МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) автоматизації та комп'ютерних систем
Кафедра інформаційних систем

«До захисту в ЕК»	**«До захисту допущено»**
Директор інституту(декан факультету)	Завідувач кафедри
_____ Форсюк А.В.	_____ Чумаченко С.М.
(підпис) (прізвище та ініціали)	(підпис) (прізвище та ініціали)
«___» _____ 20__р.	«___» _____ 20__р.

КВАЛІФІКАЦІЙНА РОБОТА
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ БАКАЛАВРА

зі спеціальності 122 «Комп'ютерні науки та інформаційні технології»
(код та назва спеціальності)
освітньо-професійної програми Комп'ютерні науки

на тему: Розроблення frontend сайту кафедри екологічної безпеки та охорони праці НУХТ

Виконав: здобувач 4 курсу, групи КН-4-5

Зарицька Віолетта Віталіївна _____
(прізвище, ім'я, по батькові повністю) (підпис)

Керівник Бойко Регіна Олегівна _____
(прізвище , ім'я та по батькові повністю) (підпис)

Консультанти Бойко Регіна Олегівна _____
(прізвище та ініціали) (підпис)
Бойко Регіна Олегівна _____
(прізвище та ініціали) (підпис)
Бойко Регіна Олегівна _____
(прізвище та ініціали) (підпис)

Рецензент Ладанюк Анатолій Петрович _____
(прізвище та ініціали) (підпис)

Засвідчую, що в цій кваліфікаційній роботі немає запозичень із праць інших авторів без відповідних посилань.

Здобувач_____
(підпис)

Київ - 2020р.

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) автоматизації та комп'ютерних систем
Кафедра інформаційних систем
Освітній ступінь бакалавр
Спеціальність 122 «Комп'ютерні науки та інформаційні технології»
(код і назва)
Освітньо-професійна програма Комп'ютерні науки
(назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри Чумаченко С.М.

"____" _____ 20___ року

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧА

Зарицькій Віолетті Віталіївні
(прізвище, ім'я, по батькові)

1. Тема роботи Розроблення frontend сайту кафедри екологічної безпеки та охорони праці НУХТ

керівник роботи доц., к.т.н. Бойко Регіна Олегівна,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від "27" квітня 2020 року № 269-кс
2. Строк подання здобувачем роботи 1 червня 2020
3. Вихідні дані до роботи
1. *Новини кафедри.*
2. *Фотографії з життя кафедри.*
3. *Інформація про кафедру.*
4. *Категорії новин.*
4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)
1. *Системний аналіз кафедри «Екологічної безпеки та охорони праці» НУХТ.*
2. *Моделі бази даних веб-сайту кафедри «Екологічної безпеки та охорони праці».*
3. *Інтерфейс веб-сайту кафедри.*
4. *Охорона праці та техніка безпеки.*
5. Перелік графічного матеріалу
1. *Структурні схеми.*
2. *Функціональна та концептуальна моделі БД.*
3. *Логічна та фізична моделі бази даних.*
4. *Приклади роботи веб-сайту (інтерфейс користувача)*
5. *Фрагменти коду.*

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	доц., к.т.н. Бойко Р.О.		
2	доц., к.т.н. Бойко Р.О.		
3	доц., к.т.н. Бойко Р.О.		

7. Дата видачі завдання 27.04.2020.

КАЛЕНДАРНИЙ ПЛАН

№ З№	Назва етапів виконання кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Системний аналіз кафедри та постановка задачі на проектування	13.03.2020 - 24.03.2020	Виконано
2	Проектування БД	25.03.2020 - 04.04.2020	Виконано
3	Створення WEB-сайту	05.04.2020 - 15.05.2020	Виконано
4	Написання інструкцій користувача	16.05.2020 - 18.05.2020	Виконано
5	Оформлення пояснювальної записки	19.05.2020 - 24.05.2020	Виконано
6	Оформлення презентації	25.05.2020 - 26.05.2020	Виконано

Здобувач _____ Зарицька В.В.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Бойко Р.О.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Головною метою даної кваліфікаційної роботи є створення front-end частини веб-сайту для кафедри «Екологічної безпеки та охорони праці» Національного університету харчових технологій з покращеним інтуїтивним інтерфейсом, який полегшить роботу з сайтом. Надаватиме можливість отримувати дані з бази даних, проектувати створені адміністратором сторінки, пункти/підпункти меню та інші подібні елементи сайту.

Запропонований WEB-сайт вбиратиме у себе всі сучасні методи розробки, схеми декомпозиції процесів існуючого WEB-сайту кафедри.

Об’єктом дослідження є кафедра «Екологічної безпеки та охорони праці» НУХТ.

Предметом створення є WEB-сайт кафедри «Екологічної безпеки та охорони праці» НУХТ.

Кваліфікаційна робота містить 93 сторінки, 7 таблиць, 20 рисунків, 5 додатків, 16 літературних джерел.

КЛЮЧОВІ СЛОВА: WEB-САЙТ, СИСТЕМА, КАФЕДРА «ЕКОЛОГІЧНОЇ БЕЗПЕКИ ТА ОХОРОНИ ПРАЦІ», ІНТЕРФЕЙС, БАЗА ДАНИХ, СТРУКТУРА.

ABSTRACT

The main purpose of this graduate work is to create a front-end part of the website for the Department of "Environmental Safety and Health" of the National University of Food Technologies with an improved intuitive interface that will facilitate work with the site. It will provide the ability to retrieve data from the database, design administrator-created pages, menu items / sub-items and other similar elements of the site.

The proposed WEB-site will include all modern methods of development, schemes of processes' decomposition of the existing WEB-site of the department.

The object of research is the Department of "Environmental Safety and Health" NUFT.

The subject of creation is the WEB-site of the department "Environmental safety and labor protection" of NUFT.

Graduate work contains 93 pages, 7 tables, 20 figures, 5 appendices, 16 literature sources.

KEY WORDS: WEBSITE, SYSTEM, DEPARTMENT OF ENVIRONMENTAL SAFETY AND HEALTH, INTERFACE, DATABASE, STRUCTURE.

ЗМІСТ

ВСТУП

Останнім часом, підприємства все більше намагаються передати свій продукт на веб-сторінки Інтернету. Головне призначення сайту — це надання інформації. Незважаючи на існування інших способів передачі даних, величезна кількість людей вибирають саме цей спосіб, як найбільш ефективний.

З кожним роком мережа Інтернет стає все більш доступним каналом зберігання і надання інформації для багатьох людей світу. До незаперечних переваг Інтернету можна віднести: швидкість і простота отримання інформації, зручність зберігання і передачі інформації, доступ з будь-якого пристрою, підключеного до всесвітньої мережі.

Завдяки розвитку інформаційних технологій роль сайтів зростає. Від змісту, організаційної структури та функціонування сайту залежить не тільки успіх взаємодії кафедри із зовнішнім світом, а й всі процеси, які проходять в середині нього.

По-перше, за допомогою сайту можна поліпшити інформаційне забезпечення користувачів.

По-друге, сайт є одним (а найчастіше основним) з рекламних та інформаційних інструментів, здатних дати перше уявлення про кафедру.

У будь-якій сучасній кафедрі існує сайт. Це один з елементів престижу, адже саме в Інтернеті потенційні студенти будуть в першу чергу шукати інформацію про кафедру.

На даний момент, існуючий веб-сайт кафедри «Екологічної безпеки та охорони праці» НУХТ не задовольняє сучасних потреб та включає в себе застарілі елементи дизайну, які ще й відображаються некоректно. Та аналіз роботи кафедри та існуючого веб-сайту сприяли правильній розробці нового сайту зі зрозумілим дизайном, який полегшить сприймання та пошук інформації користувачу.

Розділ 1. СИСТЕМНИЙ АНАЛІЗ КАФЕДРИ «ЕБОП» ТА ПОСТАНОВКА ЗАДАЧІ НА ПРОЕКТУВАННЯ

1.1. Загальна характеристика кафедри «Екологічної безпеки та охорони праці»

Об’єктом дослідження є кафедра «Екологічної безпеки та охорони праці» Національного університету харчових технологій.

Кафедра «Екологічної безпеки та охорони праці» Національного університету харчових технологій (надалі — кафедра «ЕБОП» НУХТ) була створена у 1937 році. Вона є частиною факультету «Біотехнології та екологічного контролю» та спеціалізується на здійсненні підготовки фахівців у галузі знань «Природничі науки» за освітнім ступенем «бакалавр», спеціальністю «Екологія» освітньою програмою «Екологія та екоменеджмент», «магістр» за спеціальністю «Екологія» та «Технології захисту навколишнього середовища» за освітніми програмами «Екологія та охорона навколишнього середовища» та «Екологічний контроль та аудит» відповідно [1].

З жовтня 2018 року після реорганізації та об’єднанням з кафедрою безпеки життєдіяльності розширилось коло кафедральних дисциплін, а саме «Охорона праці та безпека життєдіяльності», «Основи охорони праці», «Основи будівництва в галузі» та «Основи промислового будівництва та санітарної техніки», які викладаються студентам більшості спеціальностей університету.

Впродовж існування кафедри для студентів всіх спеціальностей НУХТ викладалися загально-наукові дисципліни, а саме «Екологія», «Основи охорони праці та безпека життєдіяльності», «Основи промислового будівництва та санітарної техніки». На самій кафедрі забезпечують викладання студентам-екологам 42 сновні фахові дисципліни.

Кафедра «ЕБОП» у НУХТ виділяється серед подібних кафедр інших університетів за декількома аспектами:

1. На кафедрі було створено навчальні спеціалізовані лабораторії, що обладнані приладами за новими стандартами та установками для студентів кафедри; також мають місце лабораторії, які призначені для екологічного контролю.

2. Викладачі кафедри «ЕБОП» без перестанку вдосконалюють принципи навчання, додаючи сучані методології навчання та наукові досягнення.

3. Між НУХТ та організаціями екологічного напряму, підприємствами харчової та переробної промисловості (ЗАТ «Оболонь», Київські хлібозаводи № 4, №10, м'ясо-переробні комбінати, Київський завод шампанських вин, ЗАТ «Росинка», Вінницький олієжировий комбінат, ТОВ «Спецтехніка», Яготинський молокозавод тощо), підприємствами з очищення стічних вод та утилізації відходів (Бортницька станція аерації, Деснянська станція водо-підготовки, сміттєспалювальний завод «Енергія»), науково-дослідними та галузевими інститутами укладено певні угоди, які надають можливість студентам проходити преддипломну практику, а також надають їм можливість вирішувати питання працевлаштування після закінчення навчання в університеті [2].

1.2. Організаціна структура факультету

1.2.1. Структура факультету «БТЕК»

Верхівку структурної схеми факультету «БТЕК» займає декан. Наступний рівень складають вчена рада факультету, заступники дкана за напрямами, деканат та рада студентського самоврядування. Повна структура управління факультетом представлена у *додатку А, рис. А1*.

Основними структурними підрозділами факультету є кафедри:

- біотехнології та мікробіології;
- фізики;
- екологічної безпеки та охорони праці;
- фізичного виховання.

Кафедри проводять навчально-виховну та методичну діяльність з однієї або кількох споріднених спеціальностей або навчальних дисциплін, а також здійснюють наукову, науково-дослідну та науково-технічну діяльність за певним напрямом.

1.2.2. Опис роботи кафедри «ЕБОП»

Головним завданням кафедри є організація та здійснення на високому рівні навчальної і навчально-методичної роботи із закріплених за кафедрою навчальних дисциплін, виховної роботи серед студентів, наукових досліджень відповідно до профілю кафедри, підготовки науково-педагогічних кадрів та підвищення їх кваліфікації.

Основними напрямками діяльності кафедри є:

- навчальна робота;
- методична робота;
- науково-дослідна робота;
- організаційна робота;
- виховна робота.

Функції кафедри: розробка навчальних програм дисциплін кафедри, постійний контроль якості навчання студентів з дисциплін кафедри, участь у науково-дослідній роботі університету, організація і супроводження веб-сторінки кафедри.

До складу кафедри входять завідувач кафедри, замісник завідувача, професори, доценти, старші викладачі, викладачі, аспіранти та навчально-допоміжний персонал. Організаційну структуру кафедри можна побачити у *додатку А, рис. А2*.

Завідувач кафедри забезпечує організацію осітнього процесу, виконання навчальних планів і програм навчальних дисциплін, контролює якість викладання дисциплін кафедри, контролює виконання правил з охорони праці та протипожежної безпеки особами, які навчаються та працюють на кафедрі.

1.2.3. Взаємодія кафедри з іншими підрозділами університету

Таблиця 1. Взаємодія кафедри з іншими підрозділами університету

Кафедра отримує **1**	**Кафедра надає** **2**
Взаємодія з управлінням університету	
• Накази, розпорядження ректора, інформаційні або службові листи проректорів за напрямками діяльності. • Матеріали про роботу Вченої ради факультету та університету і його комісій (рішення, постанови). • Матеріали з виховної роботи.	• Положення про кафедру. • Акт про готовність кафедри до нового навчального року. • Копії посадових інструкцій спів-робітників кафедри. • Плани та звіти про роботу кафедри. • Матеріали з виховної роботи кафедри.
Взаємодія з факультетом	
• Затверджені навчальні плани за напрямами підготовки. • Розклад навчальних занять, іспитів та заліків. • Проект штатного розкладу кафедри. • Списки студентів по навчальним групам	• Проекти навчальних планів. • Матеріали з обліку успішності студентів. • Штатний розклад. • Матеріали про організацію і проходження практик студентами. • Матеріали по самостійній роботі студентів. • Графік чергових трудових відпусток. • Дані та аналіз анкетування учнів по задоволеності навчальним процесом.

1	2
Взаємодія зі службами проректора по методичній роботі	
• Обсяг навчального навантаження кафедри. • Пропозиції щодо формування потоків. • Інформація по конференціях.	• Погоджений список груп в потоках. • Розподіл навчального навантаження по викладачах на паперових та електронних носіях. • Поточні зміни в навчальному навантаженні викладачів кафедри. • Плани роботи кафедри. • Список дисциплін за вибором студентів. • Проекти навчальних планів.
Взаємодія з іншими кафедрами	
• Взаємодія з поліпшення якості освітнього процесу. • Навчально-методичний комплекс дисциплін, закріплених за кафедрою.	• Взаємодія з поліпшення якості освітнього процесу. • Навчально-методичний комплекс дисциплін, закріплених за кафедрою.
Взаємодія з вченою радою факультету	
• Рекомендації щодо розробки освітніх програм. • Графік організації підвищення кваліфікації науково-педагогічних працівників у поточному році. • Перелік баз практик.	• Звіти про роботу занять кафедри. • Відомості про успішність студентів. • Відомості для заповнення модулів щорічної звітності. • Проекти наказів на розподіл студентів на практики.

1.3. Аналіз нинішнього стану WEB-сайту кафедри «ЕБОП»

Нинішній сайт кафедри «ЕБОП» створений на платформі «Wix», яка надає послуги створення WEB-сторінки за допомогою конструктора сайтів.

Конструктор сайтів — програмно-реалізована складна система для створення WEB-сторінок без знань мов програмування [4].

Сайт кафедри являє собою систему електронних документів, що складається з файлових даних та коду, які регулюють його роботу. Система має підключення до бази даних, яка надається конструктором «Wix».

Інтерфейс WEB-сайту створений за допомогою шаблону, який призначений саме для висвітлення подій, адже конструктор «Wix» надає можливість вибрати його тематику ти стиль. Структура сайту передбачає обробку та виведення даних на сторінки сайту, які зберігаються у базі даних.

При аналізі сайту з технічної точки зору було виявлено декілька позитивних сторін даної розробки, а саме:

- розроблена сторінка помилок «404», яка повідомляє корстувачів о переході на неіснуючу сторінку сайту;
- конструктор використовує нову версію HTML (HyperText Markup Language - мова розмітки документів в Інтернеті) — HTML 5, що передбачає можливість не переписувати код під нові стандарти у майбутньому;
- на сторінках не задіяні фрейми, що значно прискорює роботу сайту;
- вказано кодування мови — це допоможе запобіганню проблем з відображенням спеціальних символів;
- встановлений метод стиснення файлів Gzip дозволяє зменшити розмір WEB-сторінки і любих інших типових WEB-файлів приблизно до 30% від їх початкового розміру до їх передачі — використовується для більш швидкої передачі файлів у мережу.

На жаль, на сьогоднішній день цих якостей не достатньо для того, щоб бути повноцінним конкурентноспособним інтернет-ресурсом кафедри

«ЕБОП» і є низка проблем, які пов’язані з кодом, дизайном та оптимізацією. Тому необхідне вирішення цих проблем шляхом розробки нового WEB-сайту.

