МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТЕХНОЛОГІЙ

ННІ/факультет	Інститут економіки і бізнес освіти
Кафедра	Економіки та цифрового бізнесу
Спеціальність	«Комп'ютерні науки»
Форма навчання	Заочна

КВАЛІФІКАЦІЙНА РОБОТА

Борисенка Івана Олексійовича

(прізвище, ім'я, по батькові здобувача)

на тему Створення веб-порталу для оперативного інформування про надзвичайні ситуації та заходи пожежної безпеки

(повна назва теми)

за матеріалами _____

(повна назва бази дослідження)

науковий керівник Кандидат економічних наук, доцент _____ Соловйова В. В

(наук. ступінь, вчене звання) *(підпис)* *(прізвище, ініціали)*

Робота допущена до захисту в ЕК
Протокол засідання кафедри
від 09 червня 2025 р. № 12
Завідувач кафедри _____
(підпис)

К.е.н., доцент В.М. Радько
Наук. ступінь, вчене звання *Ініціали, прізвище*

Кривий Ріг – 2025

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ЕКОНОМІКИ ТА БІЗНЕС-ОСВІТИ
(повне найменування вищого навчального закладу)

Кафедра економіки та цифрового бізнесу
Освітній ступінь бакалавр
Спеціальність 122 Комп'ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри _____ В.М. Радько

"07" квітня 2025 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА ЗДОБУВАЧУ

_____ Борисенка Івана Олексійовича_____

1. Тема роботи Створення веб-порталу для оперативного інформування про надзвичайні ситуації та заходи пожежної безпеки_____

науковий керівник роботи Соловйова В.В._____,
затверджені наказом вищого навчального закладу від «04» квітня 2025 р. № 224-ст (д/ф)
№ 151-ст (з/ф)

2. Строк подання здобувачем роботи 31.05.2025р.

3. Зміст кваліфікаційної роботи бакалавра, об'єкт, предмет та мета дослідження:

Розділ 1 Аналіз існуючих рішень та вимог до системи створення веб-порталу для оперативного інформування населення про надзвичайні ситуації та заходи пожежної безпеки
Розділ 2 Проектування веб-порталу для оперативного інформування населення про надзвичайні ситуації та заходи пожежної безпеки_____
Розділ 3 Реадіхація веб-порталу для оперативного інформування населення про надзвичайні ситуації та заходи пожежної безпеки_____

Об'єкт дослідження – процес створення веб-порталу для оперативного інформування про надзвичайні ситуації та заходи пожежної безпеки.

Предмет дослідження - створення веб-порталу для оперативного інформування про надзвичайні ситуації та заходи пожежної безпеки.

Мета кваліфікаційної роботи бакалавра – Метою дослідження є розробка та впровадження веб-порталу для оперативного інформування населення про надзвичайні ситуації та пожежну безпеку, а також обґрунтування методичних підходів до його створення. Фінальним результатом має стати функціонуючий веб-портал, здатний своєчасно доводити до користувачів критичну інформацію про загрози та надавати рекомендації щодо дій, підвищення рівня обізнаності громадян у сфері безпеки.

4. Дата видачі завдання 04.04.2025р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи бакалавра	Строк виконання етапів роботи	Відмітка керівника про виконання етапів (дата, підпис)
1	Підготовка розділу 1	до 28.04.2025р.	25.04.2025
2	Підготовка розділу 2	до 16.05.2025р.	15.05.2025
3	Підготовка розділу 3	до 30.05.2025р.	29.05.2025
4	Реєстрація завершеної дипломної роботи	до 31.05.2025р.	30.05.2025
5	Отримання відгуку від наукового керівника	03-04.06.2025р.	04.06.2025
6	Отримання зовнішньої рецензії	05-06.06.2025р.	06.06.2025
7	Перевірка кваліфікаційної роботи на плагіат	02-09.06.2025р.	04.06.2025
8	Попередній захист кваліфікаційної роботи на кафедрі	03.06.2025р.	03.06.2025
9	Допуск кафедрою кваліфікаційної роботи до захисту	09.06.2025р.	09.06.2025
10	Підготовка студента до захисту в ЕК	до 17.06.2025р.	17.06.2025

Завдання підготував науковий керівник _____ Соловйова В.В._____
(підпис) (прізвище та ініціали)

Завдання одержав здобувач _____ Борисенко І. О.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Робота містить 90 сторінок, 82 малюнка, 32 джерела, 0 таблиць, 1 додаток.

Об’єкт дослідження: процес створення веб-порталу для оперативного інформування про надзвичайні ситуації та заходи пожежної безпеки.

Предмет дослідження - створення веб-порталу для оперативного інформування про надзвичайні ситуації та заходи пожежної безпеки.

Мета: розробка та впровадження веб-порталу для оперативного інформування населення про надзвичайні ситуації та пожежну безпеку, а також обґрунтування методичних підходів до його створення. Фінальним результатом має стати функціонуючий веб-портал, здатний своєчасно доводити до користувачів критичну інформацію про загрози та надавати рекомендації щодо дій, підвищення рівня обізнаності громадян у сфері безпеки.

В результаті дослідження було розроблено функціональний веб-портал, що забезпечує своєчасне донесення інформації про надзвичайні ситуації та рекомендації щодо заходів пожежної безпеки. Портал дозволяє оперативно публікувати повідомлення та новини, містить інтерактивні модулі. Новизна роботи полягає у поєднанні на одному веб-ресурсі функцій екстреного оповіщення населення з інформаційно-навчальними матеріалами з пожежної безпеки, а також у застосуванні оповіщення.

Область застосування: результати цієї роботи можуть бути використані системах цивільного захисту, загальне користування, також веб-додаток може полегшити відстеження за надзвичайними ситуаціями в регіоні та країні, що у свою чергу, сприятиме підвищенню рівня безпеки населення, своєчасному реагуванню на загрози та зниження негативних наслідків надзвичайних ситуацій.

ІНФОРМАЦІЙНА СИСТЕМА, ВЕБ-ДОДАТОК, ВЕБ-ІНТЕРФЕЙС, СТРУКТУРА ДАНИХ, ОПЕРАТИВНЕ ПОВІДОМЛЕННЯ.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

ДСНС - Державна служба України з надзвичайних ситуацій

МВС - Міністерство внутрішніх справ України

МОЗ - Міністерство охорони здоров’я України

НС - Надзвичайна ситуація

API - Application Programming Interface (інтерфейс прикладного програмування)

UI - User Interface (інтерфейс користувача)

UX - User Experience (досвід користувача)

HTML - HyperText Markup Language

CSS - Cascading Style Sheets

JSON - JavaScript Object Notation (формат обміну даними)

URL - Uniform Resource Locator (уніфікований локатор ресурсу)

IE 9 - Internet Explorer 9

ВСТУП

У сучасних умовах воєнного стану постає питання оперативного інформування населення набуває особливої актуальності. Від швидкого та достовірного доступу до інформації залежить безпека громадян, ефективність дій у разі загрози, а також зменшення негативних наслідків надзвичайних ситуацій. Система оповіщення населення повинна бути зручною, швидкою, надійною та доступною для різних верств населення. Одним із таких рішень є створення спеціалізованого веб-порталу, що дозволяє у режимі реального часу надавати населенню актуальну інформацію про небезпеки та заходи безпеки.

Дослідження базується на різноманітних джерелах. Використано нормативно-правові акти України у сфері цивільного захисту (закони, положення про оповіщення тощо), статистичні дані та звіти ДСНС про надзвичайні ситуації і пожежі, а також веб-технологій і безпеки життєдіяльності. Крім того перевірено онлайн-ресурси з інформацією про НС та пожежну безпеку для врахування існуючих напрацювань.

Об’єктом дослідження є процес створення веб-порталу для оперативного інформування про надзвичайні ситуації та заходи пожежної безпеки.

Предметом дослідження є створення веб-порталу для оперативного інформування про надзвичайні ситуації та заходи пожежної безпеки.

Метою дослідження є розробка та впровадження веб-порталу для оперативного інформування населення про надзвичайні ситуації та пожежну безпеку, а також обґрунтування методичних підходів до його створення. Фінальним результатом має стати функціонуючий веб-портал, здатний своєчасно доводити до користувачів критичну інформацію про загрози та надавати рекомендації щодо дій, підвищення рівня обізнаності громадян у сфері безпеки.

Завданням дослідження є проаналізувати стан систем оповіщення населення про надзвичайні ситуації і пожежну безпеку. Розробити макет та структуру веб-порталу відповідно до вимог. Створити веб-портал, наповнивши його контентом та інтегрувати механізм оперативного оновлення інформації. І

самий ключовий етап це протестувати роботу веб-порталу щоб оцінивши його ефективність.

Загальнонаукові методи аналізу, синтезу і порівняння використано для опрацювання джерел та вивчення існуючих рішень; системний підхід – для проектування порталу як елемента системи цивільного захисту; методи прототипування і моделювання – на етапі розробки програмного рішення.

Результатом виконання роботи стане створення повноцінного веб-порталу, готового до використання за призначенням. Портал забезпечуватиме швидкий та зручний доступ до інформації про надзвичайні ситуації та заходи пожежної безпеки для широкого кола користувачів. Розроблений ресурс сприятиме підвищенню рівня безпеки населення та ефективності роботи служб цивільного захисту.

РОЗДІЛ 1
АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ ТА ВИМОГ ДО СИСТЕМИ СТВОРЕННЯ ВЕБ-ПОРТАЛУ ДЛЯ ОПЕРАТИВНОГО ІНФОРМУВАННЯ НАСЕЛЕННЯ ПРО НАДЗВИЧАЙНІ СИТУАЦІЇ ТА ЗАХОДИ ПОЖЕЖНОЇ БЕЗПЕКИ

1.1 Аналіз веб-сайтів державних служб для оперативного інформування населення про надзвичайні ситуації та заходи пожежної безпеки

Офіційний сайт Державної служби України з надзвичайних ситуацій є веб-порталом для інформування населення про виникнення надзвичайних ситуацій, заходи пожежної безпеки, а також діяльність служби загалом. Аналіз даного ресурсу проводиться за такими критеріями: структура новин, швидкість оновлення інформації, дизайн інтерфейсу, функціонал сповіщення та доступність для користувачів.

На головній сторінці сайту одразу представлений блок із найважливішими оновленнями - “Останні новини”. Повідомлення подані у хронологічному порядку, де зазначені дата та час публікації та короткий заголовок події. При переході до розділу “Новини” користувач отримує доступ до розгорнутих описів інцидентів: пожеж, ДТП, техногенних аварій, пошуково-рятувальних операцій та інших надзвичайних подій.

Кожна новина супроводжується, в першу чергу текстовим описом події, офіційними фотографіями(з водяним знаком), за можливістю відеоматеріалом, а також рекомендаціями або інструкціями для населення.

Новини структуровані за категоріями: “Пожежі”, “Рятувальні операції”, “Навчання та тренування”, “Реагування на НС” тощо. Однак чіткої системи тематичної фільтрації безпосередньо на сторінці новин немає - для пошуку потрібної інформації доводиться використовувати загальний пошук по сайту.

Великим плюсом даного порталу є те що більшість новин подаються зрозумілою мовою, без надлишкової термінології, що робить інформацію доступною для широкого кола користувачів.

ДСНС України приділяє значну увагу оперативності висвітлення надзвичайних подій. На сайті оновлення новин здійснюється цілодобово, особливо в періоди підвищеної небезпеки - під час обстрілів, сезонних пожеж, паводків або техногенних аварій.

Новини публікуються протягом 1-3 годин після фактичної події. Такий підхід дозволяє громадянам своєчасно отримувати актуальні новини і відповідно реагувати на зміну ситуації у своєму регіоні.

Слід зазначити, що в екстрених випадках на сайті може з’являтися спеціальний банер або окрема новина з позначкою “Терміново”, що акцентує увагу користувачів на критичній інформації.

Наступним кроком оцінки став інтерфейс сайту, він є типовим для державних порталів України. Він оформлений у офіційній кольоровій гамі - білий фон із елементами синього.

Навігація по сайту інтуїтивно зрозуміла, головне меню розташоване у верхній частині сайту. Важливі розділи виділені окремими вкладками (“Оперативна інформація за останню добу”, “Останні новини”, “Надзвичайні події”, “Актуально”) також воно має верхнє меню для швидкого переходу між розділами

Сайт підтримує адаптивний дизайн, що дозволяє коректно переглядати його з мобільних телефонів та планшетів. Проте при перегляді на деяких мобільних пристроях помітні дрібні недоліки у відображенні елементів - зокрема, деякі великі банери не масштабується автоматично під розмір екрану. Також присутній режим для людей із порушенням зору.

На самому сайті функція реєстрації для отримання сповіщень (наприклад, через email-підписку) або миттєвих push-нотифікацій відсутня. Також сайт не використовує спливаючі повідомлення (pop-up alerts) для швидкого інформування відвідувачів про нові критичні події.

Проаналізувавши сайт можна виділити його переваги як висока оперативність оновлення інформації, адаптивний дизайн для мобільних пристроїв (телефони та планшети), а також використання сучасних засобів екстреного оповіщення таких як Cell Broadcast та чат-боти. Але також не зважаючи на переваги було також виділено і недоліки, а саме невеликі технічні недоліки при мобільному перегляді, відсутність фільтрів для пошуку новин за типом НС безпосередньо на сайті, і саме головне це дуже багато елементів та не дуже інтуїтивно зрозумілий дизайн.

Сайт ДСНС України загалом виконує свою основну функцію - оперативне інформування населення про надзвичайні ситуації та заходи пожежної безпеки. Він є надійним джерелом офіційної інформації, доступним для широкого кола користувачів.

Однак з урахуванням сучасних вимог до цифрових комунікацій можна рекомендувати розширення функціоналу сайту: впровадження механізмів персонального оповіщення користувачів та оптимізувати під мобільні версії. Це дозволило б підвищити ефективність інформування та забезпечити ще кращий захист населення під час надзвичайних ситуацій.

Офіційний сайт Міністерство внутрішніх справ є інформаційним порталом, призначеним для оперативного інформування громадян про діяльність органів системи МВС, зокрема Національної поліції, ДСНС, Державної прикордонної служби, Нацгвардії та інших підрозділів. Через сайт здійснюється публікація офіційних новин, пресрелізів, нормативно-правових актів, звітів і розпорядчих документів. Крім того, сайт виконує функцію публічного звітування та є важливим інструментом у забезпеченні прозорої діяльності Міністерства внутрішніх справ.

Новини МВС розміщено у розділі “Прес Центр - Новини” у вигляді хронологічного списку повідомлень. Кожна публікація має точну дату і час та тематичні теги, наприклад “Обстріли”, “Вшанування пам'яті”, “Герої МВС” тощо. Теги відображаються над заголовком і дозволяють фільтрувати схожі матеріали. У кінці кожної статті є блок “Схожі матеріали”, який містить

посилання на інші новини за тією ж темою. Структура рубрик видно і в навігаційній панелі – є окремий пункт “Пресцентр” з підменю “Новини”, “Анонси”, “Фото/Відео” тощо.

Швидкість оновлення інформації. Сайт МВС оперативно реагує на надзвичайні події. Увесь прес-реліз із оперативними зведеннями подій публікується щоденно, часто кілька разів на день. Зазвичай інформація про аварії, руйнування або поранених з’являється на сайті МВС буквально впродовж кількох годин після події. Таким чином сайт показує високу оперативність: події одного дня висвітлюються у формі послідовних оновлень зі свіжими часовими позначками.

Дизайн і адаптивність інтерфейсу**.** Інтерфейс МВС реалізовано за сучасними стандартами доступності. Присутня двомовність - є кнопка вибору англійської мови . Для людей з порушеннями зору реалізована окрема версія сайту: поруч із мовним перемикачем відображається іконка “Людям із порушенням зору”. Адаптивність підтверджує й наявність “бургер-меню” - іконки для мобільної навігації. До того ж у верхньому правому куті розміщено посилання на телефонний, що зазвичай відкриває мобільне меню. Таким чином сайт МВС зручний для користувачів на різних пристроях і враховує базову доступність. Але він має дуже багато елементів та незрозумілу навігацію сайту.

На сайті МВС немає власних систем push-сповіщень чи SMS-оповіщення. Натомість він інтегровано з соціальними мережами: в шапці є посилання на офіційні акаунти у Facebook, Twitter, YouTube, Instagram та Telegram. Завдяки цьому користувачі можуть підписатися на Telegram-канал МВС або інші канали, аби отримувати оперативні повідомлення про надзвичайні події. Крім того, на сайті є окремий розділ для повідомлення про корупцію.

Офіційний сайт Міністерства охорони здоров’я України є офіційним порталом, що забезпечує інформування населення про стан системи охорони здоров’я, державну політику у сфері медицини, а також актуальні події, пов’язані з охороною здоров’я. Портал МОЗ є джерелом інформації для працівників медичної галузі, журналістів та громадян, які шукають перевірену інформацію

про профілактику, вакцинацію, лікування та реформування системи охорони здоров'я.

Новини Міністерства охорони здоров'я зібрано у розділі "Прес-центр". За офіційною інформацією МОЗ, на оновленому сайті передбачено 10 основних розділів - серед них "Про міністерство", "Громадянам", "Медичним працівникам", "Освіта", та "Прес-центр". Розділ "Прес-центр" має підменю "Останні новини", "Анонси", "Інтерв'ю", "Контакти пресслужби" тощо. Тому всі актуальні оголошення та зведення щодо медицини шукають саме в категорії "Останні новини". Сторінка з новинами демонструє перелік повідомлень із заголовками та датами, але на відміну від МВС та ДСНС не так інтенсивно маркує їх тематичними тегами; найважливіші теми можуть виділятися окремими рубриками. На відміну від МВС та ДСНС, сайт МОЗ більше орієнтований на постійні тематичні оновлення медичної політики, тому повідомлення про окремі інциденти там менш виражені.

Інформація на сайті МОЗ оновлюється з регульованою періодичністю. Для тем воєнного часу - наприклад, руйнувань лікарень чи гуманітарної допомоги - пости публікуються після узгодження даних, тому зазвичай із затримкою в кілька годин або день. Таким чином швидкість оновлення можна охарактеризувати як помірну: актуальні медичні статистики публікуються щоденно, тоді як новини надзвичайного характеру виходять із затримкою через потребу в перевірці.

Оновлений сайт МОЗ позиціонується як сучасний ресурс із зручним інтерфейсом. Він адаптований до різних пристроїв - меню згортається до " "бургер-іконки", а контент масштабовано для читабельності на мобільних екранах. Співробітники МОЗ заявляли, що структура сайту принципово змінена і оптимізована для громадян: створено окремі блоки "Громадянам", "Медичним працівникам" тощо, спрощено навігацію. Важливим елементом є доступність: зокрема, є версія сайту для людей із порушеннями зору. Ці зміни роблять контент більш доступним. Також на сайті МОЗ, передбачено багатомовний інтерфейс. Загалом дизайн досить простий і мінімалістичний, з великими кнопками для

важливих розділів і гнучкою верстkою – що зручно для громадян, зокрема на мобільних пристроях.

ДСНС, МВС та МОЗ - мають сучасний вигляд і практичні інструменти навігації. Сайт ДСНС та МВС більше орієнтований на швидке висвітлення подій безпекового характеру: оперативно оновлюється декілька разів на день і має чітку двомовну структуру з функціями доступності. Сайт МОЗ націлений на зручність громадян і медиків: він організований за темами, акцентує увагу на доступності контенту та утримує інформацію за узгодженою схемою. Оперативні сповіщення на порталах ДСНС та МВС реалізовані через інтеграцію з соціальними каналами та месенджерами, тоді як штатної системи пуш-оповіщення сайти безпосередньо не пропонують. У підсумку можна констатувати, що ДСНС, МВС, та МОЗ забезпечують громадянам необхідну інформаційну підтримку - МВС та ДСНС швидким реагуванням на надзвичайні ситуації, а МОЗ - структурованими сервісами громадської охорони здоров’я та доступністю своїх інтернет-ресурсів.

1.2 Визначення цільової аудиторії та її потреб для оперативного інформування населення про надзвичайні ситуації та заходи пожежної безпеки

У межах розробки веб-порталу для оперативного інформування населення про надзвичайні ситуації та заходи пожежної безпеки важливим етапом є чітке визначення цільової аудиторії та розуміння її основних потреб. Важливо розуміти правильне врахування особливостей користувачів дозволяє створити ефективний інструмент комунікації, що сприятиме підвищенню рівня обізнаності громадян що забезпечить їх безпеку.

Цільова аудиторія порталу охоплює максимально широкий спектр користувачів. Портал орієнтований на всіх громадян України, незалежно від віку, місця проживання чи професійної діяльності. Особливо актуальним є його використання в умовах війни, коли оперативне отримання достовірної

інформації може мати значення для життя та безпеки людей. Оскільки ситуація в країні залишається напруженою, головними користувачами сайту виступатимуть громадяни, які потребують актуальної інформації про загрози та правила поведінки під час надзвичайних ситуацій.

Потреби цільової аудиторії формуються виходячи з основної мети веб порталу - забезпечення швидке та доступне інформування населення. Основними запитами аудиторії є оперативне отримання термінових повідомлень про надзвичайні ситуації, зокрема обстріли, пожежі, техногенні аварії тощо. Також мати чіткі та зрозумілі інструкції щодо дій у разі різних типів НС. Не мало важливою ціллю аудиторії є можливість швидко знайти контактну інформацію екстрених служб, задля отримання консультації або допомоги через гарячу лінію чи онлайн-чат. І ключова потреба це є наявність чіткої та простої у використанні навігації для швидкого доступу до новин та інструкцій без зайвого перегляду великої кількості розділів.

Особливості використання порталу також враховують сучасні тенденції користування інтернетом в Україні. Основними пристроями для доступу до порталу є телефони та персональні комп'ютери.

Таким чином, на основі проведеного аналізу можна визначити такі ключові вимоги до веб-порталу:

1. оперативне оновлення інформації у режимі реального часу;
2. простий, інтуїтивно зрозумілий інтерфейс;
3. наявність інструкцій для населення щодо дій у надзвичайних ситуаціях;
4. інтеграція гарячої лінії або онлайн-чат-бота для консультацій;

Розробка порталу, що відповідає цим вимогам, дозволить ефективно виконувати завдання оперативного інформування населення в умовах підвищеного ризику надзвичайних ситуацій.

1.3 Вимоги до порталу для оперативного інформування населення про надзвичайні ситуації та заходи пожежної безпеки

Функціональні вимоги описують конкретні сервіси, які надає веб-портал. Система має реалізовувати можливість публікації та управління новинами: адміністратор повинен мати інструменти додавання, редагування і видалення публікацій (текстових статей, фото- та відеоматеріалів, офіційних документів тощо). Необхідно забезпечити ефективну фільтрацію контенту за категоріями, тегами, датою тощо для зручності пошуку та сортування новин.

Публікація новин і контенту**:** додавання, редагування та видалення матеріалів (текст, медіафайли, документи). Призначати тематичні теги та обрані новини.

Навігація та структура контенту: логічна організація розділів порталу, дворівневе меню з чіткими, однозначними назвами.

Картографічна інтеграція: відображення місць подій на інтерактивній карті за допомогою стороннього API. Карта повинна бути інтегрована у відповідний розділ порталу.

Інтеграція чат-бота: взаємодія з користувачем через чат-бот (наприклад, у Telegram) для оперативного консультування за поширеними питаннями. Чат-бот має бути доступним 24/7 та надавати відповіді в режимі реального часу.

Інформаційні вимоги визначають склад і структуру даних, що зберігаються і обробляються системою. Портал має містити такі види інформації: динамічні новини та повідомлення, офіційні документи, аналітичні матеріали, контакти та довідкову інформацію організації. Контент має бути маркований метаданими (дата, автор, категорія, теги) для подальшої фільтрації та пошуку. Структура інформації повинна бути зрозумілою та узгодженою: глибина ієрархії розділів не перевищує 4 рівнів, назви розділів короткі і логічні. Пошук по порталу має здійснюватися повнотекстово за ключовими словами у заголовках і зміст публікацій.

