.

Використання хмарних баз даних Microsoft Azure дозволяє зменшити витрати на підтримку серверної інфраструктури, підвищити доступність сервісів та спростити процес розробки й розгортання програмного забезпечення.

2.9 Хмарні бази даних PostgreSQL

PostgreSQL – це об’єктно-реляційна система керування базами даних з відкритим вихідним кодом, яка широко використовується для створення сучасних інформаційних систем і вебзастосунків [31].

Хмарні бази даних PostgreSQL дозволяють розміщувати та адмініструвати бази даних у хмарному середовищі без необхідності

підтримувати власну серверну інфраструктуру. Такий підхід забезпечує високу доступність, масштабованість і надійність системи.

Хмарні сервіси PostgreSQL надаються багатьма платформами, зокрема Amazon Web Services, Google Cloud і Microsoft Azure. Вони забезпечують автоматичне резервне копіювання, оновлення, масштабування ресурсів і моніторинг продуктивності бази даних. Завдяки цьому розробники можуть зосередитися на створенні функціональності застосунку без необхідності в адмініструванні серверів.

PostgreSQL підтримує широкий набір можливостей для роботи з даними, зокрема SQL-запити, транзакції, індексацію, тригери, збережені процедури та механізми контролю цілісності даних. Система також підтримує JSON, що дозволяє ефективно працювати як зі структурованими [32-34], так і з напівструктурованими даними.

Однією з важливих переваг PostgreSQL є підтримка розширень, які дають змогу додавати додаткові функції без зміни ядра системи. Наприклад, розширення PostGIS використовується для роботи з геопросторовими даними, а pgCrypto – для шифрування інформації. Це робить PostgreSQL універсальним рішенням для різних типів проєктів.

Хмарні реалізації PostgreSQL забезпечують високий рівень безпеки завдяки механізмам автентифікації, шифруванню даних та контролю доступу. Крім того, підтримується реплікація даних і автоматичне відновлення після збоїв, що підвищує стабільність роботи системи.

Завдяки поєднанню відкритого вихідного коду, широкої функціональності та можливостей хмарних технологій PostgreSQL є ефективним рішенням для створення масштабованих і надійних вебзастосунків та інформаційних систем.

PostgreSQL має вбудовані інтерфейси libpq та ECPG, які забезпечують можливість підключення програм до бази даних. Крім цього, існує значна кількість сторонніх бібліотек і драйверів, що дозволяють інтегрувати

PostgreSQL із різними мовами програмування, зокрема Java, C++, Python, Ruby, Node.js та іншими.

Популярність PostgreSQL пояснюється низкою важливих переваг. Насамперед система є безкоштовною та має відкритий вихідний код, що дає змогу вільно використовувати її в проєктах різного масштабу. PostgreSQL підтримує роботу з багатьма мовами програмування, що робить його універсальним рішенням для розробки програмного забезпечення.

Ще однією важливою характеристикою є висока масштабованість системи, яка дозволяє ефективно працювати як із невеликими застосунками, так і з великими інформаційними системами. PostgreSQL також забезпечує підтримку цілісності даних завдяки механізмам транзакцій, обмежень і контролю доступу. Додатково система створює надійне та стабільне середовище для зберігання й обробки інформації, що робить її популярним вибором для сучасних вебзастосунків і корпоративних систем.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ВЕБЗАСТОСУНКУ ДЛЯ КООРДИНАЦІЇ ВОЛОНТЕРСЬКОЇ ДІЯЛЬНОСТІ ТА ПІДТРИМКИ ВНУТРІШНЬО ПЕРЕМІЩЕНИХ ОСІБ

Розробка плану створення вебзастосунку є важливим етапом, який дозволяє організувати процес роботи та забезпечити досягнення поставлених цілей. Такий план допомагає визначити основні напрями розробки, зрозуміти функціональність майбутньої системи та узгодити процес створення застосунку.

Перш за все, планування дозволяє чітко сформулювати мету та основні завдання вебзастосунку, що допомагає розробнику краще зрозуміти потреби користувачів і визначити функціональні та нефункціональні вимоги до системи. Крім цього, планування сприяє правильному розподілу ресурсів, зокрема часу, програмних засобів та інструментів розробки.

Важливою складовою є керування термінами виконання робіт. Наявність чіткого плану дозволяє визначити послідовність етапів розробки, встановити пріоритети та контролювати виконання поставлених завдань. Також це дає змогу своєчасно виявляти можливі проблеми та мінімізувати ризики під час створення вебзастосунку.

Процес створення вебзастосунку складається з кількох основних етапів. На початковому етапі виконується аналіз вимог, визначаються цілі системи та потреби користувачів. Після цього здійснюється проєктування структури застосунку, бази даних та інтерфейсу користувача.

Наступним етапом є розробка функціональності вебзастосунку. На цьому етапі налаштовується робоче середовище, реалізуються клієнтська та серверна частини системи, а також забезпечується взаємодія з базою даних і зовнішніми сервісами.

Після завершення основної розробки проводиться тестування застосунку, зокрема перевірка коректності роботи інтерфейсу користувача та

основних функцій системи. Далі вебзастосунок розгортається на сервері або хостингу, виконується налаштування середовища та забезпечується захист даних.

Після впровадження система потребує подальшого супроводу та підтримки. Це включає виправлення помилок, оновлення функціональності, удосконалення роботи системи та підвищення рівня безпеки.

Методологія Agile є одним із найпоширеніших підходів до розробки сучасних вебзастосунків. Вона базується на принципах гнучкості, поетапної розробки та постійної взаємодії між командою розробників і замовником. Основною особливістю Agile є поділ процесу створення програмного забезпечення на короткі ітерації (спринти), під час яких реалізується певний набір функціональних можливостей. Після завершення кожного спринту проводяться оцінка отриманих результатів, збір зворотного зв’язку та визначення подальших кроків розвитку проєкту.

Завдяки такому підходу команда може оперативно реагувати на зміни у вимогах, виправляти виявлені недоліки та поступово вдосконалювати функціональність вебзастосунку. Це особливо важливо для проєктів, де вимоги можуть змінюватися в процесі розробки або виникає потреба швидко адаптувати продукт до нових умов ринку чи побажань користувачів. Agile також сприяє підвищенню якості програмного забезпечення через регулярне тестування, постійний контроль виконаних завдань і тісну співпрацю всіх учасників проєкту.

Структура вебзастосунку визначає спосіб організації його компонентів, взаємозв’язки між окремими модулями та принципи розподілу функціональності. Зазвичай сучасні вебзастосунки будуються за багаторівневою архітектурою, яка включає клієнтську частину (Frontend), серверну частину (Backend) та базу даних. Клієнтська частина відповідає за взаємодію з користувачем, відображення інформації та обробку подій інтерфейсу. Серверна частина забезпечує обробку бізнес-логіки, виконання запитів, керування даними та взаємодію з іншими сервісами. База даних

використовується для зберігання, оновлення та отримання інформації, необхідної для роботи системи.

Правильно спроєктована структура вебзастосунку забезпечує масштабованість, підтримуваність і надійність системи. Розподіл функціональності між окремими модулями дозволяє спростити процес розробки, тестування та подальшого супроводу програмного забезпечення. Крім того, модульний підхід полегшує внесення змін і додавання нових можливостей без значного впливу на вже реалізовані компоненти системи. Поєднання Agile-методології та продуманої архітектури створює умови для ефективної розробки вебзастосунків, що відповідають сучасним вимогам до якості, продуктивності та зручності використання.

3.1 Розробка дизайну інтерфейсу користувача

Під час розробки дизайну інтерфейсу основна увага приділялася забезпеченню максимальної зручності, доступності та зрозумілості вебзастосунку для широкого кола користувачів. Особливий акцент було зроблено на створенні інтерфейсу, який буде однаково комфортним для людей різних вікових категорій, зокрема дітей, дорослих і людей похилого віку. Для досягнення цієї мети було використано просту та логічну структуру навігації, зрозумілі елементи керування, контрастне оформлення та читабельні шрифти, що сприяють швидкому освоєнню функціоналу без потреби в тривалому навчанні.

Інтерфейс вебзастосунку було спроєктовано відповідно до принципів інтуїтивного користувацького досвіду (User Experience, UX), завдяки чому користувач може легко знаходити необхідні функції та виконувати потрібні дії з мінімальною кількістю кроків. Усі елементи керування розміщені з урахуванням сучасних рекомендацій щодо ергономіки та зручності взаємодії, що дозволяє зменшити ймовірність помилок під час роботи із системою.

Окрім цього, важливою складовою процесу розробки стала адаптивність інтерфейсу. Вебзастосунок коректно працює та відображається на різних типах пристроїв, зокрема на персональних комп’ютерах, ноутбуках, планшетах та смартфонах. Для цього було застосовано адаптивну верстку, яка автоматично підлаштовує розміщення та розміри елементів інтерфейсу відповідно до параметрів екрана користувача.

Особлива увага приділялася забезпеченню коректної роботи вебзастосунку на мобільних пристроях із невеликим розширенням екрана та обмеженими технічними характеристиками. Під час розробки враховувалися особливості застарілих моделей смартфонів, що дозволило зберегти зручність користування навіть за умов невисокої продуктивності пристрою. У результаті було створено універсальний інтерфейс, який забезпечує стабільну роботу, комфортне сприйняття інформації та позитивний користувацький досвід незалежно від типу пристрою та рівня технічної підготовки користувача. На основі наведених вимог і характеристик було розроблено макет вебзастосунку (рис. 3.1).