1.4. Розроблення функціональної моделі процесів кафедри «як-є»

Так як на кафедрі вже існує WEB-сайт, то було прийнято рішення зробити саме його дослідження та аналіз. Це необхідно задля виявлення недоліків існуючого сайту кафедри «ЕБОП» та виконується з метою визначення можливих шляхів удосконалення WEB-сайту за допомогою функціональної моделі, створеної з використанням CASE-засобу AllFusion Process Modeler [5].

Дана модель має статус «AS-IS» — «як є» та має 1 рівень декомпозиції.

Контекстна діаграма, що має назву «Організація роботи frontend сайту кафедри Екологічної безпеки та охорони праці НУХТ», складається з двох рівнів декомпозиції і її можна переглянути у *додатку Б, рис. Б1*.

Вхідними даними являються відвідувачі сайту та збережені події (події, які були збережені у базі даних адміністратором на керуючій панелі та ще не були додані до сторінок сайту)

Вихідні дані представлені проінформованим студентом.

Механізми — це база даних, веб-браузер та адміністратор сайту.

Керування відбувається такими елементами: документація та правила оформлення веб-сайту.

Після контекстної діаграми йде її декомпозиція, яка складається з чотирьох подій. Діаграму першого рівня декомпозиції можна побачити у *додатку Б, рис. Б2*.

Дані, які були збережені у базі даних на керуючій панелі, отримуються сайтом з БД для подальшої обробки інформації. Після їх отримання адміністратор повинен створити (або скорегувати) елементи сайту під нові дані за певними правилами овермлення веб-сайтів. Це займає досить багато часу та є непрактичним для сучасних реалій, бо вже є технології, які автоматично створюють сторінки по написанному шаблону.

Після отримання згенерованих елементів починається вигрузка сайту з оновленими даними. Це також лишні дії, які гальмують процес створення нових елементів та ще й на певний час робить сайт непрацюючим. Після вигруження нова інформація вже відображається на сайті, та відвідувачі через браузер можуть її переглянути. В підсумку студенти отримують нові новини та стають проінформованними.

1.4.1. Виявлені проблеми

В результати моделювання веб-сайту кафедри, який використовується зараз, були виявлені такі проблеми:

1. Елементи сторінки не створюються автоматично.
2. Відсутність пошукової оптимізації сайту.
3. Відсутність коректного виведення нових елементів.
4. Сайт не оновлюється динамічно.
5. Сайт не адаптується під мобільні пристрої.
6. Відсутність можливості фільтрації даних на сторінці.

1.4.2. Задачі автоматизації

1. Розробка автоматичного створення елементів сайту при оновлені даних у БД.
2. Встановлення пошукової оптимізації сайту.
3. Забезпечити коректним виведенням нових елементів.
4. Автоматизувати оновлення сайту.
5. Забезепечити сайт адаптацією на мобільних пристроях.
6. Розробити фільтрацію даних на сторінці.

1.5. Аналіз існуючих аналогів розробки WEB-сайтів

CMS (content management system або система управління контентом) — програмна основа для розробки і редагування сайту. Якщо говорити простіше: це конструктор, який дозволяє створити веб-ресурс і наповнювати

його статтями, фотографіями, відео і іншими даними. CMS, як і будь-який інший конструктор, може бути дуже простим: деталі великі, без якихось складних елементів, бо їх відносно небагато, тож заплутатися просто неможливо (як і нереально побудувати щось дійсно унікальне). А може бути і досить витонченим: маса різних деталей, які поєднати в щось одне не так то й легко. Важливо відзначити, що коли появляется певний рівень вправності, сотні і тисячі компонентів перетворюються в потужні оригінальні рішення. Розглянемо найбільш популярні CMS системи.

1.5.1. WordPress

WordPress — це система управління вмістом сайту CMS з відкритим вихідним кодом, поширювана під ліцензією General Public License (GPL) версії 2 [6]. Написана на мові програмування для побудови веб-додатків Hypertext Preprocessor (PHP), в якості бази даних використовує My structured query language (MySQL). Сфера застосування — від блогів до досить складних новинних ресурсів і навіть інтернет-магазинів. Вбудована система «тем» і «плагінів» разом із вдалою архітектурою дозволяє конструювати практично будь-які проекти. WordPress забезпечує комфортний і нетрудомісткий процес розробки сайту.

Головні переваги WordPress

1. Простота. Створення сайту на WordPress не вимагає практично ніяких спеціальних навичок. Більш того: багато хостинг-провайдери позволяють встановити WordPress в один клік. Опублікувати новий контент на цій CMS не складніше, ніж друкувати щось в програмі MS Word. Про роботу з WordPress написані мільйони статей на різних мовах, зняті детальні відео, опубліковані тисячі книг. На будь-яке питання щодо WordPress можна знайти відповідь в лічені хвилини.

2. Маса безкоштовних можливостей. Настроюються теми для створення унікального дизайну, плагіни для розширення функціоналу — за все це не

потрібно платити ні копійки і все це можна знайти, не виходячи з адміністративної панелі сайту.

3. Постійний розвиток. WordPress створили професійні розробники. Вони ж (спільно з тисячами волонтерів-ентузіастів у всьому світі) до сих пір беруть участь в постійних поліпшеннях «движку». Після кожного релізу CMS стає швидше, зручніше, безпечніше. Коли вибирають WordPress в якості платформи для свого сайту, всі точно знають, що проект не згорнеться в найближчі кілька років — він тільки може стати краще.

Недоліки WordPress:

1. Порівняно обмежені можливості налаштувань. Це зроблено спеціально. Розробники не хочуть заплутувати користувачів. Новачки це цінують. Зате більш-менш продвинуті користувачі не завжди можуть реалізувати якусь витончену задумку.

2. Сайт на WordPress легко створити, але також легко і зіпсувати. Значна частина відео-уроків і гайдів по WordPress записана людьми, які дуже далекі від веб-розробки. Наприклад, вони радять як вносити правки в код, але не попереджають (бо не знають), що всі правки зникнуть після поновлення ядра CMS або теми. Необхідно бути дуже уважним, коли намагаєтеся щось поліпшити на сайті, переглядаючи сумнівні відео на YouTube.

1.5.2. Joomla

Joomla є безкоштовною системою для створення веб-сайтів. Це проект з відкритим вихідним кодом, який, як і більшість подібних проектів, не стоїть на місці. Він дуже успішно розвивається, протягом ось уже семи років, і користується популярністю у мільйонів користувачив по всьому світу [7].

Joomla — друга за популярністю CMS в світі. Як і WordPress, вона безкоштовна, неймовірно проста, доступна і надійна. По суті, вона володіє всіма якостями WordPress. Головна відмінність Joomla від більш успішного відкритого рішення — з нею трохи складніше впоратися. Її не так просто встановити, налаштувати і наповнити контентом.

Не так просто, як WordPress, але теж дуже легко. Також можна обійтисяь без спеціальних знань в області PHP і HTML. Для створення простого сайту досить подивитися кілька відео уроків на YouTube.

Головні переваги CMS Joomla

1. Для цієї платформи існують зручні багатофункціональні програми для створення інтернет-магазинів різної складності: VirtueMart, Joomshopping, Tienda і т.п. Для WordPress теж є спеціальні рішення для електронної комерції (наприклад, WooCommerce). Все ж Joomla вважається набагато більш підходящим вибором, якщо планується розробляти інформаційну WEB-сторінку. Створити повнофункціональну WEB-сторінку на Joomla простіше і швидше, ніж на WordPress.

2. Сайт на Joomla можна оптимізувати без додаткових плагинів. Користувач з легкістю може прописати мета-теги і налаштувати URL (Uniform Resource Locator — уніфікований покажчик ресурсу).

3. В «движку» відразу є кешування. Це прискорює завантаження сторінок, підвищує позиції сайту.

4. Сайт на Joomla можна редагувати, не заходячи в адміністративну панель. Доступне front-end редагування контенту.

Недоліки CMS Joomla

Об'єктивно недоліки у Joomla такі ж, як і у WordPress — занадто багато навчальної інформації, серед якої даремні і навіть шкідливі поради та уроки; трохи обмежені можливості (хоч і не так обмежені, як у WordPress), велика кількість бажаючих знайти слабкі місця в коді і навчитися зламувати сайти на цій CMS.

1.5.3. Wix

Це умовно безкоштовна онлайн платформа, яка дозволяє створити веб-сайти і їх мобільні версії. Wix надає готові професіональні шаблони дизайну і HTML5-редактор, який працює за принципом Drag-and-Drop [8]

Шаблони налаштовуються у таких категоріях:

- додавання нових функцій та медіа;
- можливість зміни стилю;
- можливість зміни кольорів;
- можливість зміни тексту;
- додавання фонових зображень;
- додавання кнопок та інше.

Існує колекція безкоштовних зображень, форм і іконок.

Wix працює по бізнес-моделі freemium, пропонуючи можливість створити сайт безкоштовно і розвивати його, збільшуючи функціонал. Наприклад, тарифи Premium дозволяють підключити до сайту власний домен, прибрати банери Wix, додати онлайн-магазин, отримати додаткове місце для зберігання даних, купони на рекламу та інше.

Усередині редактора також інтегрований App Marker, в якому представлені додатки, створені різними компаніями з використанням автоматизованої технології веб-розробки Wix. App Market пропонує безкоштовні та платні програми і дозволяє інтегрувати на сайт такі функції як галереї фотографій, блоги, плейлисти, онлайн-спільноти, розсилки електронних листів і файлові менеджери. В App Market можна знайти готові рішення від Google, Instagram, LiveChat, Shopify та інших провідних компаній.

Ключові переваги Wix:

- безкоштовний хостинг;
- підключення власного домену;
- оптимізація для мобільних пристроїв;
- додавання зовнішнього HTML-коду;
- захист сторінок;
- модулі блогу, інтернет-магазину, списку, галереї зображень,
- відео та аудіо;
- постинг в соціальних мережах;
- блоки статистики.

Wix хвалять за зовнішню привабливість, яка робить роботу з конструктором приємною. Але це далеко не всі переваги сервісу. Також можна відзначити:

- адаптивність шаблонів;
- зручний інтерфейс візуального редактора, з яким зможе швидко розібратися навіть новачок;
- магазин додатків з двома сотнями віджетів;
- величезна база знань, яка допомагає розібратися з інтерфейсом конструктора;
- можливість додавання коду HTML на сторінки;
- єдина панель управління інтернет-магазином, через яку можна відстежувати замовлення, отримувати платежі і управляти доставкою;
- повна статистик по сайту — кількість переглядів сторінок, джерела трафіку, позиція в пошуковій видачі і т.д.

Якщо повернутися до зовнішньої складової конструктора, то можна ще раз відзначити що шаблонів на Wix дійсно багато. Вони грамотно розподілені за категоріями і типами сайтів, плюс підтримується пошук по ключевим словам і фільтри «Нові», «Популярні», «Порожні».

Перш ніж вибрати макет, можна подивитися, як він буде виглядати на моніторі та на екрані мобільного пристрою. Якщо жоден з варіантів дизайну не підійшов, можна відфільтрувати пропозиції, використовуючи розділ «Порожні», і створити власний шаблон з нуля. На Wix можна створювати односторінкові сайти. Готові шаблони для Лендінзі знаходяться в одноймменному розділі.

Головна претензія до Wix — високі ціни і незрозумілий поділ преміум-планів. Connect Domain, після оплати якого на сайті зберігається реклама конструктора, навряд чи комусь потрібен; eCommerce для створення інтернет-магазину не пропонує нічого надприродного, а ціну має таку, ніби сам збере сайт за день.

Проблемою для веб-майстрів початківців може стати перевантажений інтерфейс. Його легко пояснити функціональністю конструктора, але на

освоєння сервісу знадобиться не один день: треба буде не тільки розібратися з тим, що де знаходиться, але й навчитися правильно застосовувати інструменти. На Wix можна зіпсувати шаблон, тому що можливостей для ре-вання у користувача дуже багато.

Малоінформативним залишається розділ «Підтримка». Опис того, як виконуються в редакторі основні дії знайти можна, але без скріншотів буває важко зрозуміти, куди натискати. Можливо, це марна зачіпка, але якщо сервіс позиціонується як конструктор для новачків, то логічно очікувати макси-мально зрозумілого представлення інформації.

1.5.4. 1С-Бітрікс

«1С-Бітрікс» — популярна комерційна CMS від російських розробників. Створювалася в першу чергу для високонавантажених проектів: інфор-маційних порталів, інтернет-магазинів, сайтів великих компаній і державних організацій. Найбільшого поширення набула в електронній комерції.

В архітектурі продукту реалізований принцип MVC, поділ логіки від подання. Управління структурою відбувається за допомогою Інфоблоки, вони порівнянні з базою даних, кожен інфоблок це якась коробка яку можемо налаштовувати саме під ту інформацію яку хочемо в ній зберігати. Вся візуальна частина знаходиться окремо що дає гнучкість в управлінні дизайном сайту.

Розширення в 1С-Бітрікс як безкоштовні, так і платні. Всі розширення знаходяться в магазині Маркеплейс: купивши рішення ми отримуємо підтримку розробників і зворотний зв'язок.

Технічна підтримка Бітрікс оперативно вирішує завдання, є докумен-тація і навчальний матеріал з управління двигуном як для користувача, так і розробника. 1С-Бітрікс має свою спільноту розробників, форуми, блоги, де допоможуть вирішити ті чи інші завдання.

Високий ступінь захисту самої платформи забезпечується за рахунок вбудованого модуля проактивного захисту. В неї входить двохетапна

авторизація, захист сесії, обмеження по IP і багато додаткових фішок, які не дадуть просто так зламати сайт.

Розділи і сторінки сайту мають зрозумілу структуру, реалізовано за принципом папок і файлів, як на комп'ютері. Завдяки технології Ермітаж є можливість редагувати сторінки, змінювати налаштування сайту прям через візуальну частину сайту, не заходячи в адміністративний розділ, що у багато разів полегшує її адміністрування [9].

Переваги 1С-Бітрікс

1. Коробкова версія включає всі необхідні модулі для роботи з сайтом. У кожній редакції продукту свій набір модулів — чим дорожче, тим ширше функціонал.

2. Зручна і інтуїтивно зрозуміла адміністративна панель дозволяє легко управляти контентом сайту.

3. Система гнучка і кастомізуюча — можна створити і підтримувати проект практично будь-якої складності і будь-яких масштабів. Регулярні оновлення забезпечують стабільну роботу і відкривають нові можливості.

4. Є магазин готових рішень «Маркетплейс 1С-Бітрікс», де можна знайти безліч розробок, які не просто розширюють штатні можливості системи, так і дозволяють створити готовий сайт в найкоротші терміни без залучення технічних фахівців.

5. Модуль «Пошукова оптимізація» відкриває широкі можливості для просування в пошукових системах.

6. Інтеграція з «1С: Підприємство». Можна відстежувати всі покупки через сайт компанії, а статус замовлення — в особистому кабінеті. Також можлива синхронізація каталогу через «1С» — вся інформація про товари буде оновлюватися автоматично [10].

7. Сайти, створені на платформі проходять моніторинг якості. Проходження моніторингу хоч і не дає стовідсоткової гарантії, але дозволяє переконатися, що в процесі розробки не було допущено критичних помилок.

8. Система «Композитний сайт» — унікальна, запатентована технологія, яка прискорює завантаження ресурсу за рахунок оптимізації процесу доставки контенту і створення кешованих копій сторінок [11].

Головні недоліки

1. Вимогливість до ресурсів. Сервер повинен бути досить потужним. Однак багато хостерів пропонують доступні тарифи для «1С-Бітрікс», що знімає проблему.

2. Надмірність коду і складна архітектура. Для підтримки сайту і доробок потрібен програміст, який вивчив систему.

3. Для впровадження і зв'язку з «1С: Підприємство» необхідно залучати професіоналів.

4. Вартість розробки і підтримки. Крім вартості самого «движку», необхідно врахувати, що вартість роботи програмістів на «1С-Бітрікс» зазвичай вище середнього.

5. Платне оновлення системи. Термін оновлення лише 1 рік.

На «1С-Бітрікс» добре працюють складні, великі і високонавантажені проекти. Для маленького сайту краще вибрати CMS простіше — це дозволить заощадити гроші на розвиток продукту і сервісу.

1.5.5. Порівняння систем-аналогів

Порівнянння систем аналогів наведені у таблиці 1.

Вимоги, за якими проводолось порівняння CMS систем:

1. Легкість установки — система встановлена так, що навіть людина далека від програмування має змогу з легкістю установити програму. Не пред’являє особливих вимог до хостингу та працює на більшості ОС.

2. Підтримка розробників — при певних умовах (наприклад, при певному пакеті) система надає можливість скористатися допомогою та отримати більш професійну розробку сайту. У майбутньому це забезпечить сайт замовленими оновленнями без втручання в код користувача.

3. Безпека — гарантує надійність при користуванні системою: може включати в себе фаєрвол, антивірус та захист від DDOS атак.

4. Простота освоєння системи — мануали та інструкції знаходяться у відкритому доступі, що не викликає особих проблем з освоєнням системи.