Типи контенту: тексти (новини, статті), фотографії, відео, офіційні документи. Усі документи мають зберігатися в єдиній базі з можливістю фільтрації (за темою, типом, датою) та повнотекстового пошуку.

Організація інформації: логічне групування матеріалів за рубриками та тегами; реалізація новинної стрічки (хронологічна стрічка останніх подій). Кожен розділ порталу має відповідати ключовим задачам користувачів і бути інтуїтивно зрозумілим.

Пошукова підсистема: реалізація функції пошуку по всьому контенту порталу з видачею результатів. Результати пошуку мають надавати список знайдених публікацій із сортуванням по темі чи даті.

Система повинна задовольняти потреби кінцевих користувачів з обох ролей. Інтерфейс має бути дружнім навіть для непрофільного користувача: забезпечувати простоту навігації, мінімум необхідних дій та введення даних. Користувачі мають мати можливість фільтрувати та сортувати контент, використовувати пошук і звертатися до чат-бота.

В системі передбачаються дві ролі користувачів: Адміністратор та користувач. Адміністратор має отримати розширені права управління контентом і налаштуваннями порталу. Звичайному користувачу надаються мінімальні необхідні права - насамперед це перегляд інформації, пошук, використання фільтрів і чат-бота. Відповідно до принципу найменших привілеїв, адміністратор має широкий набір прав (керування публікаціями, категоріями, тегами, меню), а звичайні користувачі - лише потрібні для їхньої взаємодії з порталом.

Інтерфейс веб-порталу має бути інтуїтивно зрозумілим і сприяти ефективному виконанню завдань користувачами. Ключові елементи керування повинні бути простими і знайомими, навігація - чіткою (з можливістю повернення на головну сторінку з будь-якої частини порталу). Важливо мінімізувати кількість кроків та обсяг інформації, яку повинен вводити користувач для вирішення основних задач. Дизайн має відповідати стандартам офіційних урядових ресурсів: бути легким для сприйняття, привабливим і адаптуватися до фірмового стилю (рекомендовано використовувати

ліцензований шрифт e-Ukraine). Кольори та елементи інтерфейсу мають забезпечувати достатній контраст і читабельність. Необхідно передбачити адаптивну верстку: окремі прототипи розробляють для десктопної версії інтерфейсу, щоб сайт був доступним на екранах різного розміру. Усі графічні ресурси (іконки, зображення, шрифти) повинні бути ліцензовані, оптимізовані для швидкого завантаження, а анімація — помірною і приємною.

Висновки до розділу 1

У першому розділі роботи було проведено аналіз рішень щодо інформування населення про надзвичайні ситуації та заходи пожежної безпеки. Дослідження показало, що зазначені ресурси виконують важливу інформаційну функцію, оперативно публікують новини, мають адаптивний дизайн та базові інструменти доступності. Проте було виявлено також недоліки:гнучкість інтерфейсу, відсутність персоналізованих сповіщень та обмеженість інструментів для швидкої взаємодії з користувачем.

На основі цільової аудиторії визначено, що веб-портал має бути орієнтованим на широке коло громадян, хто потребує швидкого доступу до достовірної інформації про загрози, а також чітких інструкцій щодо поведінки під час НС.

У результаті аналізу сформульовано функціональні, інформаційні та інтерфейсні вимоги до веб-порталу. Зокрема, передбачено реалізацію системи керування новинами, фільтрацію контенту, інтеграцію чат-бота, адаптивний дизайн, повнотекстовий пошук і навігацію. Важливим аспектом є також поділ користувачів на дві ролі - адміністратора та користувача, що дозволяє оптимізувати доступ до функціоналу та забезпечити безпечну роботу системи.

Таким чином, розділ закладає підґрунтя для подальшої розробки архітектури веб-порталу, що відповідатиме сучасним стандартам інформування та дозволить підвищити ефективність комунікації з населенням.

РОЗДІЛ 2
ПРОЕКТУВАННЯ ВЕБ-ПОРТАЛУ ДЛЯ ОПЕРАТИВНОГО ІНФОРМУВАННЯ НАСЕЛЕННЯ ПРО НАДЗВИЧАЙНІ СИТУАЦІЇ ТА ЗАХОДИ ПОЖЕЖНОЇ БЕЗПЕКИ

Процес розробки будь-якого веб-додатку неможливий без попереднього етапу проектування. Саме на цьому етапі визначаються ключові можливості майбутньої системи, її структура, зовнішній вигляд, логіка взаємодії користувача з інтерфейсом, а також технічні вимоги до реалізації.

Проектування веб-порталу для оперативного інформування населення про надзвичайні ситуації передбачає створення зручного, швидкого та зрозумілого у користуванні сайту, який би забезпечував доступ до актуальної інформації. Основну увагу буде зосереджено на простоті користування, чіткості подачі матеріалів, а також на ефективному додаванні контенту через адміністратора.

У цьому буде розроблено структуру сайту, користувацькі сценарії взаємодії, етапи створення дизайну в середовищі Figma, а також підготовку технічного завдання, що стало основою для реалізації системи.

2.1 Створення структури сайту та користувацьких сценаріїв для оперативного інформування населення про надзвичайні ситуації та заходи пожежної безпеки

В першу чергу треба визначитись з архітектурою сайту, вона має бути чіткою, багатосторінкову, та зрозумілу для користувача та адміністратора. Головна сторінка має бути інформаційно вмістною та мати навігаційне меню, це робить сайт набагато зрозумілішим для користувача. Проаналізувавши інші інформаційні ресурси та вимоги, можна сказати що для такого виду сайту треба такі сторінки та розділи сайту:

Головна сторінка**:** буде містити основну інформацію про веб-портал та його функції. На ній буде представлено опис мети сайту, перелік напрямів інформації (новини, рекомендації, оперативні повідомлення тощо) і переваги

ресурсу для користувачів. Ця сторінка буде допомагати користувачу швидко зрозуміти структуру ресурсу та знайти потрібний розділ.

Розділ “Новини”**:** це одна з найголовніших сторінок вона буде відображати стрічку новин пов’язаних з надзвичайними ситуаціями. Новини мають бути згруповані за категоріями для зручності навігації. Користувач може переглядати всі новини або відфільтрувати їх за обраною категорією. Кожна новина буде представлена заголовком, коротким описом та, за потреби, фотографією або піктограмою; передбачено можливість переходу до повного тексту новини на окремій сторінці.

Розділ “Інструкції та рекомендації”: буде містити офіційні рекомендації, інструкції та поради щодо дій у різноманітних надзвичайних ситуаціях. Інформація структурована за тематикою (наприклад, дії при пожежі, при виявленні підозрілих предметів, правила поведінки під час повітряної тривоги тощо) і подана у зручному для читання форматі. Цей розділ виконує освітню функцію, надаючи громадянам перевірені алгоритми дій для забезпечення власної безпеки від офіційних ресурсів.

Розділ “Оперативна інформація”**:** забезпечує користувачів актуальними даними про поточні надзвичайні події в режимі реального часу. На цій сторінці може бути інтегровано інтерактивну карту тривог, що автоматично оновлюється та відображає небезпечні зони чи інциденти. Також у розділі будуть виводитись оперативні повідомлення та попередження, які надходять з офіційних джерел. Це дозволяє відвідувачам порталу своєчасно отримувати критично важливу інформацію.

Загальна архітектура сайту буде побудована таким чином, щоб забезпечити інтуїтивну навігацію і швидкий доступ до ключових розділів. Кожна сторінка буде мати єдиний стиль оформлення і зрозумілу структуру: у верхній частині розташовано головне меню для переходу між розділами, центральна область відведена під зміст відповідного розділу, а нижня частина буде містити футер. Такий підхід дозволяє користувачу легко орієнтуватися на порталі та знаходити необхідну інформацію без зайвих зусиль.

При проектуванні інтерфейсу враховано дві основні ролі користувачів системи – звичайний користувач та адміністратор. Адміністратор буде мати окрему адміністративну панель.

Навігаційне меню: доступне усім відвідувачам без обмежень. Звичайний користувач клікає по пунктах меню (“Новини”, “Інструкції та рекомендації”, “Оперативна інформація”) для переходу до відповідного розділу. Для адміністратора меню виконує ту саму функцію навігації по основних сторінках сайту; при цьому адміністратор, як і звичайний користувач, бачить актуальний контент кожного розділу, але додатково може мати доступ до окремого меню посилання “Адмін-панель”, які недоступні звичайним користувачам.

Елементи списків новин та категорій: у розділі “Новини” будуть мати інтерфейс списку публікацій. Звичайний користувач може прокручувати стрічку новин, вибирати категорію для фільтрації новин за темою, та переходити до деталізованого перегляду обраної новини. Кожен заголовок новини та анонс є клікабельними елементами інтерфейсу: при натисканні відвідувач бачить повний текст новини. Адміністратор взаємодіє з цим же списком двома способами: по-перше, як читач, по-друге, як редактор - для адміністратора. Ці керуючі елементи дозволяють адміністратору створювати нові публікації, змінювати тексти існуючих новин чи вилучати їх.

Сторінка рекомендацій: для відвідувача це статичний контент, з яким він взаємодіє шляхом читання та перегляду. Сторінка буде містити перелік тем, клікаючи які користувач отримує детальні поради з конкретного питання. У будь-якому разі звичайний користувач має лише права на перегляд цього розділу. Адміністратор натомість може взаємодіяти з цим контентом у режимі редагування: через адмін-панель він може додавати нові інструкції, оновлювати тексти рекомендацій чи видаляти застарілу інформацію. Такі дії виконуються поза межами основного користувацького інтерфейсу, і кінцевий відвідувач їхнього процесу не бачить – він отримує вже оновлений контент.

Розділ оперативної інформації: цей інтерфейс орієнтований переважно на пасивну взаємодію з боку користувача. Звичайний користувач може переглядати

інтерактивну карту тривог, а також мати точну інформацію про загрози. Оперативні текстові повідомлення на цій сторінці автоматично з’являються у міру оновлення даних – користувач просто читає їх. Для адміністратора ця частина системи має ручне внесення інформації. Звичайні користувачі не мають прямого впливу на наповнення цього розділу, вони лише споживають інформацію.

Контактні та зовнішні посилання: ці елементи (адреси, телефони, іконки соцмереж) доступні та однаково виглядають для обох ролей. Користувач взаємодіє з ними, наприклад, натискаючи на іконку Facebook або YouTube для переходу на відповідний офіційний ресурс ДСНС, чи використовує номер телефону зі сторінки, щоб звернутися на гарячу лінію. Також в цій частині сайту буде реалізовано чат-бота в телеграм про повітряні тривоги.

Таким чином, кожен елемент спроектовано з урахуванням його цільової аудиторії. Звичайні користувачі мають змогу вільно переглядати інформацію та користуватися інтерактивними можливостями без можливості зміни контенту, тоді як адміністратор отримує адміністративну панель для створення і редагування цього контенту.

Важливо зазначити, що для кожної ролі впроваджено окрему логіку доступу і взаємодії із системою. Звичайний користувач має вільний доступ до відкритих розділів сайту без необхідності авторизації: він може переглядати всі сторінки, читати та взаємодіяти з контентом у межах наданих користувацьких функцій. Водночас адміністратор отримує розширені можливості лише після успішної авторизації. Тільки в авторизованому режимі йому відкрито окрему сторінку для управління сайтом (Див. Рис 2.1). Неавторизований користувач не може отримати доступ до адміністративних функцій – система захищає сторінки адміністрування та відповідні дії від перегляду та використання сторонніми

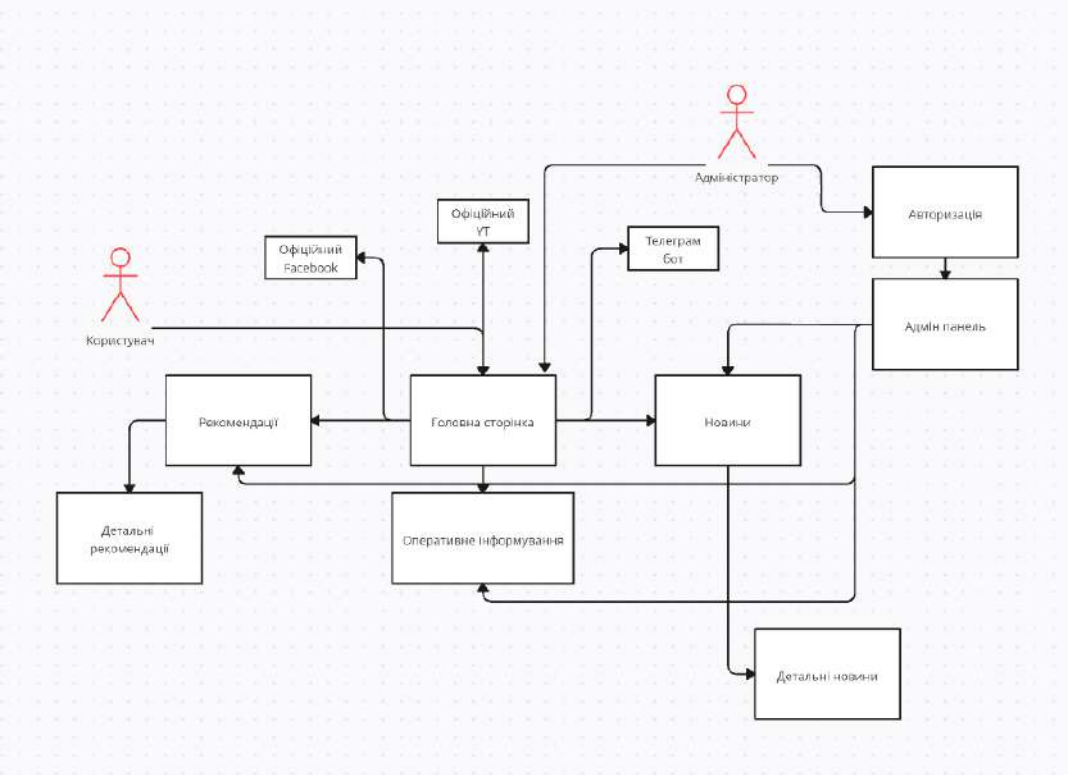


Рис. 2.1. Візуалізація архітектури сайту з розмежуванням ролей

(Розроблено автором)

Такий підхід гарантує, що публічний контент може переглядатися всіма відвідувачами, але змінювати його можуть виключно визначені адміністратори. Це забезпечує безпеку ресурсу та надійність інформації, що публікується на сайті.

2.2 Розробка UI/UX дизайну у Figma веб-порталу для оперативного інформування населення про надзвичайні ситуації та заходи пожежної безпеки

Кожен проект такого формату починається з розробки користувацького інтерфейсу, його головною метою є не тільки привабливий дизайн, а і зрозумілість у використанні. Для створення макету сайту був обраний онлайн-сервіс Figma [22], [23], [24], що є одним із найпопулярніших інструментів для

UI/UX-дизайну завдяки своїй зручності, підтримці колективної роботи та можливості створення інтерактивних прототипів.

Для створення сайту було обрано такі принципи UI/UX-дизайну, в першу чергу це простота та мінімалізм - інтерфейс позбавлений зайвих елементів, що дозволяє користувачу інтуїтивно зрозумілий інтерфейс та концентруватися на головному: отриманні інформації. Також невідкладною частиною є контрастність і читабельність - будуть обрані кольори та шрифти які забезпечать легке сприйняття тексту. І найголовніше для швидкого орієнтування в інтерфейсі є інтуїтивна навігація - усі елементи мають бути розташовані логічно і передбачувано для користувача, що спрощує взаємодію із сайтом.

Ключовим моментом розробки макету почалась з головної сторінки що є лицем сайту. Сторінка була поділена на 3 фрагменти а саме на шапку, вміст та футер. Для шапки та футеру був обраний спокійний темно синій колір, що є в стилі офіційних ресурсів, для вмісту був обраний легкий сіро-синій колір, що є дуже гарним рішенням для перегляду контенту. Після створення робочої форми, можна переходити до заповнення її. Для початку був обраний стиль шрифту, був обраний e-Ukraine - це сучасний та дуже приємний шрифт.

Було сформовано таку структуру веб-сторінки, яка відповідає вимогам сучасного дизайну. У верхній частині сторінки (шапці) ліворуч розміщується логотип ДСНС України - офіційна емблема служби. Поряд із ним розташовано назву порталу - "Державна служба України з надзвичайних ситуацій" та підзаголовок - "Інформування населення про надзвичайні ситуації". Навігаційне меню представлено кнопками "Новини", "Рекомендації" та "Інформування", що забезпечують швидкий та інтуїтивно зрозумілий перехід до відповідних розділів сайту. У правому верхньому куті розміщені іконки соціальних мереж (Telegram, Facebook, YouTube), які ведуть на офіційні сторінки служби, включаючи телеграм-бот.

Під час створення сторінки було взято за ідею реалізувати великі інтерактивні кнопки з зображеннями, які позначають основні розділи. Це сприяє кращому візуальному сприйняттю інформації. На головній сторінці створено три

основні інтерактивні блоки це "Новини", "Інструкції та рекомендації", а також "Оперативне інформування". Вони мають естетичний дизайн: закруглені кути, тінь, легке розмиття фону зображень, чіткий шрифт. При натисканні на будь-який із блоків користувач буде переходити до відповідного розділу. На сторінці також присутній текстовий блок, що містить опис веб-порталу, його основних задач та функціоналу.

Для оформлення тексту використано офіційний шрифт e-Ukraine, колір тексту - темно-сірий (#333333), що забезпечує хорошу читабельність. У нижній частині сторінки (футері) продубльовано назву порталу з підзаголовком, повторюється навігаційне меню з кнопками, а також подано контактну інформацію: адреса, телефон гарячої лінії та електронна пошта.

Таким чином ми отримали повністю готовий візуальний макет головної сторінки подано на Рис. 2.2. Це дає нам змогу оцінити інтуїтивність дизайну та зробити певні корективи в логіці сайту.

Рис. 2.2. Візуалізація макету “Головна сторінка”

(Розроблено автором)

Наступним кроком розробки макету буде створення сторінки “Інформування”. Це одна із робочих областей де користувач буде дивитись наявні загрози, чи то повітряна тривога або інші небезпечні події. Шапка та футер не змінюються на відміну від змісту. Зміст цієї сторінки буде займати карта повітряних тривог та текст, шрифт та інші аспекти не змінні. Цей макет був розроблений з урахуванням максимальної інформативності, простоти читання це можна побачити на рис. 2.3.

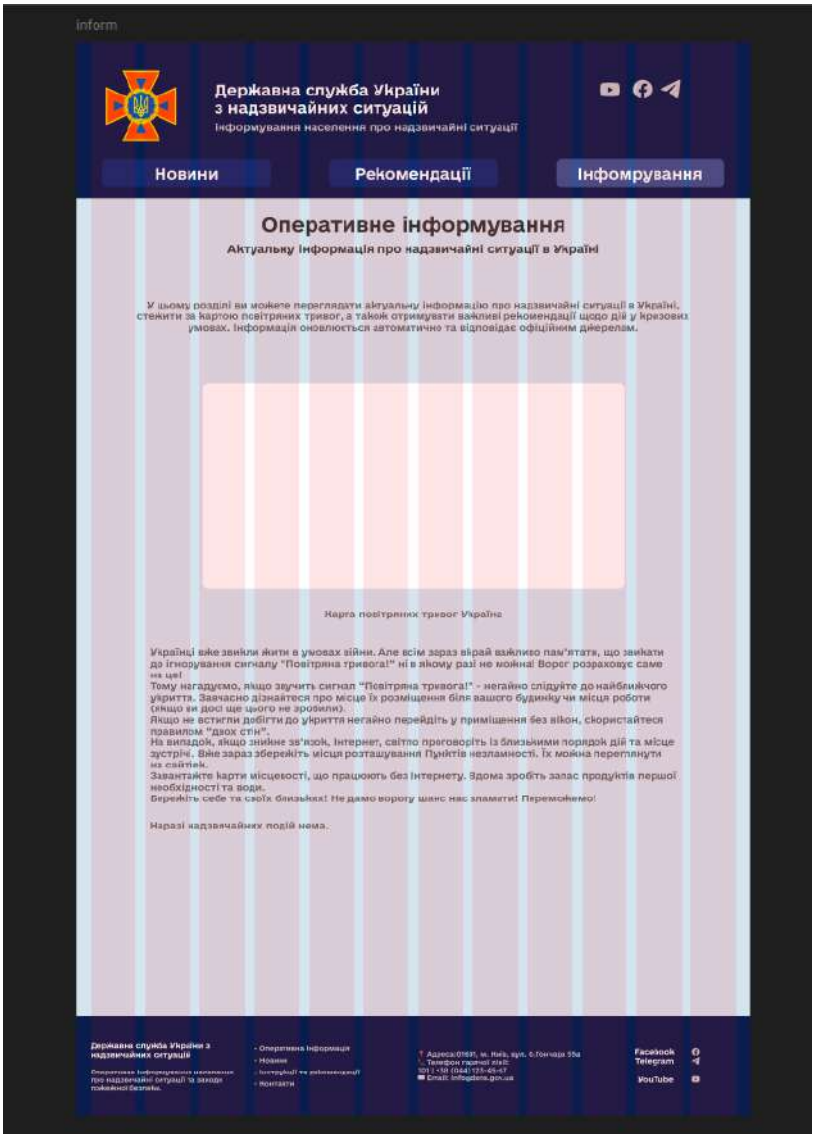


Рис. 2.3. Візуалізація макету сторінки “Інформування”

(Розроблено автором)

Ця сторінка відіграє ключову роль у виконанні головного завдання сайту - наданні оперативної та важливої інформації для захисту життя громадян. UI/UX-

проектування у Figma дозволило створити сторінку, що поєднує простоту та функціональність.

Рекомендації та інструкції це один з основних розділів сайту призначений для інформування населення про правила поведінки під час надзвичайних ситуацій. Основна мета сторінки - надання офіційних рекомендацій та інструкцій, що можуть допомогти врятувати життя в умовах НС. Дана сторінка буде показувати так само шапку та футер але наповнення її зовсім інше, це буде невелика ілюстрація, заголовок та підзаголовок. Контент поділений на компактні, повторювані блоки - картки рекомендацій. Це полегшує сприйняття і дозволяє швидко сканувати інформацію, це показано на рис. 2.4. Після натискання на неї користувача буде перекидати на повноцінну статтю, з фотоматеріалами та текстом.

Рис. 2.4. Візуалізація сторінки “Інструкції та рекомендації”

(Розроблено автором)

Розділ “Інструкції та рекомендації” - один із найважливіших на порталі, оскільки містить практичні поради для порятунку життя. Дизайн у Figma

дозволив реалізувати його у вигляді доступного, структурованого та візуально збалансованого блоку. Такий підхід значно полегшує сприйняття та допомагає користувачам швидко знайти відповідь на важливе питання.

Сторінка деталі ‘Інструкції та рекомендації’ є детальним розгортанням з рекомендацій, яка знаходиться на головній сторінці розділу “Інструкції та рекомендації” представляється як коротка картка. Перехід на цю сторінку відбувається після кліку на відповідну інструкцію. Її головна мета - надати розгорнуту, структуровану та візуально підкріплену інструкцію про дії в надзвичайних ситуаціях. Сторінка має в собі текстове наповнення та ілюстрацію контенту що підвищує сприйняття.