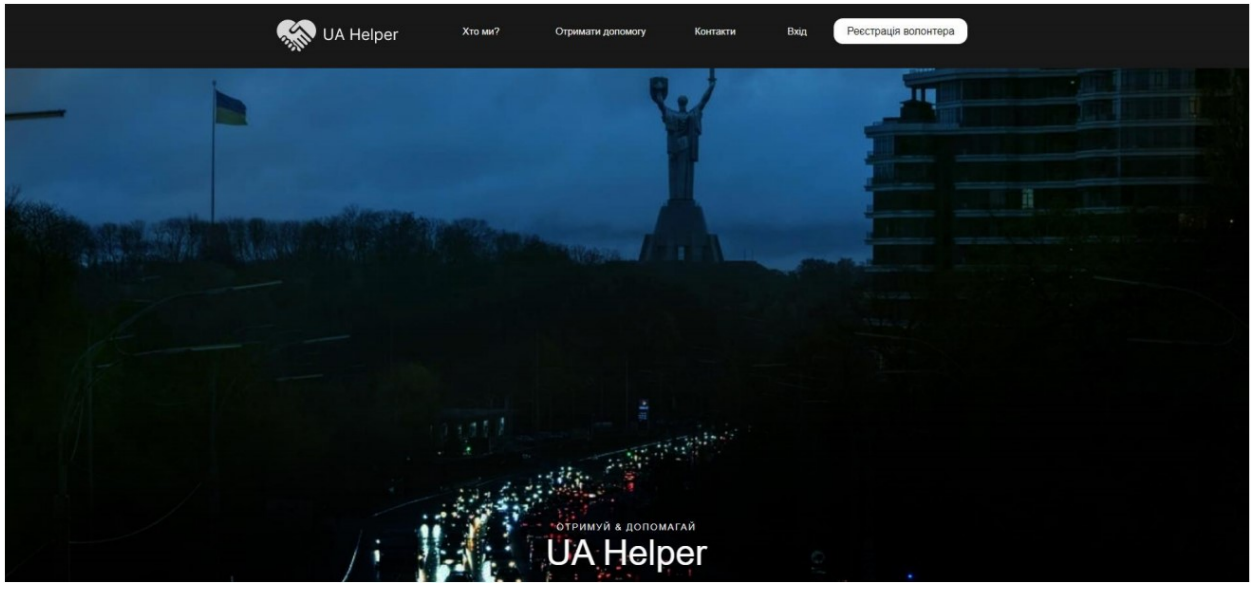

Рисунок 3.1 – Головна сторінка вебзастосунку

Відповідно до розробленої концепції вебзастосунку, головна сторінка містить інформаційну стрічку новин, яка забезпечує користувачів

актуальними матеріалами та корисною інформацією. Стрічка новин реалізована у вигляді набору інформаційних карток із коротким описом, зображенням та можливістю переходу до повного матеріалу. Такий підхід дозволяє користувачам швидко ознайомлюватися з найважливішими новинами та подіями без необхідності шукати інформацію на сторонніх ресурсах.

Розміщення новинної стрічки на головній сторінці сприяє підвищенню інформованості користувачів і забезпечує зручний доступ до актуального контенту. Інформаційні матеріали можуть оновлюватися автоматично, що дозволяє підтримувати актуальність відображених даних. Для покращення сприйняття контенту новини впорядковуються за датою публікації або рівнем популярності, а їх відображення адаптується до різних типів пристроїв.

Завдяки використанню сучасних засобів веброзробки стрічка новин забезпечує швидке завантаження матеріалів та зручну навігацію між ними. Зовнішній вигляд сторінки зі стрічкою новин наведено на рисунку 3.2.

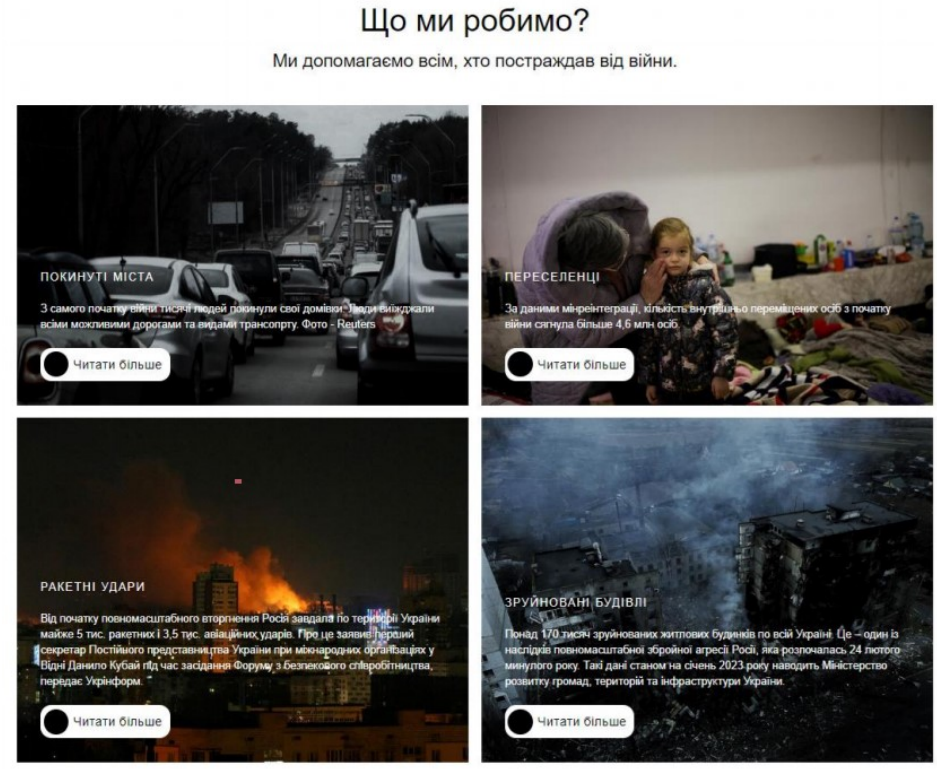

Рисунок 3.2 – Сторінка з новинами

Крім цього, у вебзастосунку передбачено окремий розділ, який містить загальну інформацію щодо підтримки внутрішньо переміщених осіб та громадян, які були змушені залишити свої домівки внаслідок повномасштабного вторгнення (рис. 3.3).

Рисунок 3.3 – Загальна інформація про допомогу для ВПО

Даний розділ створено з метою забезпечення швидкого доступу користувачів до актуальної та корисної інформації, необхідної для адаптації до нових умов проживання. У ньому зібрано відомості про можливі види державної та гуманітарної допомоги, порядок отримання соціальних виплат, особливості реєстрації статусу внутрішньо переміщеної особи, а також контакти організацій і сервісів, що надають консультаційну та правову підтримку.

Інформація структурована у зручному для сприйняття форматі та представлена таким чином, щоб користувач міг швидко знайти необхідні відомості без зайвих витрат часу [34-37]. Завдяки цьому розділ виконує не лише інформаційну, а й соціально важливу функцію, сприяючи підвищенню обізнаності громадян щодо доступних механізмів підтримки та допомоги в умовах воєнного стану.

На сторінці вебзастосунку передбачено окремий розділ, присвячений благодійним організаціям та їхнім партнерам. Цей функціональний модуль надає користувачам можливість отримати детальну інформацію про організації, які надають гуманітарну, соціальну, медичну та іншу допомогу. Для кожної організації відображаються основні відомості, зокрема назва, опис діяльності, напрями роботи, контактна інформація та доступні способи взаємодії.

Наявність такого розділу дозволяє користувачам швидко знаходити перевірені організації та звертатися до них за необхідною підтримкою. Крім того, інформація про партнерів сприяє підвищенню рівня довіри до системи та демонструє масштаби співпраці між волонтерськими, благодійними та громадськими організаціями. Відображення партнерських організацій також допомагає координувати діяльність учасників системи та уникати дублювання наданої допомоги.

Інтерфейс сторінки реалізовано таким чином, щоб користувач міг швидко переглядати відомості про організації, переходити до детальної інформації та отримувати актуальні контактні дані. Детальна інформація про благодійні організації та їхніх партнерів наведена на рисунку 3.4.

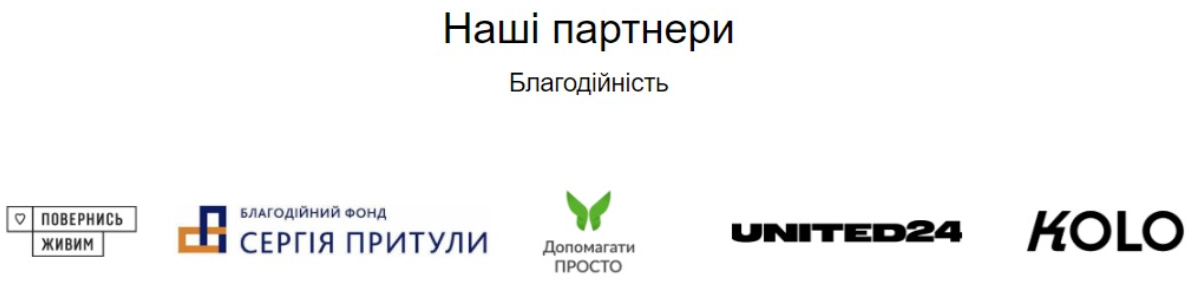

Рисунок 3.4 – Інформація про благодійні організації

Дизайн сторінки переходу до форми отримання волонтерської допомоги було розроблено з урахуванням простоти використання, доступності та зрозумілості для користувачів різного віку й рівня цифрової

грамотності (рис. 3.5). Основною метою цієї сторінки є швидке та зручне перенаправлення користувача до форми заповнення заявки на отримання необхідної допомоги.

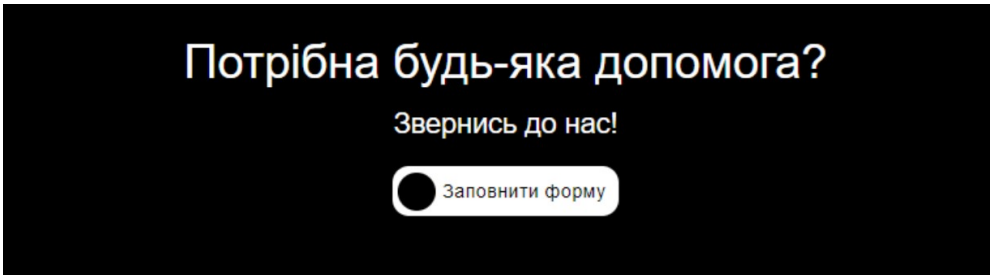

Рисунок 3.5 – Перехід на заповнення форми для отримання допомоги

Інтерфейс сторінки побудований таким чином, щоб користувач міг без труднощів знайти необхідну інформацію та перейти до заповнення форми. Для цього було використано мінімалістичний дизайн, зрозумілі елементи навігації та контрастні кнопки переходу. Особлива увага приділялася читабельності тексту, розміру елементів керування та логічному розташуванню блоків інформації.