5. Має безкоштовний пакет — система пропонує безкоштовний набір інструментів для створення простого WEB-сайту.

6. Не потребує багато ресурсів — не обов'язково сервер має бути потужним для підтримки сайту у мережі Інтернет, не витрачає багато пам'яті.

7. Підтримка HTML/CSS — можливість для користувачів із знаннями HTML/CSS редагувати та доповнювати шаблони для отримання необхідного результату.

8. Можливість використовувати шаблони — готові приклади дизайну на різні тематики, які можна використати у своєму сайті .

9. Багатофункціональність — можливості сайта можна розширити, скориставшись можливістю підключення плагінів.

10. Простота використання — зручний та інтуїтивно зрозумілий інтерфейс, який дозволяє лекго користуватися можливостями системи.

Таблиця 2. Порівняння систем-аналогів

Системи **Вимоги**	***WordPress***	***Joomla***	***Wix***	***1С-Бітрікс***
1	***2***	***3***	***4***	***5***
1. Легкість установки	+	+	+	—
2. Підтримка розробників	—	—	—	+
3. Безпека	—	—	+	+

1	2	3	4	5
4. Простота освоєння системи	+	—	—	+
5. Має безкоштовний пакет	+	+	+	—
6. Не потребує багато ресурсів	+	—	+	—
7. Підтримка HTML/ CSS	+	+	+	+
8. Можливість використовувати шаблони	+	+	+	+
9. Багатофункціо-нальність	+	—	+	+
10. Простота використання	+	+	—	—
11. Вартість	**до 1000$**	**до 1000$**	**до 600$**	**до 3533$**

WordPress — задовольняє більшість вимог, але не забезпечує WEB-сайт безпекою, що є одним із головних критерієв вибору системи.

Joomla — дана система є досить дешевою, але має багато мінусів, які дають зрозуміти, що саме ця система нам не підходить; найголовніший мінус — це незабезпечення безпекою від спроб злому та DDOS атак.

Wix — хоч дана система і надає можливість створити сайт за мінімальну плату, та для повноцінного WEB-сайту треба купляти підписку, яка матиме необхідні інструменти, а це коштує досить неоправдано дорого. Також багато часу піде на вивчення системи, що збільшить час створення WEB-сайту.

1С-Бітрікс — оптимальна система для використання, але має занадто високу вартість при недотриманні деяких вимог.

1.6. Обґрунтування доцільності проектування й розроблення WEB-сайту кафедри «ЕБОП» НУХТ

Як було зазначено у п 1.3, існуючий WEB-сайту кафедри «ЕБОП» НУХТ не має ті важливі критерії, які необхідні для того, щоб бути повноцінним конкурентноспособним інтернет-ресурсом. З навединих у п.1.5 результатів порівняння існуючих альтернативних систем CMS видно, що вони є або занадто дорогими для кафедри «ЕБОП», або не відповідають вимогам, які необхідні для створення WEB-сайту, перелік яких наведено у п.1.4.

Отже, проектування та розробка нового WEB-сайту кафедри «ЕБОП» НУХТ є доцільними, адже його створення передбачає вирішення основних проблем, які впливають на пошукову оптимізцію, зручність у використанні на різних девайсах, зрозумілість навігації та зовнішній вигляд сайту взагалом.

1.7. Концептуальна модель

Для повного розуміння змін, які будуть внесені при створенні нового веб-сайту кафедри, була створена концептуальна модель TO-BE (*додаток Б, рис. Б3-Б4*) [5].

З даної моделі можна побачити, що два процеси «Створення елементів» та «Завантаження сайту», що продемонстровані у декомпозиції функціональної моделі TO-BE (*додаток Б, рис. Б2*), на концептуальній заміщуються одним процесом «Автоматична генерація елементів веб-сайту». Це значно прискорить роботу і сайту, і адміністраторів, адже елементи будуть створюватись самостійно по завчасно підготовленому програмістом шаблону.

Через використання NoSQL бази даних Firebase Realtime Database додався ще один процес — «Форматування даних». Так як дані зберігаються у вигляді HTTP-запиту, то для правильної роботи їх треба перевести у JSON. Це не буде займати багато часу та витрачатиме мінімум ресурсів.

З’явився процес фільтрації даних, який забезпечить можливість вибору елементів новин по категоріях.

Покращення дизайну та відвідуваності сайту забезпечать правила адаптивності та норми пошукової оптимізації відповідно. Через збільшення кількості відвідувачів збільшиться й кількість потенційних студентів — абітурієнтів, які лише вибирають місце навчання.

1.8. Постановка задачі

1.8.1. Призначення та цілі створення WEB-сайту кафедри «ЕБОП»

Основними користувачами WEB-сайту будуть особи, які мають спеціальне програмне забезпечення — «браузер», доступ до мережі Інтернет та зайдуть на любу із сторінок сайту. Також важливу роль буде мати адміністратор, який зможе додавати, редагувати, видаляти контент та елементи сайту.

Призначення WEB-сайту кафедри «ЕБОП» НУХТ полягає у покращенні якості донесення інформації до користувачів (студентів, абітурієнтів, викладаців тощо) сучасними методами.

Основними цілями створення WEB-сайту є:

1. Висвітлення актуальних подій, які сталися на кафедрі.
2. Розміщення матеріалів інформаційного характеру, орієнтовані на читача.
3. Оновлення дизайну за сучасними тенденціями.
4. Залучення абітурієнтів до вибору кафедри.
5. Підтриманя та розвиток іміджу кафедри.

Головним моментом тут є створення акценту на самій кафедрі. Висвітлюються її принципи, концепція, перспективи.

1.8.2. Вимоги до створюваного WEB-сайту

Для повного функціонування WEB-сайт повинен виконувати такі вимоги:

1. WEB-сайт повинен мати захіщеність інформації.

2. WEB-сайт повинен мати адаптивний та простий інтерфейс.

3. WEB-сайт повинен бути сумісним з браузерами різних компаній та версій.

4. Для нормальної пошукової оптимізації повинен мати ключові слова (мета-теги) та спеціальні структурні теги у HTML-коді.

5. WEB-сайт повинен коректно відображати всі елементи на сторінці.

6. WEB-сайт повинен мати елемент для пошуку по конкретних запитах.

7. Для написання WEB-сайту використовувати середовище розробки Visual Studio Code.

8. Збірку проекту WEB-сайту повинна виконувати система Vue Cli.

9. WEB-сайт повинен мати довгий срок експлуатації.

10. В якості СУБД використовувати Firebase Realtime Datebase.

11. WEB-сайт повинен бути динамічним.

1.8.3. Функції, які повиннен виконувати WEB-сайт

Далі наведено основні функції, які має виконувати WEB-сайт кафедри «ЕБОП» НУХТ:

1. Автоматичне отримання даних із бази даних.
2. Генерація HTML-коду елементів сайту.
3. Підгрузка даних при скролінгу WEB-сторінки.
4. Публікація новин, статей, текстового матеріалу.

5. Фотогалерея.

6. Пошук по сайту.

7. Фільтрація новин за категорією.

1.8.4. Вхідні та вихідні дані WEB-сайту

Вхідні дані: збережені дані через адміністративну панель, відвідувач сайту.

Вихідні дані: потенційний студент, проінформаваний студент, від-фільтровані дані.

РОЗДІЛ 2. ОПИС КОМПЛЕКСУ ЗАДАЧ АВТОМАТИЗАЦІЇ

2.1. Обґрунтування вибору засобів розробки WEB-сайту

Розробка сайту кафедри «ЕБОП» буде включати в себе нові та найактуальніші web-інструменти. Вибрані засоби не лише дозволять створити багатофункціональний сучасний сайт, а ще й не потребують значних витрат коштів.

Побудова логічно-фізичної моделі даних була проведена за допомогою AllFusion ERWin Data Modeler.

Для зниження кількості помилок при проектуванні системи необхідно попередньо провести повний аналіз системи та зробити її попереднє моделювання. Для цього чудово підходять інструменти CASE-технологій.

Далі на основі створених моделей була згенерована база даних в Firebase Realtime Database — хмарна СУБД класу NoSQL, що дозволяє розробникам додатків зберігати і синхронізувати дані між декількома клієнтами. Підтримані особливості інтеграції з додатками під операційні системи Android і iOS, реалізовано API для додатків на JavaScript, Java, Objective-C і Node.js, також можливо працювати безпосередньо з базою даних в стилі REST з ряду JavaScript-фреймворків, включаючи Vue.js, AngularJS, React, Ember.js і Backbone.js. Передбачено API для шифрування даних.

Однією з особливостей Firebase Realtime Database є синхронізація у реальному часі для даних JSON — це дозволяє зберігати та синхронізувати дані між користувачами в режимі реального часу. Також завдяки цьому надається можливість користувачам отримувати доступ до своїх даних з любого девайсу.

JSON (JavaScript Object Notation) — це текстовий формат представлення даних в нотації об'єкта JavaScript. Він використовується для обміну даними. Дуже часто використовується в якості формату для передачі інформації від веб-сервера клієнту (веб-браузеру) по AJAX запиту.

Цей процес можна представити у вигляді двох кроків. На першому кроці, сервер, по запиту прийшов йому від клієнта, спочатку формує певний набір

даних в зручному форматі, який потім можна було б дуже просто упакувати в рядок JSON. Завершується робота на сервері відправкою JSON даних в якості результату клієнтові. На другому кроці, клієнт отримує в якості відповіді від сервера рядок JSON і розпаковує її, тобто переводить в JavaScript об'єкт. Після цього на клієнті виконуються подальші з ними дії, наприклад, виводяться на сторінку.

Керування сторінкою було створено за допомогою об’єкто-орієнтованї мови програмування JavaScript — вона служить для керування сценаріями перегляду Web-сторінок. Головна особливість цієї мови полягає в тому, що при його використанні є можливість змінювати властивості середовища відображення при перегляді Web-сайту і при цьому не буде відбуватися перезавантаження Web-сторінок. Наприклад, за допомогою мови JavaScript можна замінити колір фону Web-сторінки або замінити зображення, інтегроване в Web-сторінку, також є можливість створити нове вікно відображення або вивести різні повідомлення.

Щоб уникнути постійний повтор коду та зменшити цим об’єм основних файлів, було рішення використати JavaScript фреймворк — Vue.js 2.0 [12]. Головні переваги Vue.js, які вплинули на рішення його вибру, є мінімальний синтаксис шаблонів, можливість використовувати компоненти для циклічного створення елементів, легкість розширення шаблонів за допомогою підключення інших технологій, наприклад Sass, впроваджено віртуальний DOM (це означає, що фреймворк може сам зрозуміти, що змінилось в стані DOM і згодом ефективно застосувати його оновлення, зводячи до мінімуму повторний рендер і оптимізацію продуктивності додатку).

Для швидкого створення проекту на Vue.js було використано інтерфейс командної строки Vue Cli — це консольна утиліта, яка створює каркас додатку Vue.js на основі одного з наявних та вибраних шаблонів. Вона автоматизує процеси налаштування системи зборки та підключення бібліотек [13].

Запуск коду JavaScript в якості окремого додатку дозволяє Node.js — середовище виконання JavaScript. JavaScript виконує дію на стороні клієнта, а

Node.js - на сервері. За допомогою Node можна писати повноцінні програми. Node.js вміє працювати з зовнішніми бібліотеками, викликати команди з коду на JavaScript і виконувати роль веб-сервера.

Структура WEB-сайту створена за допомогою HTML5 — це стандартизована мова гіпертекстової розмітки документів Всесвітньої павутини. Більшість WEB-сторінок написані саме на HTML5. Він інтерпретується браузерами та отриманий результат у вигляді відформатованого тексту відображається на моніторі комп'ютеру та екрані мобільного телефону.

Зовнішній вигляд HTML-документу описує CSS3 (Cascading Style Sheets, каскадні таблиці стилів). Це одна з базових технологій в сучасному інтернеті. Практично жоден сайт не обходиться без CSS, тому HTML і CSS діють в єдиній зв'язці. Каскадні таблиці стилів працюють з HTML, але це зовсім інша мова. За допомогою CSS красиво оформлюються створені елементи, тобто прописуємо унікальні властивості цих HTML-елементів.

Всі перечислені вище інструменти треба якось використовувати. Тому весь код був написаний на редакторі з функціями IDE Visual Studio Code — редактор вихідного коду, розроблений Microsoft для Windows, Linux і macOS. Позиціонується як «легкий» редактор коду для кроссплатформенної розробки веб-і хмарних додатків. Включає в себе відладчик, інструменти для роботи з Git, підсвічування синтаксису. Має широкі можливості для кастомізації: призначені для користувача теми, поєднання клавіш і файли конфігурації. Розповсюджується безкоштовно[14].

2.2. Логічна та фізична модель фрагменту бази даних системи

Для створення логічної та фізичної моделі бази даних системи, в даній роботі був використаний CASE-засіб ERWin[15]. Логічна та фізична моделі фрагменту БД відповідають структурі даних розробленого WEB-сайту кафедри «ЕБОП» в Firebase Realtime Database.

Логічна або концептуальна модель бази даних — це деяка наочна діаграма, створена за допомогою прийнятих позначенню, яка детально

показує зв'язок між об'єктами та їх характеристиками. Створюється концептуальна модель для подальшого проектування бази даних та переведення її, наприклад, в реляційну базу даних. На концептуальній моделі в візуально зручному вигляді прописуються зв'язки між об'єктами даних та їх характеристиками. В даній кваліфікаційній роботі дана модель була створена за допомогою CASE-засібу ERwin.

Логічна та фізична моделі показані у *додатку В, рис. В1-В2*.

Для однаковості програмування баз даних введені такі поняття для концептуальних баз даних:

- об'єкт або сутність — це фактична річ або об'єкт (для людей), за якою користувач (замовник) хоче спостерігати;
- атрибут — це характеристика об'єкта, що відповідає його суті;
- зв'язок або відношення між об'єктами.

Лексично більш правильно говорити зв'язок між об'єктами КБД і відносини між сутностями КБД (концептуальна база даних), але зустріти можна найрізноманітніші поєднання суті, об'єкти, зв'язки і відносини (огріхи перекладів).

Фізична модель даних описує реалізацію об'єктів логічної моделі на рівні об'єктів конкретної бази даних. Вона будується на основі логічної і представлена у графічних матеріалах (Додаток Б).

Фізична модель сайту кафедри «ЕБОП» складається з 13 таблиць.

Таблиця «User» (Користувач) — містить необхідні дані про користувача та складається із таких колонок:

user_id: INTEGER;

email:VARCHAR(20);

password: VARCHAR(20);

isBlocked: BIT;

role_id: INTEGER (FK).

Таблиця «News» (Новини) — включає в себе дані, які стосуються новин та складається із таких колонок:

news_id: INTEGER;

title: VARCHAR(20);

image: CHAR(18);

user_id: INTEGER (FK);

category_id: NUMERIC (FK);

isVisible: BIT;

created: DATETIME.

Таблиця «news_category» (Категорії новин) — містить в собі назви категорій та складається із таких колонок:

category_id: NUMERIC;

name: VARCHAR(20).

Таблиця «Page» (Сторінки) — включає в себе дані, які стосуються сторінок сайту та складається із таких колонок:

page_id: INTEGER;

title: VARCHAR(20);

isVisible: BIT;

created: DATETIME;

user_id: INTEGER (FK);

category_id: NUMERIC (FK).

Таблиця «page_category» (Категорії сторінок) – включає в себе данікатегорії сторінок та складається із таких колонок:

category_id: INTEGER;

name: VARCHAR(20).

Таблиця «Gallery» (Галерея) – включає в себе дані, які стосуються галереї та складається із таких колонок:

image_id: INTEGER;

image_url: TEXT;

description: VARCHAR(20);

category_id: NUMERIC (FK);

user_id: INTEGER (FK);

Таблиця «gallery_category» (Категорія галереї) – включає в себе дані категорії галереї та складається із таких колонок:

category_id: INTEGER;

name: VARCHAR(20).

Таблиця «user_role» (Роль користувача) – включає в себе данірro роль користувача у системі та складається із таких колонок:

category_id: INTEGER;

name: VARCHAR(20).

Таблиця «Vacancy» (Вакансія) – включає в себе дані, які стосуються вакансії та складається із таких колонок:

vacancy_id: INTEGER;

title: VARCHAR(20);

content: TEXT;

created: DATETIME;

image_url: TEXT;

contact_email: VARCHAR(20);

contact_phone: VARCHAR(20);

company_id: INTEGER (FK);

category_id: INTEGER (FK);

user_id: INTEGER (FK).

Таблиця «Company» (Компанія) – включає в себе дані, які стосуються компанії та складається із таких колонок:

company_id: INTEGER;

name: VARCHAR(20);

image_url: TEXT;

description: TEXT;

contact_email: TEXT;

contact_phone: TEXT;

adress: VARCHAR(20);

user_id: INTEGER (FK).