Цей макет є вдалим прикладом того, як одна сторінка Figma може поєднувати інформаційний матеріал, візуальну підтримку та зручний для користувача UX. Такий підхід дозволяє ефективно комунікувати критично важливу інформацію - зручно, швидко та зрозуміло. Таким чином отримаємо чітку та зрозуміло інформацію. Ось декілька прикладів як це виглядає: Як врятуватися під час пожежі візуально показано на рис. 2.5. Як діяти під час сигналу повітряна тривога показано на рис. 2.6. Як надати першу допомогу: загальні правила на рис. 2.7. Рекомендації щодо дій населення під час повені показано на рис. 2.8.

Рис. 2.5. Візуалізація сторінки “Як врятуватися під час пожежі”

(Розроблено автором)

Рис. 2.6. Візуалізація сторінки “Як діяти під час сигналу повітряна тривога”

(Розроблено автором)

Рис. 2.7. Візуалізація сторінки “Як надати першу допомогу: загальні правила”

(Розроблено автором)

Рис. 2.8. Візуалізація сторінки “Рекомендації щодо дій населення під час повені”

(Розроблено автором)

Сторінка “Новини” є інформаційним хабом порталу, де зібрані всі актуальні події, пов’язані з надзвичайними ситуаціями, заходами безпеки, діяльністю рятувальників тощо. Основна мета сторінки - оперативно інформувати користувачів про важливі події зі зручним переглядом. Це одна з найважливіших сторінок з котрою буде взаємодіяти більшість користувачів. Список новин створений у вигляді карток, вона має такий вміст як превью, заголовок, дата, невеликий ввід в новину та категорія. Ці всі пункти будуть корисні для користувача для швидкої фільтрації потрібного контенту. Клік по

картці новини відкриває повний перегляд, де користувач бачить розширений текст, фото/відео, можливо, пов’язані новини або інструкції.

Усі новини, що відображаються на цій сторінці, додаються, редагуються або видаляються через адміністративну панель. Адміністратор після авторизації переходить до зовнішньої адмін-панелі, де має доступ до розділу “Новини”. В нього є такі права як додавання новин: заповнивши форму. Редагувати існуючі новини, а також видаляти застарілу або помилкову новину. Це дає адміністратору великі можливості, після збереження змін, новина автоматично з’являється на публічній сторінці у загальному списку.

Сторінка “Новини” показано на рис. 2.9 є одним із найважливіших джерел інформації на сайті. Її структура дозволяє швидко знаходити актуальні події, а взаємодія з адміністративною панеллю - гарантує оперативність та достовірність контенту. Вдале поєднання візуального стилю, текстової стиснутості та логічної структури робить цю сторінку ефективною для виконання основного завдання сайту - інформування населення.

Рис. 2.9. Візуалізація сторінки “Новини”

(Розроблено автором)

Сторінка повного перегляду новини є логічним продовженням розділу “Новини” та відкривається після натискання на заголовок або зображення новини в загальному списку. Мета цієї сторінки - надати докладний опис події з мультимедійним супроводом, що дозволяє користувачу краще зрозуміти суть ситуації.

Сторінка повного перегляду новини показана на рис 2.10 - це ключовий елемент інформаційного блоку сайту, який дозволяє детально ознайомити користувачів із подіями, що відбулися. Грамотна структура, зображення, зрозумілий текст і мультимедійна подача сприяють кращому засвоєнню критично важливої інформації.

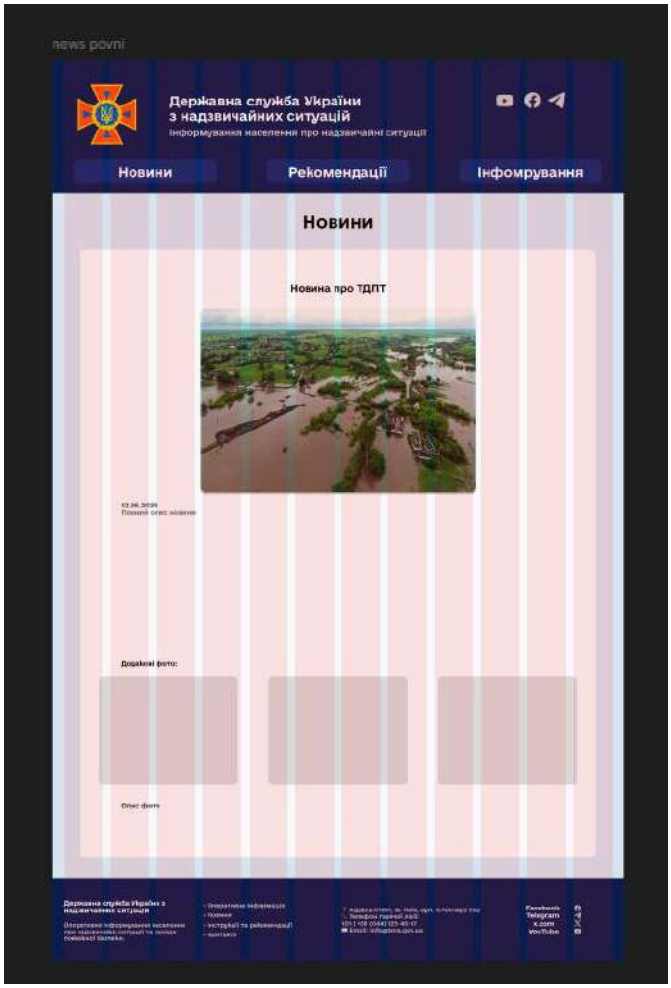


Рис. 2.10. Візуалізація сторінки “Деталі новини”

(Розроблено автором)

Таким чином у процесі проектування веб-порталу для оперативного інформування населення було створено повноцінну систему макетів у Figma, яка охоплює всі основні сторінки сайту: головну сторінку, “Новини”, “Рекомендації”, “Інформування”, а також повний перегляд новини та рекомендацій.

Кожен макет реалізовано з урахуванням сучасних принципів UI/UX-дизайну: чітка сітка, контрастна типографіка, логічна послідовність елементів, а також простота використання для різних груп користувачів. Особливу увагу було приділено для простоти інтерфейсу та інформаційності. Маємо чітку та інтуїтивно зрозумілу систему. Таким чином можемо переходити до створення зв'язків між макетами у Figma щоб чітко зрозуміти масштаби розробки.

У процесі створення веб-порталу було реалізовано інтерактивний прототип у Figma, який наочно демонструє логіку переходів між сторінками, сценарії користувацької взаємодії та загальну навігаційну архітектуру сайту. Це дозволило ще до етапу розробки перевірити зручність використання інтерфейсу, протестувати навігацію та оптимізувати потоки дій.У Figma було створено понад 10 макетів (фреймів), кожен із яких відповідає окремій сторінці або розділу сайту.

Навігація з головної сторінки. Користувач може перейти у будь-який розділ сайту, а саме Новини Рекомендації та Інформування. Всі відповідні кнопки пов'язані з відповідним розділом сайту. Для прикладу візьмемо розділ Новини кліком на кнопку або картинку веде на макет Новини, а також це все працює через футер. Такий вид переходів між макетами реалізован на кожній сторінці.

Всі взаємодії використовуються за допомогою “OnClick -> Navigate to” для демонстрації переходу між сторінками. Візуалізація всіх зв'язків виконано в синіх стрілках, як дозволяють миттєво зрозуміти логіку руху між екранами це показано на рис. 2.12.

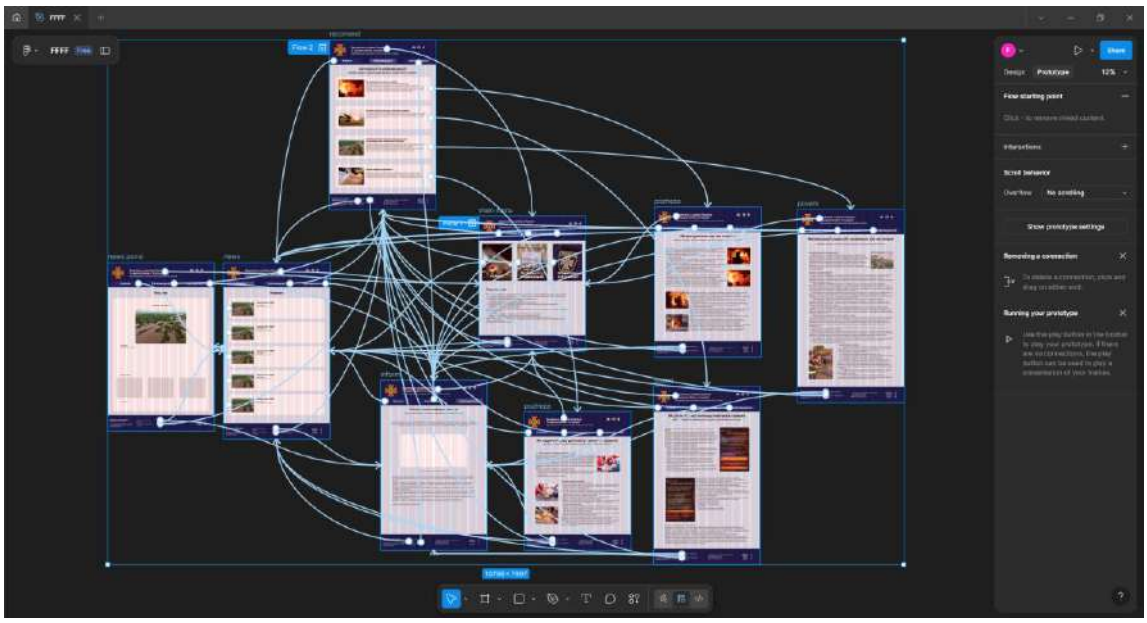


Рис. 2.12. Візуалізація всіх зв'язків між макетами

(Розроблено автором)

Таким чином це дає велику користь під час розробки, а також допоможе протестувати логіку ще до кодування, виявити зайві зв'язки між переходами, оцінити зручність інтерфейсу та визначити користувацькі сценарії.

Висновком можно сказати що інтерактивне прототипування у Figma є важливим етапом у реалізації порталу. Завдяки злагодженій структурі зв’язків, усі маршрути користувачів були перевірені й оптимізовані до початку розробки. Це суттєво зменшило ризики помилок на етапі програмування та допомогло створити логічний, цілісний і зручний веб-ресурс для інформування населення.

2.3. Підготовка технічного завдання на реалізацію веб-порталу для оперативного інформування населення про надзвичайні ситуації та заходи пожежної безпеки

У процесі реалізації веб-порталу для оперативного інформування населення про надзвичайні ситуації, важливим етапом є формування технічного завдання (ТЗ), яке забезпечує чітке розуміння цілей, вимог, обмежень і критеріїв прийняття проекту як з боку замовника, так і з боку виконавця. Нижче

представлено створення ТЗ на основі проєкту “Створення веб-порталу для оперативного інформування населення про надзвичайні ситуації та заходи пожежної безпеки”.

Для забезпечення відповідності результатів розробки були визначені такі критерії приймання.

Це в більшості новостний сайт і для нього потрібен адміністратор котрий буде додавати, редагувати та видаляти новини, оперативні повідомлення та рекомендації. Користувачу надається змога тільки бачити актуальні публікації.

1. Telegram-бот має коректно функціонувати, вкористовуючи API alerts.in.ua для інформування про повітряні тривоги.
2. Всі дані повинні зберігатися у форматі JSON.
3. Однією з умов є те щоб сайт був повністю українською мовою
4. Застосунок має безпомилково запускатися через Flask локально, а згодом і загально.
5. Код має відповідати стилістичним стандартам.

Для успішної реалізації проекту необхідно забезпечити такі ресурси:

1. Технічні засоби. До технічних засобів відноситься комп'ютер з операційною системою Windows, редактор коду Visual Studio Code, доступ для API карти тривог та бота телеграм, а також до Render для запуску сайту в загальний доступ.
2. Програмне забезпечення. Python 3.11+. Бібліотеки Flask, APScheduler, requests, telegram, json, git. Figma — для розробки UI/UX.

Для реалізації веб-порталу оперативного інформування населення про надзвичайні ситуації було використано сучасний та перевірений технологічний стек, що забезпечує функціональність.

Мови програмування: Основними мовами розробки стали Python (для серверної частини) та JavaScript (для взаємодії). Сторінки виконані за допомогою HTML5, а стилізація - з використанням CSS3.

Фреймворки та бібліотеки: Для реалізації backend-частини застосовано мікрофреймворк Flask, який забезпечує обробку запитів, маршрутизацію, роботу

з файлами та сесіями. Для реалізації Telegram-бота використовується бібліотека python-telegram-bot, а для автоматичного виконання задач — APScheduler.

Для інтеграції з картою повітряних тривог використовується офіційне API alerts.in.ua [15], що дозволяє отримувати реальні сповіщення про загрози по регіонах. Для загального доступу сайту було обрано Render.

Всі зміни в проекті відстежуються за допомогою системи Git, а код зберігається на репозиторії GitHub [29], [31].

Прототипування та створення інтерфейсу здійснено в онлайн-редакторі Figma, що дало змогу створити інтуїтивно зрозумілий, адаптивний та візуально привабливий дизайн.

З огляду на те, що продукт не передбачає складної системи користувачів або великої кількості пов’язаних таблиць, для збереження даних було обрано формат JSON-файлів, які є зручними для швидкого зчитування, редагування та зберігання невеликих обсягів структурованої інформації.

Підсумовуючи етап підготовки технічного завдання, можна сказати, що воно стало основою для подальшої реалізації веб-порталу з інформування населення про надзвичайні ситуації та заходи пожежної безпеки. У ТЗ були чітко визначені цілі, основні вимоги до функціоналу, критерії приймання результатів розробки, обмеження й необхідні ресурси.

Було сформульовано конкретні вимоги до архітектури системи, структурованості даних, технічного стеку та зовнішніх інтеграцій. Особливу увагу приділено функціонуванню Telegram-бота.

Вибір формату зберігання даних у вигляді JSON-файлів дозволяє забезпечити простоту розробки, легкість оновлення інформації та достатній рівень продуктивності для проекту. Інструменти, обрані для дизайну, розробки та хостингу, є сучасними, безкоштовними та добре документованими.

Таким чином, підготовлене технічне завдання стало ефективним інструментом для планування, управління та контролю реалізації проєкту, закладаючи основу для досягнення його основної мети — забезпечення

своєчасного, надійного та зручного інформування громадян у разі надзвичайних подій.

Висновки до розділу 2

У розділі було сформовано загальну структуру веб-порталу і визначено ролі користувача та адміністратора. Описано ключові сторінки та логіку навігації між ними. На основі виявлених потреб розроблено інтерфейс у середовищі Figma, де створено макети кожної сторінки з урахуванням зручності для користувача. Здобуті дизайн-рішення уніфіковані і спрямовані на забезпечення швидкого доступу до важливої інформації населення в надзвичайних ситуаціях.

Створено інтерактивний прототип порталу з переходами між екранами, що дозволило протестувати послідовність навігації інтерфейсу ще на етапі проектування. Цей прототип дав змогу наочно відпрацювати сценарії використання сайту і покращити UX перед початком програмування. Одночасно підготовлено детальне технічне завдання, в якому описано весь функціонал системи, архітектуру додатку, вибір технологічного стеку та критерії приймання. У ТЗ закладено вимоги до механізмів підписки на повідомлення, адміністрування контенту, а також безпеки і доступності інформації. Така комплексна специфікація гарантує, що подальша розробка буде узгодженою та цільовою.

Таким чином, проведена у розділі 2 робота узагальнює результати аналітики, дизайну й технічного планування та підкреслює логічну послідовність розробки: від побудови структури та сценаріїв до оформлення візуального дизайну й докладного технічного завдання. Узгоджені макети та інтерактивний прототип служать надійною основою для реалізації, а детальне ТЗ гарантує, що всі функціональні вимоги будуть реалізовані у процесі програмування. Виконана робота свідчить про те,шо можна перейти до практичної стадії розробки.

РОЗДІЛ 3
РЕАЛІЗАЦІЯ ВЕБ-ПОРТАЛУ ДЛЯ ОПЕРАТИВНОГО ІНФОРМУВАННЯ НАСЕЛЕННЯ ПРО НАДЗВИЧАЙНІ СИТУАЦІЇ ТА ЗАХОДИ ПОЖЕЖНОЇ БЕЗПЕКИ

3.1. Вибір технологій розробки сайту для оперативного інформування населення про надзвичайні ситуації та заходи пожежної безпеки

Для створення веб-порталу оперативного інформування населення про надзвичайні ситуації, важливо було вибрати мову програмування, яка здатна забезпечити високу функціональність, простоту в розробці, сумісність із різними пристроями та браузерами. Ретельно оцінивши можливі варіанти, було обрано JavaScript, з огляду на низку переваг цієї технології, які особливо підходять для реалізації даного проекту.

JavaScript - високорівнева мова програмування, яка підтримує імперативний, функціональний, подієво-орієнтований підходи [2], [11], [12]. Вона має динамічну типізацію та застосовується для запису послідовних операцій. Такі послідовності зазвичай інтерпретуються, а не компілюються, а тому не потребують додаткових програм чи інструментів перетворення в інший рівень кодування. Кожен веб-застосунок чи сайт побудований з використанням трьох технологій - HTML, CSS та JavaScript. Остання виступає мозком розробки й відповідає за інтерактивність й взаємодію з користувачем.

HTML, CSS, JavaScript - це три кити, на яких працюють веб-застосунки. HTML - каркас, CSS - приємне візуальне оформлення (в CSS3 з’явилась можливість реалізовувати й анімації), а JavaScript - логіка, інтерактивність та взаємодія з користувачем. Коли на сторінці ви бачите динамічні елементи, з якими можна взаємодіяти - вмикати аудіо- або відео програвач, будувати маршрут на карті тощо - можете бути впевнені, що не обійшлося без JavaScript. Тож Front-end або JavaScript-інженеру (розробнику) потрібні всі три технології.

Основні особливості цієї мови - динамічність, гнучкість роботи з функціями та універсальність. Вона підтримується всіма сучасними браузерами,

легко інтегрується з версткою (HTML) та дає змогу налаштувати комунікацію з сервером. Має також багато переваг:

1. Тип даних змінної в JavaScript визначається автоматично під час присвоєння значення. Це означає, що одна змінна може зберігати числа, текст, об’єкти та інші типи даних без необхідності заздалегідь задавати цей тип.
2. У JavaScript функції є гнучкими. Їх можна не тільки викликати, але й передавати іншим функціям як аргументи, повертати з функцій, а також привласнювати значенням змінних. Це дозволяє створювати більш гнучкий та зрозумілий код.
3. Методологія об’єктно-орієнтованого програмування дає змогу представити програму у вигляді сукупності об’єктів.
4. Частина логіки додатку може виконуватись на стороні користувача (у браузері). Це означає, що не всі запити мають оброблятись сервером, що зменшує навантаження на сервер, знижує час очікування для користувача і робить сайт більш інтерактивним.
5. JavaScript має розвинену інфраструктуру та активну спільноту. Так, веб-розробники можуть працювати з великою кількістю бібліотек і фреймворків як React, Angular і Vue, декількома пакувальниками, як Webpack, Gulp, та допоміжними бібліотеками як Lodash, axios, та іншими.

З одного боку, JavaScript - це просто, оскільки вже в браузері можна побачити результат. З іншого - це постійні оновлення, за якими потрібно слідкувати та аналізувати.

Варто вказати й на декілька обмежень. Основне - це робота з файловою системою, тобто недоступне зчитування файлів. Крім того, JavaScript не підтримує віддалений доступ до системи, а тому мову незручно використовувати для мережевих застосунків.

Особливу увагу потрібно звертати на захист на стороні клієнта, адже деякі недогляди й помилки в коді можуть використати зловмисники. Крім того, код JavaScript також може по-різному інтерпретуватися браузерами, а старі версії як

IE 9 взагалі не підтримує. Через те, що деякі браузери зчитують JavaScript-код дещо інакше, сайт чи веб-застосунок може некоректно відображатися чи працювати. Динамічну типізацію також іноді вважають недоліком. Але це можна виправити, використовуючи TypeScript (транспайлер для JavaScript).

HTML (HyperText Markup Language) у проекті виконує роль основи структури інтерфейсу [3], [13]. За допомогою HTML визначається розмітка сторінки - розташування заголовків, абзаців, форм та інших елементів. Саме HTML відповідає за групування контенту, забезпечуючи ясну організацію інформації. Завдяки стандартній підтримці у всіх браузерах HTML гарантує коректне відображення сторінки на різних платформах. Крім того, в реалізації великих документів HTML-коду може знадобитися багато розмітки навіть для простих елементів, що ускладнює підтримку та модифікацію структури. Таким чином, HTML є базовим каркасом веб-порталу - простим і зрозумілим, але не здатним самостійно забезпечити інтерактивність чи візуальне оформлення.

CSS (Cascading Style Sheets) відповідає за зовнішній вигляд інтерфейсу [4], [5], [6]. Ця мова стилізації дозволяє задавати кольори, шрифти, відступи і загалом оформлення будь-яких HTML-елементів. За допомогою CSS відокремлюється структура від подання - усі правила оформлення містяться в окремому стилістичному шарі. Завдяки цьому можна централізовано змінювати дизайн: наприклад, налаштувати кольори фону кнопок чи розмір шрифтів у багатьох місцях одночасно. CSS також забезпечує гнучкість макета - за допомогою таких модулів, як Flexbox або Grid, компоненти інтерфейсу автоматично підлаштовуються під різні розміри екрана. Використання медіа-запитів дозволяє створити адаптивний дизайн: при зменшенні ширини вікна макет може змінюватися. У підсумку CSS дає змогу отримати сучасний та привабливий інтерфейс, який однаково добре підлаштовується під різні розміри екранів. Таким чином, CSS є незамінним для оформлення веб-порталу і створення адаптивного дизайну, проте вимагає ретельного перевірки в різних середовищах.

HTML**:** забезпечує семантичну структуру сторінки та є стандартом, що підтримується всіма браузерами, простий і зрозумілий для розробника. Він

легкий і швидко завантажується, адже передається у вигляді компактного тексту. Водночас HTML є статичним - не генерує динамічного контенту сам по собі, тому для будь-якої логіки чи взаємодії потрібні додаткові скрипти або технології. Крім того, для формування навіть простого інтерфейсу іноді доводиться писати багато розмітки.

CSS**:** дозволяє гнучко і централізовано оформлювати інтерфейс - задавати кольори, шрифти, анімації та розташування елементів. За його допомогою легко створювати адаптивні макети (через Flexbox/Grid та медіа-запити). При цьому CSS-селектори і спадкування роблять підтримку стилів ефективнішими: багато сторінок може успадковувати базовий стиль із одного місця. До недоліків можна віднести необхідність контролювати каскадність правил і кросбраузерність: різні браузери можуть по-різному інтерпретувати деякі властивості, а складні переплетення селекторів можуть ускладнювати відлагодження стилів.

На основі цього можна зробити висновок, що кожен з розглянутих інструментів має чітко визначену роль і свої переваги: HTML формує скелет сторінки, CSS задає стиль і адаптивність, а JavaScript дає функціонал опрацювання. Ці всі компоненти потребують об’єднання в єдине ціле, щоб отримати повноцінний, багатофункціональний та зручний веб-портал.

Для реалізації серверної частини веб-порталу було обрано мову програмування Python - одну з найпопулярніших мов у сучасній розробці. Основними критеріями вибору стали: зрозумілий синтаксис, велика кількість бібліотек, активна спільнота та підтримка мікро фреймворків, які чудово підходять для проектів середнього масштабу.

Python є мовою високого рівня, що поєднує простоту вивчення з широкими можливостями реалізації проектів. Його популярність у веб-розробці пояснюється підтримкою таких фреймворків, як Flask, Django, FastAPI, які дозволяють швидко створювати серверні рішення. У цьому проекті буде використано Flask - легкий та гнучкий мікрофреймворк, який забезпечить потрібну структуру маршрутизації та дозволив реалізувати необхідну серверну логіку з мінімальним обсягом коду.