Також у вебзастосунку реалізовано функціонал інтерактивної мапи, який забезпечує користувачам зручний доступ до інформації про розташування волонтерських штабів, пунктів видачі гуманітарної допомоги, центрів підтримки внутрішньо переміщених осіб та інших важливих об’єктів соціальної інфраструктури. Даний модуль розроблений із використанням сучасних вебтехнологій та коректно функціонує на різних типах пристроїв, зокрема персональних комп’ютерах, ноутбуках, планшетах і смартфонах, що забезпечує його доступність для широкого кола користувачів.

Основним призначенням інтерактивної мапи є спрощення процесу пошуку найближчих пунктів допомоги та отримання актуальної інформації про їхнє місцезнаходження. Користувачі можуть переглядати об’єкти безпосередньо на карті, отримувати відомості про їхню адресу, контактні дані, графік роботи та перелік доступних послуг. Це дозволяє швидко

знаходити необхідні ресурси та ефективніше планувати звернення по допомогу.

Для підвищення зручності використання передбачено можливість масштабування карти, переміщення між різними регіонами та перегляду детальної інформації про кожен пункт допомоги за допомогою спеціальних інформаційних маркерів. Завдяки інтеграції картографічних сервісів користувачі можуть отримувати актуальні географічні дані та орієнтуватися щодо розташування необхідних об’єктів у режимі реального часу.

Використання інтерактивної мапи значно підвищує ефективність взаємодії між внутрішньо переміщеними особами, волонтерами та благодійними організаціями, забезпечуючи швидкий доступ до важливої інформації та сприяючи оперативному отриманню необхідної допомоги. Зовнішній вигляд сторінки з інтерактивною мапою наведено на рисунку 3.6.

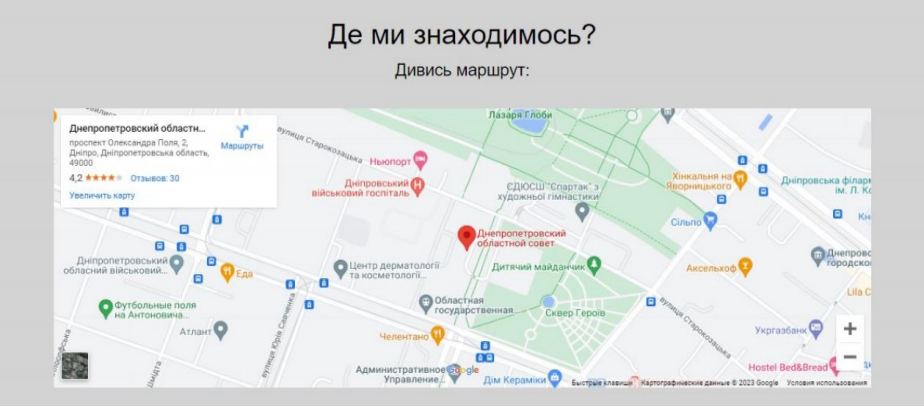

Рисунок 3.6 – Мапа волонтерських штабів

3.2 Реалізація системи авторизації

Реалізація системи авторизації є важливим етапом розробки вебзастосунку, оскільки вона забезпечує контроль доступу до функціональності системи та захист персональних даних користувачів.

Основною метою авторизації є перевірка прав доступу користувача після проходження процесу автентифікації.

У вебзастосунку система авторизації реалізована таким чином, щоб забезпечити простий і безпечний доступ до функцій системи. Після введення користувачем облікових даних виконується перевірка правильності логіна та пароля. У разі успішної перевірки користувач отримує доступ до відповідних розділів і можливостей вебзастосунку залежно від своєї ролі та рівня доступу.

Для підвищення безпеки під час реалізації авторизації використовуються механізми шифрування паролів і захисту сесій користувачів, що дозволяють мінімізувати ризики несанкціонованого доступу до системи та забезпечити конфіденційність інформації.

Інтерфейс сторінки авторизації було розроблено максимально простим і зрозумілим для користувачів. Форма входу містить лише необхідні поля для введення даних, що спрощує процес взаємодії із системою. Також передбачено обробку помилок введення та відображення відповідних повідомлень у разі некоректного введення даних.

Система авторизації вебзастосунку розроблена з урахуванням вимог щодо зручності використання, безпеки та доступності на різних типах пристроїв. Інтерфейс сторінки входу є адаптивним та коректно відображається на персональних комп’ютерах, ноутбуках, планшетах і смартфонах. Це забезпечує користувачам можливість отримувати доступ до функціоналу системи незалежно від пристрою, який вони використовують, та місця перебування.

Під час проєктування механізму авторизації особлива увага приділялася захисту персональних даних користувачів і контролю доступу до інформаційних ресурсів системи. Для входу до облікового запису користувач повинен пройти процедуру автентифікації шляхом введення реєстраційних даних, після чого система перевіряє їхню коректність та надає відповідний рівень доступу.

Логіка роботи вебзастосунку передбачає, що можливість авторизації та подальшого доступу до службового функціоналу надається виключно користувачам із підтвердженим статусом волонтера. Такий підхід дозволяє забезпечити контрольований доступ до інформації про заявки, отримувачів допомоги та інших внутрішніх ресурсів системи. Водночас особи, які потребують допомоги, можуть користуватися відкритими функціями вебзастосунку без отримання доступу до службових інструментів волонтерів.

Запровадження рольової моделі доступу сприяє підвищенню рівня безпеки системи, запобігає несанкціонованому перегляду конфіденційної інформації та забезпечує захист персональних даних користувачів. Крім того, такий механізм дозволяє чітко розмежувати функціональні можливості різних категорій користувачів відповідно до їхніх повноважень та завдань у межах вебзастосунку. Інтерфейс сторінки авторизації наведено на рисунку 3.7.

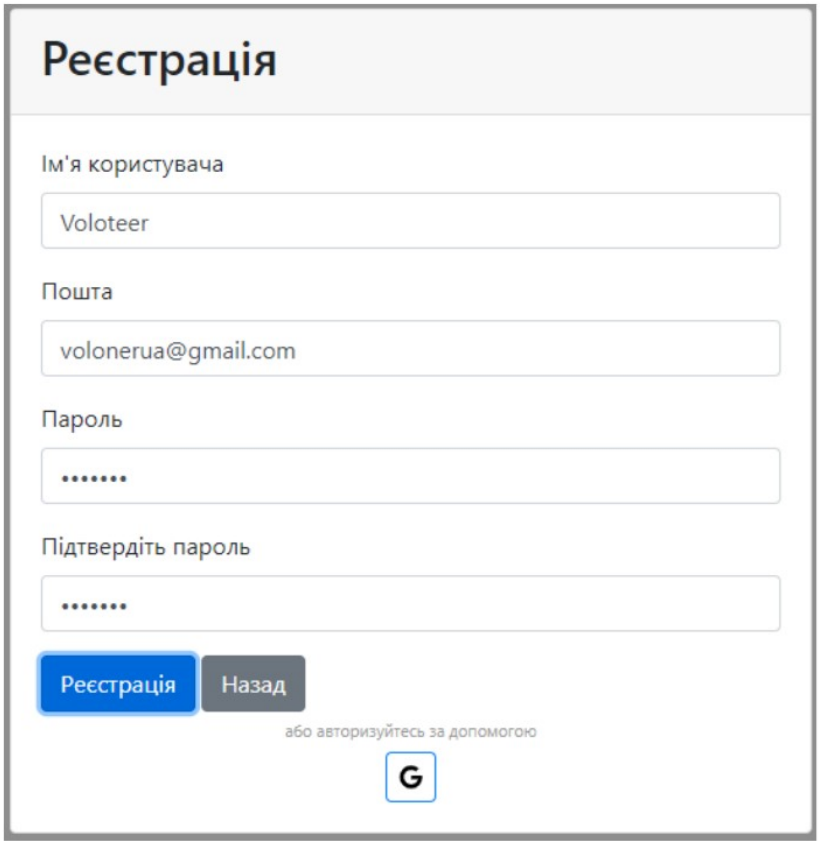

Рисунок 3.7 – Реєстрація волонтера

Перед отриманням доступу до системи користувач повинен пройти процедуру реєстрації, під час якої необхідно вказати основні персональні дані та інформацію, що підтверджує його волонтерську діяльність. Після заповнення реєстраційної форми всі введені дані автоматично надсилаються адміністратору вебзастосунку для подальшої перевірки.

Адміністратор перевіряє достовірність наданої інформації та ухвалює рішення щодо надання користувачу статусу волонтера. У разі успішного підтвердження облікового запису користувач отримує можливість авторизуватися в системі та отримати доступ до функціоналу перегляду заявок від людей, які потребують допомоги.

Такий механізм дозволяє забезпечити контроль доступу до конфіденційної інформації, мінімізувати ризик несанкціонованого використання системи та підвищити рівень безпеки вебзастосунку. Крім цього, перевірка даних адміністратором сприяє формуванню більш надійного середовища для взаємодії між волонтерами та користувачами, які звертаються по підтримку.

Після заповнення та надсилання реєстраційної форми волонтера система перевіряє введені дані та створює заявку на реєстрацію нового користувача. У разі успішного завершення цієї процедури користувачу відображається інформаційне повідомлення, яке підтверджує прийняття заявки до обробки. Такий механізм забезпечує зворотний зв’язок із користувачем та інформує його про коректне виконання операції реєстрації.

У повідомленні зазначається, що заявка успішно зареєстрована в системі та передана для подальшої обробки. Крім того, користувачу автоматично присвоюється унікальний ідентифікатор, який використовується для однозначної ідентифікації його облікового запису та пов’язаних із ним даних у базі даних вебзастосунку. Наявність такого ідентифікатора дозволяє забезпечити коректне зберігання інформації, спростити адміністрування системи та організувати подальшу взаємодію між користувачем і програмним продуктом.