Таблиця «vacancy_category» (Категорії вакансій) – включає в себе категорії вакансій та складається із таких колонок:

category_id: INTEGER;

name: VARCHAR(20).

Таблиця «Event» (Події) – включає в себе дані, які стосуються подій на кафедрі та складається із таких колонок:

event_id: CHAR(18);

title: VARCHAR(20);

description: TEXT;

image_url: TEXT;

location: VARCHAR(20);

start_datetime: DATETIME;

end_datetime: DATETIME;

category_id: INTEGER (FK);

user_id: INTEGER (FK).

Таблиця «event_category» (Категорії подій) – включає в себе категорії подій та складається із таких колонок:

category_id: INTEGER;

name: VARCHAR(20).

Тепер на основі отриманої фізичної моделі генеруємо базу даних у Firebase Realtime Database (використавши спеціальний плагін для генерації у правильному для цієї бази даних — форматі JSON), перед цим створивши порожню базу даних.

Firebase надає в режимі реального часу базу даних як службу. Ця служба надає розробникам застосунки API, який дозволяє синхронізувати дані цих застосунків між клієнтами та зберігати їх у хмарі Firebase. Розробники, які використовують Realtime Database, можуть захищати свої дані за допомогою правил безпеки, що застосовуються на сервері.

Оскільки наша база даних представляє собою JSON структуру і таблиця може мати мати будь-яку структуру, то нам необхідно зробити спеціальні JavaScript класи які будуть мати правильну структуру та валідувати дані. Для цього можно використати спецільний інструмент, який конвертує SQL код в JavaScript клас.

Рис. 1. Приклад генерації JavaScript класу на основі SQL коду створення таблиці

Як можна побачити з рис. 1, отриманий клас представляє собою структуру таблиці, яка буде записана до БД та використана для відображення записаної у цю таблицю інформації.

Всі дані в Firebase Realtime Database зберігаються як об'єкти JSON. Базу даних можна вважати як хмарне дерево JSON. На відміну від бази даних SQL, тут немає ні таблиць, ні записів. Коли додаються дані до дерева JSON, він стає вузлом у існуючій структурі JSON із пов'язаним ключем. Можна надати власні ключі, такі як ідентифікатори користувачів або семантичні імена, або вони можуть бути надані користувачу за допомогою push ()[16].

2.3. Створення інтерфейсу користувача

Веб-інтерфейс має за останні роки бажано має виконувати вимоги, які зазначені нижче:

1. Мінімалізм і простота — і сайти, і додатки стають складніше всередині і простіше зовні, цього вчать всі передові гравці і найбільше — Гугл. Користувач повинен зробити мінімум дій для отримання бажаного результату.

2. Інтерактивність і асинхронність — інтерфейси стають динамічними, пропадає перезавантаження сторінок і зміна екранів, підгрузка відбувається поступово, фрагментами. Сторінка поступово модифікує екран, відгукуючись на дії користувача.

3. Контекстність — виведення інформації та контроли для виклику операцій з'являються там, де це логічно очікується і показуються тільки коли це необхідно — треба економити екран та увагу користувача.

4. Синхронізація та комміт (збереження/фіксація змін в програмному коді) — все частіше з'являються додатки, в яких сервер генерує події за своєю ініціативою, це дозволяє синхронізувати екран користувача з поточним станом даних в БД або в пам'яті сервера.

5. Спрощений мобільні інтерфейс — з поширенням мобільних пристроїв, забезпечених нормальними браузерами, з'явилася необхідність окремо розробляти інтерфейси для малих дозволів із підтримкою тачскрін.

6. Підтримка стандартів — входить в моду вирішувати завдання із застосуванням нових можливостей і специфікацій, але з поверненням до старих технологій, наприклад, звук і відео вже хочеться програвати через html5, але флеш нас страхує, або при відсутності Local Storage ми зберігаємо стан на сервері, просто буде більше запитів, візуалізація так само спрощується при показі в старому браузері, але додаток продовжує працювати, а виглядає простіше.

2.3.1. Структура файлу та робота з Vue.js

За допомогою технології роботи прогресивного фреймворку Vue.js можна забезпечити просте створення макету з можливістю використовувати його на інших сторінках без використання одного й того самого коду та

швидке оновлення даних на сайті. Для цього скористаємось однією з особливостей Vue.js — компонентами. Файл з розширенням *.vue має свою структуру, яка складається з парних тегів <template>, <script> та <style>.

<template>

• Мова за замовчуванням: HTML.

• Кожен * .vue файл може містити максимум один блок <template>.

• Вміст буде вилучено як рядок і використано як опція template для скомпільованого компонента Vue.

<script>

• Мова за замовчуванням: JavaScript.

• Кожен * .vue файл може містити максимум один блок <script>.

• Скрипт виконується в оточенні CommonJS (як звичайний .js модуль, зібраний за допомогою webpack), що дозволяє вам використовувати require () для підключення інших залежностей. А разом з підтримкою ES2015, можна використовувати також синтаксис import і export.

• Скрипт повинен експортувати об'єкт з настройками компонента Vue.js. Експорт розширеного конструктора, створеного через Vue.extend (), також підтримується, але просто об'єкт краще.

<style>

• Мова за замовчуванням: CSS.

• Може бути кілька тегів <style> в одному * .vue файлі.

• Тег <style> може мати scoped або module атрибути для можливості забезпечити інкапсуляцію стилів в поточному компоненті. Кілька тегів <style> з різними режимами інкапсуляції можуть бути використані в одному компоненті.

• За замовчуванням, вміст буде вилучено і динамічно вставлено в тег <head> документа звичайним тегом <style> за допомогою style-loader. Також можливо налаштувати webpack щоб витягувати стилі всіх компонентів в один CSS-файл.

Головний компонент сайту називається App.vue та має досить компактний код (рис. 2).

App.vue

src > App.vue > {} "App.vue"

You, 8 days ago | 2 authors (Viza and others)

```
<template>
  <div id="app">
    <!-- <keep-alive> -->
      <router-view></router-view>
    <!-- </keep-alive>  -->

  </div>
</template>

<script>
  export default {
    name: 'App',
    components: {}
  }
</script>

<style lang="scss">

</style>
```

Рис. 2. Приклад коду компонента App.vue

За допомогою маршрутизатора <router-view> можна переходити по сторінкам сайту. Це впроваджено за допомогою задання певним маршрутам того компоненту, на який треба перейти у файлі router.js (*додаток Г, 7Г*). Для використання компоненів треба їх імпортувати у необхідний файл, задати при цьому ім’я та потім об’явити.

2.3.2. Створення елементів макету сайту

Макет сторінки сайту «ЕБОП» складається з таких елементів:

- інформаційний хедер;
- головний хедер;
- меню
- місце для контенту;
- футер.

Всі ці елементи так чи інакше повторюються на інших сторінках сайту та кожен з них є окремим компонентом.

Інформаційний хедер

Для написання інформаційного хедеру спочатку був створений файл ContactHeader.vue. Далі була створена структура компоненту, яка була описана у п.2.3.1.

У тезі <template> має бути один батьківський блок, тому був створений блок з класом «contacts». У цьому клас прописані стилі для забезпечення збільшення блоку при наводі курсору на нього. Потім створили ще один блок, який є класом-контейнером сайту (class="kektainer"). Його стилі розміщують контейнер по центру екрану користувача та визначає ширину на якій буде показано контент.

Стилі класу «kektainer» є глобальними (рис. 3), тобто були прописані у головному компоненті без ізоляції, тому вони можуть встановлюватись у інших компонентах. Але по дизайну необхідно, щоб його дочірні елементи розміщувались по центру вертикальної лінії, тому цей стиль був перевизначений у компоненті ContactHeader.vue. Через значення scoped, який був зазначений тезі <script scoped> було задано правки до глобального стилю kektainer, які будуть працювати лише у цьому компоненті(рис. 4). Зазначимо, що стилі при перевизначенні не змінюються повністю, а лише нові додаються до існуючих.

```
.kektainer {
  width: 1100px;
  margin: 0 auto;
}
```

Рис. 3. Глобальні стилі класу «kektainer»

```
.kektainer {
    display: flex;
    height: 100%;
    align-items: center;
}
```

Рис. 4. Перевизначені стилі класу «kektainer»

Далі створили два окремі блоки, які відповідають за поле пошуку та контактну інформацію відповідно. У блоці пошуку був встановлений клас «search». У блоці контактної інформації — клас «social». Всі стилі компоненту ContactHeader.vue можна побачити у *додатку Д, 1Д*.

Блок з класом «search» містить в собі іконка лупи та поле input. Поле має клас «search-input» та воно не показується користувачу. Іконка — клас «el-icon-search». У класі іконки було прописані стилі, які дозволяють при наводі курсором на іконку показати поле вводу input.

Блок з класом «social» має два ссилкові теги, які містять в собі картинки-іконки соціальних сторінок (інстаграм, фейсбук). При натисканні на ці картинки-іконки відкривається нова вкладка із силкою, яка прописана у атрибуті href. Для даних картинок були прописані стулі, які дозволили картинкам трохи збільшувати розмір при наводі на них курсору.

Головний хедер

Загальний порядок створення головного хедеру:

1. Створення компоненту Header.vue.
2. Розробка структури хедеру.
3. Створення блоків у готовій структурі.
4. Написання стилів до кожного блоку.

Інші елементи, які зазначені вище, створюються за подібною схемою. Але при цьому у кожного з них є свої особливості оформлення структури та стилів.

Код створених зазначених елементів показаний у *додатку Д*.

2.3.3. Підключення бази даних до сайту

При створенні сайту з використанням JavaScript можна швидко створити підключення до бази даних. Для цього JavaScript реалізувала функцію fetch(). Дана функція у нашому випадку приймає силку на дані у форматі JSON та записує їх у змінну. Так як дані не одразу отримуються у JSON, а лише

оболонка з параметрами, то необхідно їх відформатувати для подальшого використання.

Майже на кожній сторінці веб-сайту отримуються дані з БД. Для прикладу реалізації отримання даних показано код компоненту Article.vue (рис. 5).

```
const resp = await fetch(`https://us-central1-nuft-kebop.cloudfunctions.net/news?startAt=
                    ${startAt}&itemsOnPage=${itemsOnPage}&q=${q}`);
const result = await resp.json();
```

Рис. 5. Отримання даних з БД у компоненті Article.vue

Всі отримані дані в подальшому проходят переініціалізацію за допомогою створеного методу у кожному компоненті окремо. Це зроблено для полегшення роботи з даними (рис. 6).

```
transformNews(news) {
    return _.map(news, item => ({
        photo: !_.isUndefined(item.photo) ? item.photo.url : 'https://www.ticketpro.by/storage/img/no-image.png',
        title: item.title,                    any
        category: !_.isUndefined(item.category.title) ? item.category.title : 'Без категорії',
        description: item.description,
        id: item.id,
        createDate: moment(item.created).format("YYYY-MM-DD HH:mm")
    }))
```

Рис. 6. Переініціалізація даних у компоненті Article.vue

Також можна побачити, що при неіснуванні отриманого фото відбувається підстановка через тернарний оператор. Так як дані представлені масивом об’єктів, то зроблено перебір об’єктів для переініціалізації кожної властивості за допомогою методу map(arr, item).

2.3.4. Структура, фільтрація та функції елементу новин

Реалізація елементу новин представлена у компоненті Articles.vue, який використовується на саті найчастіше у «контенті сторінки». Переглянути повний код компоненту можна у додатку Д, 5Д.

Articles.vue має два дочірніх компоненти: DateCategory.vue та NewsCategory.vue. У DateCategory.vue визначаються дата створення новини та вибрана категорія. У NewsCategory.vue — самі категорії, при виборі яких

відображаються новини за відповідною тематикою. Підключення можна побачити на рис. 7-8.

```
import DateCategory from './DateCategory.vue'
import NewsCategory from './NewsCategory.vue'
```

Рис. 7. Імпорт дочірніх компонентів у батьківський Articles.vue

```
components: {
    DateCategory,
    NewsCategory
},
```

Рис. 8. Ініціалізація дочірніх компонентів

Переініціалізовані дані, що показані на рис. 6, використовуються у коді <template> компоненту як змінні. Для їх використання у коді прописуються подвійні скобки: {{ назваЗмінної }}. Або ж, якщо вони будуть знаходітись у значенні атрибуту, то перед цим атрибутом необхідно поставити двокрапку (скорочення від v-bind) для того, йоб зв’язати його з Vue.js (:атрибут = «назваЗмінної»).

Для забезпечення отримання певної кількості новин реалізована функція loadNews({start = 0, itemsOnPage = 6, q = ``}) (рис. 11). Атрибути були ініціалізовані за доромогою вкладенної деструктуризації, яка дозволяє присвоїти значення по дефолту:

- start — це номер елементи масиву, з якого необхідно почати вигрузку даних;
- itemsOnPage — кількість елементів, яку необхідно вигрузити;
- q — фраза для пошуку по назві елементу.

Така функція, або подібна до неї, реалізована для всіх компонентів, які отпримують інформацію з БД.

```
    async loadNews({startAt = 0, itemsOnPage = 6, q = ''}) {
        this.loading = true;
        const resp = await fetch(`https://us-central1-nuft-kebop.cloudfunctions.net/news?startAt=
                                  ${startAt}&itemsOnPage=${itemsOnPage}&q=${q}`);
        const result = await resp.json();
        this.loading = false;
        return this.transformNews(result.data);
    }
},

async mounted() {
    this.articles = await this.loadNews({q: this.search, itemsOnPage: this.itemsOnPageImp});
},
```

Рис. 9. Приклад функції для отримання певної кількості елементів

Для отримання даних для пошуку новин було встановлено input-елемент (рис. 10). Для динамічного отрамання даних у Vue.js є атрибут v-model. Змінна search, яка була записана у цей атрибут, динамічно отримує дані з інпута та передає у метод, який при зміні певного елементу (у нашому випадку — змінна search) автоматично виконується знову (рис. 11).

```
<div>
    <el-input placeholder="Пошук новин" v-model="search"></el-input>
</div>
```

Рис. 10. Приклад елементу input у компоненті Articles.vue

```
watch: {
    async search(searchValue) {
        this.articles = await this.loadNews({q: searchValue});
    }
}
```

Рис. 11. Watch-метод для перезапуску функції loadNews()

Також, якщо новин за пошуком було не знайдено, то про це буде повідомлено (рис. 12).

```
</div>
<div v-if="!loading && !articles.length"><h3>Нічого не знайдено</h3></div>
iv>
```

Рис. 12. Реалізація виключення при пошуку у компоненті Articles.vue

Тобто, якщо відбулася загрузка новин, то булева змінна loading набуває значення false (від самого початку визначене значення true). Також якщо масив не матиме елементів взагалі, то також буде показано повідомлення «Нічого не знайдено».

Є можливість переглянути новини за їх категорією. Порядок виконання коду при виборі певної театики:

1. У компоненті NewsCategory кастомний select-елемент передає значення до методу, який вже відправляє дані до батьківського компоненту.

2. За допомогою функції selectedCategory(option) змінна проходить по кожному об’єкту масиву та при рівності записується до масиву selectByCategory (рис. 13).

```
selectedCategory(option) {
    this.selected = option.categoryType;
    this.selectByCategory = [];
    let vm = this;
    this.articles.map((item) => {
        if(item.category === option.categoryType) {
            vm.selectByCategory.push(item)
        }
    });
    if (this.selected === 'all_categories') {
        this.selectByCategory = this.articles;
    }
}
```

Рис. 13. Функція selectedCategory(option) у компоненті Articles.vue

3. Після цього через зміну масиву selectByCategory запуститься фільтр filterArticles() (тому що він знаходиться у елементі computed), який виведе цей масив на сторінці користувача (якщо масив пустий, то повернеться масив article)(рис. 14).

```
computed: {
    filterArticles() {
        if(this.selectByCategory.length) {
            return this.selectByCategory
        }
        else {
            return this.articles
        }
    }
},
```

Рис. 14. Фільтр filterArticles() компоненті Articles.vue

2.3.5. Автоматична підгрузка елементів сторінки

Дана можливість буде показана на прикладі сторінки «Фотогалерея». Для цього було трохи видозмінено функцію loadPhotos({startAt = 0, itemsOnPage = this.itemsPerPage}) (рис. 15). Початкове значення itemsPerPage = 12, тобто спочатку на сторінці буде показано лише 12 фотографій. При прокрутці сторінки до кінця відбудеться підгрузка ще 12 фотографій (або стількох, скільки залишилось у масиві).

```
async loadPhotos({startAt = 0, itemsOnPage = this.itemsPerPage}) {
    this.loading = true;

    await new Promise(resolve => setTimeout(() => {resolve()}, 2000))

    const resp = await fetch(`https://us-central1-nuft-kebop.cloudfunctions.net/gallery
                    ?start=${startAt}&itemOnPage=${itemsOnPage}`);

    const result = await resp.json();
    this.loading = false;

    if(!result.success)
        this.isEndReached = true;
    else
        this.isEndReached = false;

    return this.transformPhotos(result.data);
},
```

Рис. 15. Функція loadPhotos() у компоненті GaleryPhoto.vue

Для правильної підгрузки фотографій було створено ще одну змінну start з початковим значенням 0 (start = 0). При досягненні кінця сторінки отримуєм дані з <template>, а саме з рядків:

* *
<intersect @enter="load"> </intersect>

Після цього передаються дані у функцію load(), яка предсталена на рис. 16. Спочатку відбувається перевірка чи досягнуто кінця сторінки. Далі змінна start перезаписується додаванням itemsPerPage (якщо спочатку фотографії загружались із позиції 0, то після визову функції будуть загружатись фото з

позиції 12). Після знову визивається функція loadPhotos() для подальшої підгрузки елементів.

```
async load() {
    if(this.isEndReached) return;

    this.start += this.itemsPerPage;
    const data = await this.loadPhotos({startAt: this.start});
    this.photos = [...this.photos, ...data];
},
```

Рис. 16. Функція load() у компоненті GaleryPhoto.vue

Повний код компоненту GaleryPhoto.vue можна побачити у *додатку Д, 6Д.*

2.3.6. Адаптивність сайту

Для створення адаптивного дизайну сайту «ЕБОП» були використані медіа запити.