Python є кращим вибором для реалізації серверної частини проекту завдяки простоті, гнучкості, широким можливостям обробки даних і підтримці інтеграцій. Незважаючи на окремі обмеження, Python є сучасним, зручним та практичним рішенням для серверної логіки у подібних веб-порталах.

3.2 Реалізація структури сайту для оперативного інформування населення про надзвичайні ситуації та заходи пожежної безпеки

В наш час суспільство потребує швидкого, достовірного та зручного доступу до інформації про надзвичайні ситуації. Особливо в умовах воєнного стану, надзвичайних погодних умов, техногенних катастроф та інших критичних подій, які можуть нести загрозу для життя та здоров’я громадян. Саме тому створення спеціалізованого веб-порталу для оперативного інформування населення стало актуальним завданням, що лягло в основу цієї дипломної роботи.

Етап реалізації структури сайту є ключовим у процесі створення інформаційного ресурсу, адже саме він формує каркас системи, на якому лежить весь функціонал, від візуального представлення до обробки даних. Від якості реалізації залежить зручність користування порталом, доступу до публікацій, а також можливість адміністрування.

У даному проєкті структура сайту формувалася з урахуванням попередньо розробленого макету в Figma та технічного завдання, в якому були чітко визначені основні модулі системи, функціональні блоки, типи користувачів, а також вимоги до зберігання, обробки й виводу інформації.

Було враховано потреби як звичайного користувача, так і адміна. Перший має доступ до перегляду оперативних повідомлень, новин та рекомендацій, другий - можливість створення, редагування та видалення відповідних публікацій через адміністративну панель.

Архітектура проекту реалізована за принципом розділення на клієнтську та серверну частину. Клієнтська частина створена з використанням HTML5, CSS3 та JavaScript, що забезпечує адаптивність, зручність навігації та

інтерактивність інтерфейсу. Серверна частина побудована на мікрофреймворку Flask (Python) [18], [25], який дозволяє ефективно обробляти HTTP-запити, керувати сесіями, зчитувати та оновлювати дані у форматі JSON [16], [17], [21]. Такий підхід дозволив забезпечити збереження даних без використання повноцінної СУБД, що є доцільним для середніх обсягів контенту.

Окремо варто зазначити інтеграцію з зовнішніми сервісами, зокрема API платформи alerts.in.ua для отримання інформації про повітряні тривоги, а також використання Telegram-бота для дублювання сповіщень у месенджері. Така функціональність значно підвищує ефективність інформування та охоплення аудиторії.

Загалом, структура сайту буде реалізована відповідно до принципів модульності. Кожен функціональний блок - від новин до карт тривог - є логічно відокремленим. Використані технології є сучасними, відкритими та активно підтримуються ком'юніті, що також сприяє довгостроковість проекту.

Реалізація починається зі створення головної сторінки сайту, а саме index.html. Головна сторінка сайту виконує роль центральної навігаційної порталу сайту, з якого користувач переходить до основних інформаційних розділів веб-порталу: новин, оперативного інформування та рекомендацій. Це перший інтерфейс, який бачить користувач, тому дизайн і структура мають бути інтуїтивно зрозумілими та візуально привабливими.

Структура сторінки поділяється на 4 основні блоки, кожен з котрих має свої функції. Початок сайту це шапка (header) сторінки. Це основне меню сайту котре буде збережено на всіх інших сторінках. Його функція це навігація між іншими сторінками та перенаправлення на соціальні мережі (Facebook, YouTube та телеграм-бот).

Дизайн сторінки був створений з образа макету Figma. Реалізації шапки сайту почалася з того що було створено головний блок де будуть знаходитись всі елементи шапки. З лівої частини шапки знаходиться лого(емблема ДСНС), заголовок “Державна служба України з надзвичайних ситуацій”, підзаголовок “інформування населення про надзвичайні ситуації”, три функціональні

навігаційні кнопки “Новини”, “Рекомендації” та “Інформування”, а також з правої частини шапки було створено три навігаційні емблеми соціальних мереж “Facebook”, “Telegram-bot” та “YouTube”. Таким чином ми отримали гарну шапку показану на рис. 3.1, та з охайним кодом html, що показано на рис. 3.2

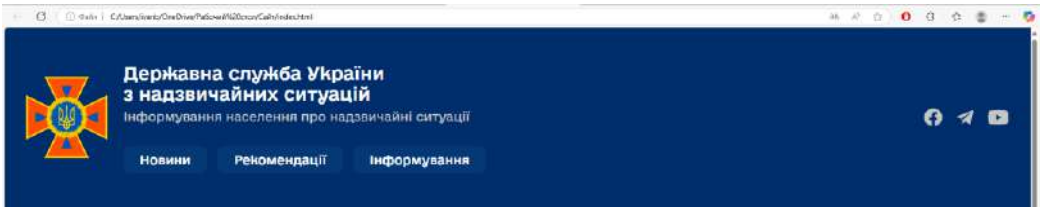


Рис. 3.1. Візуалізація частини сайту “Header”

(Розроблено автором)

```
index.html > html > body > header.main-header > div.left-block > div.titles
<!DOCTYPE html>
<html lang="uk">
<head>
  <meta charset="UTF-8">
  <title>ДСНС України</title>
  <link rel="stylesheet" href="style.css">
  <link href="https://fonts.cdnfonts.com/css/e-ukraine-head" rel="stylesheet">
</head>
<body>
  <header class="main-header">
    <div class="left-block">
      <a href="index.html">
        <img src="static/icons/logo.svg" alt="ДСНС України" class="logo">
      </a>
      <div class="titles">
        <h1 class="main-title">Державна служба України<br>з надзвичайних ситуацій</h1>
        <p class="subtitle">Інформування населення про надзвичайні ситуації</p>
        <div class="menu-buttons">
          <a href="news.html" class="menu-button">Новини</a>
          <a href="recommendations.html" class="menu-button">Рекомендації</a>
          <a href="info.html" class="menu-button">Інформування</a>
        </div>

      </div>
    </div>

    <div class="right-social">
      <a href="https://www.facebook.com/DSNS.GOV.UA/"><img src="static/icons/facebook.svg" alt="Facebook"></a>
      <a href="https://t.me/Air_Alerts_Urkraine_bot"><img src="static/icons/telegram.svg" alt="Telegram"></a>
      <a href="https://www.youtube.com/mnsgovua"><img src="static/icons/youtube.svg" alt="YouTube"></a>
    </div>
  </header>
```

Рис. 3.2. Візуалізація коду HTML для шапки сайту

(Розроблено автором)

Наступним блоком сайту є його наповнення. Сторінка має три великих функціональних кнопок з фото та текстом, які переводять користувача на певну сторінку. Трохи нижче цього фрагменту знаходиться опис порталу, що дає користувачу розуміння про що цей сайт. Все це оформлено охайно та стильно показано на рис. 3.3. Таким чином отримаємо фрагмент сайту, який має функціональні кнопки та опис порталу, фрагмент коду показано на рис. 3.4.

Рис. 3.3. Візуалізація шапки та наповнення сторінки

(Розроблено автором)

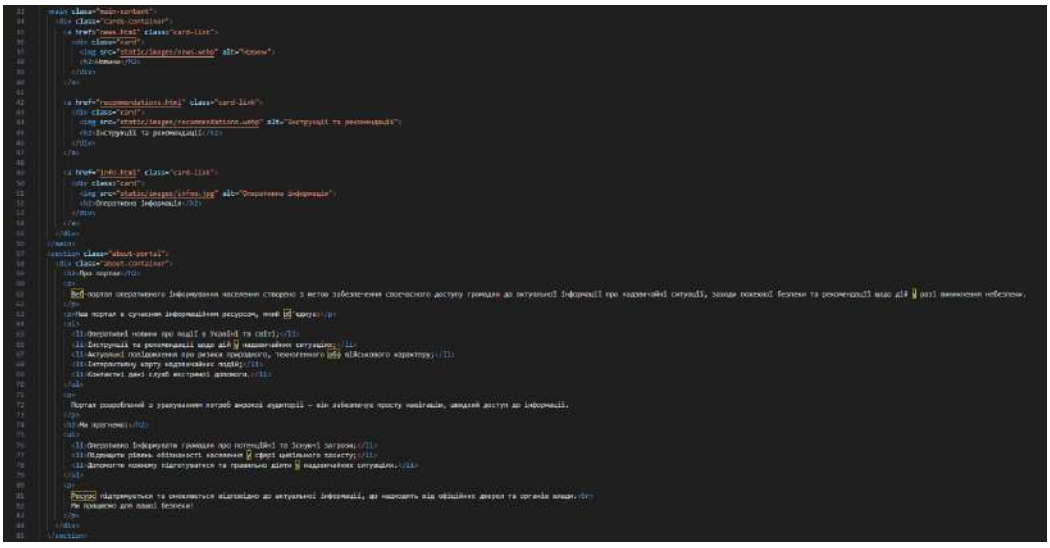


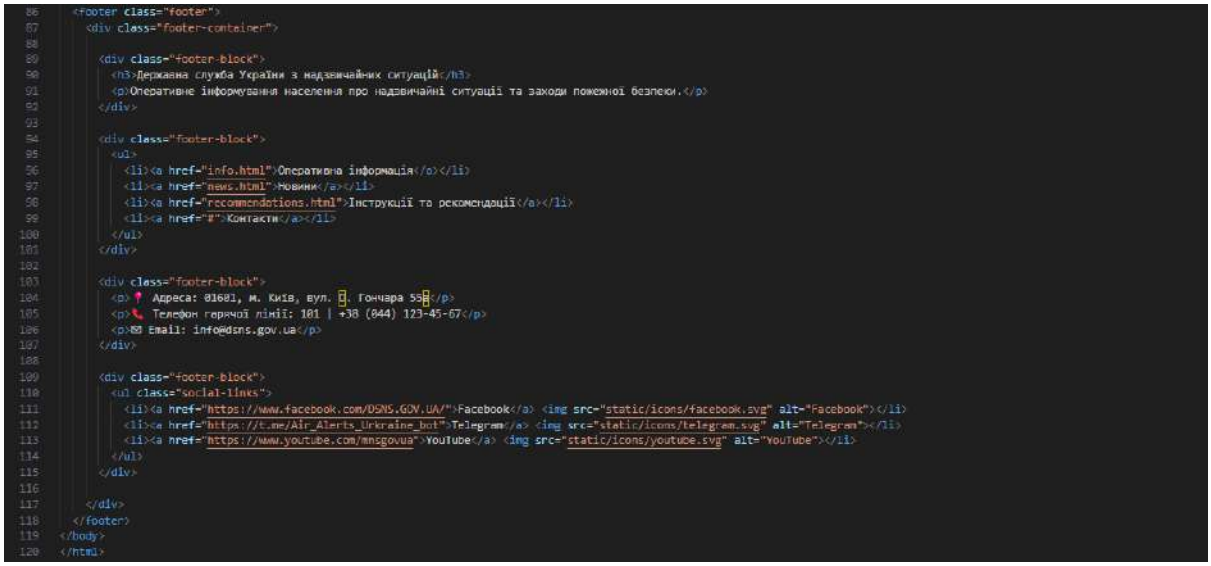
Рис. 3.4. Візуалізація коду HTML для основної частини сторінки

(Розроблено автором)

В нижній сторінки сайту було розроблено функціональний футер котрий буде як і шапка відображатись на інших сторінках. З лівої частини футеру є заголовок сайту та підзаголовок. Також він має чотири функціональні кнопки для переходу між сторінками “Новини” “Оперативна інформація” “Інструкції та рекомендації” “Контакти”, правіше від кнопок є відділ з адресою, телефон та електронна пошта, всім цим завершує функціональні кнопки для перенаправлення на соціальні мережі. Всі ці фрагменти візуально гарно виглядають, показано на рис. 3.5 та мають свій функціональний код показано на рис. 3.6.

Рис. 3.5. Візуалізація функціонального футеру

(Розроблено автором)



```
<footer class="footer">
  <div class="footer-container">

    <div class="footer-block">
      <h3>Державна служба України з надзвичайних ситуацій</h3>
      <p>Оперативне інформування населення про надзвичайні ситуації та заходи пожежної безпеки.</p>
    </div>

    <div class="footer-block">
      <ul>
        <li><a href="info.html">Оперативна інформація</a></li>
        <li><a href="news.html">Новини</a></li>
        <li><a href="recommendations.html">Інструкції та рекомендації</a></li>
        <li><a href="#">Контакти</a></li>
      </ul>
    </div>

    <div class="footer-block">
      <p>📍 Адреса: 01601, м. Київ, вул. О. Гончара 55б</p>
      <p>📞 Телефон гарячої лінії: 101 | +38 (044) 123-45-67</p>
      <p>✉ Email: info@dsns.gov.ua</p>
    </div>

    <div class="footer-block">
      <ul class="social-links">
        <li><a href="https://www.facebook.com/DSNS.GOV.UA/">Facebook</a> <img src="static/icons/facebook.svg" alt="Facebook"></li>
        <li><a href="https://t.me/Air_Alerts_Ukraine_bot">Telegram</a> <img src="static/icons/telegram.svg" alt="Telegram"></li>
        <li><a href="https://www.youtube.com/mnsgovua">YouTube</a> <img src="static/icons/youtube.svg" alt="YouTube"></li>
      </ul>
    </div>

  </div>
</footer>
</body>
</html>
```

Рис. 3.6. Візуалізація коду функціонального футеру

(Розроблено автором)

Таким чином, головна сторінка сайту була реалізована відповідно до розробленого дизайн-макету у Figma, дотримуючись принципів зручності, доступності та інформативності. Яка навігаційну та презентаційну функцію.

Вона включає розділені блоки: шапку з меню, основний контент з інтерактивними елементами та футер з контактною інформацією. Усі елементи мають єдине стилістичне оформлення, що відповідає стандартам сучасного веб-дизайну. Головна сторінка закладає фундамент для всієї структури сайту, забезпечуючи зрозумілу навігацію та привабливу подачу вмісту. Її реалізація є важливою частиною створення повноцінного інформаційного порталу.

Наступним кроком реалізації структури стала сторінка “Оперативне інформування”. Це є однією з найголовніших та корисних сторінок. Вона має в собі функціональні можливості у вигляді інтерактивних елементів як карта повітряних тривог та оперативні оповіщення.

Реалізація сторінки почалась с того що ми копіюємо шапку головної сторінки, щоб зберегти однотипність дизайну та зекономити час. Далі йде основний фрагмент сайту, де буде відображатись карта повітряних тривог, опис та оперативні оповіщення.

В верхній частині сторінки починається з того що є заголовок а саме надписа “Оперативне інформування” та підзаголовок “Актуальна інформація про надзвичайні ситуації в Україні”. Нижче підзаголовка йде короткий вступний текст функції цієї сторінки та карта повітряних тривог. Після цього короткий текст і під ним знаходяться оперативні повідомлення. Оперативні повідомлення будуть виводитись за допомогою JavaScript, все це буде реалізовано в адмін панелі. Таким чином отримаємо дуже гарний результат на рис 3.7 та функціональний дизайн на рис. 3.8. І як і на головній сторінки в кінці є футер.

Рис. 3.7. Візуалізація сторінки “Оперативна інформація”

(Розроблено автором)

Рис. 3.8. Візуалізація коду HTML для сторінки “Оперативна інформація”

(Розроблено автором)

Сторінка “Оперативне інформування” є однією з найголовніших частин веб-сайту, оскільки забезпечує доступ до актуальної інформації про надзвичайні ситуації. Реалізація дозволяє користувачам у зручній формі переглядати поточну карту повітряних тривог в Україні, а також оперативні повідомлення.

Використання готової шапки та футера з головної сторінки сприяє єдності стилю та зручності навігації, що покращує користувацький досвід. Інтеграція інтерактивної карти за допомогою вбудованого iframe дозволяє безперервно відображати інформацію з надійного джерела, а функція оперативних

повідомлень, що буде реалізована через JavaScript і оновлюється в адмін-панелі, забезпечує контент в режимі реального часу. Таким чином, реалізована сторінка не лише відповідає сучасним стандартам, а й виконує важливу функцію - забезпечення громадян оперативною інформацією під час небезпечних ситуацій.

Сторінка новин та делалі новини - це комбо з двох сторінок як не може працювати одне без одного. Одна з них показує список новин, а інша показує повний текст новини зі списку.

Додавання новин у список та формування деталей новини робиться за допомогою адмін панелі реалізовану через JavaScript. Адміністратор пише заголовок, підзаголовок, завантажує фото превью, додає категорії, пише повний текст новини та додає додаткові фото.

Початок сторінки новини зустрічає нас, як і інші сторінки шапкою, з заголовку новини, після чого йде фільтри новин. Сама фільтрація новин йде по тексту та по категоріям це зображено на рис. 3.9. Під фільтром знаходиться список новин і після нього йде футер це зображено на рис. 3.10.

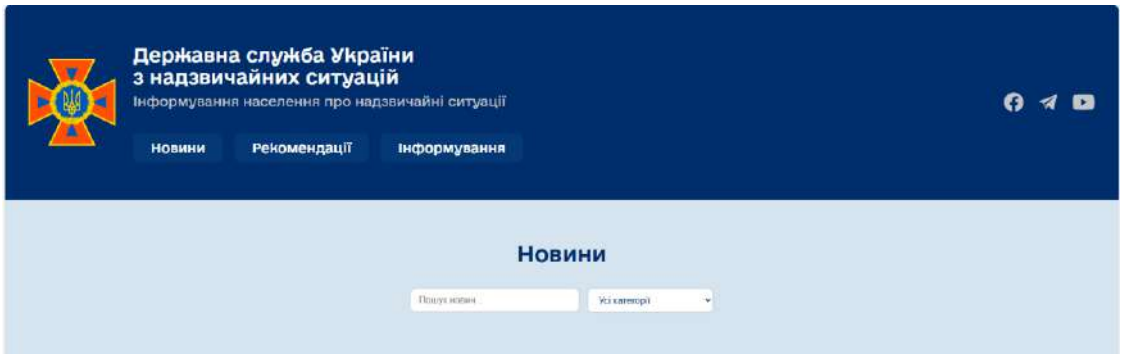


Рис. 3.9. Візуалізація початку списка з новинами

(Розроблено автором)

```
<main class="main-content recommendations-wrapper">
  <div class="recommendations-container">
  <h2 class="recommendations-heading">Новини</h2>
<section class="news-filters">
  <input type="text" id="searchInput" placeholder="Пошук новин..." class="search-box">
  <select id="filterCategory" class="filter-select">
    <option value="">Усі категорії</option>
    <option value="Пожежа">Пожежа</option>
    <option value="Стихійні лиха">Стихійні лиха</option>
    <option value="ДТП">ДТП</option>
    <option value="Клас безпеки">Клас безпеки</option>
    <option value="Поліція">Поліція</option>
    <option value="Надзвичайна ситуація">Надзвичайна ситуація</option>
    <option value="Попередження">Попередження</option>
    <option value="Рятувальні операції">Рятувальні операції</option>
  </select>
</section>
<main>
  <section class="news-list" id="newsList"></section>
</main>
</div>
</main>
```

Рис. 3.10. Візуалізація коду сторінки “Новини”

(Розроблено автором)

Сам список формується з дати, превью, заголовка, підзаголовком та категорій (наприклад: ДТП, Пожежа, Поліція). Таким чином отримуємо дуже охайний вигляд новин показано на рис 3.11.

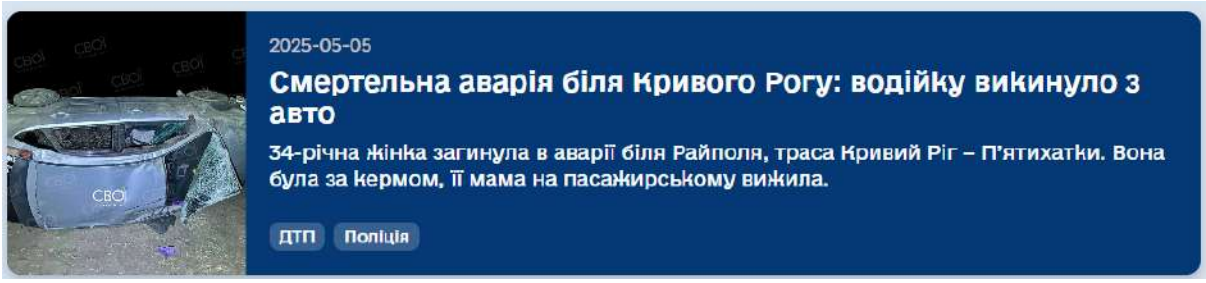


Рис. 3.11. Візуалізація як виглядає новина в списку

(Розроблено автором)

Деталі новини формується адміністратором разом зі предоглядом новини. Сторінка новини, як і інші мають шапку та футер, зустрічає нас блоком в якому зображено заголовком новини, датою, категоріями, текстом описаним новину та додаткові фото. Це робить сторінку деталі новини максимально інформативною та зручною це показано на рис. 3.12. Майже все для виводу детальної новини робить скрипт JavaScript і через нього і робиться дизайн візуалізація коду як це працює на рис. 3.13.

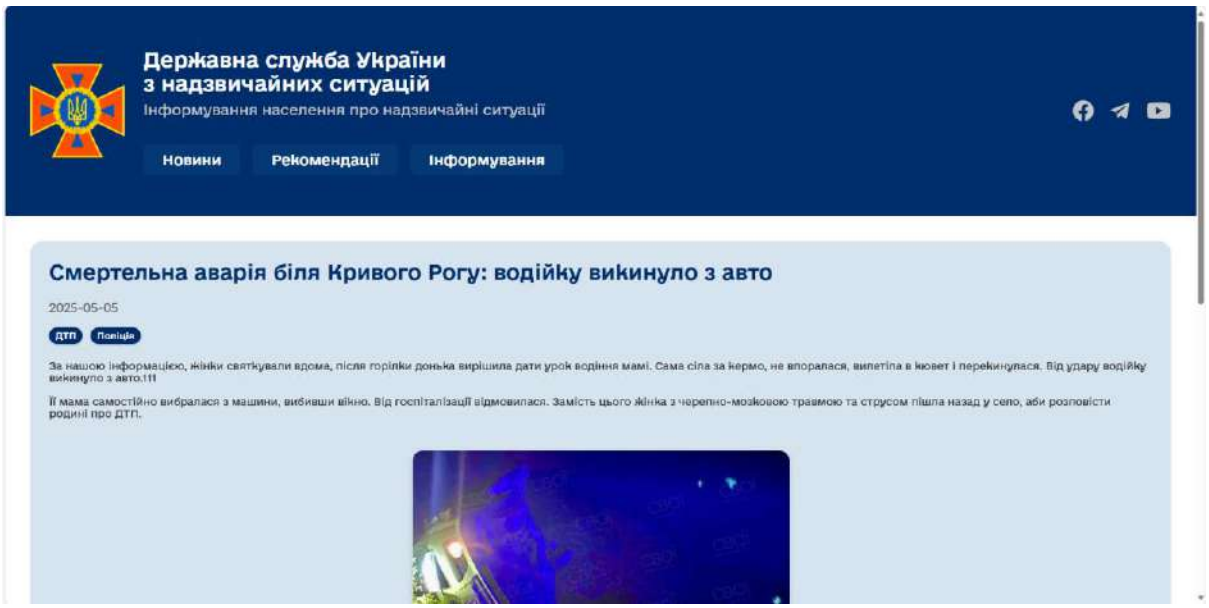


Рис. 3.12. Візуалізація детальної новини

(Розроблено автором)

```
37  <main>
38    <section class="news-detail" id="newsDetail">
39      <!-- Деталі новини підвантажуються через JS -->
40    </section>
41  </main>
42
43  <script src="admin.js"></script>
```

Рис. 3.13. Візуалізація коду HTML для детальної новини

(Розроблено автором)

Розділ “Новини” у поєднанні зі сторінкою детальної новини утворює єдину, логічно зв’язану систему. Завдяки реалізації списку новин із коротким описом, категоріями, превью та датою - користувачі мають змогу швидко ознайомитись з актуальною подією. Перехід на сторінку детальної новини забезпечує повноцінне ознайомлення з усією інформацією, включаючи текст, додаткові фото та контекст події.