Після успішної реєстрації заявку можна передати на етап перевірки та підтвердження статусу волонтера. Лише після завершення процедури верифікації користувач отримує можливість авторизуватися в системі та використовувати функціональні можливості, передбачені для волонтерів. Такий підхід сприяє підвищенню рівня безпеки вебзастосунку та забезпечує достовірність інформації про зареєстрованих користувачів.

Інтерфейс повідомлення про успішне створення заявки на реєстрацію волонтера наведений на рисунку 3.8.

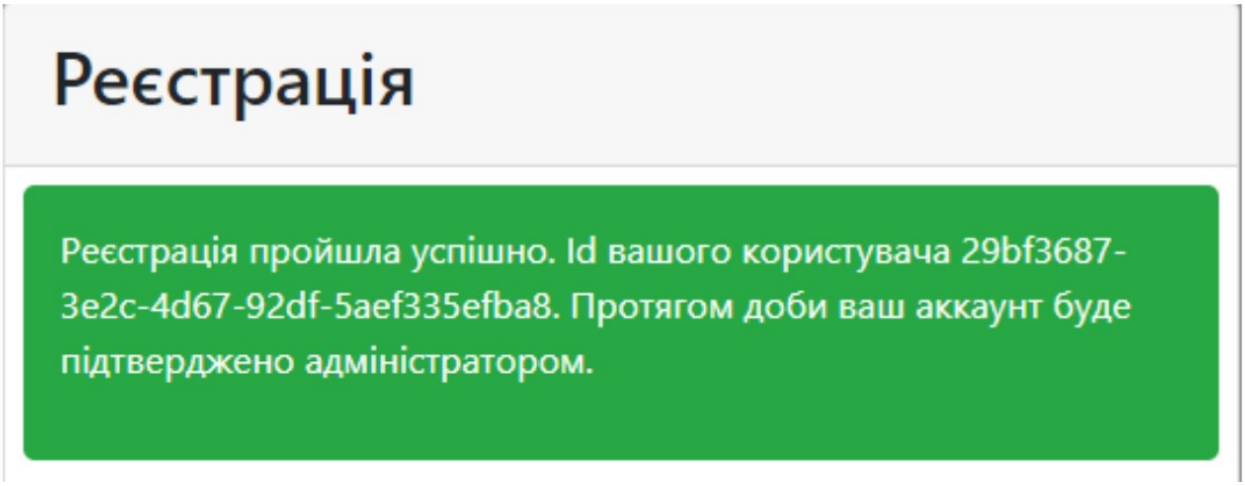

Рисунок 3.8 – Інформаційне повідомлення після успішного заповнення та надсилання форми реєстрації волонтера

Також користувач отримує повідомлення про те, що надані ним дані будуть перевірені адміністратором вебзастосунку. Після успішної перевірки обліковий запис волонтера буде підтверджено, а доступ до функціональності системи стане можливим. Орієнтовний час обробки заявки та підтвердження сторінки становить до однієї доби.

Такий механізм дозволяє забезпечити додатковий рівень безпеки системи, перевірити достовірність введених даних та обмежити доступ сторонніх осіб до інформації про заявки користувачів, які потребують допомоги. Крім цього, використання унікального ідентифікатора спрощує процес адміністрування та подальшу взаємодію з користувачами системи.

Після проходження перевірки та підтвердження облікового запису адміністратором вебзастосунку волонтер отримує доступ до сторінки зі списком усіх заявок від людей, які потребують допомоги (рис. 3.9). Цей функціонал дозволяє волонтерам оперативно переглядати нові звернення, аналізувати потреби користувачів та організовувати процес надання допомоги.

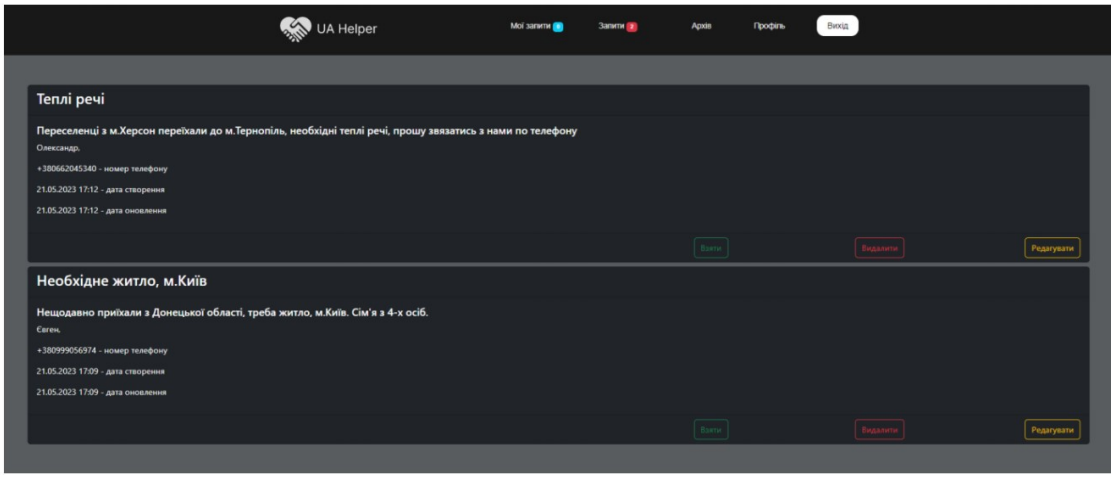

Рисунок 3.9 – Робоча сторінка волонтера

На сторінці відображається перелік заявок із основною інформацією про користувачів та описом необхідної допомоги. Для покращення координації між волонтерами реалізовано можливість взяти заявку в роботу. Після цього інші волонтери можуть бачити, що конкретним зверненням уже займається відповідальна особа, що дозволяє уникнути дублювання роботи та підвищує ефективність взаємодії між учасниками системи.

Також у вебзастосунку реалізовано функціонал редагування заявок, який забезпечує можливість внесення змін до раніше створених записів. Необхідність використання цього механізму може виникати у випадках, коли під час первинного заповнення заявки користувач не вказав усі необхідні відомості, припустився помилки або коли згодом з’явилася потреба уточнити інформацію щодо необхідної допомоги, контактних даних чи інших важливих параметрів звернення.

Можливість редагування дозволяє підтримувати актуальність інформації, що зберігається в системі, та забезпечує більш якісну взаємодію між користувачами й волонтерами. Після внесення змін оновлені дані автоматично зберігаються в базі даних і стають доступними для подальшого перегляду та опрацювання. Це сприяє підвищенню достовірності інформації та зменшує ймовірність виникнення непорозумінь під час виконання заявки.

Для волонтерів функція редагування є важливим інструментом під час обробки звернень, оскільки дозволяє оперативно уточнювати інформацію, отриману від користувачів, та підтримувати її в актуальному стані. Завдяки цьому забезпечується більш точне визначення потреб заявника, ефективніше планування ресурсів та підвищується якість надання допомоги.

Реалізація механізму редагування заявок підвищує гнучкість роботи системи, сприяє підтриманню актуальності даних та забезпечує більш ефективний процес опрацювання звернень на всіх етапах їхнього виконання.

Інтерфейс сторінки розроблено таким чином, щоб забезпечити швидкий доступ до основних функцій, зручну навігацію та комфортну роботу з великою кількістю заявок, що значно спрощує процес взаємодії між волонтерами та людьми, які потребують підтримки.

Крім цього, у вебзастосунку реалізовано можливість перегляду вже опрацьованих заявок, які автоматично переміщуються до архіву після завершення роботи з ними. Такий функціонал дозволяє зберігати історію звернень, здійснювати повторний перегляд інформації та контролювати виконані запити користувачів.

Архів заявок спрощує процес систематизації даних і дає можливість волонтерам швидко знаходити раніше оброблені звернення у разі потреби. Це також дозволяє аналізувати ефективність роботи системи та вести облік наданої допомоги.

Окрім роботи із заявками, користувачам із роллю волонтера доступна функція редагування власного профілю (рис. 3.10). Волонтер може змінювати особисту інформацію, оновлювати контактні дані та редагувати

окремі параметри облікового запису. Така можливість забезпечує актуальність інформації в системі та покращує комунікацію між учасниками вебзастосунку.

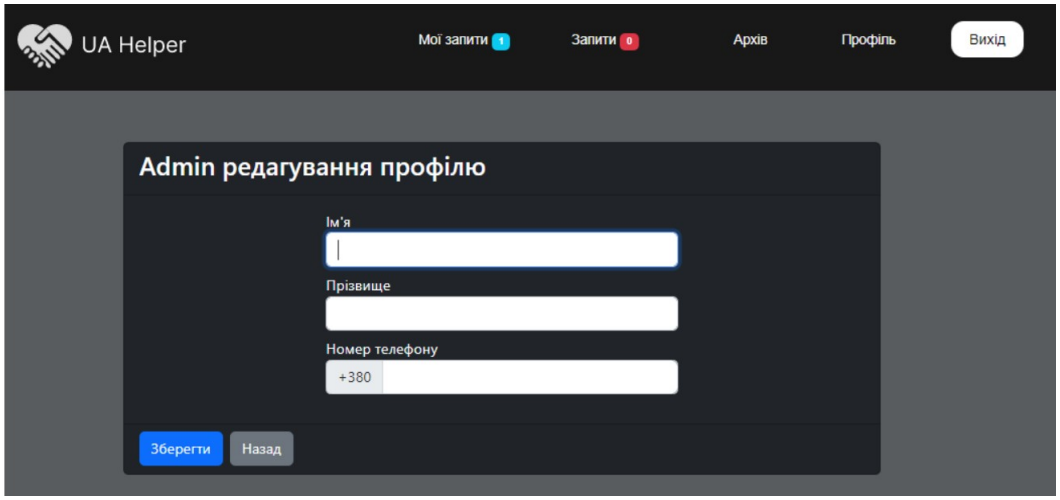

Рисунок 3.10 – Форма редагування профілю волонтера

Інтерфейс сторінки архіву та налаштувань профілю розроблений із урахуванням принципів зручності використання, інтуїтивної навігації та швидкого доступу до основних функцій вебзастосунку. Головною метою даної сторінки є надання користувачу можливості переглядати історію власної активності в системі, а також керувати персональними налаштуваннями облікового запису.