Медіа запити (media queries) — це правила CSS, які дозволяють управляти стилями елементів в залежності від значень технічних параметрів пристроїв. Іншими словами, це конструкції, які дозволяють визначати на підставі деяких умов які стилі необхідно використовувати на веб-сторінці, а які ні.

Для створення медіа запиту використовувались позначення @media. Далі прописувались умови або функції, при яких змінюються стилі, а вже потім у тілі змінювались самі стилі елементів. При цьому назви класів, які використовувались раніше, не змінювались.

Використовувались умови, які впливають на ширину екрану як комп’ютера, так і мобільного девайсу. Також окремо прописувались стилі для екранів гаджетів (планшетів, телефонів), які перебувають у горизонтальному режимі (landscape).

2.3.7. Пошукова оптимізація сайту

Пошукова оптимізація сайту — це комплекс заходів спрямованих на підняття позицій сайту в результатах видачі пошукових систем по певних запитах.

Для хорошої пошукової оптимізації всі мета-теги, теги заголовків <h1>-<h6>, атрибут alt у тегу <img> (альтернативна назва), <title> та <description> були заповнені. На кожній сторінці у <title> міститься її назва, наприклад «Головна сторінка», або «Новини». У <description> встановлюється опис поточної сторінки, на якій перебуває користувач. Ці два теги є важливими, адже саме вони видаються пошуковою машиною (Google, Opera і т.д.) в якості короткого опису сторінки.

Був доданий файл robots.txt — даний файл-помічник пошукових систем повідомляє, які сторінки не потрібно індексувати (копіювати) пошуковим системам. Наприклад, сторінки зворотного зв'язку, фільтрації або пошуку на сайті. Загалом, в даному файлі необхідно закрити для пошуку ті сторінки, які не несуть корисної і потрібної інформації для цільового відвідувача.

Також для гарної пошукової оптимізації було розроблено зручний інтерфейс та забезпечено сайт швидкою загрузкою даних.

Для майбутньої підтримки пошукової оптимізації необхідно на сторінках сайту використовувати лише унікальний текст, адже чим вища унікальність — тим вище місте, на якому знаходиться сайт після пошукових запитів користувачів у пошукових машинах

2.4. Інтерфейс користувача

У сучасному світі WEB-сайт повинен мати інтуїтивний та зрозумілий для користувача дизайн.

Уявіть, що ви заходите на сайт і без проблем знаходите потрібну інформацію і швидко оформлюєте замовлення. Веб-ресурс повністю для вас зрозумілий, незважаючи на те, що ви відвідуєте його вперше. Такий сайт має

інтуїтивний веб-дизайн, що забезпечує відвідувачів можливістю легкого досягнення мети за короткий час. При його створенні враховується досвід користувачів і особливості сприйняття інформації.

Інтуїтивний веб-дизайн непомітний для користувачів. Зате його відсутність відразу кидається в очі. Якщо відвідувачі сайту можуть без проблем вирішувати поставлені перед ними завдання, якщо вони витрачають мінімум часу на пошук відповідей, якщо їм зручно і легко користуватися всіма функціями — значить сайт відповідає вимогам людей і досягає поставлених цілей. Інтуїтивний веб-дизайн підвищує ефективність просування і розкрутки сайту. Кожен успішний сайт повинен мати саме таке оформлення.

Інтуїтивна навігація забезпечує швидкими та зрозумілими переходами по сайту, без додаткових пошуків відповідей.

На рис. 17 можна побачити три відділи: невеличкий блок з ссилками на соціальні мережі та пошук по сайту, меню та слайдер. Спочатку блок контактів має менші розміри, та збільшується при наводі на нього курсора — це зроблено задля привернення уваги користувача. Також збільшуються при наводі іконки соціальних мереж.

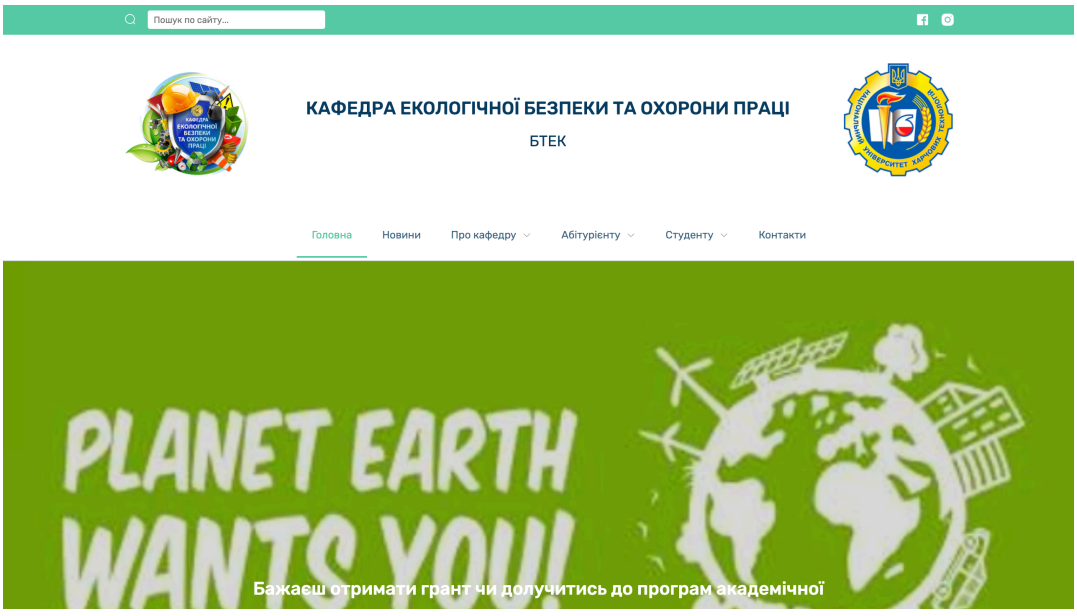

Рис. 17. Головна сторінка веб-сайту

Навігацію між сторінками забезпечує меню (рис. 18), яке може мати подвійну структуру. При скролінгу меню залишається зверху сторінки, тому доступ до нього є постійним.

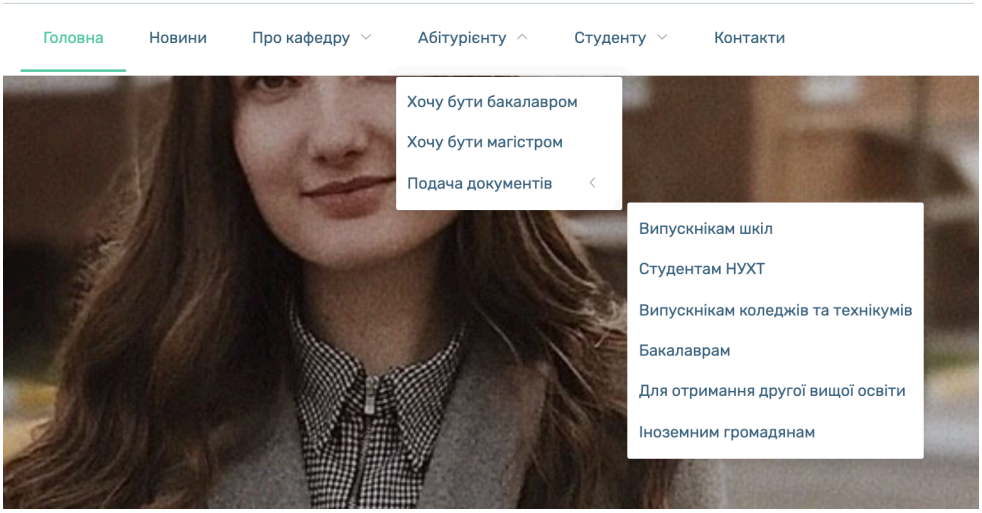

Рис. 18. Головне меню веб-сайту

Слайдер (рис. 19) містить в собі інформаційні картки, у яких буде зазначена важлива інформація або новини для користувачів. Є змога перейти на сторінку за кнопкою «Дізнатись більше» задля отримання детальної інформації. Для навигації між картками треба клацнути на одну із стрілочок, які знаходяться вздовж середини слайду зліва (назад) та справа (вперед). Або можна просто навести курсор на одну із полосок знизу і відповідний слайд одразу з’явиться.

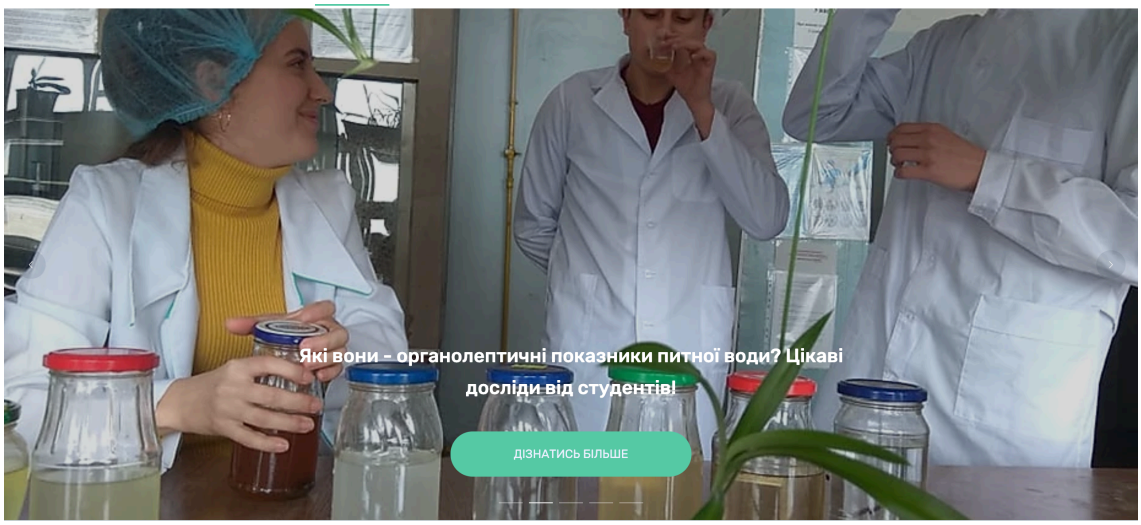

Рис. 19. Слайдер веб-сайту

Якщо з’явиться потреба відвідати соціальні мережі кафедри «ЕБОП», то достатньо натиснути на кнопку потрібної вам мережі на контактному хедері (рис. 17) або у футері (*додаток Г, рис. Г1*). Також у футері розміщена сама затребувана інформація, а саме адреса знаходження кафедри, електронна пошта та інша контактна інформація.

На головній сторінці є поле з трьома найактуальнішими новинами (рис. 20). Для переходу до окремої з них необхідно просто навести на неї, що супроводиться змільшенням картинки та потім клацнути. Якщо треба переглянути більше новин, то достатньо нажати на кнопку «Побачити більше».

На кожній новині під її назвою також зазначений час створення та назву категорію, за якою можна здійснювати фільтрацію на сторінці «Новини». Для пошуку за категорією достатньо натиснути на поле «Вибрати категорію» та вибрати одну із запропонованих у списку. Після вибору висвітлюються лише ті новини, категорію яких було вибрано. Приклад виконання вибору новин за певною тематикою можна переглянути у *додатку Г, рис. Г7*.

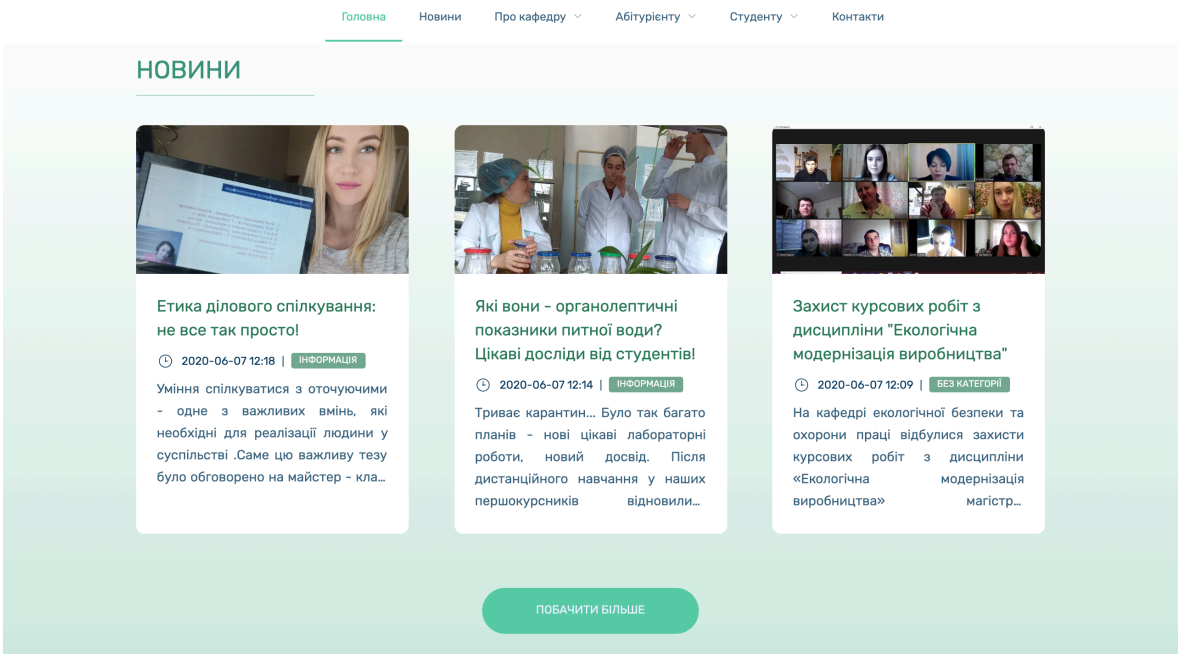

Рис. 20. Блок новин на головній сторінці

Також присутня моливість пошуку новини за її назвою. Для цього достатньо натиснути на поле «Пошук…», який знаходиться у контактному

хедері (рис. 17) або біля блоку новин. Далі просто треба ввести любу фразу, яку забажаєте, та отримати результат пошуку. Приклад пошуку можна побачити у додатку Г, рис. Г6. Якщо результатів по введеній фразі не буде, то з’явиться напис «Нічого не знайдено».

Якщо з’явиться необхідність переглянути фотографії, адже саме фотознімки відображають справжнє життя кафедри «ЕБОП», то для цього є окремий підпункт меню «Фотогалерея», який знаходиться у пункті «Студенту». Після переходу на сторінку з’являться фотографії, на які можна просто навести для отримання якогось уточнення, наприклад, що за подія сталась в тот день, чи при яких умовах зроблене фото (додаток Г, рис. Г8). Для перегляду фотографії у збільшеному розмірі достатньо просто клікнути на неї. Результат можна побачити у *додатку Г, рис. Г9*.

До фотогалереї також можна перейти і з головної сторінки веб-сайту. Для цього на сторінці є слайдер, у якому розміщені актуальні фотографії кафедри. Цей слайдер знаходить знизу сторінки, перед футером. Вигляд слайдеру фотографій можна переглянути у додатку Г, рис. Г

Більше прикладів інтерфейсу можна побачити у *додатку , рис. Г2*.

2.5. Розрахунок техніко-економічного ефекту

1. Визначення розміру оплати праці.

1.1. Вид системи*: інформаційний сайт кафедри*

1.2. Стадії:

- ескізний проект;
- ТЗ, технічний проект;
- робочий проект;
- впровадження.