Таке розділення контенту підвищує зручність використання ресурсу. Реалізація фільтрації новин за ключовими словами та категоріями дозволяє користувачу швидко знаходити потрібну інформацію.

Адміністративна частина системи відіграє ключову роль у забезпеченні функціонування цього розділу, дозволяючи адміністратору оперативно

публікувати події, редагувати та видаляти новини. Таким чином, блок новин не лише підвищує інформативність порталу, а й забезпечує його актуальність і надійність. Реалізація сторінок “Новини” та “Деталі новини” створює повноцінний, зручний та функціональний інформаційний розділ, який є невід’ємною частиною всього веб-порталу.

Одна із ключових напрямків сайту є також інструкції та рекомендації дій під час НС. Це є дуже корисним розділом порталу де будуть знаходитись актуальні рекомендацій. Так само як із новинами він створений з двох сторінок. Основна сторінка де є список рекомендацій та інструкцій, а також і детальна стаття кожної з тем.

Основна сторінка зі списком інструкцій оформлення так само як і новини, але тільки з невеликими корекціями. А саме, список рекомендація оформлена фото (превью), датою, заголовком і підзаголовком це показано на рис. 3.14. Це дає користувачу чітке розуміння про що йдеться в статті. Так само як із новинами додавання рекомендації та візуал прописаний в JavaScript візуалізація коду показана на рис. 3.15.

Рис. 3.14. Візуалізація сторінки “Офіційні рекомендації”

(Розроблено автором)

```
36  <main class="main-content recommendations-wrapper">
37    <div class="recommendations-container">
38      <h2 class="recommendations-heading">Офіційні рекомендації</h2>
39      <section id="recommendationsList" class="news-list"></section>
40    </div>
41  </main>
42  <script src="admin.js"></script>
```

Рис. 3.15. Візуалізація коду HTML для сторінки “Офіційні рекомендації”

(Розроблено автором)

Деталі рекомендації майже так само виглядають як і деталі новини, але з невеликими змінами. Сторінка починається з того що користувача зустрічає, заголовок, дата, повний текст рекомендації\інструкції та фото але під кожним фото є невеликий текст який описує фото це показано на рис. 3.16. Ці всі пункти дають користувачеві чітку та зрозумілу інформацію, що сприяє швидкому засвоєнню через візуальний контент ілюстрація показана на рис. 3.17.

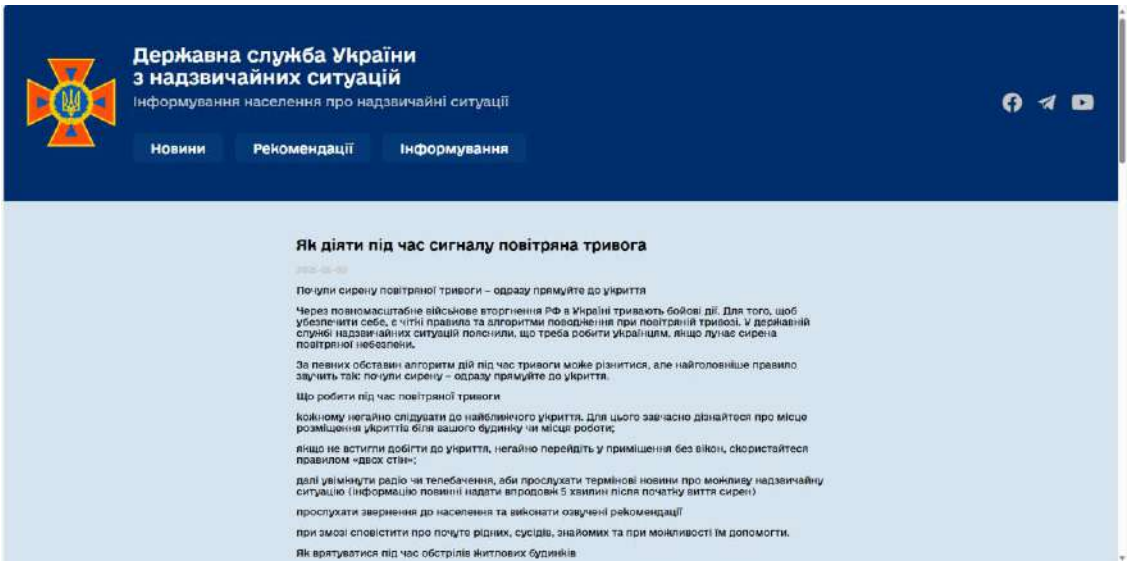


Рис. 3.16. Візуалізація однієї з інструкції

(Розроблено автором)

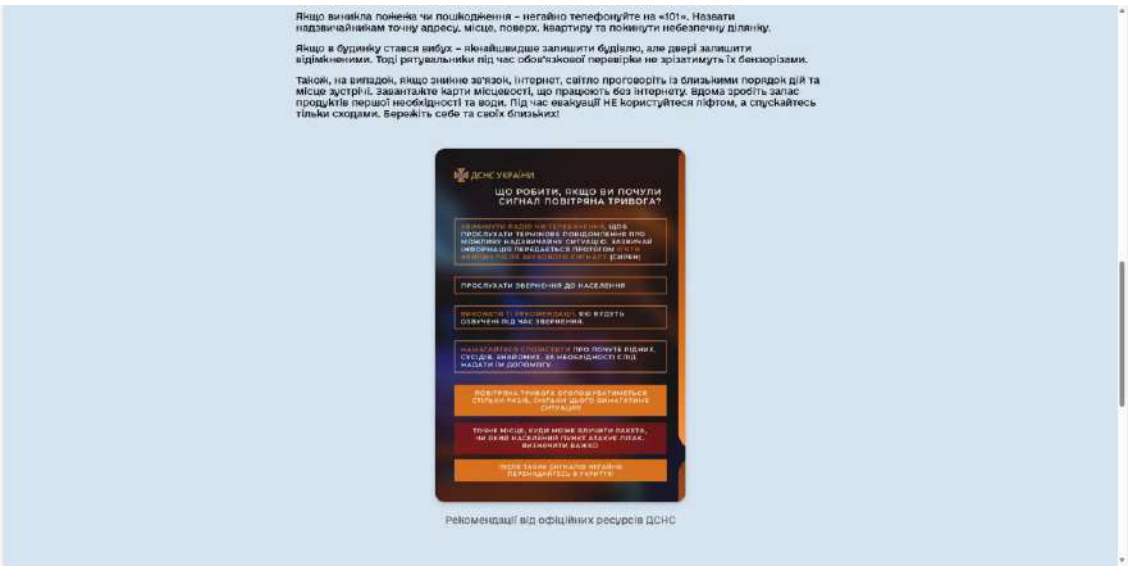


Рис. 3.17. Візуалізація фото та підписа під ним

(Розроблено автором)

Так само як і з іншими все функціонує через JavaScript, тому весь візуал прописується в скрипті це показано на рис. 3.18.

```
37  <main class="main-content">
38      <section id="recommendationDetail" style="max-width: 900px; margin: 0 auto;"></section>
39  </main>
40  <script src="admin.js"></script>
41  <footer class="footer">
```

Рис. 3.18. Візуалізація коду HTML для детальної рекомендації

(Розроблено автором)

Розділ “Інструкції та рекомендації” є частиною порталу, адже надає користувачам корисні поради щодо дій під час надзвичайних ситуацій. Його реалізовано з двох сторінок: загального списку та детальної статті. Такий підхід дозволяє користувачу швидко переглядати теми й заглиблюватися в деталі за потреби.

Кожна рекомендація має прев’ю, заголовок, дату та короткий опис, а в детальній версії - повний текст, фото та підписи під ними, що покращує сприйняття інформації. Адміністратор може легко додавати та редагувати рекомендації або їх видаляти.

Таким чином, розділ забезпечує простий доступ до важливої інформації та виконує освітню функцію для підвищення безпеки населення.

Авторизація - це перехідний та ключовий елемент між звичайним переглядом та панеллю адміна. Панель авторизації запитує логін та пароль це показано на рис. 3.19 у разі невірного паролю видає помилку візуалізація цього показується на рис. 3.20, у разі успіху перекидає на панель адміністратора.

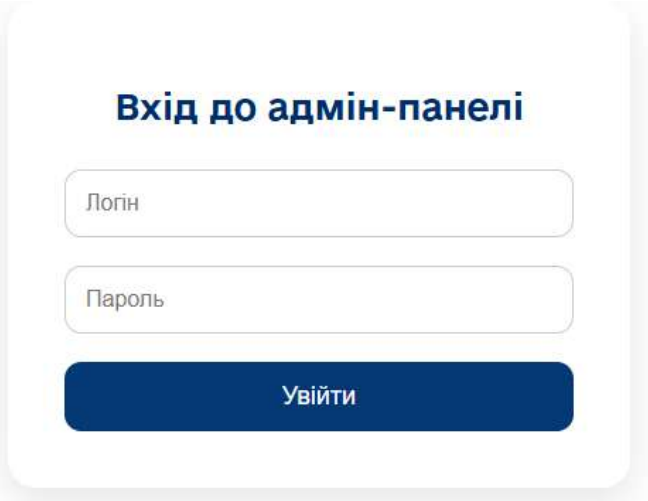


Рис. 3.19. Візуалізація панелі “Авторизації” до адмін панелі

(Розроблено автором)

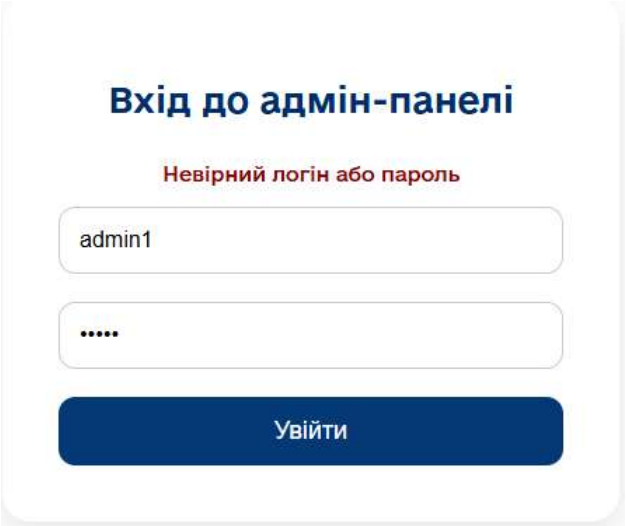


Рис. 3.20. Візуалізація невірного входу

(Розроблено автором)

Весь функціонал авторизації реалізован за допомогою функцій JavaScript. Це ключовий елемент який надає доступу до адміністративної панелі. Таким чином код самого HTML виглядає так як на рис. 3.21.

```
login.html > html
<!DOCTYPE html>
<html lang="uk">
<head>
  <meta charset="UTF-8">
  <title>Вхід до адмін-панелі</title>
  <link rel="stylesheet" href="style.css">
  <link href="https://fonts.cdnfonts.com/css/e-ukraine-head" rel="stylesheet">
</head>
<body>
  <div class="login-container">
    <div class="login-box">
      <h2>Вхід до адмін-панелі</h2>
      <div id="error" class="error-message"></div>
      <input type="text" id="username" placeholder="Логін">
      <input type="password" id="password" placeholder="Пароль">
      <button onclick="login()">Увійти</button>
    </div>
  </div>
  <script src="admin.js"></script>
</body>
</html>
```

Рис. 3.21. Візуалізація коду HTML для панелі Авторизації в адміністративну панель

(Розроблено автором)

Авторизації є важливим елементом системи безпеки сайту, оскільки фільтрує звичайних користувачів від адмін функціоналу. Він забезпечує доступ лише для уповноважених осіб, які мають логін і пароль. Реалізація авторизації через JavaScript у поєднанні з серверною перевіркою дозволяє ефективно контролювати вхід, виводити повідомлення про помилку при невірних даних та перенаправляти користувача до адмін-панелі у разі успіху.

Таким чином, авторизація виконує роль захисту до адміністративної частини сайту та забезпечує надійність управління вмістом порталу.

Панель адміна - це ключова функціональна сторінка із всіх, через неї працюють всі основні функції сайту. Адмін панель має такі функції, для оперативних повідомлень: додавати оперативні оповіщення це показано на рис. 3.22. Для новин повністю заповнює починаючи від превью та закінчуючи категоріями візуалізація цього показано на рис 3.23. Має функцію редагування та видалення новини. Для інструкцій та рекомендацій: повністю заповнює статтю починаючи від превью та закінчуючи додатковими фото та текстом під

фото візуалізація панелі показано на рис. 3.24. Має функцію редагування та видалення новини.

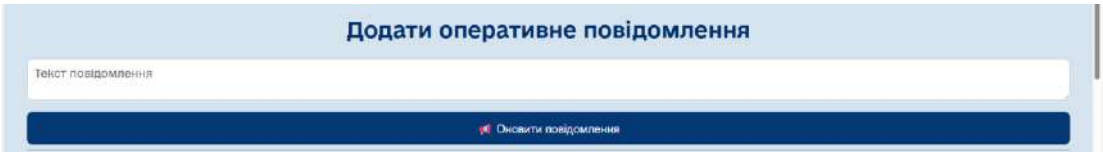


Рис. 3.22. Візуалізація додавання оперативного повідомлення

(Розроблено автором)

Рис. 3.23. Візуалізація додавання новин

(Розроблено автором)

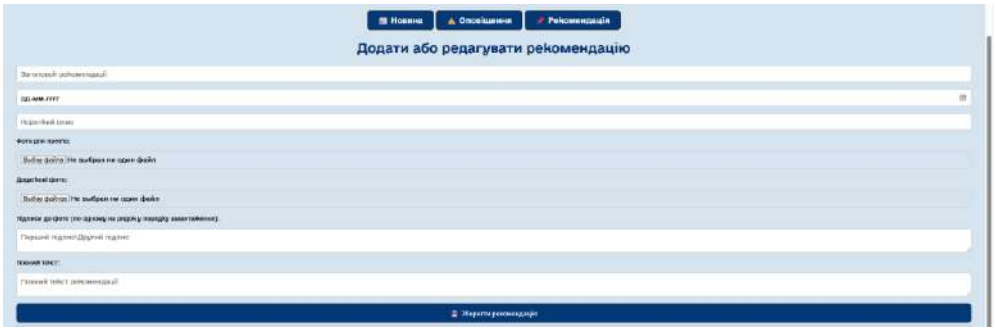


Рис. 3.24. Візуалізація додавання рекомендацій\інструкцій

(Розроблено автором)

Це інструмент для взаємодії між адміном та контентом в сайті. Адміністративна панель взаємодіє з усіма частинами сайта крім головної сторінки.

Також адміністративна панель має можливість перегляду списка новин та рекомендацій в вигляді таблиці це показано на рис. 3.25. Ці таблиці показують

ID новини або рекомендації, заголовок та дату, а також має функціональні кнопки для редагування та видалення контенту. Це все реалізовано за допомогою функцій JavaScript.

Рис. 3.25. Візуалізація таблиць Список новин та Список рекомендацій

(Розроблено автором)

Адмін панель є центральним інструментом управління, через який реалізуються всі ключові функції: створення, редагування та видалення новин, рекомендацій і оперативних повідомлень. Інтерфейс панелі розроблено таким чином, щоб адміністратор міг швидко та зручно керувати контентом, не взаємодіючи з кодом.

Завдяки підтримці візуального редагування, перегляду списків та можливості оновлення контенту в режимі реального часу - адмін панель забезпечує гнучкість, ефективність і зручність управління. Усі ці дії інтегруються зі структурою сайту, підтримуючи актуальність та повноту інформації для користувачів.

Таким чином ми отримаємо грамотну структуру сайту, що забезпечує функціональність та зручність у використанні. Всі сторінки сайту пов'язані правильно та зрозуміло реалізовані це показано на рис. 3.26.

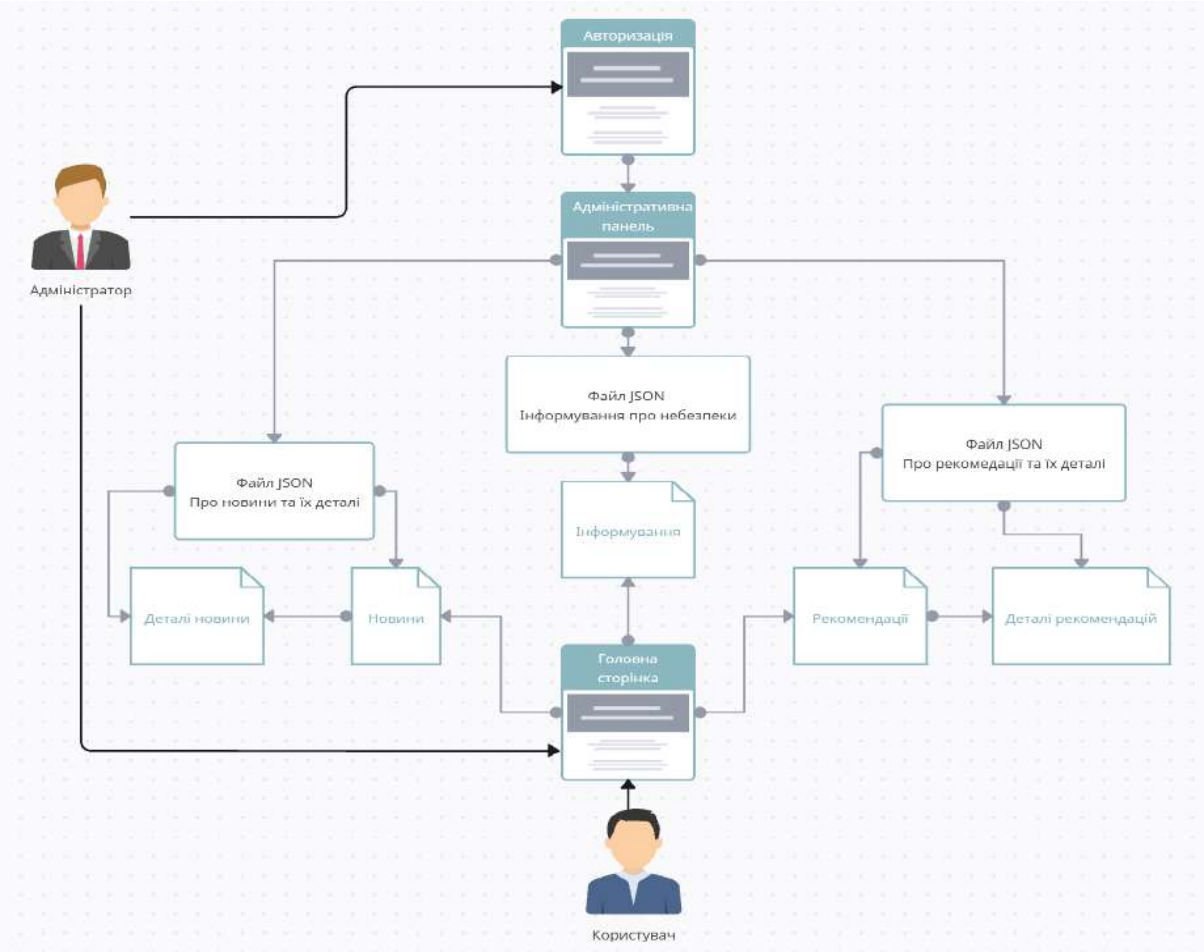


Рис. 3.26. Візуалізація структури сайту і як вона працює

(Розроблено автором)

На цей час маємо готовий макет для реалізації самого функціоналу сайту. Весь макет виглядає гармонічно та структуровано.

3.3. Розробка адмін-панелі для додавання новин для оперативного інформування населення про надзвичайні ситуації та заходи пожежної безпеки

Адмін-панель - це ключовий інструмент управління контентом сайту, що дозволяє адміністратору самостійно додавати, редагувати та видаляти рекомендації, новини та оповіщення. Однією з найважливіших функцій панелі є додавання новин та оперативних оповіщень, оскільки саме вони забезпечують користувачів актуальними повідомленнями про надзвичайні події та дії служб.

Цей підрозділ описує процес створення функціонального модуля. Реалізація передбачає повне створення адміністративної панелі, основні функції якої створення новин, редагування та видалення, а також таким самим методом будуть реалізовані оповіщення та рекомендації і інструкції. Вся інформація зберігається у форматі JSON, що дозволяє зручно обробляти дані як на стороні клієнта, так і на сервері.

У результаті реалізації цього модуля адміністратор отримує зручний інтерфейс для швидкого оновлення інформації на сайті без необхідності технічного втручання в код.

Початок розробки адміністративної панелі починається з вікна авторизації. Це ключовий перехідний елемент між звичайним переглядом та редагуванням. Вже був створений HTML файл з вікном авторизації залишается тільки функціональна частина. Щоб все працювало коректно був створений server.py.

Початок береться з login.html є два поля, а саме #username і #password та кнопка “Увійти”. При натисканні на кнопку виконується функція login() сама функція виглядає так як показано на рис. 3.27.

```
// ================= [АВТОРИЗАЦІЯ] =================
function login() {
  const username = document.getElementById('username').value.trim();
  const password = document.getElementById('password').value.trim();
  const errorDiv = document.getElementById('error');

  fetch('/login', {
    method: 'POST',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify({ username, password })
  })
    .then(res => {
      if (res.ok) return res.json();
      return res.json().then(data => { throw new Error(data.message); });
    })
    .then(() => {
      window.location.href = '/admin.html';
    })
    .catch(err => {
      errorDiv.textContent = err.message;
    });
}
```

Рис. 3.27. Візуалізація функції login()

(Розроблено автором)

Функція працює за таким принципом, а саме зчитує значення полів логін та пароль, надсилає запит POST на сервер за маршрутом /login. Якщо всі дані

вірні то відповідає кодом 200 та перенаправляє користувача в адмін панель, а якщо логін або пароль неправильні, відображається повідомлення про помилку у блоці #error.

Сам сервер працює так, він приймає JSON-запит з логіном та паролем, порівнює отримані дані з фіксованими обліковими даними. Після чого якщо всі дані вірні то у сесії встановлюється прапорець logged_in = True, та відповідає кодом 200. Якщо є помилка в написанні логіну чи паролю сервер відповідає кодом 401 Unauthorized та повідомленням в HTML “Невірний логін або пароль”. Таким чином реалізація авторизації через сервер є працюючою, візуалізація коду сервера показана на рис. 3.29.

```
# === АВТОРИЗАЦІЯ ===
ADMIN_USERNAME = 'admin'
ADMIN_PASSWORD = 'admin123'

@app.route('/login', methods=['POST'])
def login():
    data = request.json
    username = data.get('username')
    password = data.get('password')

    if username == ADMIN_USERNAME and password == ADMIN_PASSWORD:
        session['logged_in'] = True
        return jsonify({'success': True}), 200
    else:
        return jsonify({'success': False, 'message': 'Невірний логін або пароль'}), 401

@app.route('/logout', methods=['POST'])
def logout():
    session.clear()
    return jsonify({'success': True})

@app.route('/check_auth')
def check_auth():
    return jsonify({'authenticated': session.get('logged_in', False)})

@app.route('/admin.html')
def protected_admin():
    if not session.get('logged_in'):
        return redirect('/login.html')
    return send_from_directory('.', 'admin.html')
```

Рис. 3.29. Візуалізація коду сервера для авторизації

(Розроблено автором)

Функція виходу з адмін-панелі реалізована через кнопку та подію JavaScript, яка надсилає запит на сервер для завершення сесії. На сторінці адміністративної панелі розміщена кнопка “Вихід”. Ця кнопка має унікальний id="logoutBtn", завдяки чому її можна знайти в DOM і прикріпити обробник подій.