Сторінка архіву містить перелік раніше створених та оброблених заявок із можливістю перегляду їхнього поточного статусу, дати створення та детальної інформації. Такий підхід дозволяє користувачам відстежувати історію взаємодії із системою та контролювати виконання поданих звернень. Для підвищення зручності передбачено механізми пошуку та фільтрації записів, що значно спрощують роботу з великою кількістю даних.

Розділ налаштувань профілю забезпечує можливість редагування особистої інформації користувача, зміни контактних даних, керування параметрами облікового запису та налаштуваннями безпеки. Усі елементи

інтерфейсу згруповані за функціональним призначенням, що забезпечує логічну структуру сторінки та спрощує взаємодію користувача із системою.

Особлива увага під час проєктування інтерфейсу була приділена адаптивності та доступності. Сторінка коректно відображається на різних типах пристроїв, включаючи персональні комп'ютери, планшети та смартфони, що забезпечує комфортне використання вебзастосунку незалежно від роздільної здатності екрана.

Код реалізації системи авторизації серверної частини вебзастосунку, розроблений мовою програмування C#, наведений на рисунку 3.11.

```
public async Task<ApplicationUser> AutoProvisionUser(string provider, string userId,
List<Claim> claims)

    {
        // create a list of claims that we want to transfer into our store
        var filtered = new List<Claim>();
        foreach (var claim in claims)
            // if the external system sends a display name - translate that to the
standard OIDC name claim
            if (claim.Type == ClaimTypes.Name)
                filtered.Add(new Claim(JwtClaimTypes.Name, claim.Value));
            // if the JWT handler has an outbound mapping to an OIDC claim use that
            // copy the claim as-is
            else
                filtered.Add(claim);

        // if no display name was provided, try to construct by first and/or last name
        if (filtered.All(x => x.Type != JwtClaimTypes.Name))
        {
            var first = filtered.FirstOrDefault(x => x.Type ==
JwtClaimTypes.GivenName)?.Value;
            var last = filtered.FirstOrDefault(x => x.Type ==
JwtClaimTypes.FamilyName)?.Value;
            if (first != null && last != null)
                filtered.Add(new Claim(JwtClaimTypes.Name, first + " " + last));
            else if (first != null)
                filtered.Add(new Claim(JwtClaimTypes.Name, first));
            else if (last != null) filtered.Add(new Claim(JwtClaimTypes.Name, last));
        }

        // check if a display name is available, otherwise fallback to subject id
        var name = filtered.FirstOrDefault(c => c.Type == ClaimTypes.GivenName)?.Value
?? userId;
        var email = filtered.FirstOrDefault(c => c.Type == ClaimTypes.Email)?.Value;
        var lastName = filtered.FirstOrDefault(c => c.Type ==
ClaimTypes.Surname)?.Value;
```

Рисунок 3.11 – Код реалізації авторизації бек-енд частини мовою C#

Реалізований механізм забезпечує перевірку облікових даних користувача, генерацію токенів автентифікації, керування сесіями та контроль доступу до захищених ресурсів системи. Завдяки використанню сучасних механізмів автентифікації забезпечується належний рівень безпеки персональних даних користувачів та захист від несанкціонованого доступу.

3.3 Реалізація взаємодії волонтера та переселенця

Реалізація взаємодії волонтера та переселенця є одним із ключових функціональних елементів вебзастосунку, оскільки саме вона забезпечує ефективний процес надання допомоги та координації дій між користувачами системи. Основною метою даного механізму є організація швидкого та зручного обміну інформацією між особами, які потребують підтримки, та волонтерами, готовими її надати.

У вебзастосунку взаємодія реалізована за допомогою системи заявок на допомогу. Переселенець після авторизації в системі має можливість створити заявку, в якій зазначає необхідну інформацію щодо своїх потреб. Залежно від типу допомоги користувач може вказати категорію звернення, описати ситуацію, зазначити контактні дані для зв’язку та додати додаткову інформацію, що може бути корисною для волонтера під час опрацювання заявки.

Після заповнення форми та підтвердження створення заявки всі введені дані зберігаються в базі даних вебзастосунку. Кожній заявці автоматично присвоюється унікальний ідентифікатор, що дозволяє відстежувати її стан протягом усього життєвого циклу. Створена заявка стає доступною для перегляду волонтерами, які мають відповідні права доступу в системі.

Волонтери можуть переглядати список активних заявок, здійснювати пошук за різними параметрами та обирати звернення, які відповідають їхнім можливостям і напрямкам діяльності. Для кожної заявки відображається детальна інформація про потреби користувача, що дозволяє оцінити складність завдання та прийняти рішення щодо його виконання. Після вибору заявки волонтер може взяти її в роботу, що автоматично змінює її статус та інформує інших волонтерів про те, що звернення вже опрацьовується.

Для забезпечення прозорості процесу надання допомоги система підтримує відстеження статусів заявок. На різних етапах обробки заявка

може перебувати у станах «створена», «в роботі», «виконана» або «скасована». Зміна статусу відображається як для волонтера, так і для переселенця, що дозволяє обом сторонам контролювати процес виконання звернення.

Додатково система забезпечує можливість обміну повідомленнями між учасниками взаємодії. Це дозволяє уточнювати деталі заявки, повідомляти про хід її виконання та оперативно вирішувати питання, які можуть виникнути під час надання допомоги. Використання такого підходу значно підвищує ефективність комунікації та сприяє швидшому задоволенню потреб переселенців.

Таким чином, реалізований механізм взаємодії волонтера та переселенця забезпечує централізоване управління заявками, зручний обмін інформацією між користувачами та контроль за процесом надання допомоги, що підвищує ефективність роботи всієї системи.Волонтер, у свою чергу, має можливість переглядати список доступних заявок, обирати конкретне звернення та брати його в роботу. Після цього заявка закріплюється за певним волонтером, що дозволяє уникнути дублювання дій і забезпечує прозорість процесу надання допомоги.

У процесі опрацювання заявки волонтер може уточнювати інформацію, редагувати окремі поля та фіксувати результати виконаних дій. Це дозволяє підтримувати актуальність даних і забезпечує якісну комунікацію між учасниками системи.

Після завершення роботи із заявкою вона переводиться до архіву, де зберігається для подальшого перегляду та аналізу. Такий підхід забезпечує повний цикл взаємодії між переселенцями та волонтерами, а також дає змогу контролювати ефективність надання допомоги.

Для реалізації взаємодії між користувачами було створено спрощену форму подання заявки, яка дозволяє максимально швидко залишити звернення на отримання допомоги. Особливістю даного підходу є відсутність

необхідності попередньої реєстрації, а також підтвердження облікового запису через електронну пошту чи номер телефону.

Форма створення заявки розроблена з урахуванням принципів простоти, доступності та зручності використання. Основною метою під час її проєктування було мінімізувати час, необхідний для заповнення, та забезпечити можливість швидкого подання звернення користувачами незалежно від їхнього рівня володіння цифровими технологіями. Особливу увагу приділено спрощенню процесу введення даних, що є важливим для осіб, які перебувають у складних життєвих обставинах та потребують оперативної допомоги.

Форма містить лише мінімально необхідний набір полів для формування заявки. Користувачеві необхідно вказати заголовок звернення, який коротко відображає суть проблеми, надати опис ситуації для детальнішого розуміння потреб, а також зазначити власне ім’я та контактний номер телефону для подальшого зв’язку. Така структура дозволяє зібрати достатній обсяг інформації для початкової обробки звернення без перевантаження користувача надмірною кількістю полів.

Після заповнення форми введені дані передаються на серверну частину вебзастосунку, де проходять перевірку та зберігаються в базі даних. Створена заявка стає доступною для перегляду волонтерами, які можуть ознайомитися з її змістом, оцінити потреби користувача та розпочати процес надання необхідної допомоги. Завдяки лаконічній структурі форми забезпечується швидке надходження звернень до системи та скорочується час між поданням заявки й початком її опрацювання.

Реалізований підхід дозволяє зробити процес створення заявки максимально зрозумілим та доступним для всіх категорій користувачів, підвищуючи ефективність взаємодії між особами, які потребують допомоги, та волонтерами. Інтерфейс форми створення заявки наведено на рисунку 3.12.

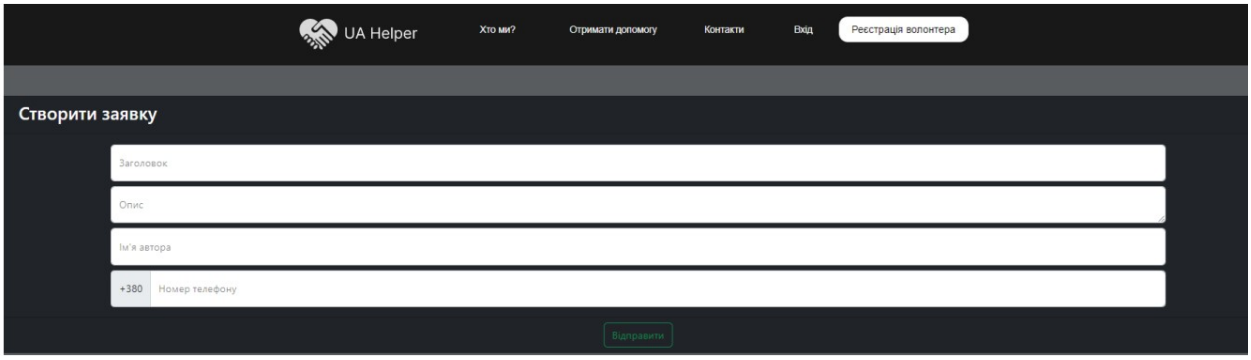

Рисунок 3.12 – Форма заповнення заявки для ВПО

Завдяки спрощеному інтерфейсу користувач може оперативно залишити запит навіть у стресових або обмежених за часом умовах. Це особливо важливо у випадках, коли необхідно швидко отримати допомогу без додаткових процедур авторизації.