1.3. Ступінь новизни розроблюваної задачі: В

1.4. Група складності алгоритму за їх характеристикою — 2

1.5. Вид інформації (табл. 3)

Таблиця 3. Дані про вхідну та вихідну інформацію сайту «ЕБОП»

Вид інформації	Позначення	К-сть наборів даних
Кількість видів змінної інформації	ЗІ	M=7
Кількість видів нормативно-довідкової інформації	НДІ	N=3
Кількість баз даних	БД	P=1
Обробка в режимі реального часу	РЧ	Так
Забезпечення телекомунікаційної обробки даних і управління віддаленими об’єктами	ТОУ	Ні

1.6. Визначення витрат часу:

Таблиця 4. Визначення витрат часу сайту кафедри «ЕБОП»

Вид системи	Стадія розробки системи	
Управління відвідуванням	Ескізний проект T_1	Технічне завдання T_2
	В	В
	T_1=70	T_2=43

1.7 Визначення витрат часу для стадії «Технічний проект» (T_3)

Дані для визначення:

- Кількість форм вхідної інформації: $B_1 = 2$
- Кількість форм вихідної інформації $B_2 = 5$
- Кількість баз даних $B_3 = 1$
- Базове значення витрат часу для стадії «технічний проект» $T_{Б3} = 72$
- Базове значення витрат часу для стадії «робочий проект» $T_{Б4} = 221$
- Базове значення витрат часу для стадії «впровадження» $T_{Б5} = 73$

1.7.1 Розрахунок витрат часу для стадії «Технічний проект» (T_3)

$$К_п = (1.0*2 + 0.72*5 + 2.08*1) / 8 = 0,96$$

Таблиця 5. Коефіцієнти $К_1$, $К_2$, $К_3$ для стадії «Технічний проект»

Вид використанної інформації	Ступінь новизни
	В
$К_1$ (ЗІ)	1.0
$К_2$ (НДІ)	0.72
$К_3$ (БД)	2.08

Таблиця 6. Коефіцієнт ступеню новизни проекту k0

Стадія розробки системи	Вид оброки	Ступінь новизни
		В
Технічний проект	РЧ	1.26
Робочий проект	РЧ	1.32
Впровадження	РЧ	1.21

$$Т_3 = Т_{Б3} * К_П * К_0 = 72 * 0,96 * 1.26 = 54,5076$$

1.7.2 Розрахунок витрат часу для стадії «Робочий проект» ($Т_4$)

$$К_П = (1.1*2 + 0.58*5 + 0.48*1) / 8 = 0,697$$

Таблиця 7. Коефіцієнти k_1, k_2, k_3 для стадії “Робочий проект ”

Вид використаної інформації	Група складності алгоритму	Ступінь новизни
		В
k_1 (ЗІ)	2	1.1
k_2 (НДІ)	2	0.58
k_3 (БД)	2	0.48

$$К_С = 1, 16$$

$$Т_4 = Т_{Б4} * К_П * К_0 * К_С = 221 * 0,697 * 1,32 *1,16 = 236,031$$

1.7.3 Розрахунок витрат часу для стадії «Впровадження» ($Т_5$)

Поправочні коефіцієнти мають такі ж значення, як і при обрахунку $Т_4$, тому:

$$T_5 = T_{Б5} * К_П * К_0 * К_С = 73* 0,697 * 1, 21 * 1,16 = 71,468$$

1.8 Визначення загальних витрат часу на розробку системи

$$T = T_1 + T_2 + T_3 + T_4 + T_5 = 507.59$$

1.9 Визначення чисельності виконавців

$$Ф = 75 : М = 3$$

$$Ч = Т / Ф = 507.59 / 75 = 6.768$$

1.10 Оплата праці виконавців

$$V_1 = Ч*М*ЗП = 101517,931$$

2. Витрати, пов’язані з розробкою програми на ПК

2.1 Розрахунок річного фонду часу роботи ПК в годинах

$$Т_{пк} = 2000 - (6 * 8 + 5 * 12) = 1892$$

$$Т`_{пк} = 1892 * (450 / 2000) = 425.7$$

2.2. Поточні витрати на експлуатацію $V"_1$

$$V"_1 = З_{ОП} + З_{АМ} + З_{ЕЛ} + З_{МАТ}$$

2.2.1 Амортизаційні відрахування:

Балансова вартість ПК:

$$Ц_{пк} = 10000 * (1 + 0.12) = 11200 (грн)$$

$$H_A=5$$

$$З_{ам}=11200 / 5 = 2240 (грн)$$

2.2.2 Витрати на електроенергію *:*

Потужність ПК: $Р_{пк} = 0.25$ кВт

Фонд корисного часу роботи ПК: $Т'_{пк} = 425.7$ год

Вартість 1 кВт електроенергії для підприємств: $Ц_{ел} = 1.2$ грн\кВт

Коефіцієнт інтенсивного використання ПК: А=0.9

$$Зел = Р_{пк} * Т'_{пк} * Ц_{ел} * А = 0.25 * 425.7 * 1.2 * 0.9 = 114.939 (грн)$$

2.2.3 Витрати на поточний ремонт і технічне обслуговування ПК:

$$Зр = Ц_{пк} * 0.06 = 11200 * 0.06 = 672 \text{ (грн)}$$

2.2.4 Непрямі витрати, пов'язані з експлуатацією ПК:

$$З_{мат} = Ц_{пк} * 0.05 = 11200 * 0.05 = 560 \text{ грн}$$

Заробітна плата обслуговуючого персоналу $З_{оп} = 3500$ *грн*

2.2.5 Поточні витрати на експлуатацію

$$V''_1 = 3500 + 2240 + 114.939 + 672 + 560 = 7086.939 \text{ (грн)}$$

2.3 Загальні витрати на розробку програмного забезпечення:

$$V_1 = V`_1 + V''_2 = 101517.931 + 7086.939 = 108604.87 \text{ (грн)}$$

3. Витрати на придбання і установку ПК

$$V_2 = Ц_{пк} = 11200 \text{ грн}$$

4. Витрати на підготовку приміщення і навчання персоналу

Приміщення підходить, тому $V_3 = 0$ грн

Навчання персоналу $V_4 = 1000$ грн

5. Загальна вартість розробки і впровадження системи

$$V = V_1 + V_2 + V_3 + V_4 = 108604.87 + 11200 + 0 + 1000 = 120804.87 \text{ (грн)}$$

Враховуючи норму амортизаційних втрат:

$$V_р = 120804.87 / 5 = 24160.974$$

6. Коефіцієнт економічної ефективності розробки:

Річний прибуток за рахунок збільшення кількості замовлень становить приблизно 202900.

$$К_{еф} = П_р / V_р = 202900 / 24160.974 = 8.4$$

Термін окупності системи:

$$Т_{ок} = 1 / К_{еф} = 1 / 8.4 = 0.12 \text{ (~2 місяці)}$$

РОЗДІЛ 3. ОХОРОНА ПРАЦІ

Всі гігієнічні вимоги до організації праці із застосуванням комп'ютерів умовно можна розділити на три групи:

1) вимоги до приміщень, де організована і здійснюється подібна діяльність з комп'ютером;

2) вимоги до обладнання комп'ютерного місця;

3) вимоги до організації праці або навчального процесу з використанням комп'ютерної техніки.

Для обробки інтер'єру приміщень, в яких експлуатуються комп'ютери, забороняється використовувати полімерні будівельні матеріали (деревостружкові плити, шаруватий паперовий пластик, синтетичні станів і килимові покриття та ін.), що виділяють у повітря шкідливі хімічні речовини.

Особливі вимоги пред'являються до мікроклімату приміщень, в яких користувачі працюють з комп'ютерами: оптимальна температура нормативно повинна становити від 18 до 25 ° С, а відносна вологість повітря — від 40 до 60% при його русі в 0,1-0,2 м / с.

Крім гігієнічних вимог до мікроклімату необхідно також забезпечити гранично допустимі рівні шуму комп'ютерної техніки на робочих місцях користувачів. Зокрема, згідно з ними у всіх приміщеннях рівень шуму на робочому місці не повинен перевищувати 50 ДБА, в офісних і виробничих приміщеннях, де робота з комп'ютером не є основною, 60 ДБА.

Сьогодні на перший план в якості джерел виникнення у користувачів ПК різного роду хронічних захворювань, які називають псевдо «комп'ютерними», виходять статичні навантаження на організм людини внаслідок малорухливого характеру роботи, а також розвиваються на цьому тлі гіпокінезія (нестача руху), гіподинамія (недолік фізичного навантаження) і гіповолемія (порушення перерозподілу крові).

Тому сучасні вимоги до організації режиму праці та відпочинку користувачів комп'ютерів покликані захистити людину не від машини, а від самого себе, від функціональних стереотипів поведінки, здатних стати причиною виникнення цілого комплексу захворювань.

10 найважливіших гігієнічних вимог при роботі з комп’ютером:

1. Покладіть комп'ютер або його монітор до вікна боком, щоб світло на нього падало зліва.

2. При організації та обладнанні робочого місця купуйте меблі відповідно до зростання користувача комп'ютера.

3. Щодня перед початком роботи обов'язково прибирайте пил на робочому місці.

4. Перед початком і після закінчення роботи, а також в обідню перерву проводьте провітрювання приміщення, де працює комп'ютер.

5. Щодня проводьте вологе прибирання в приміщенні, де працює комп'ютер.

6. При безперервній роботі з комп'ютером кожні 2 години робіть перерву на 15 хвилин для відпочинку і виконання комплексу фізкультурно-оздоровчих вправ.

8. При роботі з комп'ютером відстань від очей користувача до монітора повинна становити 600-700 мм, але не менше 500 мм.

9. Слідкуйте за поставою: спина повинна бути пряма, руки в ліктях повинні бути зігнуті під прямим кутом.

10. Регулярно проходьте профілактичний лікарський огляд.

ВИСНОВКИ

В даній кваліфікаційній роботі було розроблено сайт для кафедри екологічної безпеки та охорни праці НУХТ, який було виконано у середовищі Microsoft Visual Code та використанні HTML5, CSS3, JavaScript, Vue.js, Vue Cli, Node.js технології для написання інтерфейсу. Для написання логічної частини яка зв’язувала базу даних та інтерфейс був використаний сервій Firebase Functions, яки дозволив написаня фукнцій для маніпулювання даними з бази даних на мові JavaScript. Також був використаний CASE-засіб AllFusion-ErwinProcessModelerVer для створення логічної та фізийної моделі даних. Реалізована база даних у Firebase Realtime Database.

Створений сайт має інтуїтивний дизайн, надає можливість пошуку, вибору новин за категорією. Генерує автоматично та циклічно оформлює елементи з бази даних, які добавив адміністратор. Є динамічна підгрузка даних при скролінгу сторінки, що не перезавантажить сервер великою кількістю даних.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Кафедра екологічної безпеки та охорони праці — НУХТ — [Електронний ресурс]. — Режим доступу: https://nuft.edu.ua/fakultet-btek/kafedra-ebop/

2. Про нас — Кафедра екологічної безпеки та охорони праці [Електронний ресурс]. — Режим доступу: https://ecologynuft.wixsite.com/ecologynuft/pro-kafedru

3. Про затвердження Положення про вищі навчальні заклади МВС [Електронний ресурс]. — Режим доступу: https://zakon.rada.gov.ua/laws/show/z0193-08

4. Конструктор сайтов — Википедия — [Електронний ресурс]. — Режим доступу: https://ru.wikipedia.org/wiki/Конструктор_сайтов

5. Маклаков С.В. BPwin и ERwin. CASE-средства разработки информационных систем. - М.: ДИАЛОГ-МИФИ, 1999 — 256 с.

6. My WordPress [Електронний ресурс]. — Режим доступу: http://babulyapartner.ru/mywordpress/chto-takoe-wordpress/

7. TLTCam [Електронний ресурс]. — Режим доступу: https://tltcam.ru/rukovodstva/103-chto-takoe-joomla

8. CMS Magazine [Електронний ресурс]. — Режим доступу: https://cmsmagazine.ru/instrument/wix-cms/

9. Интерфейс «Эрмитаж» — Бітрікс 24 — [Електронний ресурс]. — Режим доступу: https://www.bitrix.ua/products/cms/features/hermitage.php

10. Огляд системи 1С:Підприємство 8 [Електронний ресурс]. — Режим доступу: http://1c.ua/ua/v8/

11. Композитний сайт [Електронний ресурс]. — Режим доступу: https://www.bitrix.ua/composite/

12. Introduction — Vue.js — [Електронний ресурс]. — Режим доступу: https://vuejs.org/v2/guide/

13. Overview — Vue CLI — [Електронний ресурс]. — Режим доступу: https://cli.vuejs.org/guide/

14. Visual Studio Code [Електронний ресурс]. — Режим доступу: https://ru.wikipedia.org/wiki/Visual_Studio_Code

15. М'якшило О. М. Моделювання баз даних засобами CASE-технології ERWin: Конспект лекцій з дисципліни «Структурне моделювання систем» для студ. спец. 6.080400 «Інформаційні управляючі системи та технології» напряму 0804 «Комп'ютерні науки» всіх форм навчання. – К.: НУХТ, 2008. – 60с

16. Structure Your Database — Firebase Realtime Database — [Електронний ресурс]. — Режим доступу: https://firebase.google.com/docs/database/web/structure-data

Додаток А. Структурні схеми

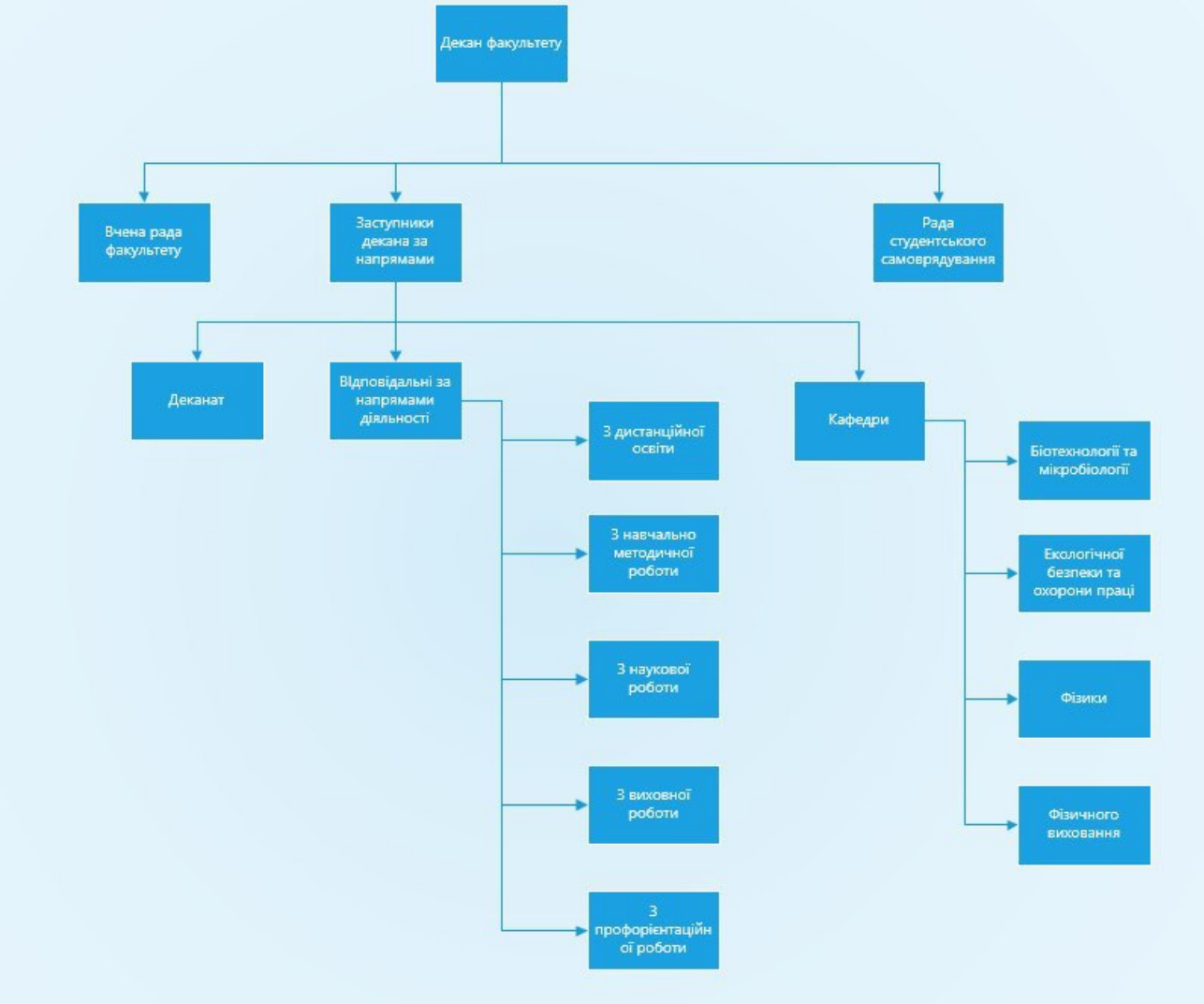

Рис. 1. Структурна схема факультету «БТЕК»