Скрипт який показано на рис. 3.30 виконується після завантаження сторінки, починається з того що він шукає кнопку id="logoutBtn" додає подію

click, після чого надсилає POST-запит на серверний маршрут /logout. Якщо відповідь успішна, користувача перенаправляє в login.html.

```
524  // ================= [ВИХІД] =================
525  document.addEventListener('DOMContentLoaded', function () {
526    const logoutBtn = document.getElementById('logoutBtn');
527    if (logoutBtn) {
528      logoutBtn.addEventListener('click', function () {
529        fetch('/logout', { method: 'POST' })
530          .then(res => res.json())
531          .then(() => {
532            window.location.href = '/login.html';
533          })
534          .catch(() => alert('Помилка при виході'));
535      });
536    }
537  });
```

Рис. 3.30. Код JavaScript, що реалізує вихід з системи

(Розроблено автором)

Натомість сервер обробляє запит на вихід /logout, це показано на рис. 3.31. Очищає сесію користувача (session.clear()), тим самим скасовуючи авторизацію, і повертає JSON-відповідь {success: true}.

```
44  @app.route('/logout', methods=['POST'])
45  def logout():
46      session.clear()
47      return jsonify({'success': True})
```

Рис. 3.31. Серверний маршрут logout у файлі server.py

(Розроблено автором)

Цей функціонал дозволяє адміністратору ввести текст актуального повідомлення та зберегти його в файл alerts_data.json, який потім буде використовуватись на публічній сторінці “Оперативне інформування”.

На сторінці адміністратора є окремий блок із формою, це показано на рис. 3.32, де адміністратор може ввести текст повідомлення. Форма має id="alertForm" - щоб її можна було обробити через JavaScript. Кнопка type="submit" надсилає форму без перезавантаження сторінки.

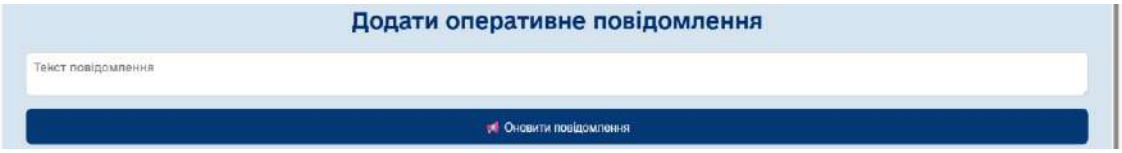


Рис. 3.32. Оперативне повідомлення внесення даних в адмін панелі

(Розроблено автором)

Скрипт який показан на рис. 3.33 додає подію на відправлення форми, після натискання кнопки виконується запит POST на /update_alert. Текст повідомлення передається в запиті. Після чого у відповідь від сервера користувач бачить повідомлення про успіх або помилку. Якщо був успіх форма очищується автоматично.

```
163  // ================== [ОПОВІЩЕННЯ] ==================
164  const alertForm = document.getElementById('alertForm');
165    if (alertForm) {
166      alertForm.addEventListener('submit', function (e) {
167        e.preventDefault();
168        const text = document.getElementById('alertText').value.trim();
169        fetch('/update_alert', {
170          method: 'POST',
171          headers: { 'Content-Type': 'application/json' },
172          body: JSON.stringify({ text })
173        })
174          .then(res => res.json())
175          .then(data => {
176            alert(data.message || 'Оновлено');
177            alertForm.reset();
178          })
179          .catch(() => alert('Помилка при оновленні'));
180      });
181    }
```

Рис. 3.33. JavaScript додавання оповіщення

(Розроблено автором)

На сервері тим часом Flask обробляє запит /update_alert. Сервер приймає JSON з полем text, записує цей текст у файл alerts_data.json, а потім на публічній сторінці info.html є контейнер, у який динамічно підставляється актуальне повідомлення з alerts_data.json, візуалізація самого коду показана на рис. 3.34.

```python
@app.route('/update_alert', methods=['POST'])
def update_alert():
    data = request.json
    text = data.get('text', '')
    with open(ALERT_FILE, 'w', encoding='utf-8') as f:
        json.dump({'text': text}, f, ensure_ascii=False, indent=2)
    return jsonify({'message': 'Оповіщення оновлено'}), 200

@app.route('/alerts_data.json')
def get_alert_data():
    return send_from_directory('.', 'alerts_data.json')
```

Рис. 3.34. Серверна частина додавання оповіщення

(Розроблено автором)

Тобто, коли адміністратор оновлює повідомлення через адмін-панель, воно миттєво з'являється у користувачів сайту при завантаженні сторінки.

Реалізація додавання оперативного повідомлення є простою та ефективною: Адміністратор заповнює форму, далі JS надсилає запит на Flask-сервер, після чого сервер оновлює файл alerts_data.json, і в результаті повідомлення показується на публічній сторінці сайту. Це дозволяє максимально швидко інформувати населення про надзвичайні події.

Система управління новинами у веб-порталі реалізована через адмін-панель. Вона забезпечує повноцінну роботу з публікаціями: адміністратор має змогу додавати новини, редагувати вже існуючі записи, видаляти непотрібні, а також переглядати список наявних матеріалів у вигляді таблиці. Всі новини зберігаються у файлі news_data.json, що дозволяє працювати з ними без використання повноцінної бази даних.

Початок бере admin.html він містить всі необхідні поля для заповнення новини: заголовок, дата, короткий опис, повний текст, завантаження превью-зображення, додаткові фото, вибір пов'язаної новини для секції "Читайте також", а також категорії події, візуально код показано на рис. 3.35. Категорії реалізовані через multiple-select і дозволяють одразу задати тематику новини, таку як "Пожежа", "ДТП", "Надзвичайна ситуація" тощо.

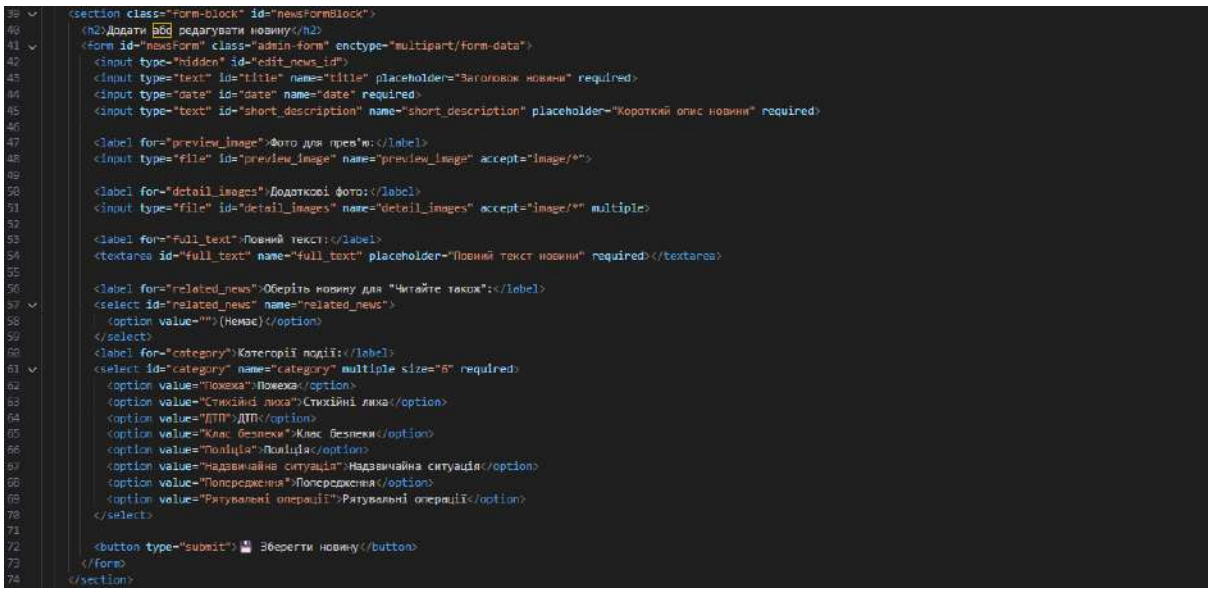


Рис 3.35. Візуалізація коду admin.html

(Розроблено автором)

Форма має приховане поле edit_news_id, що використовується для визначення: чи створюється нова новина, чи редагується існуюча. Якщо це нова новина, то поле залишається порожнім, а у випадку редагування сюди підставляється ID вибраного запису, після чого всі поля заповнюються існуючими записами певної новини.

Клієнтська логіка, реалізована у файлі admin.js, відповідає за обробку форми та взаємодію з сервером. Коли сторінка завантажується, автоматично викликається функція loadNewsList(), яка виконує запит до файлу news_data.json і формує список наявних новин, це показано на рис. 3.36. Цей список відображається у вигляді HTML-таблиці, де для кожного запису виводиться його ID, заголовок, дата, а також кнопки для редагування та видалення.

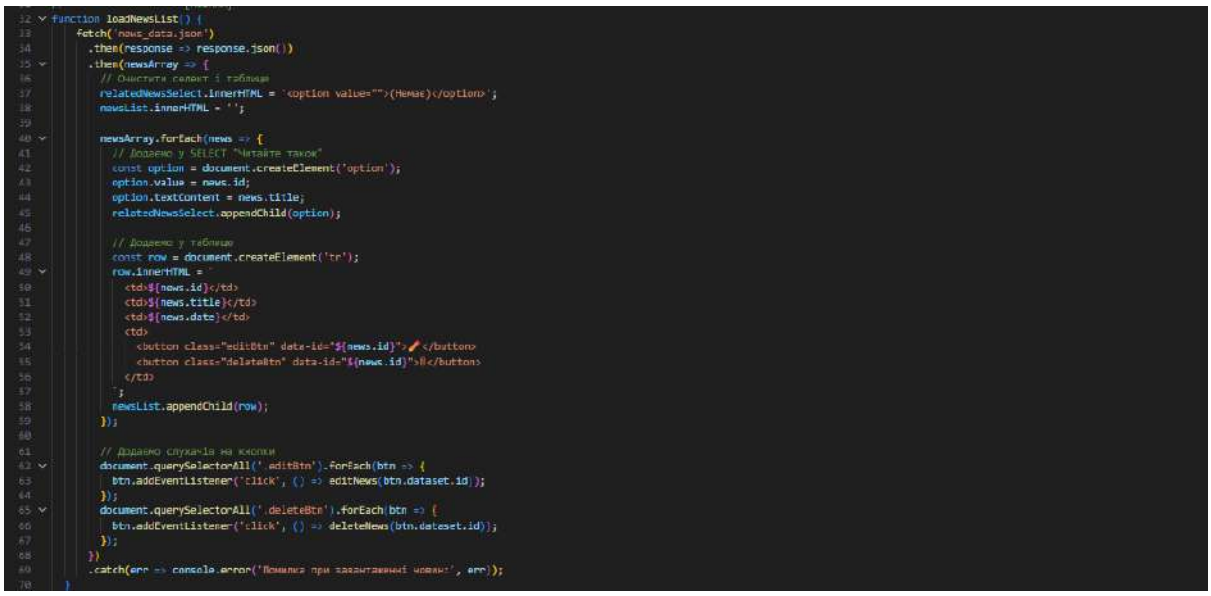


Рис. 3.36. Функція loadNewsList()

(Розроблено автором)

Якщо натиснути кнопку редагування, викликається функція editNews(id), яка завантажує відповідний запис із JSON, підставляє його значення в форму, та дозволяє відредагувати текст, змінити дату, опис або навіть категорії, візуалізація логіки коду показано на рис. 3.37. У момент редагування у приховане поле додається ID новини, щоб сервер знав, що потрібно оновити існуючий запис, а не створити новий, логіка сервера показано на рис. 3.38.

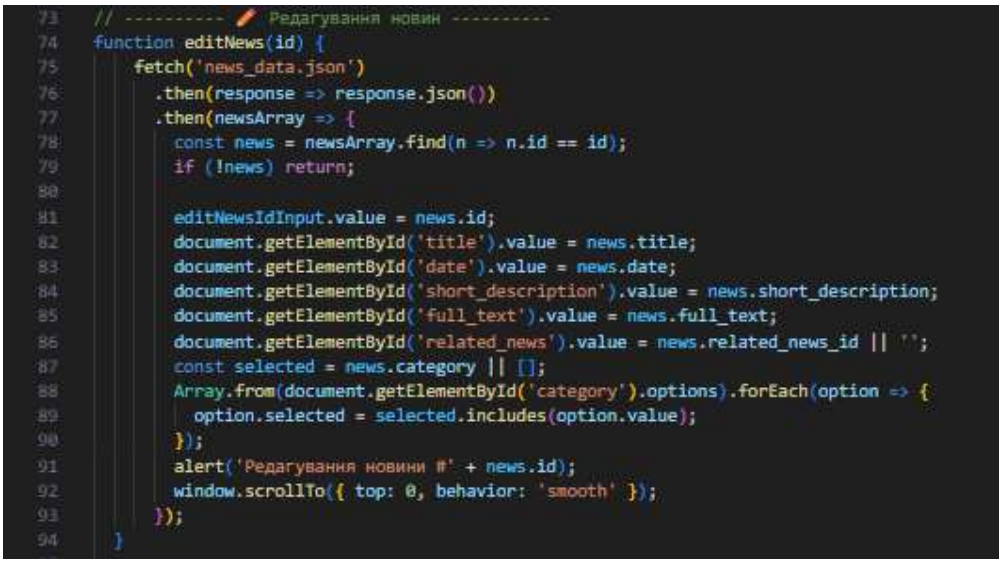
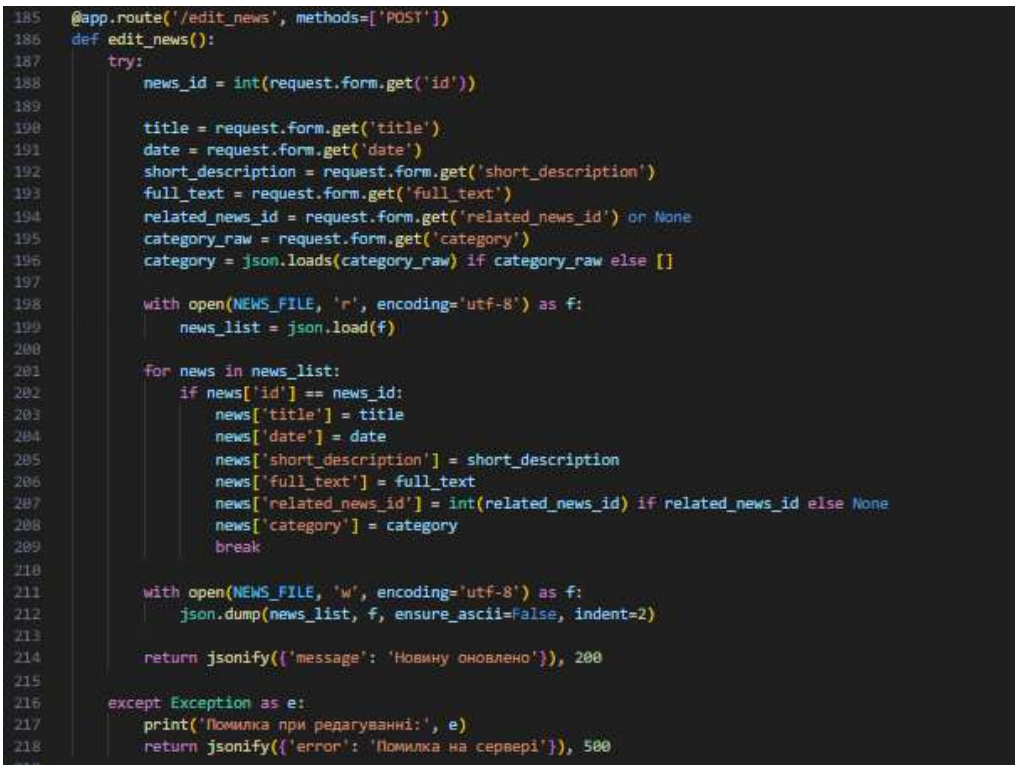


Рис. 3.37 Функція editNews(id)

(Розроблено автором)



```python
@app.route('/edit_news', methods=['POST'])
def edit_news():
    try:
        news_id = int(request.form.get('id'))

        title = request.form.get('title')
        date = request.form.get('date')
        short_description = request.form.get('short_description')
        full_text = request.form.get('full_text')
        related_news_id = request.form.get('related_news_id') or None
        category_raw = request.form.get('category')
        category = json.loads(category_raw) if category_raw else []

        with open(NEWS_FILE, 'r', encoding='utf-8') as f:
            news_list = json.load(f)

        for news in news_list:
            if news['id'] == news_id:
                news['title'] = title
                news['date'] = date
                news['short_description'] = short_description
                news['full_text'] = full_text
                news['related_news_id'] = int(related_news_id) if related_news_id else None
                news['category'] = category
                break

        with open(NEWS_FILE, 'w', encoding='utf-8') as f:
            json.dump(news_list, f, ensure_ascii=False, indent=2)

        return jsonify({'message': 'Новину оновлено'}), 200

    except Exception as e:
        print('Помилка при редагуванні:', e)
        return jsonify({'error': 'Помилка на сервері'}), 500
```

Рис. 3.38 Серверна частина редагування новин

(Розроблено автором)

При натисканні кнопки “Зберегти” виконується подія submit, код події показано на рис. 3.39, на формі. JavaScript формує об'єкт FormData - це спеціальна структура, яка дозволяє надсилати як текстові дані, так і файли. Якщо в полі edit_news_id є значення, тобто редагується новина, запит надсилається на маршрут /edit_news, інакше - на /add_news. Разом з цим надсилаються усі поля: заголовок, дата, короткий опис, повний текст, зв'язана новина, категорії, зображення для превью та галереї.

Рис. 3.39. Подія submit

(Розроблено автором)

Серверна частина, реалізована у файлі server.py, приймає цей запит через маршрут /add_news або /edit_news. У випадку додавання новини сервер обробляє зображення, зберігає їх у папці uploads, генерує унікальний ID для нової новини (шляхом вибору найбільшого з існуючих ID), формує словник із усіма полями новини та додає його до списку у файлі news_data.json. Потім новий масив записується назад у JSON-файл.

У випадку редагування, сервер спочатку знаходить новину за переданим ID у масиві, оновлює її значення тими, що були передані з форми, й також перезаписує JSON-файл.

Видалення новини реалізовано через маршрут /delete_news, це показано на рис. 3.40. Коли користувач натискає кнопку видалення, з’являється підтвердження через confirm(). Якщо користувач погоджується, виконується POST-запит із ID новини, яка має бути видалена. Сервер видаляє її з масиву та зберігає оновлений файл без цього запису.

```python
# === ВИДАЛЕННЯ НОВИНИ ===
@app.route('/delete_news', methods=['POST'])
def delete_news():
    try:
        news_id = int(request.form.get('id'))

        with open(NEWS_FILE, 'r', encoding='utf-8') as f:
            news_list = json.load(f)

        news_list = [news for news in news_list if news['id'] != news_id]

        with open(NEWS_FILE, 'w', encoding='utf-8') as f:
            json.dump(news_list, f, ensure_ascii=False, indent=2)

        return jsonify({'message': 'Новину успішно видалено!'}), 200

    except Exception as e:
        print('Помилка при видаленні:', e)
        return jsonify({'error': 'Помилка на сервері'}), 500
```

Рис. 3.40. Візуалізація серверного запиту на видалення новини

(Розроблено автором)

Окремо реалізовано відображення новин у публічній частині сайту. На сторінці news.html завантажується список новин із news_data.json, який виводиться у вигляді карток із зображенням, заголовком, коротким описом, датою та категоріями. Передбачено можливість фільтрації за ключовими словами та за категоріями подій, самкод можно побачити на рис. 3.41.

```
// ================= [ФІЛЬТРИ ТА ПОШУК НОВИН] =================
document.addEventListener('DOMContentLoaded', function() {
  const listContainer = document.getElementById('newsList');
  const searchInput = document.getElementById('searchInput');
  const filterCategory = document.getElementById('filterCategory');
  let allNews = [];

  if (!listContainer) return;

  fetch('news_data.json')
    .then(response => response.json())
    .then(newsList => {
      allNews = newsList;
      renderNews();
    });

  function renderNews() {
    const keyword = searchInput.value.toLowerCase();
    const selectedCategory = filterCategory.value;
    listContainer.innerHTML = '';

    const filtered = allNews.filter(news => {
      const matchesText = news.title.toLowerCase().includes(keyword) ||
                          news.short_description.toLowerCase().includes(keyword) ||
                          news.full_text.toLowerCase().includes(keyword);

      const matchesCategory =
        selectedCategory === '' ||
        (news.category && news.category.includes(selectedCategory));

      return matchesText && matchesCategory;
    });

    filtered.forEach(news => {
      const card = document.createElement('a');
      card.href = `news_detail.html?id=${news.id}`;
      card.className = 'news-card';
      card.setAttribute('data-category', (news.category || []).join(','));
      card.innerHTML = `
        <img src="${news.preview_image}" alt="${news.title}">
        <div class="news-card-content">
          <div class="news-date">${news.date}</div>
          <h3>${news.title}</h3>
          <p>${news.short_description}</p>
          <div class="news-categories">
            ${(news.category || []).map(cat => `<span class="category-badge">${cat}</span>`).join('')}
          </div>
        </div>
      `;
      listContainer.appendChild(card);
    });
  }

  searchInput.addEventListener('input', renderNews);
  filterCategory.addEventListener('change', renderNews);
});
```

Рис. 3.41. Функція фільтрації новин за категоріями та ключовими словами

(Розроблено автором)

При натисканні на новину користувач переходить на сторінку news_detail.html, яка завантажує повну версію новини за допомогою параметра ?id=... у URL. Відображаються повний текст, дата, категорії та фото, це показано на рис. 3.42. Якщо новина має пов’язаний запис (related_news_id), внизу сторінки додається блок “Читайте також” з посиланням на іншу новину.

```
// ================ [ДЕТАЛІ НОВИНИ] ================
document.addEventListener('DOMContentLoaded', function() {
    const params = new URLSearchParams(window.location.search);
    const newsId = params.get('id');

    fetch('news_data.json')
      .then(response => response.json())
      .then(newsList => {
        const newsItem = newsList.find(item => item.id == newsId);
        const container = document.getElementById('newsDetail');

        if (newsItem) {
          // Основна новина
          let contentHtml = `
            <div class="news-full">
              <h1>${newsItem.title}</h1>
              <div class="news-date">${newsItem.date}</div>
              <div class="news-categories">
              ${(newsItem.category || []).map(cat => `<span class="news-category-chip">${cat}</span>`).join('')}
            </div>
          <div style="margin-top: 20px; white-space: pre-line;">${newsItem.full_text}</div>
          `;

          // Фото
          if (newsItem.images && newsItem.images.length > 0) {
            contentHtml += `<div class="news-gallery">`;
            newsItem.images.forEach(img => {
              contentHtml += `<img src="${img}" alt="Фото новини" class="news-full-img">`;
            });
            contentHtml += `</div>`;
          }

          container.innerHTML = contentHtml;

          // ====== Додаємо "Читайте також" ========
          if (newsItem.related_news_id) {
            const relatedNews = newsList.find(item => item.id == newsItem.related_news_id);
            if (relatedNews) {
              const relatedContainer = document.createElement('div');
              relatedContainer.classList.add('related-news-block');
              relatedContainer.innerHTML = `
                <p><strong>Читайте також:</strong>
                  <a href="news_detail.html?id=${relatedNews.id}">
                  ${relatedNews.title}
                  </a>
                </p>
              `;
              container.appendChild(relatedContainer);
            }
          }
        } else {
          container.innerHTML = '<p>Новину не знайдено.</p>';
        }
      })
      .catch(error => console.error('Помилка завантаження деталей новини:', error));
});
```

Рис. 3.42. Функція відображення повної новини, а також “Читайте також”

(Розроблено автором)

Таким чином, реалізація новин охоплює повний цикл: від створення через зручну панель адміністратора до публічного перегляду з можливістю фільтрації та детального ознайомлення. Усі дані зберігаються у форматі JSON, що спрощує доступ і дозволяє легко підтримувати проект без використання складних баз даних.