Окрім цього, форма адаптована для використання на різних пристроях, що забезпечує зручність її заповнення як на комп'ютерах, так і на мобільних телефонах. Такий підхід підвищує доступність сервісу та сприяє швидкому реагуванню на запити користувачів.

З метою зменшення кількості некоректно заповнених заявок та уникнення невалідних повідомлень у вебзастосунку було реалізовано систему валідації форми. Вона забезпечує перевірку введених користувачем даних у режимі реального часу та допомагає уникнути помилок під час заповнення (рис. 3.13).

Рисунок 3.13 – Валідація полів в формі створення заявки ВПО

Механізм валідації інформує користувача про правильність введених даних за допомогою підказок і повідомлень про помилки. У разі некоректного заповнення окремих полів система виділяє їх і надає рекомендації щодо виправлення, що значно спрощує процес подання заявки та підвищує її якість.

Особлива увага приділялася перевірці обов’язкових полів, таких як ім’я користувача, контактний номер телефону, заголовок і опис звернення. Це дозволяє забезпечити повноту та достовірність отриманої інформації, яка надалі використовується волонтерами для опрацювання заявок.

Завдяки впровадженій валідації підвищується зручність використання вебзастосунку, зменшується кількість помилкових або неповних заявок, а також покращується загальна ефективність роботи системи взаємодії між користувачами.

3.4 Тестування вебзастосунку

Одним із ключових етапів розробки вебзастосунку було забезпечення його адаптивності. Адаптивність вебзастосунку – це здатність інтерфейсу коректно відображатися та стабільно функціонувати на пристроях із різними розмірами екранів, зокрема на персональних комп’ютерах, ноутбуках, планшетах і смартфонах. Основною метою адаптивного дизайну є забезпечення зручного користувацького досвіду незалежно від типу пристрою.

Важливість адаптивності обумовлена кількома факторами. Насамперед, це підвищення зручності для користувачів, оскільки інтерфейс автоматично підлаштовується під розмір екрана, що дозволяє комфортно переглядати контент і взаємодіяти з функціоналом системи. Це забезпечує доступність вебзастосунку для широкого кола користувачів.

Крім того, адаптивний дизайн позитивно впливає на пошукову оптимізацію, оскільки сучасні пошукові системи надають перевагу сайтам, які коректно працюють на мобільних пристроях. Це підвищує видимість вебзастосунку в результатах пошуку та сприяє залученню більшої кількості користувачів.

Ще однією перевагою є економія часу та ресурсів на розробку, оскільки замість створення окремих версій для різних пристроїв використовується єдиний адаптивний інтерфейс, який автоматично підлаштовується під доступний простір екрана.

Також адаптивність дозволяє розширити аудиторію користувачів, оскільки забезпечує можливість використання вебзастосунку з будь-якого пристрою, підключеного до Інтернету.

У процесі тестування було перевірено роботу вебзастосунку на різних пристроях із різними розмірами екранів. Зокрема, під час відкриття сторінки на пристрої Apple iPhone 12 Pro Max (рис. 3.14) було підтверджено, що шапка сайту та основні елементи інтерфейсу коректно адаптуються до мобільної версії, забезпечуючи зручну навігацію та читабельність.

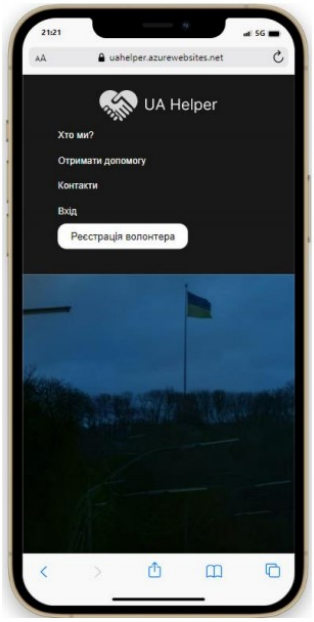

Рисунок 3.14 – Тестування застосунку на мобільній версії (Apple Iphone 12 PRO MAX)

Оскільки вебзастосунок коректно відображається на попередньому етапі тестування, а всі блоки, шрифти, зображення та стилі працюють належним чином і не порушують структуру інтерфейсу, можна переходити до наступного етапу перевірки.

Далі було проведено тестування форми для створення заявки на отримання допомоги переселенцем із використанням смартфона Xiaomi 12, результати якого наведено на рисунку 3.15. Основною метою цього етапу стало оцінювання працездатності форми на мобільному пристрої, перевірка коректності відображення елементів інтерфейсу, а також визначення рівня зручності введення даних користувачем.

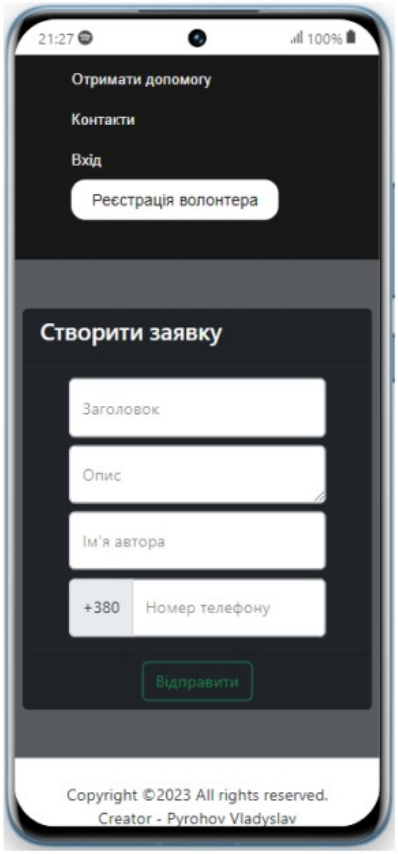

Рисунок 3.15 – Тестування створення заявки ВПО на допомогу використовуючи смартфон XIAOMI 12

Під час тестування особлива увага приділялася адаптивності форми до розмірів екрана мобільного пристрою. Було перевірено правильність

розташування полів введення, кнопок керування та інформаційних елементів, а також зручність навігації між ними. Крім того, оцінювалася коректність роботи віртуальної клавіатури, швидкість введення інформації та відсутність помилок під час надсилання заявки до серверної частини системи.

Результати тестування показали, що форма коректно адаптується до екрана смартфона та забезпечує комфортну взаємодію користувача із системою. Усі елементи інтерфейсу відображалися без спотворень, а поля введення залишалися достатньо великими та зручними для заповнення за допомогою сенсорного екрана. Процес створення заявки не викликав труднощів і не потребував додаткових дій з боку користувача.

Також було перевірено коректність валідації введених даних та передачі інформації на сервер. Усі обов’язкові поля успішно проходили перевірку, а створені заявки коректно зберігалися в базі даних і ставали доступними для подальшого опрацювання волонтерами. Це підтверджує надійність реалізованого механізму створення звернень та його готовність до використання на мобільних пристроях.

Отримані результати свідчать про те, що форма створення заявки забезпечує належний рівень адаптивності, функціональності та зручності використання, що є важливою умовою для ефективної взаємодії користувачів із вебзастосунком через мобільні пристрої.

У результаті тестування було підтверджено, що форма повністю зберігає свою функціональність на мобільному пристрої. Поля для введення даних коректно масштабуються під розмір екрана, а користувач може без труднощів заповнювати заголовок, опис звернення, ім’я та контактний номер телефону.

Також перевірено роботу інтерактивних елементів, зокрема кнопок для відправки форми та повідомлень про валідацію. Інтерфейс залишається зручним для взаємодії, а всі елементи не виходять за межі екрану та не перекривають один одного.

Отримані результати підтверджують, що вебзастосунок є повністю адаптивним і забезпечує стабільну роботу форми для створення заявки на різних мобільних пристроях.

Також проведено тестування вебзастосунку на здатність коректно відображати та обробляти код у різних веббраузерах. Для цього використано інструмент аналізу продуктивності та якості вебсторінок, який широко застосовується розробниками – https://pagespeed.web.dev/ (рис. 3.16).

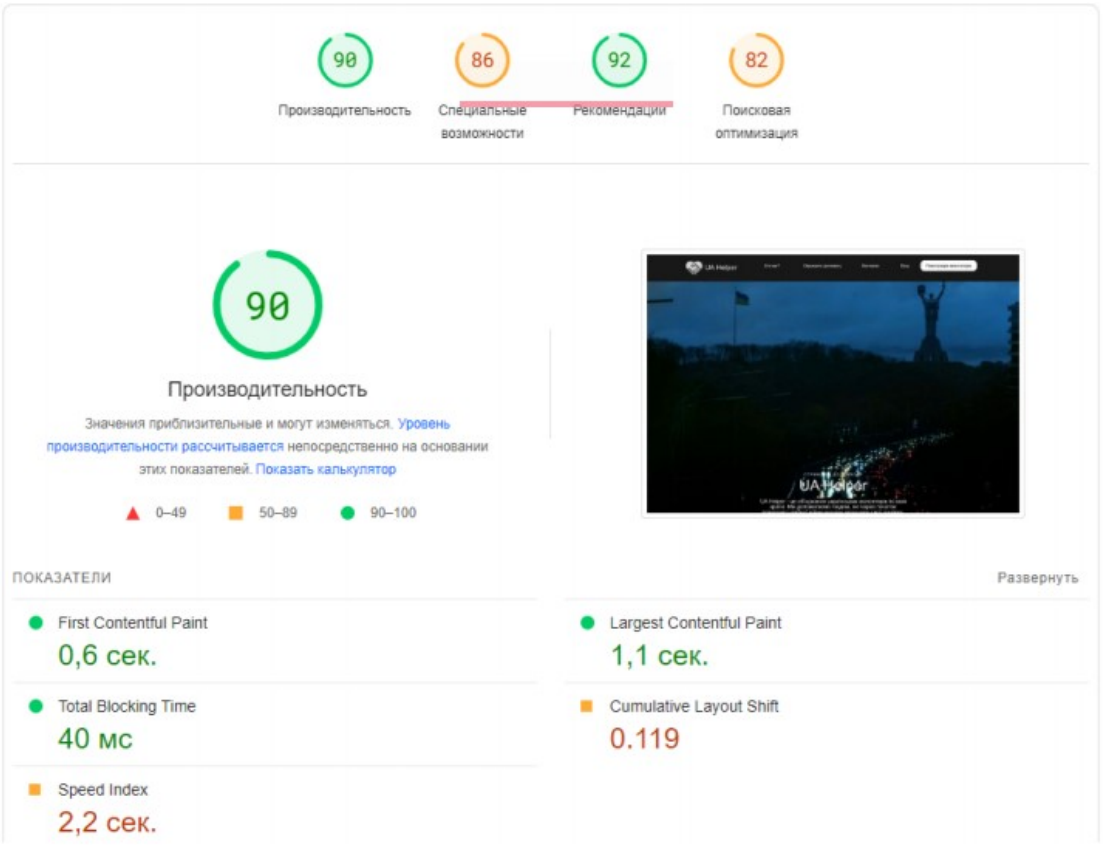

Рисунок 3.16 – Результати тестування вебзастосунку та його відображення на десктопній (комп’ютерній/ноутбучній) версії

Даний сервіс дозволяє оцінити швидкість завантаження сторінки, оптимізацію ресурсів, адаптивність інтерфейсу та загальну ефективність роботи вебзастосунку в різних браузерних середовищах. У процесі тестування система аналізує HTML, CSS та JavaScript код, а також перевіряє, наскільки коректно сторінка відображається для кінцевого користувача.