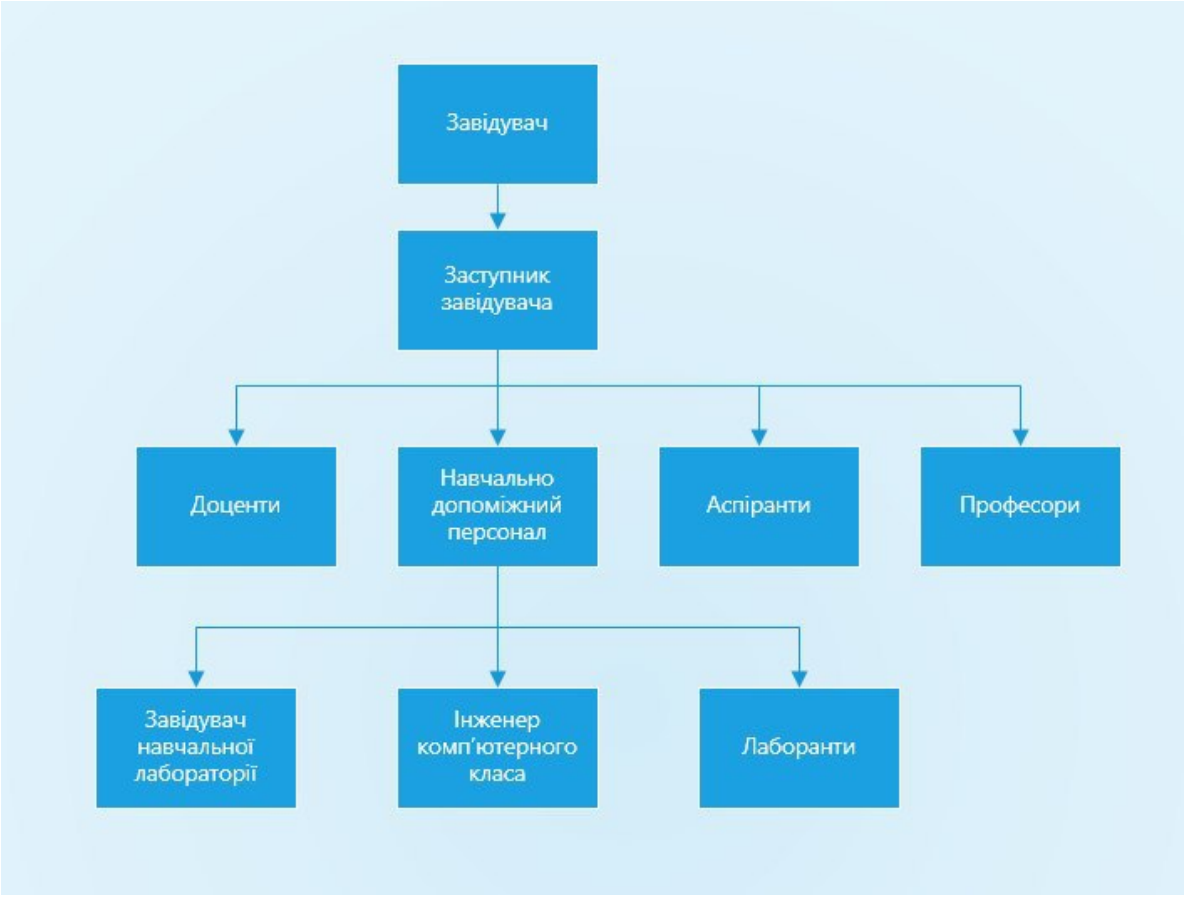

Рис. 2. Організаційна структура кафедри «ЕБОП»

Додаток Б. Функціональна та концептуальна моделі

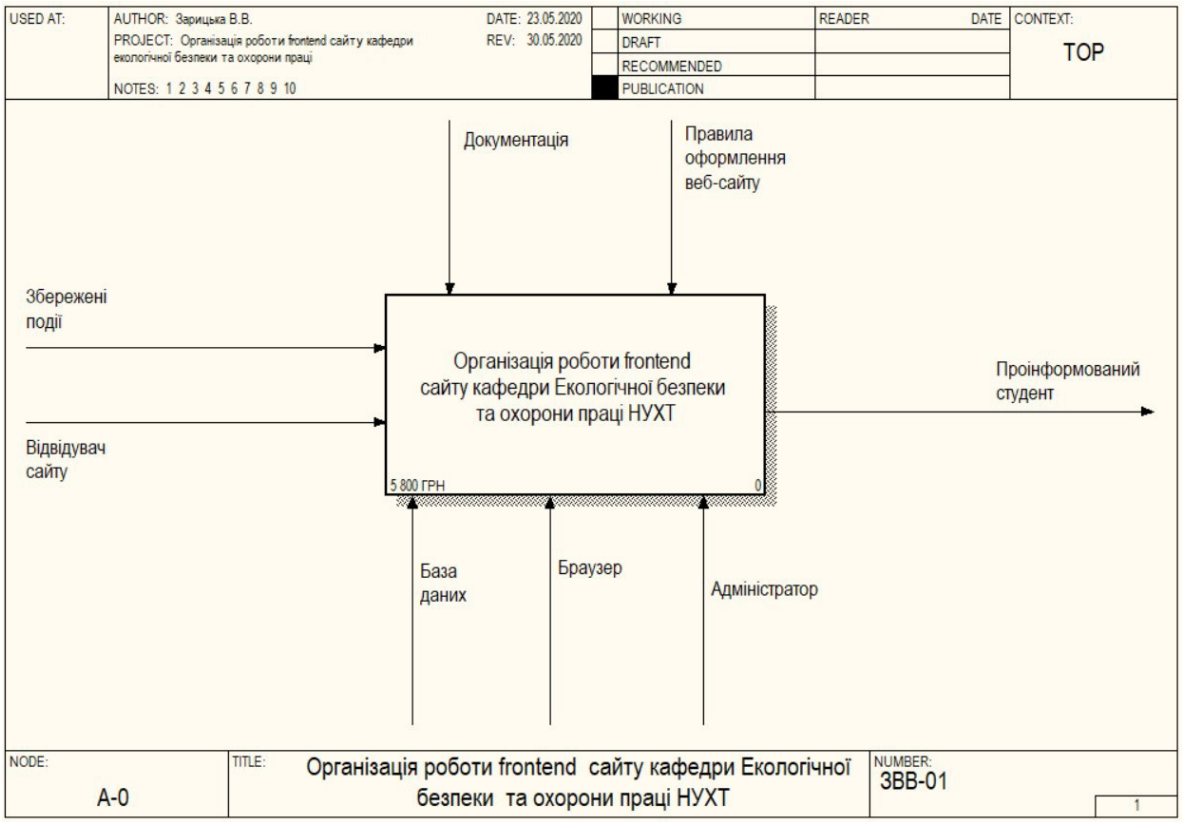

Рис. Б1. Функціональна модель AS-IS «Організації роботи frontend сайту кафедри «Екологічної безпеки та охорони праці» НУХТ»