Система рекомендацій реалізована як окремий розділ адмін-панелі, що дозволяє адміністратору публікувати корисні поради, інструкції або інформативні матеріали з фото, заголовком, текстом і підписами до зображень. Вся інформація зберігається у файлі recommendations_data.json.

На стороні admin.html реалізована велика форма з полями для введення заголовка, дати, опису, повного тексту, завантаження превью-зображення, кількох додаткових фото та підписів до них, це показано на рис. 3.43. Для кожної рекомендації можна задати текстові підписи до кожного зображення, які користувач вводить через textarea (кожен підпис з нового рядка). У формі також є приховане поле edit_recom_id, що дозволяє визначити, чи рекомендація створюється нова, чи редагується існуюча.

```
<section class="form-block" id="recommendationFormBlock">
  <h2>Додати або редагувати рекомендацію</h2>
  <form id="recommendationForm" class="admin-form" enctype="multipart/form-data">
    <input type="hidden" id="edit_recom_id">

    <input type="text" id="recom_title" name="title" placeholder="Заголовок рекомендації" required>
    <input type="date" id="recom_date" name="date" required>
    <input type="text" id="recom_description" name="description" placeholder="Короткий опис" required>

    <label for="recom_preview">Фото для прев'ю:</label>
    <input type="file" id="recom_preview" name="preview_image" accept="image/*">

    <label for="recom_images">Додаткові фото:</label>
    <input type="file" id="recom_images" name="images" accept="image/*" multiple>

    <label for="recom_captions">Підписи до фото (по одному на рядок у порядку завантаження):</label>
    <textarea id="recom_captions" name="captions" placeholder="Перший підпис\Другий підпис" required></textarea>

    <label for="recom_full_text">Повний текст:</label>
    <textarea id="recom_full_text" name="full_text" placeholder="Повний текст рекомендації" required></textarea>

    <button type="submit">💾 Зберегти рекомендацію</button>
  </form>
</section>
```

Рис. 3.43. HTML код додавання та редагування рекомендацій

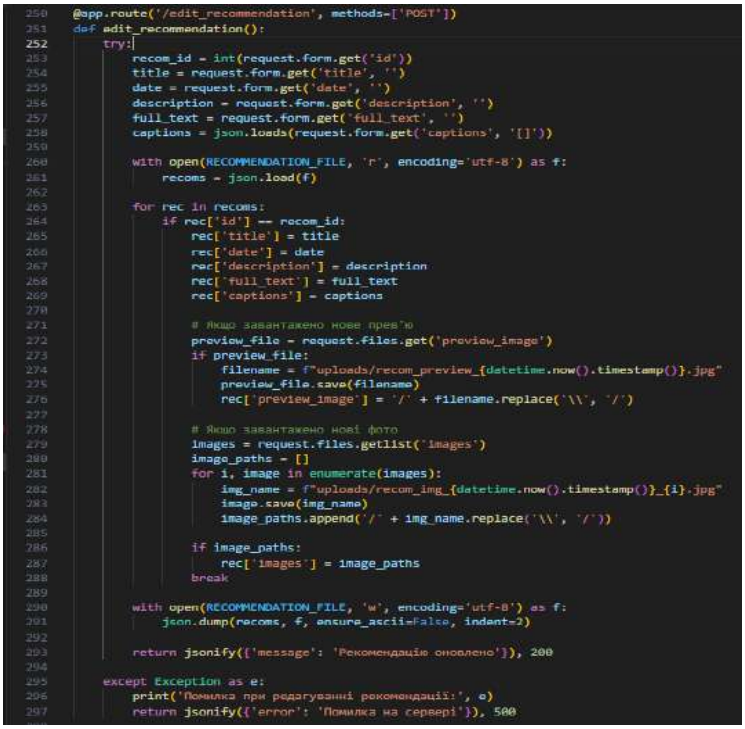
(Розроблено автором)

У admin.js при натисканні кнопки “Зберегти” виконується збір усіх даних форми у вигляді FormData, зокрема і файлів, це можна побачити на рис. 3.44. Якщо edit_recom_id має значення, тоді форму надсилають на серверний маршрут /edit_recommendation візуалізація коду на рис. 3.45, інакше - на /add_recommendation, показано на рис. 3.46.

```
// ================= [РЕКОМЕНДАЦІЇ] =================
const recommendationForm = document.getElementById('recommendationForm');
if (recommendationForm) {
  recommendationForm.addEventListener('submit', function (e) {
    e.preventDefault();

    const formData = new FormData();
    const isEdit = document.getElementById('edit_recom_id').value !== '';

    if (isEdit) {
      formData.append('id', document.getElementById('edit_recom_id').value);
    }

    formData.append('title', document.getElementById('recom_title').value.trim());
    formData.append('date', document.getElementById('recom_date').value);
    formData.append('description', document.getElementById('recom_description').value.trim());
    formData.append('full_text', document.getElementById('recom_full_text').value.trim());

    // Фото прев'ю
    const preview = document.getElementById('recom_preview').files[0];
    if (preview) {
      formData.append('preview_image', preview);
    }

    // Фото і підписи
    const images = document.getElementById('recom_images').files;
    const captionsRaw = document.getElementById('recom_captions').value.trim().split('\n');

    for (let i = 0; i < images.length; i++) {
      formData.append('images', images[i]);
    }

    // Передаємо підписи як JSON
    formData.append('captions', JSON.stringify(captionsRaw));

    const url = isEdit ? '/edit_recommendation' : '/add_recommendation';

    fetch(url, {
      method: 'POST',
      body: formData
    })
    .then(res => res.json())
    .then(data => {
      alert(data.message || 'Рекомендацію збережено');
      recommendationForm.reset();
      document.getElementById('edit_recom_id').value = '';
      loadRecommendations();
      // Якщо буде список, можна перезавантажити loadRecommendations()
    })
    .catch(() => alert('Помилка при збереженні рекомендації'));
  });
}
```

Рис. 3.44. Функція збереження нової новини

(Розроблено автором)



```
@app.route('/edit_recommendation', methods=['POST'])
def edit_recommendation():
    try:
        recom_id = int(request.form.get('id'))
        title = request.form.get('title', '')
        date = request.form.get('date', '')
        description = request.form.get('description', '')
        full_text = request.form.get('full_text', '')
        captions = json.loads(request.form.get('captions', '[]'))

        with open(RECOMMENDATION_FILE, 'r', encoding='utf-8') as f:
            recoms = json.load(f)

        for rec in recoms:
            if rec['id'] == recom_id:
                rec['title'] = title
                rec['date'] = date
                rec['description'] = description
                rec['full_text'] = full_text
                rec['captions'] = captions

                # Якщо завантажено нове прев'ю
                preview_file = request.files.get('preview_image')
                if preview_file:
                    filename = f"uploads/recom_preview_{datetime.now().timestamp()}.jpg"
                    preview_file.save(filename)
                    rec['preview_image'] = '/' + filename.replace('\\', '/')

                # Якщо завантажено нові фото
                images = request.files.getlist('images')
                image_paths = []
                for i, image in enumerate(images):
                    img_name = f"uploads/recom_img_{datetime.now().timestamp()}_{i}.jpg"
                    image.save(img_name)
                    image_paths.append('/' + img_name.replace('\\', '/'))

                if image_paths:
                    rec['images'] = image_paths
                break

        with open(RECOMMENDATION_FILE, 'w', encoding='utf-8') as f:
            json.dump(recoms, f, ensure_ascii=False, indent=2)

        return jsonify({'message': 'Рекомендацію оновлено'}), 200

    except Exception as e:
        print('Помилка при редагуванні рекомендації:', e)
        return jsonify({'error': 'Помилка на сервері'}), 500
```

Рис. 3.45. Серверний маршрут редагування рекомендацій /edit_recommendation

(Розроблено автором)

```python
# === DODAVANYA RECOMENDCIY ===
@app.route('/add_recommendation', methods=['POST'])
def add_recommendation():
    (variable) description: str
    description = request.form.get('description', '')
    full_text = request.form.get('full_text', '')
    captions = json.loads(request.form.get('captions', '[]'))

    preview_file = request.files.get('preview_image')
    preview_path = ''
    if preview_file:
        filename = f"uploads/recom_preview_{datetime.now().timestamp()}.jpg"
        preview_file.save(filename)
        preview_path = '/' + filename.replace('\\', '/')

    images = request.files.getlist('images')
    image_paths = []
    for i, image in enumerate(images):
        img_name = f"uploads/recom_img_{datetime.now().timestamp()}_{i}.jpg"
        image.save(img_name)
        image_paths.append('/' + img_name.replace('\\', '/'))

    # Готуємо нову рекомендацію
    with open(RECOMMENDATION_FILE, 'r', encoding='utf-8') as f:
        all_recoms = json.load(f)

    new_id = max([r['id'] for r in all_recoms], default=0) + 1

    new_recom = {
        "id": new_id,
        "title": title,
        "date": date,
        "description": description,
        "full_text": full_text,
        "preview_image": preview_path,
        "images": image_paths,
        "captions": captions
    }

    all_recoms.append(new_recom)

    with open(RECOMMENDATION_FILE, 'w', encoding='utf-8') as f:
        json.dump(all_recoms, f, ensure_ascii=False, indent=2)

    return jsonify({"message": "Рекомендацію додано успішно"}), 200
```

Рис. 3.46 Серверний маршрут редагування рекомендацій /add_recommendation

(Розроблено автором)

Серверна частина server.py обробляє запити наступним чином. При створенні нової рекомендації сервер отримує усі поля, зберігає файли зображень у папку uploads, формує новий об’єкт-рекомендацію, надає їй унікальний id, і зберігає цей об'єкт у recommendations_data.json, це показано на рис. 3.47.

```python
@app.route('/uploads/<path:filename>')
def uploaded_files(filename):
    return send_from_directory(UPLOAD_FOLDER, filename)
```

Рис. 3.47. Збереження файлів зображень у папку uploads

(Розроблено автором)

Якщо запит надходить на /edit_recommendation, сервер знаходить існуючу рекомендацію за ID, оновлює всі її поля. Якщо були завантажені нові зображення

або прев'ю - старі перезаписуються. Весь об'єкт рекомендації оновлюється і зберігається назад у JSON-файл.

Функція видалення реалізована через маршрут /delete_recommendation, це показано на рис. 3.48. Адмін може зі списку рекомендацій видалити обрану рекомендацію, після чого JSON-файл оновлюється без цієї записи.

```python
@app.route('/delete_recommendation', methods=['POST'])
def delete_recommendation():
    try:
        recom_id = int(request.form.get('id'))

        with open(RECOMMENDATION_FILE, 'r', encoding='utf-8') as f:
            recoms = json.load(f)

        recoms = [r for r in recoms if r['id'] != recom_id]

        with open(RECOMMENDATION_FILE, 'w', encoding='utf-8') as f:
            json.dump(recoms, f, ensure_ascii=False, indent=2)

        return jsonify({'message': 'Рекомендацію успішно видалено!'}), 200

    except Exception as e:
        print('Помилка при видаленні рекомендації:', e)
        return jsonify({'error': 'Помилка на сервері'}), 500
```

Рис. 3.48. Реалізація серверного маршруту видалення рекомендацій

(Розроблено автором)

Для перегляду рекомендацій на публічній частині сайту, JavaScript на сторінці recommendations.html виконує запит до recommendations_data.json, створює картки з прев'ю, заголовком, датою і описом, та підставляє їх у DOM, це показано на рис. 3.49.

```javascript
document.addEventListener('DOMContentLoaded', function () {
  fetch('recommendations_data.json')
    .then(res => res.json())
    .then(data => {
      const container = document.getElementById('recommendationsList');
      data.reverse().forEach(item => {
        const block = document.createElement('a');
        block.href = `recommendation_detail.html?id=${item.id}`;
        block.className = 'news-card';
        block.innerHTML = `
          <img src="${item.preview_image}" alt="${item.title}">
          <div class="news-card-content">
            <div class="news-date">${item.date}</div>
            <h3>${item.title}</h3>
            <p>${item.description}</p>
          </div>
        `;
        container.appendChild(block);
      });
    })
    .catch(() => console.error('Не вдалося завантажити рекомендації'));
});
```

Рис. 3.49. JavaScript виводу списка рекомендацій в публічну частину сайту

(Розроблено автором)

Після натискання на будь-яку картку користувач переходить на сторінку recommendation_detail.html, де через ?id=... у URL завантажується повний контент рекомендації, код візуально показаний на рис. 3.50. Відображається заголовок, дата, повний текст, а також зображення з підписами під кожним фото. Усі зображення мають однаковий стиль і красиво форматуються у вигляді галереї з поясненням до кожного елементу.

```
document.addEventListener('DOMContentLoaded', function () {
  const container = document.getElementById('recommendationDetail');
  const params = new URLSearchParams(window.location.search);
  const id = params.get('id');

  fetch('recommendations_data.json')
    .then(res => res.json())
    .then(data => {
      const item = data.find(r => r.id == id);
      if (!item) {
        container.innerHTML = '<p>Рекомендацію не знайдено.</p>';
        return;
      }

      let html = `
        <h1>${item.title}</h1>
        <div class="news-date">${item.date}</div>
      `;

      const paragraphs = item.full_text.split(/\r?\n/).filter(p => p.trim() !== '');

      html += `<div class="recommendation-text">` +
                paragraphs.map(p => `<p>${p}</p>`).join('') +
              `</div>`;

      if (item.images && item.images.length > 0) {
        html += `<div class="news-gallery">`;
        item.images.forEach((img, i) => {
          const caption = item.captions && item.captions[i] ? item.captions[i] : '';
          html += `
            <div style="margin-bottom: 30px; text-align: center;">
              <img src="${img}" alt="Фото" class="news-full-img">
              <p style="margin-top: 8px; color: #555;">${caption}</p>
            </div>
          `;
        });
        html += `</div>`;
      }

      container.innerHTML = html;
    })
    .catch(() => {
      container.innerHTML = '<p>Помилка завантаження рекомендації.</p>';
    });
});
```

Рис. 3.50. JavaScript виводу повної рекомендації в публічній частині сайту

(Розроблено автором)

Коли адміністратор вносить зміни або додає нову рекомендацію, після успішного збереження JavaScript викликає функцію loadRecommendations(), яка

візуалізована на рис. 3.51, яка оновлює таблицю списку рекомендацій. Таблиця містить ID, заголовок, дату та кнопки для редагування і видалення.

```
// ---------- 📥 Завантаження рекомендацій ----------
function loadRecommendations() {
  fetch('recommendations_data.json')
    .then(res => res.json())
    .then(data => {
      const tbody = document.getElementById('recomList');
      tbody.innerHTML = '';

      data.forEach(rec => {
        const row = document.createElement('tr');
        row.innerHTML = `
          <td>${rec.id}</td>
          <td>${rec.title}</td>
          <td>${rec.date}</td>
          <td>
            <button class="editRecomBtn" data-id="${rec.id}">🖍️</button>
            <button class="deleteRecomBtn" data-id="${rec.id}">🗑</button>
          </td>
        `;
        tbody.appendChild(row);
      });

      document.querySelectorAll('.editRecomBtn').forEach(btn => {
        btn.addEventListener('click', () => editRecommendation(btn.dataset.id));
      });

      document.querySelectorAll('.deleteRecomBtn').forEach(btn => {
        btn.addEventListener('click', () => deleteRecommendation(btn.dataset.id));
      });
    });
}
```

Рис. 3.51. JavaScript оновлення списка рекомендацій в публічну частину сайту

(Розроблено автором)

Таким чином, реалізація функціоналу рекомендацій охоплює повний цикл: створення, редагування, перегляд, видалення, а також прив’язку зображень і підписів. Це зручно для створення навчально-інформаційного контенту, яким легко управляти без баз даних - усього через JSON-файл і сучасний інтерфейс.

У підсумку реалізація адмін панелі для управління контентом веб-порталу стала ефективна, гнучка і зручна у використанні. Завдяки поєднанню HTML-форм, JavaScript і серверної логіки на Flask, адміністратор отримує інструмент, що дозволяє повністю додавати та редагувати наповнення сайту - новини, оперативні повідомлення та інструкції та рекомендації.

Використання формату JSON як основи для зберігання даних дозволило уникнути складних баз даних, спростивши структуру і зробивши її доступною для модифікації навіть користувачам без глибоких технічних знань. Інтерфейс реалізовано інтуїтивно: для кожного типу контенту передбачено зручні блоки додавання, редагування, перегляду та видалення. Авторизація забезпечує

базовий рівень безпеки, дозволяючи доступ до адмін-панелі лише уповноваженим особам.

Таким чином, реалізований модуль адмін-панелі що став ключовою частиною системи сайту, що забезпечує актуальність публікацій, а також оперативне інформування громадян у разі надзвичайних ситуацій.

3.4. Тестування веб-порталу для оперативного інформування населення про надзвичайні ситуації та заходи пожежної безпеки

Тестування веб-порталу це фінальний крок для завершення роботи. Таким чином підходимо до того що треба поставити сайт на хостинг. Для цього був обраний безкоштовний хмарний хостинг Render [19], [28], [30], який забезпечує просту інтеграцію з GitHub, автоматичний запуск Flask-додатків та зручне керування процесом розгортання.

Першим кроком для публікації порталу на хостингу Render стало завантажженя в GitHub. Після успішного завантаження ми маємо повний вміст проекту в репозиторії GitHub, це показано на рис. 3.52.

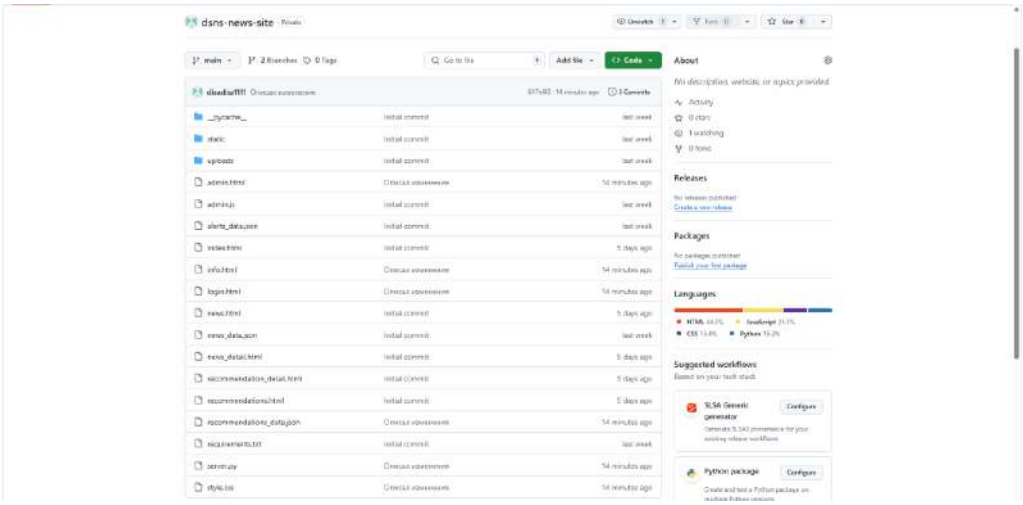


Рис. 3.52. Репозиторій GitHub з доданими файлами

(Розроблено із використанням GitHub)

Наступним кроком стало створення веб-хостингу в Render. Цей хостинг був обраний з тим що він може працювати з Flask та має безкоштовний хостинг.

Таким чином ми отримаємо ідеальний сервіс для хостингу на перший час. Після налаштування хостингу, вже можна запустити сайт.

Після запуску отримали посилання на веб-портал: https://dsns-news-site.onrender.com/ . Сайт вже працює. Можна переходити до тестування.

Після переходу за посиланням нас зустрічає головна сторінка, це показано на рис. 3.53. Перевіривши всі функціональні кнопки, помилок не було виявлено.

Рис. 3.53 Вже працююча головна сторінка

(Розроблено автором)

Можна переходити до наступної сторінки, а саме “Оперативна інформація”. Перейшовши на сторінку, отримали бажаний результат, карта повітряних тривог працює, оперативні повідомлення є, працездатність показана на рис. 3.54. Але для перевірки працездатності, треба змінити повідомлення за допомогою адмін панелі і разом перевірити працездатність панелі авторизації.

Рис. 3.54. Відображення сторінки оперативне інформування

(Розроблено автором)

Перейшовши за посиланням на адміністративну панель, було автоматично перенаправлено на панель з авторизацією. Перевіривши введення неправильного паролю та логіну було отримано повідомлення про невірний логін та пароль, це показано на рис. 3.55.

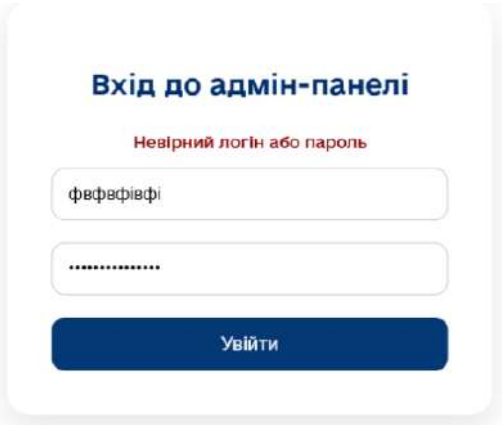


Рис. 3.55. Повідомлення “Невірний логін або пароль”

(Розроблено автором)

Після успішного вводу достовірних даних, користувача перенаправляє до адмін панелі. Там зустрічає три великих кнопки “Новина”, “Оповіщення” та “Рекомендація”. Зараз цікавить оперативне повідомлення. Обравши потрібний блок, отримаємо відповідне поле з вводом тексту. Для тестування було обрано повідомлення “Тестування оперативного повідомлення”. Після натискання на “Оновити повідомлення”, випливає вікно з успіхом “Повідомлення оновлено” це видно на рис. 3.56. Залишається тільки перевірити сторінку інформування.

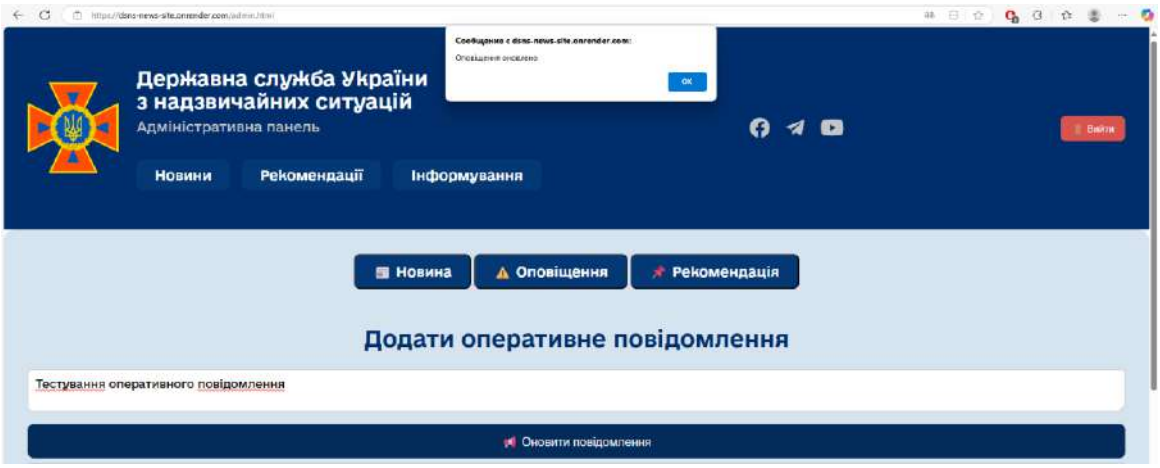


Рис. 3.56. Перевірка працездатності функціоналу додавання оперативного повідомлення

(Розроблено автором)

Перевіривши сторінку інформування. Внизу сторінки було відображено оперативне повідомлення, це видно на рис. 3.57. Таким чином отримаємо працюючий функціональний елемент сайту.