Результати перевірки дозволяють визначити рівень оптимізації вебзастосунку та виявити можливі проблемні місця, які можуть впливати на продуктивність або швидкість завантаження. Крім того, інструмент надає рекомендації щодо покращення коду, оптимізації зображень і підвищення загальної швидкодії системи. Завдяки використанню PageSpeed Insights було підтверджено, що вебзастосунок коректно обробляється сучасними браузерами та забезпечує стабільне відображення інтерфейсу для користувача, що свідчить про відповідність розробленого рішення сучасним вимогам веброзробки.

Результати тестування загалом є позитивними, оскільки показник продуктивності перевищує 90 балів. Це свідчить про те, що вебзастосунок добре оптимізований і коректно відображається у браузерах на персональних комп’ютерах та ноутбуках.

Наступним етапом оцінювання якості програмного продукту стало тестування мобільної версії вебзастосунку, результати якого наведено на рисунку 3.17.

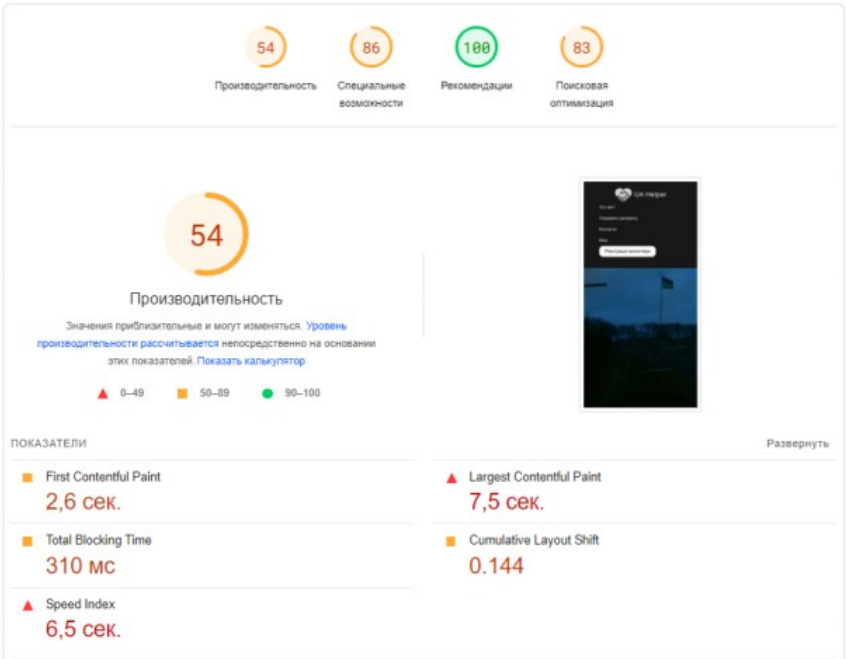

Рисунок 3.17 – Результати тестування вебзастосунку та його відображення на мобільних пристроях (смартфонах)

Основною метою цього тестування була перевірка коректності відображення інтерфейсу на мобільних пристроях, оцінка швидкодії системи та визначення рівня зручності використання функціональних можливостей застосунку на екранах невеликого розміру.

Під час тестування було проаналізовано адаптивність інтерфейсу до різних роздільних здатностей екрана, коректність розташування елементів керування, швидкість завантаження сторінок, а також зручність навігації та взаємодії користувача з основними функціями системи. Особлива увага приділялася перевірці роботи форм введення даних, інтерактивної мапи, системи авторизації та механізму створення заявок, оскільки саме ці компоненти є найбільш важливими для кінцевих користувачів.

За результатами проведеного аналізу встановлено, що мобільна версія вебзастосунку забезпечує коректне відображення основних елементів інтерфейсу та зберігає повну функціональність системи. Усі сторінки успішно адаптуються до різних розмірів екранів, що забезпечує комфортне використання застосунку на смартфонах і планшетах. Водночас отримані показники продуктивності виявилися дещо нижчими порівняно з результатами тестування на десктопних пристроях.

Зниження окремих показників швидкодії можна пояснити особливостями середовища, у якому було розгорнуто вебзастосунок. Для проведення тестування використовувалася хмарна платформа Microsoft Azure у межах безкоштовного тарифного плану, що передбачає обмеження щодо доступних обчислювальних ресурсів, обсягу оперативної пам’яті та продуктивності процесора. Такі обмеження можуть впливати на швидкість обробки запитів, час завантаження сторінок та загальну реакцію системи під час виконання ресурсоємних операцій.

Крім того, на продуктивність мобільної версії впливають особливості самих мобільних пристроїв, які зазвичай мають меншу обчислювальну потужність порівняно з настільними комп’ютерами та ноутбуками. Додатковим фактором є якість мережевого з’єднання, яка може змінюватися

залежно від умов використання мобільного інтернету та навантаження на мережу.

Незважаючи на зазначені обмеження, вебзастосунок продемонстрував стабільну та коректну роботу під час тестування. Усі основні функції, включаючи авторизацію користувачів, створення та обробку заявок, роботу інтерактивної мапи та перегляд інформаційних матеріалів, виконувалися без критичних помилок. Отримані результати свідчать про те, що розроблений вебзастосунок забезпечує прийнятний рівень продуктивності та може ефективно використовуватися в реальних умовах експлуатації. У разі переходу на платний тарифний план або використання більш потужної серверної інфраструктури очікується додаткове покращення показників швидкодії та загальної продуктивності системи.Отримані результати підтверджують, що реалізований адаптивний інтерфейс забезпечує коректну роботу програмного продукту на мобільних пристроях, а використані технологічні рішення дозволяють підтримувати належний рівень зручності та доступності для користувачів навіть за умов обмежених серверних ресурсів.

Крім того, певний вплив на швидкість завантаження та загальну продуктивність має наявність великої кількості зображень і графічних елементів із значними розмірами файлів. Це ускладнює обробку сторінок браузером, особливо на мобільних пристроях із меншою обчислювальною потужністю та обмеженим інтернет-з’єднанням.

Незважаючи на деяке зниження показників продуктивності, під час розробки вебзастосунку пріоритетною метою було забезпечення високого рівня зручності використання та доступності інтерфейсу для кінцевих користувачів. Особлива увага приділялася читабельності контенту, зрозумілості навігації та якості візуального оформлення, оскільки значна частина користувачів взаємодіє із системою в умовах обмеженого часу та потребує швидкого доступу до необхідної інформації.

З огляду на це, під час тестування та оптимізації не застосовувалися агресивні методи зменшення якості графічних матеріалів або суттєвого

скорочення розміру зображень. Хоча такі заходи могли б позитивно вплинути на швидкість завантаження сторінок, вони водночас могли б погіршити сприйняття інформації користувачами, знизити якість візуального контенту та негативно вплинути на загальний користувацький досвід. Тому було прийнято рішення зберегти баланс між продуктивністю системи та якістю відображення інтерфейсу.

Результати проведеного тестування підтвердили, що вебзастосунок забезпечує коректну роботу основних функціональних можливостей на мобільних пристроях та зберігає належний рівень зручності використання. Усі ключові компоненти інтерфейсу адаптуються до різних розмірів екранів, а навігація та взаємодія з функціоналом залишаються інтуїтивно зрозумілими для користувачів.

Таким чином, можна зробити висновок, що навіть за умов дещо нижчих показників продуктивності порівняно з десктопною версією вебзастосунок залишається функціональним, стабільним і зручним для використання на мобільних пристроях. Отримані результати свідчать про відповідність програмного продукту основним вимогам щодо адаптивності, доступності та якості користувацького інтерфейсу.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи було проведено аналіз предметної області програмного забезпечення, який є підсумковим результатом дослідження та розробки. Робота охоплює повний цикл створення вебзастосунку:

– обґрунтовано актуальність створення вебзастосунку для підтримки волонтерів та внутрішньо переміщених осіб. На сучасному етапі в Україні відсутнє повністю цифрове рішення для пошуку та організації волонтерських місій, яке передбачає реєстрацію користувачів зі статусом волонтера;

– сформовані вимоги до програмного забезпечення, що стали основою для подальшого вибору технологій реалізації. З метою охоплення максимальної кількості користувачів розроблюваний продукт реалізовано у вигляді вебзастосунку з клієнтською та серверною частинами;

– обрано мову програмування JavaScript як один із найзручніших інструментів для розробки сучасних вебінтерфейсів;

– проведено аналіз інструментів і фреймворків, у результаті якого обрано Angular завдяки його перевагам, зокрема компонентному підходу, продуктивності та відповідності вимогам проєкту.

На основі порівняння інтегрованих середовищ розробки було обрано IDE WebStorm як найбільш функціональне для реалізації клієнтської частини. Для серверної частини застосунку використано технологію ASP.NET Core Web API – сучасний інструмент для створення серверних рішень мовою C#.

Під час аналізу систем керування базами даних із найпоширеніших варіантів – PostgreSQL та MySQL – було обрано PostgreSQL, оскільки вона забезпечує високу продуктивність, надійність та не має низки обмежень, притаманних альтернативним рішенням.

Після завершення етапу аналізу було розроблено макети інтерфейсу вебзастосунку, а також сформовано схему роботи системи та структуру бази

даних. На основі створених макетів реалізовано клієнтську частину вебзастосунку з необхідними сторінками, формами реєстрації та основним функціоналом, а також серверну частину, що забезпечує роботу з даними та взаємодію з базою даних.

Завершальним етапом стала перевірка працездатності системи та усунення виявлених недоліків. Після проведеного тестування вебзастосунок можна вважати завершеним програмним продуктом, який відповідає поставленим вимогам і забезпечує можливість швидкого та ефективного отримання допомоги внутрішньо переміщеними особами через перевірених волонтерів.

Результати роботи апробовано у вигляді двох тез доповідей:

– під час Міжнародного молодіжного форуму «Радіоелектроніка і молодь у XXI столітті» [38];

– під час X Всеукраїнської мультидисциплінарної студентської наукової конференції «Розвиток сучасної науки: актуальні питання теорії та практики» [39].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Мезенцев, К. В. (2020). ВНУТРІШНЬО ПЕРЕМІЩЕНІ ОСОБИ У МІСТАХ: МІКРОГЕОГРАФІЯ ВИМУШЕНОГО ПЕРЕСЕЛЕННЯ. Реєстраційне посвідчення в УкрІНТЕІ МОН № 777 від 18 грудня 2019 р., 19.

2. Гаврилюк, О. К. (2020). ПАТТЕРН «ТИМЧАСОВОЇ КОНЦЕНТРАЦІЇ» ВПО? ФАКТОРИ ТА УМОВИ ПРОСТОРОВОГО РОЗПОДІЛУ ВПО В УКРАЇНІ ПРОТЯГОМ ОСТАННЬОЇ ПЕНТАДИ. Реєстраційне посвідчення в УкрІНТЕІ МОН № 777 від 18 грудня 2019 р., 157.

3. Редзюк, В., & Дармостук, Д. (2024). Цифрові платформи для надання державних послуг: досвід України та світу. Публічне управління: концепції, парадигма, розвиток, удосконалення, (9), 168-175.

4. Makarenko, P. O. (2025). Цифрова трансформація державних послуг: історія створення платформи «Дія» та порівняльний аналіз міжнародних аналогів (2019–2022 pp.). Studies in history and philosophy of science and technology, 34(1), 154-170.

5. Маматова, Т. В., & Чикаренко, І. А. СОЦІАЛЬНА МОБІЛІЗАЦІЯ НА КРАУДСОРСИНГОВИХ ОНЛАЙН ПЛАТФОРМАХ ПІД ЧАС ВІЙНИ І В ПОСТВОЄННИЙ ПЕРІОД. Редакційна колегія, 161.

6. Бєлкіна-Ковальчук, О. ГУМАНІТАРНА ДОПОМОГА ЯК НАПРЯМ СОЦІАЛЬНОГО ЗАБЕЗПЕЧЕННЯ НАСЕЛЕННЯ УКРАЇНИ. НАУКОВА РЕДАКЦІЯ, 15.

7. Alio, M., Alrihawi, S., Milner, J., Noor, A., Wazefadost, N., & Zigashane, P. (2020). By refugees, for refugees: Refugee leadership during COVID-19, and beyond. International Journal of Refugee Law, 32(2), 370-373.

8. Тимоцко, Р. (2021). Управління фандрейзинговою діяльністю неприбуткових організацій (на прикладі відділу розвитку Пласту–Національної скаутської організації України).

9. Bauer, T. N. (2013). Onboarding: The power of connection. Onboarding White Paper Series. SuccessFactors.

10. Zhukovska, V. M., Mykolaichuk, I., Marnyalo, A. M., & Shoma, M. S. (2021). РЕКРУТМЕНТ ЯК ТЕХНОЛОГІЯ ЕФЕКТИВНОГО ЗАЛУЧЕННЯ Й ОНБОРДИНГУ ПЕРСОНАЛУ (Recruitment as a Technology of Effective Attraction and Onboarding of Staff). Жуковська ВМ, Миколайчук ІП, Марняло АМ, Шома МС Рекрутмент як технологія ефективного залучення й онбордингу персоналу. Бізнес Інформ, (12), 257-262.

11. Sturridge, C., Girling-Morris, F., Spencer, A., Kara, A., & Chicet, C. (2023). Funding refugee-led organisations.

12. Мельник, А. В. (2026, May). Трансформація парадигм програмної архітектури: порівняльний аналіз монолітних, мікросервісних та безсерверних рішень. In XX Міжнародна науково-практична конференція «Latest Technologies and Modern Society: Automation of Processes and Improvement of Quality of Life» (pp. 68-75).

13. Турбай, Є. О., & Поночовний, Ю. Л. (2025). Розроблення вебсервісу для управління каталогом товарів з авторизацією та системою фільтрації на основі React та MySQL.

14. Wang, V., Salim, F., & Moskovits, P. (2013). The websocket protocol. In The Definitive Guide to HTML5 WebSocket (pp. 33-60). Berkeley, CA: Apress.

15. України, З. (2010). Про захист персональних даних. Відомості Верховної Ради України, (34), 2297-17.

16. Самагальська, Ю. Я. (2024). Захист персональних даних в мережі Інтернет.

17. Гребенніков, В. (2018). Комплексні системи захисту інформації. Проектування, впровадження, супровід. ЛитРес.

18. Легка, О. В. (2021). Актуальні питання захисту персональних даних: вітчизняний та міжнародний досвід. Правова позиція, 2(31), 74-79.

19. Мисленкова, В. О. (2024). Реалізація державної політики у сфері соціального захисту населення в умовах дії воєнного стану.

20. Chaudhary, M., & Kumar, A. (2014). PhpStorm Cookbook. Packt Publishing Ltd.

21. Del Sole, A., & Sole, D. (2019). Visual Studio Code Distilled. Berkeley: Apress.

22. Kanjilal, J. (2013). ASP. NET Web API. Packt Publishing.

23. Staiano, F. (2022). Designing and Prototyping Interfaces with Figma: Learn essential UX/UI design principles by creating interactive prototypes for mobile, tablet, and desktop. Packt Publishing Ltd.

24. Duckett, J., & Schlüter, J. (2011). Html & Css. New York: Wiley.

25. Liberty, J. (2005). Programming C#: building. NET applications with C. " O'Reilly Media, Inc.".

26. Turner, A. (2001, May). Angular analysis. In Proceedings of the 3rd international symposium on space syntax (Vol. 30, pp. 30-11).

27. Collier, M., & Shahan, R. (2015). Microsoft azure essentials-fundamentals of azure. Microsoft Press.

28. Shafronenko, A., Bodyanskiy, Y., Pliss, I., & Patlan, K. (2019, June). Fuzzy clusterization of distorted by missing observations data sets using evolutionary optimization. In 2019 9th International Conference on Advanced Computer Information Technologies (ACIT) (pp. 217-220). IEEE.

29. Bodyanskiy, Y. V., Shafronenko, A., & Klymova, I. (2021, April). Adaptive Recovery of Distorted Data Based on Credibilistic Fuzzy Clustering Approach. In COLINS (pp. 6-15).

30. Hu, Z., Bodyanskiy, Y. V., Tyshchenko, O. K., & Shafronenko, A. (2019, July). Fuzzy clustering of incomplete data by means of similarity measures. In 2019 IEEE 2nd Ukraine Conference on Electrical and Computer Engineering (UKRCON) (pp. 957-960). IEEE.

31. Momjian, B. (2001). PostgreSQL: introduction and concepts (Vol. 192, p. 2001). New York: Addison-Wesley.

32. Bodyanskiy, Y. V., Shafronenko, Y. O., Brodetskyi, F. A., & Tanianskyi, O. S. (2025). ШВИДКА НЕЙРОННА МЕРЕЖА ТА ЇЇ АДАПТИВНЕ НАВЧАННЯ В ЗАДАЧАХ КЛАСИФІКАЦІЇ. Radio Electronics, Computer Science, Control, (3), 45-51.

33. Shafronenko, A., Bodyanskiy, Y. V., Klymova, I., & Holovin, O. (2020, May). Online credibilistic fuzzy clustering of data using membership functions of special type. In CMIS (pp. 744-753).

34. Shafronenko, A., Bodyanskiy, Y. V., & Pliss, I. (2023). Credibilistic Fuzzy Clustering Method Based on Evolutionary Approach of Crazy Wolfs in Online Mode. In CMIS (pp. 141-150).

35. Bodyanskiy, Y. V., Shafronenko, A., & Klymova, I. (2021, April). Adaptive Recovery of Distorted Data Based on Credibilistic Fuzzy Clustering Approach. In COLINS (pp. 6-15).

36. Hu, Z., Bodyanskiy, Y. V., Tyshchenko, O. K., & Shafronenko, A. (2019, July). Fuzzy clustering of incomplete data by means of similarity measures. In 2019 IEEE 2nd Ukraine Conference on Electrical and Computer Engineering (UKRCON) (pp. 957-960). IEEE.

37. Shafronenko, A., Bodyanskiy, Y., Pliss, I., & Popov, S. (2020, September). Evolving neo-fuzzy system for distorted data online processing. In 2020 10th International Conference on Advanced Computer Information Technologies (ACIT) (pp. 352-355). IEEE.

38. Деговцов В.О. (2026) Теоретико-прикладні аспекти розробки спеціалізованих інформаційних систем для координації волонтерської діяльності. 30-й Міжнародний молодіжний форум «Радіоелектроніка і молодь у XXI столітті». Зб. матеріалів форуму. Т. 7. Харків: ХНУРЕ. 2026. 48-49

39. Деговцов В.О. (2026) Розробка вебзастосунку для координації волонтерської діяльності та підтримки внутрішньо переміщених осіб «Розвиток сучасної науки: актуальні питання теорії та практики». м. Тернопіль. 2026. 450-451.