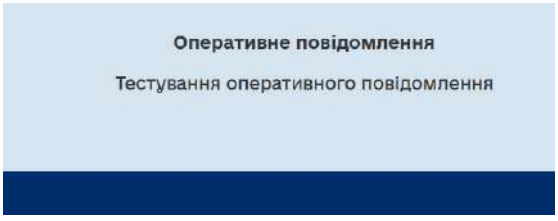


Рис. 3.57. Тестування оперативного повідомлення

(Розроблено автором)

Наступним кроком перевірки буде сторінка з новинами. Для початку треба додати новину, цей елемент робиться через адміністративну панель.

Заповнивши всі строки в адміністративній панелі. Натискаємо зберегти новину, після чого отримуємо повідомлення про “Новину успішно додано” це показано на рис. 3.58.

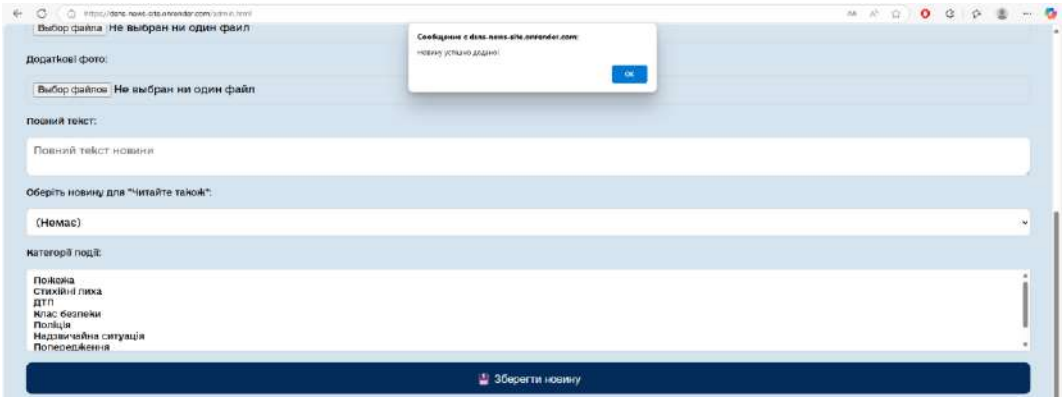


Рис. 3.58. Успішне додання новини

(Розроблено автором)

Після переходу на сторінку новини ми бачимо успішно додану новину, це показано на рис. 3.59. Всі ключові елементи відображаються чітко. Перейшовши в саму новину, бачимо повну новину з додатковими фото та “Читайте також…”, це успішно видно на рис. 3.60.

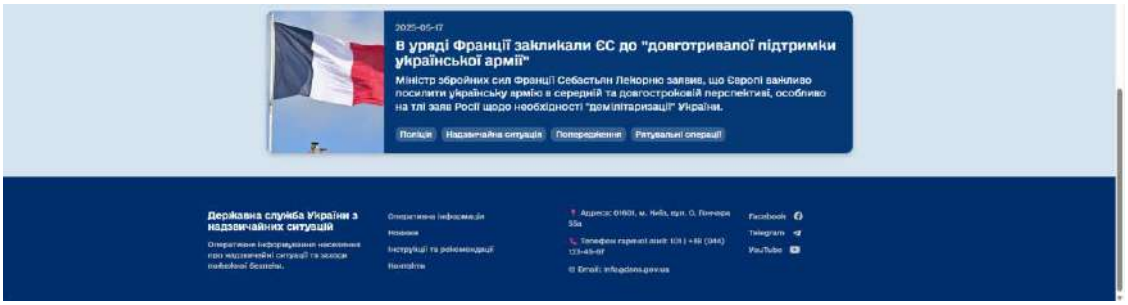


Рис. 3.59. Успішно додана новина у списку

(Розроблено автором)

Рис. 3.60. Успішне відображення повної новини, а також “Читайте також…”

(Розроблено автором)

Таким чином отримуємо повністю функціональні та робочі сторінки новини та адмін панелі. Вони чітко все додають та редагують, а також видаляють.

Наступним кроком буде додавання рекомендацій та інструкцій. Початок так як з новинами береться з адмін панелі. Обравши потрібний розділ додавання та редагування, заповнивши форму. Заповнивши всі пункти натискаємо зберегти рекомендацію. Отримуємо повідомлення про успіх додавання рекомендації, це показано на рис. 3.61.

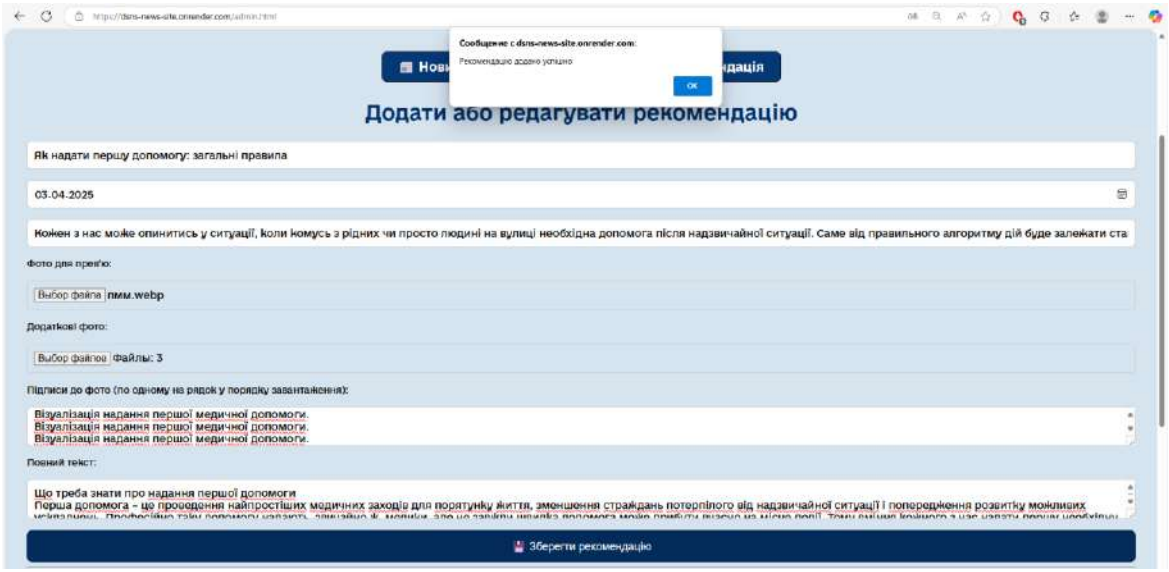


Рис. 3.61. Успішне додавання рекомендації

(Розроблено автором)

Переходимо в розділ рекомендації та бачимо успішно додано рекомендацію в сторінці, це показано на рис. 3.62. Перейшовши на неї бачимо повну рекомендацію з текстом, заголовком, фото та підписи під ними, візуалізація показана на рис. 3.63.

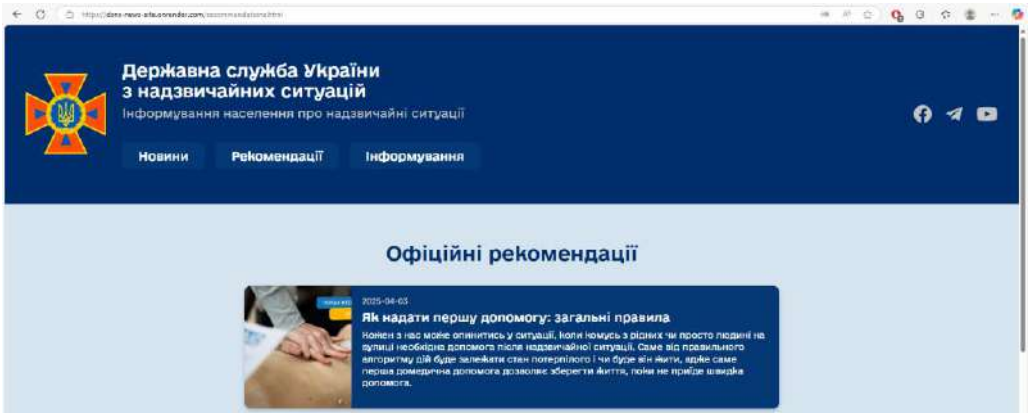


Рис. 3.62. Відображення рекомендації у списку

(Розроблено автором)

Рис. 3.63. Відображення повної рекомендації

(Розроблено автором)

У процесі тестування веб-порталу було проведено повну перевірку всіх основних функціональних компонентів, починаючи з авторизації адміністратора

і завершуючи публічним відображенням контенту на сайті. Успішне розміщення проекту на хостингу Render через GitHub підтвердило працездатність серверної логіки та клієнтської частини в реальному середовищі.

Жодних критичних помилок під час взаємодії з веб-інтерфейсом виявлено не було. Система працює стабільно, швидко реагує на дії користувача, а всі зміни, внесені через адмін-панель, миттєво оновлюються на публічній частині сайту.

Таким чином, веб-портал повністю відповідає технічному завданню, реалізовані функції працюють на практиці, а структура та логіка додатку забезпечують зручне та ефективне інформування населення.

3.5. Можливості подальшого розвитку веб-порталу для оперативного інформування населення про надзвичайні ситуації та заходи пожежної безпеки

Проект dsns-news-site вже реалізує основні функції оперативного інформування населення: додавання новин, рекомендацій, відображення карти повітряних тривог, а також просту адміністративну панель для редагування контенту. Проте для підвищення ефективності, масштабу та зручності використання платформи є кілька реалістичних напрямків її розвитку.

По перше у плані збереження даних доцільно перейти на базу даних (наприклад, SQLite або PostgreSQL) замість JSON-файлів. Це дозволить реалізувати пошук, сортування, статистику переглядів, архіви новин та рекомендацій.

По друге доцільним є розширення картографічної інтеграції - наприклад, відображення на карті пунктів укриттів, зон ризику, евакуаційних маршрутів. Це перетворить портал на повноцінний інформаційний навігатор у кризових ситуаціях.

По третє, корисним буде впровадження коментарів і зворотного зв’язку від користувачів. Це дасть змогу користувачам не лише читати інформацію, а й

залишати думки, повідомляти про події, уточнювати деталі. Реалізація зворотного зв’язку можлива через просту форму або інтеграцію з месенджерами.

Також у перспективі можна реалізувати багатомовність, зокрема підтримку англійської для міжнародної аудиторії або внутрішньо переміщених осіб, які не володіють українською.

Таким чином, подальший розвиток проекту полягає не лише в додаванні функціоналу, а й у створенні повноцінної платформи для кризового інформування, яка зможе масштабуватись у реальних умовах надзвичайних ситуацій.

Висновки до розділу 3

У цьому розділі обґрунтовано вибір технологій для створення веб-порталу та реалізовано його основні компоненти. Було застосовано стандартні веб-технології HTML, CSS і JavaScript, що забезпечують гнучкий та адаптивний інтерфейс. Серверна частина розроблена на мові Python із використанням мікрофреймворку Flask - легкого і гнучкого рішення для створення веб-додатків. Реалізовано структуру порталу, що включає головну сторінку та окремі розділи: новини, рекомендації та оперативне інформування населення про надзвичайні ситуації. Кожен із цих розділів має відповідні елементами інтерфейсу та навігації, що гарантують доступність та інформативність даних.

Було створено адміністративну панель з авторизацією, яка дозволяє адміну додавати, редагувати та видаляти публікації. Для зберігання даних використано формат JSON. Портал було протестовано в хмарному середовищі Render і успішно запущено онлайн. Render надає можливість швидкого розгортання Flask-додатків за кілька кроків. Такий підхід гарантує стабільну роботу порталу та його доступність для користувачів.

Портал успішно функціонує відповідно до вимог технічного завдання і забезпечує своєчасне інформування населення. Швидкість донесення інформації

під час надзвичайної ситуації має вирішальне значення - система повинна передавати оновлення миттєво.

Окрім виконаних завдань, портал має перспективи подальшого розвитку. Зокрема передбачено перехід від JSON-сховища до реляційної бази даних для підвищення продуктивності зберігання та зручності пошуку інформації. Розширення картографічних функцій - інтеграція інтерактивної карти - дозволяє наочно відображати зони надзвичайних подій і небезпек, що підвищить ситуаційну обізнаність населення. Важливим кроком буде також впровадження багатомовності - портал може надавати інформацію рідною мовою різних груп користувачів, що зменшить мовний бар’єр і підвищить довіру населення. Окремо доцільно реалізувати механізми зворотного зв’язку з користувачами, які допоможуть актуалізувати дані відповідно до потреб і зроблять сервіс ще більш ефективним і доступним.

ВИСНОВКИ

У ході виконання даної кваліфікаційної роботи повністю досягнуто поставлену мету - розроблено сучасний веб-портал для оперативного інформування населення про надзвичайні ситуації та заходи пожежної безпеки. В процесі роботи було проаналізовано існуючі системи оповіщення, спроектовано дизайн порталу та реалізовано всі заплановані функціональні можливості. Результатом є функціонуюча інформаційна система, що відповідає визначеним вимогам і забезпечує належний рівень надійності та зручності у використанні.

Розроблений веб-портал містить адмін панель з авторизацією, яка дозволяє адміністратору додавати, редагувати та видаляти інформаційні матеріали, зокрема новини, рекомендації та оперативні повідомлення. Всі дані реалізовано у форматі JSON, що спростило розгортання проекту та не потребує впровадження повноцінної бази даних. Особливої уваги заслуговує інтеграція з картою повітряних тривог, яка надає актуальну інформацію щодо поточної ситуації в регіонах України у режимі реального часу. Інтерфейс має простий дизайн, що забезпечує коректне відображення контенту та має зручну навігацію. Публічна частина сайту дозволяє користувачам переглядати новини з фільтрацією за категоріями, а також отримувати доступ до повного змісту матеріалу.

Усе перелічене що є в системи успішно реалізовано та протестовано. Портал розгорнуто на безкоштовному хмарному хостингу Render та перевірено в умовах реальної експлуатації. Результати тестування підтвердили стабільну роботу всіх компонентів, правильне функціонування адмін панелі, своєчасне оновлення оперативної інформації та коректну інтеграцію з картою тривог. Таким чином, розроблена система повністю відповідає вимогам, поставленим на етапі проектування, та може використатися практично.

Створений портал підвищує ефективність та швидкість донесення важливої інформації до громадян. В умовах воєнного стану це рішення має особливу актуальність. Служба цивільного захисту отримує простий у

використанні інструмент для оперативного сповіщення населення, а самі громадяни - перевірене джерело інформації щодо подій і дій у разі надзвичайної ситуації. Це значно підвищує обізнаність населення і, зрештою, сприяє збереженню життя та здоров’я.

Разом з тим, веб-портал має потенціал для подальшого розвитку проекту. Перспективним є впровадження модулів сповіщення користувачів про небезпеку через SMS, email або push-нотифікації. У випадку зростання обсягу даних можливий перехід на повноцінну систему баз даних, що підвищить масштабованість і швидкодію. Удосконалення дизайну та забезпечення веб-доступності, а також впровадження двофакторної автентифікації та резервного копіювання сприятимуть підвищенню безпеки та зручності використання ресурсу.

Таким чином, виконана кваліфікаційна робота підтверджує доцільність використання сучасних веб-технологій для розв’язання актуальної соціальної проблеми - оперативного інформування населення про надзвичайні ситуації. Розроблений портал ефективно реалізує поставлені завдання та може бути використаний як основа для подальших удосконалень і масштабування у сфері цивільного захисту.

ВИКОРИСТАНІ ДЖЕРЕЛА

1. Datta, U. Create HTML User Interface for Python using Eel Library [Електронний ресурс]. - 2020. - Режим доступу: https://utsav-datta.medium.com/create-html-user-interface-for-python-using-eel-library-bab101cc0f99
2. Wikipedia JavaScript [Електронний ресурс]. - 2025. - Режим доступу: https://ru.wikipedia.org/wiki/JavaScript
3. Wikipedia HTML [Електронний ресурс]. - 2025. - Режим доступу: https://ru.wikipedia.org/wiki/HTML
4. Wikipedia CSS [Електронний ресурс]. - 2025. - Режим доступу: https://ru.wikipedia.org/wiki/CSS
5. CSS: Cascading Style Sheets [Електронний ресурс]. - 2025. - Режим доступу: https://developer.mozilla.org/en-US/docs/Web/CSS
6. W3Schools. CSS Підручник [Електронний ресурс]. - 2023. - Режим доступу: https://w3schoolsua.github.io/css/index.html#gsc.tab=0
7. Bai L Функції у JavaScript [Електронний ресурс]. - 2022. - Режим доступу: https://xn--80adth0aefm3i.xn--j1amh/%D0%BF%D1%83%D0%B1%D0%BB%D1%96%D0%BA%D0%B0%D1%86%D1%96%D1%8F/59
8. HTML - JavaScript довідка [Електронний ресурс]. - 2023. - Режим доступу: https://xn--80adth0aefm3i.xn--j1amh/HTML
9. Introduction to web APIs [Електронний ресурс]. - 2025. - Режим доступу: https://developer.mozilla.org/en-US/docs/Learn_web_development/Extensions/Client-side_APIs/Introduction
10. W3Schools. Адаптивний веб-дизайн [Електронний ресурс]. - 2025. - Режим доступу: https://w3schoolsua.github.io/css/css_rwd_intro.html#gsc.tab=0
11. JavaScript. MDN Web Docs – документація мови JavaScript [Електронний ресурс]. - 2025. - Режим доступу: https://developer.mozilla.org/en-US/docs/Web/JavaScript

12. HTML (HyperText Markup Language). MDN Web Docs – документація мови розмітки HTML [Електронний ресурс]. - 2025. - Режим доступу: https://developer.mozilla.org/en-US/docs/Web/HTML
13. CSS (Cascading Style Sheets). MDN Web Docs [Електронний ресурс]. - 2025. - Режим доступу: https://developer.mozilla.org/en-US/docs/Web/CSS
14. API Docs | Alerts.in.ua API Docs [Електронний ресурс]. - 2023. - Режим доступу: https://devs.alerts.in.ua/
15. Robbins J. Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics. Sebastopol [Електронний ресурс]. - 2018. - Режим доступу: https://learningwebdesign.com
16. Acsany P. Working With JSON Data in Python [Електронний ресурс]. - 2024. - Режим доступу: https://realpython.com/python-json/
17. Андрєєв А. Що таке JSON. Усе про цей формат передачі даних в інтернеті [Електронний ресурс]. - 2022. - Режим доступу: https://apix-drive.com/ua/blog/useful/scho-take-json
18. Render. Deploy a Flask App on Render [Електронний ресурс]. - 2025. - Режим доступу: https://render.com/docs/deploy-flask
19. Flask Documentation. Security Considerations [Електронний ресурс]. - 2023. - Режим доступу: https://flask.palletsprojects.com/en/stable/web-security/
20. Hardy-Françon G. Flask-tastic Admin Panel: A Step-by-Step Guide to Building Your Own [Електронний ресурс]. - 2025. - Режим доступу: https://www.forestadmin.com/blog/flask-tastic-admin-panel-a-step-by-step-guide-to-building-your-own-2/
21. HOSTiQ. Як зробити прототип сайту? Кращі сервіси для прототипування сайтів [Електронний ресурс]. - 2019. - Режим доступу: https://hostiq.ua/blog/ukr/site-prototype/
22. Oriol Banús. Making a website mockup in Figma (LogRocket Blog) [Електронний ресурс]. - 2023. - Режим доступу: https://blog.logrocket.com/ux-design/making-website-mockup-

figma/#:~:text=A%20website%20mockup%20is%20a,and%20organize%20the%20website%E2%80%99s%20development

23. Каміла Амескан. “Рай для дизайнера або що таке Figma” (блог Kukurudza) [Електронний ресурс]. - 2023. - Режим доступу: https://kukurudza.com/blog/shho-take-figma/#:~:text=Figma%20%E2%80%94%20%D1%89%D0%BE%20%D1%86%D0%B5%3F
24. Flask Documentation (Pallets Projects). https://flask.palletsprojects.com/en/stable/
25. Philipp Acsany. Working With JSON Data in Python (Real Python) [Електронний ресурс]. - 2024. - Режим доступу: https://realpython.com/python-json/#:~:text=The%20JSON%20format%20can%20come,Python%20has%20got%20you%20covered
26. Anthony Herbert. How To Add Authentication to Your App with Flask-Login (DigitalOcean) [Електронний ресурс]. - 2021. - Режим доступу: https://www.digitalocean.com/community/tutorials/how-to-add-authentication-to-your-app-with-flask-login
27. Colin Kraczkowsky. Building a secure admin interface with Flask-Admin and Flask-Security (Medium) [Електронний ресурс]. - 2020. - Режим доступу: https://ckraczkowsky.medium.com/building-a-secure-admin-interface-with-flask-admin-and-flask-security-13ae81faa05
28. Flask-Admin Documentation. Офіційна документація Flask-Admin [Електронний ресурс]. - 2019. - Режим доступу: https://flask-admin.readthedocs.io/en/stable/#:~:text=Why%20Flask,friendly%20interface
29. Miguel Guevara. “GitHub: Not Only For Developers & Why You Should Use It” [Електронний ресурс]. - 2021. - Режим доступу: https://www.miguelgguevara.com/github-not-only-for-developers-why-you-should-use-

it/#:~:text=GitHub%20is%20a%20service%20for,can%20host%20them%20on%20GitHub

30. GitHub Docs [Електронний ресурс]. - 2025. - Режим доступу: https://docs.github.com/ru/get-started/start-your-journey/hello-world
31. Render Documentation. Deploy a Flask App on Render – офіційна документація хмарного сервісу Render [Електронний ресурс]. - 2025. - Режим доступу: https://render.com/docs/deploy-flask
32. Aarush Acharya. Deploy your Flask App on Render (Medium) [Електронний ресурс]. - 2023. - Режим доступу: https://medium.com/@acharyaaarush879/deploy-your-flask-app-on-render-c452a7406fb2

ЗГОДА здобувача вищої освіти

Державного університету економіки і технологій
про перевірку кваліфікаційної роботи на прояви
академічного плагіату
та розміщення в Репозитарії Університету

Я, Борисенко Іван Олексійович, підтримую політику Державного університету економіки і технологій з академічної доброчесності і відкритого доступу.

Засвідчую, що кваліфікаційна бакалаврська робота «Створення веб-порталу для оперативного інформування про надзвичайні ситуації та заходи пожежної безпеки» виконана самостійно та не містить академічного плагіату. Я не надавав і не одержував недозволену допомогу під час підготовки цієї роботи. Робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

Із чинним Положенням про запобігання та виявлення академічного плагіату в роботах здобувачів вищої освіти Державного університету економіки і технологій ознайомлений. Чітко усвідомлюю, що в разі виявлення у кваліфікаційній роботі порушення норм академічної доброчесності робота не допускається до захисту або оцінюється незадовільно.

Також я поінформований, що відповідно до «Положення про Репозитарій (електронну базу даних) Державного університету економіки і технологій» зазначена робота буде розміщена в Електронному архіві Університету (Репозитарії ДУЕТ). З умовами такого розміщення ознайомлений(на).

Дата	підпис	ініціали, прізвище (власноруч)
10.06.2025		І.О. Борисенко