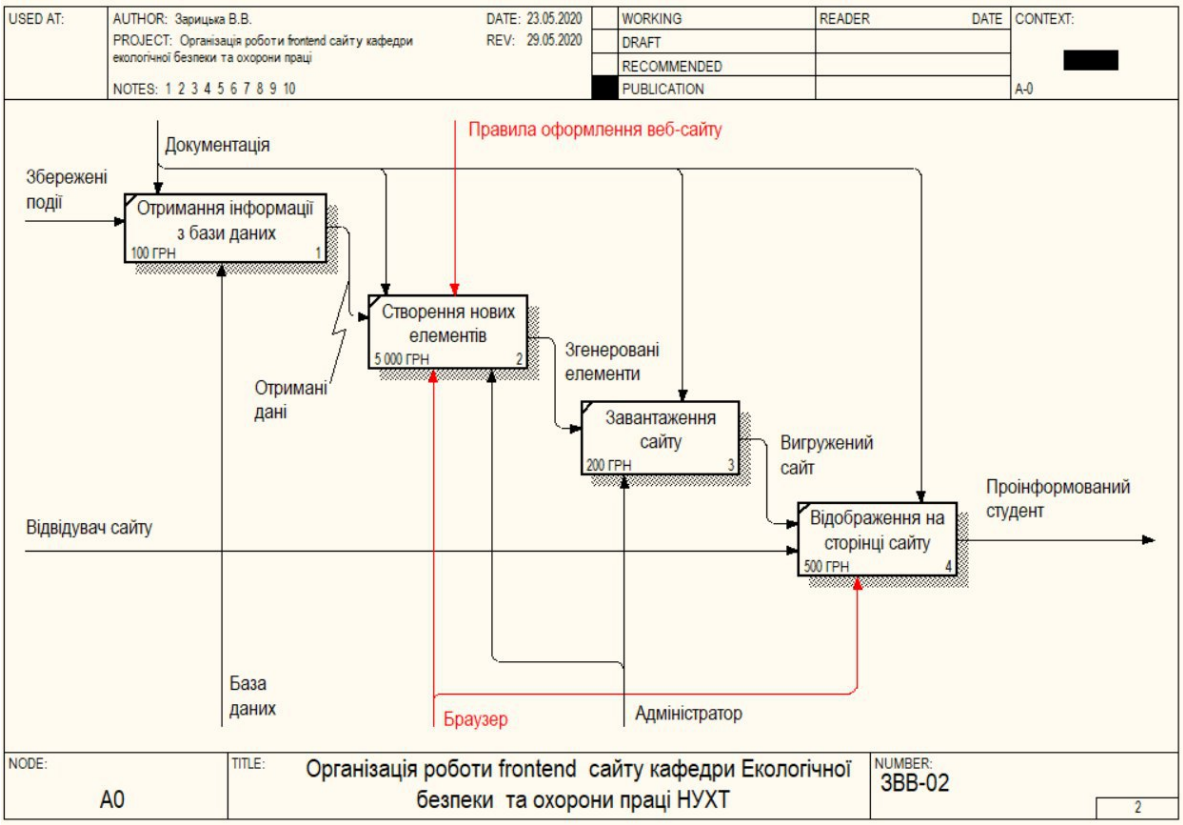

Рис. Б2. Перший рівень декомпозиції контекстної діаграми

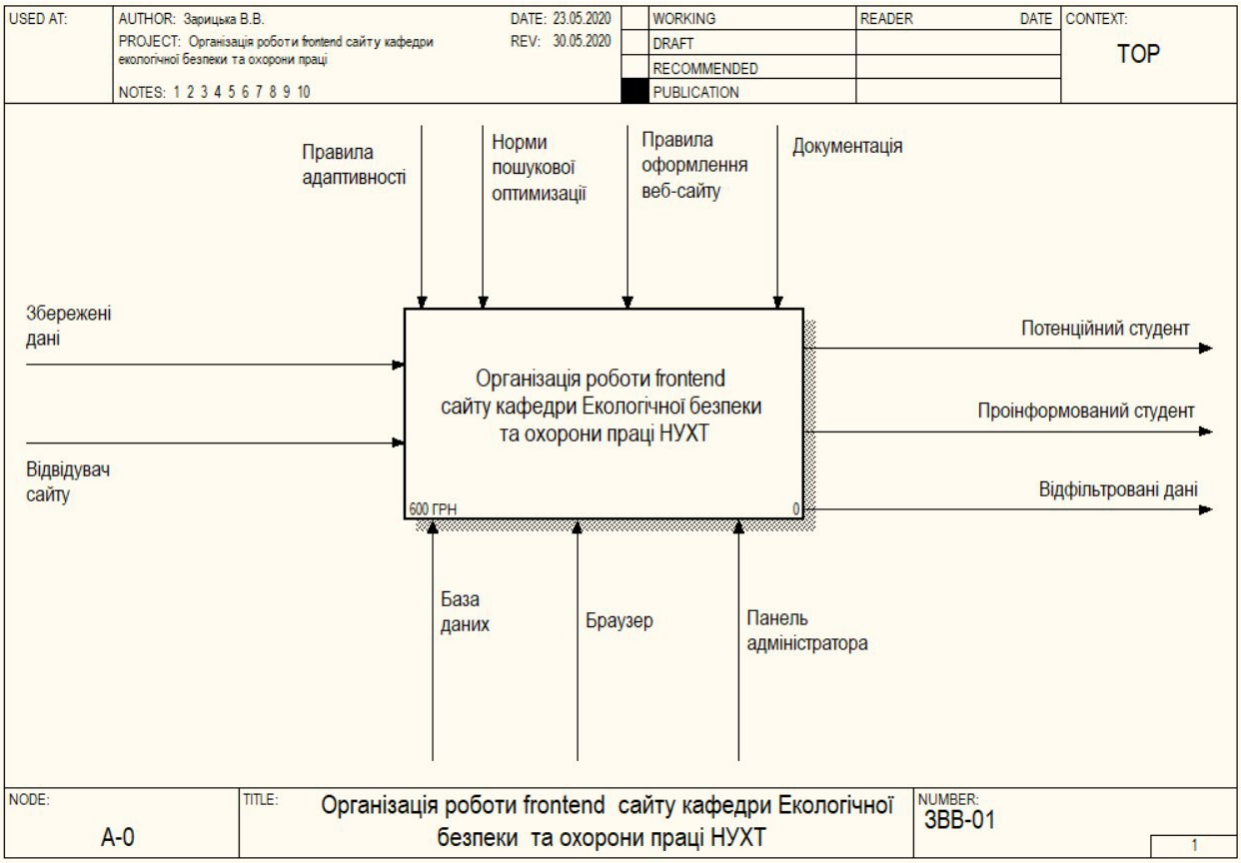

Рис. Б3. Концептуальна модель TO-BE

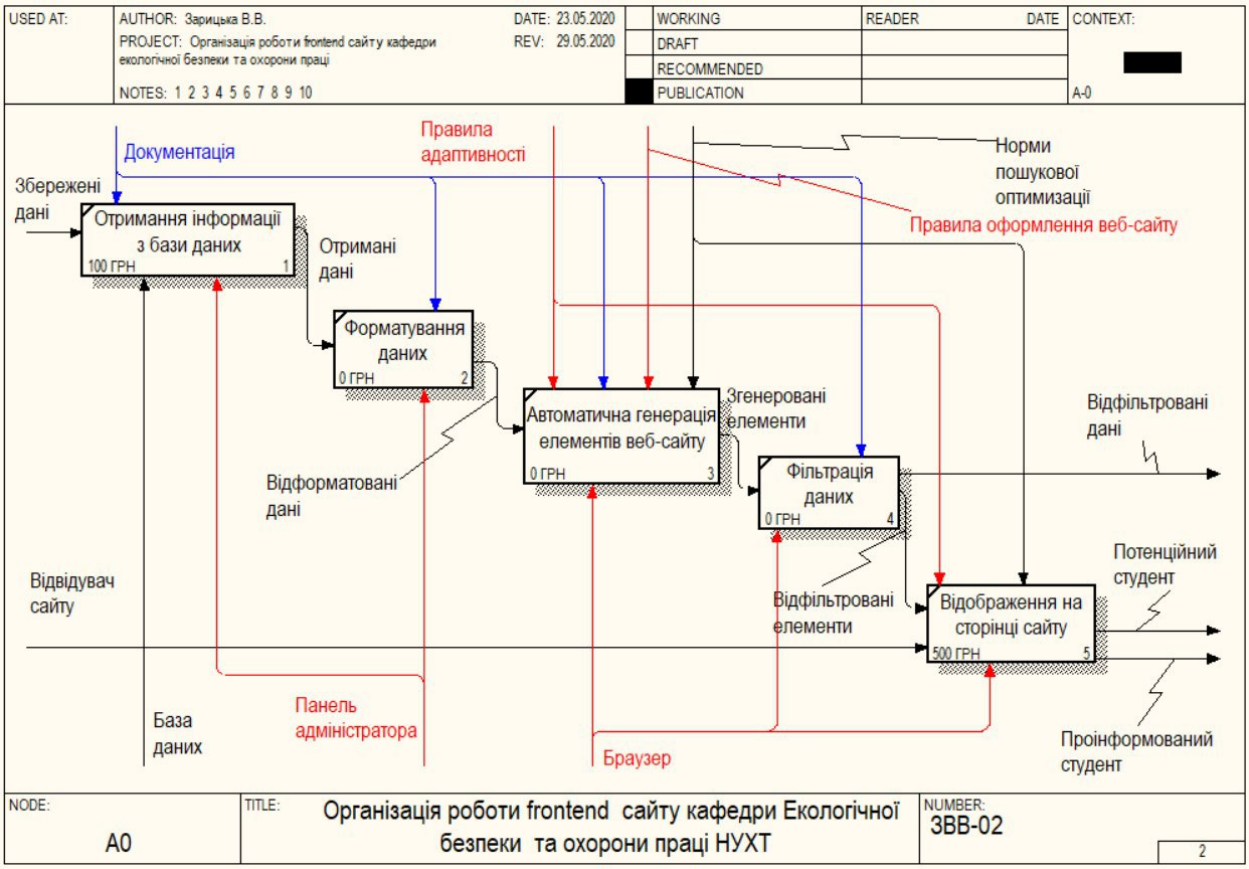

Рис. Б4. Перший рівень декомпозиції TO-BE

Додаток В. Логічна та фізична моделі бази даних

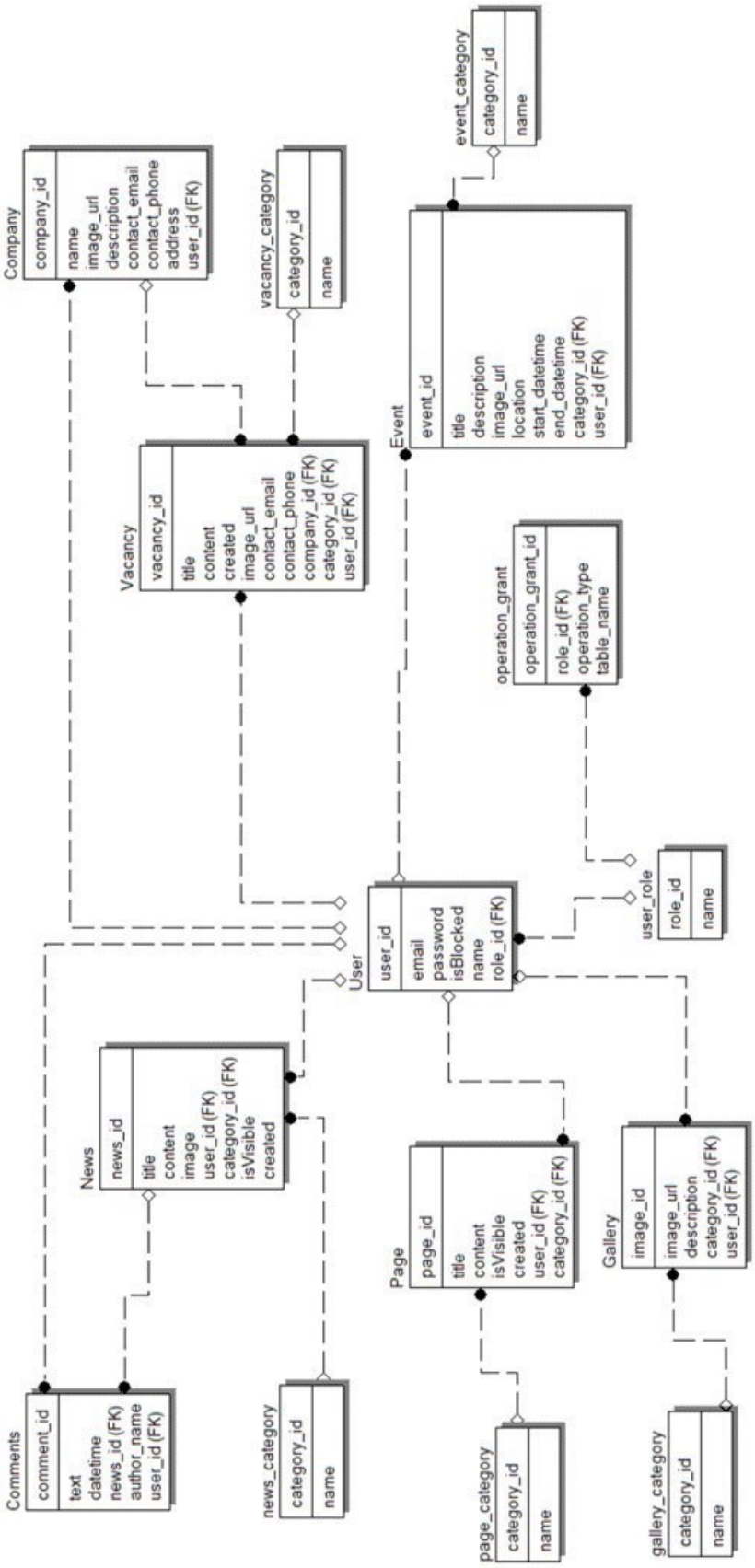

Рис. В1. Логічна модель бази даних

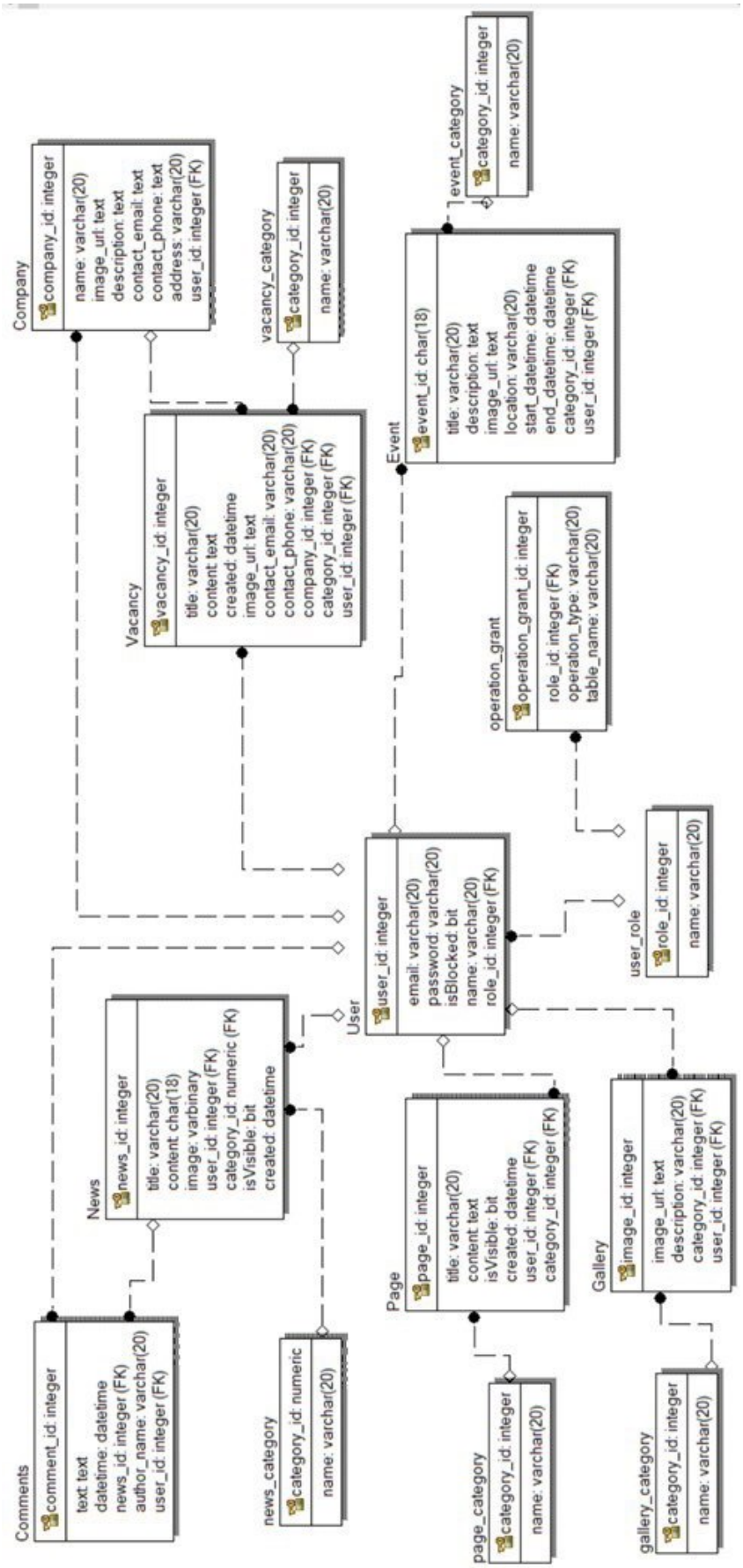

Рис. В2. Фізична модель бази даних

Додаток Г. Приклади роботи веб-сайту (інтерфейс користувача)

Рис. Г1. Футер сайту

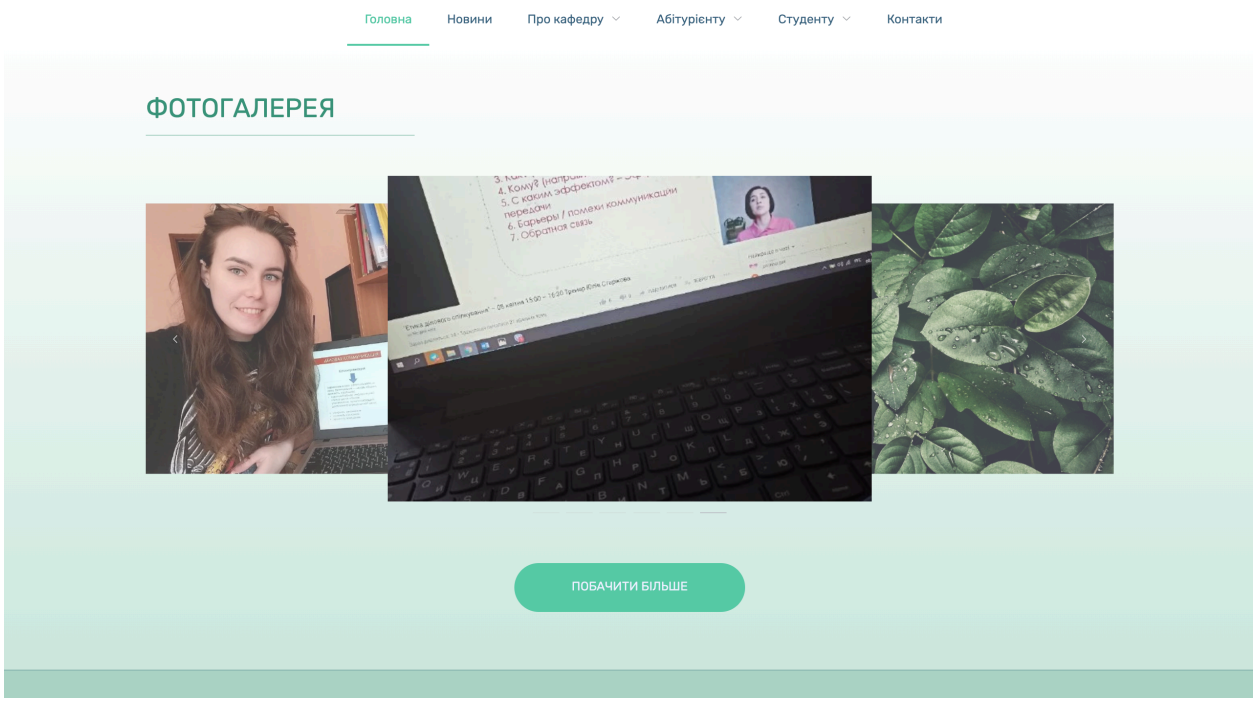

Рис. Г2. Слайдер фотографій головної сторінки

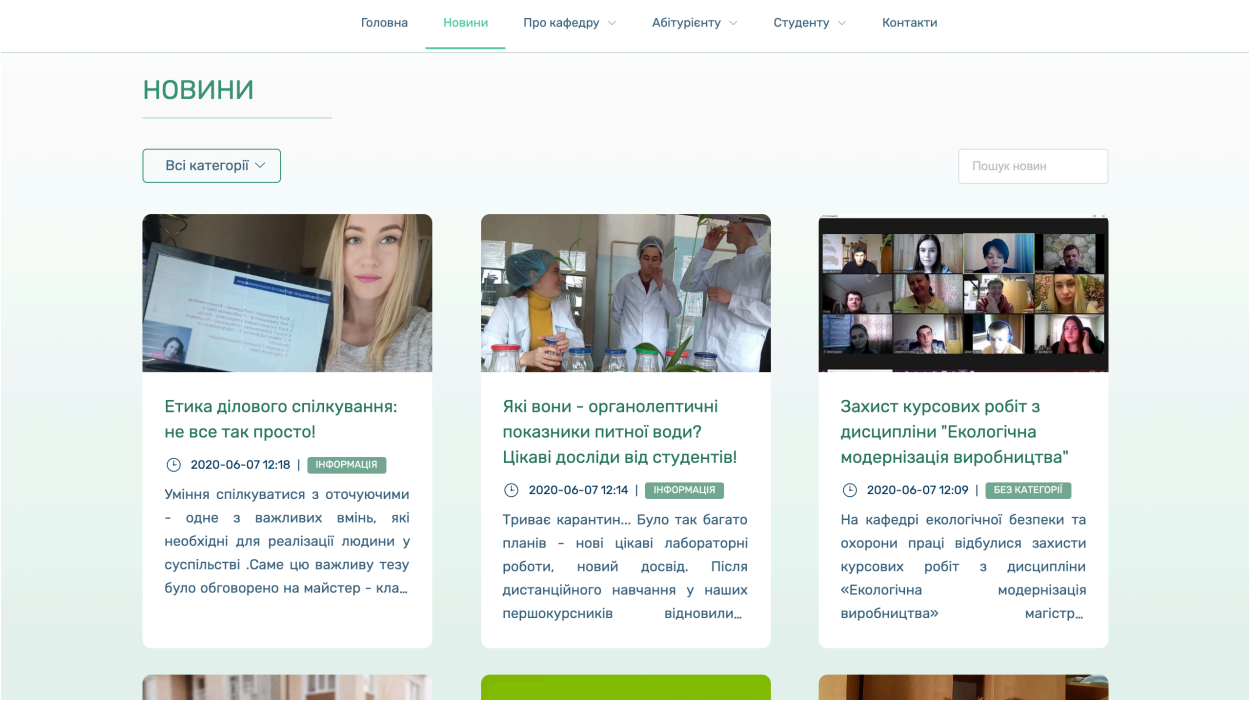

Рис. Г3. Сторінка новин

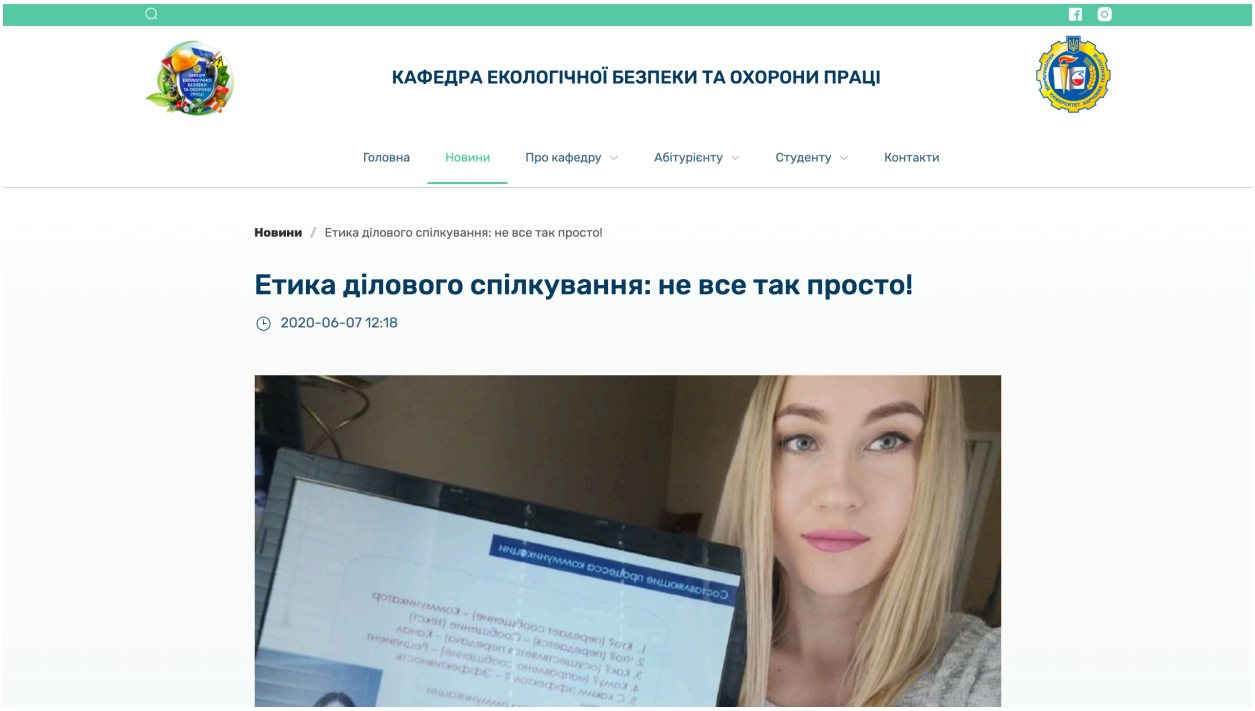

Рис. Г4. Сторінка статті

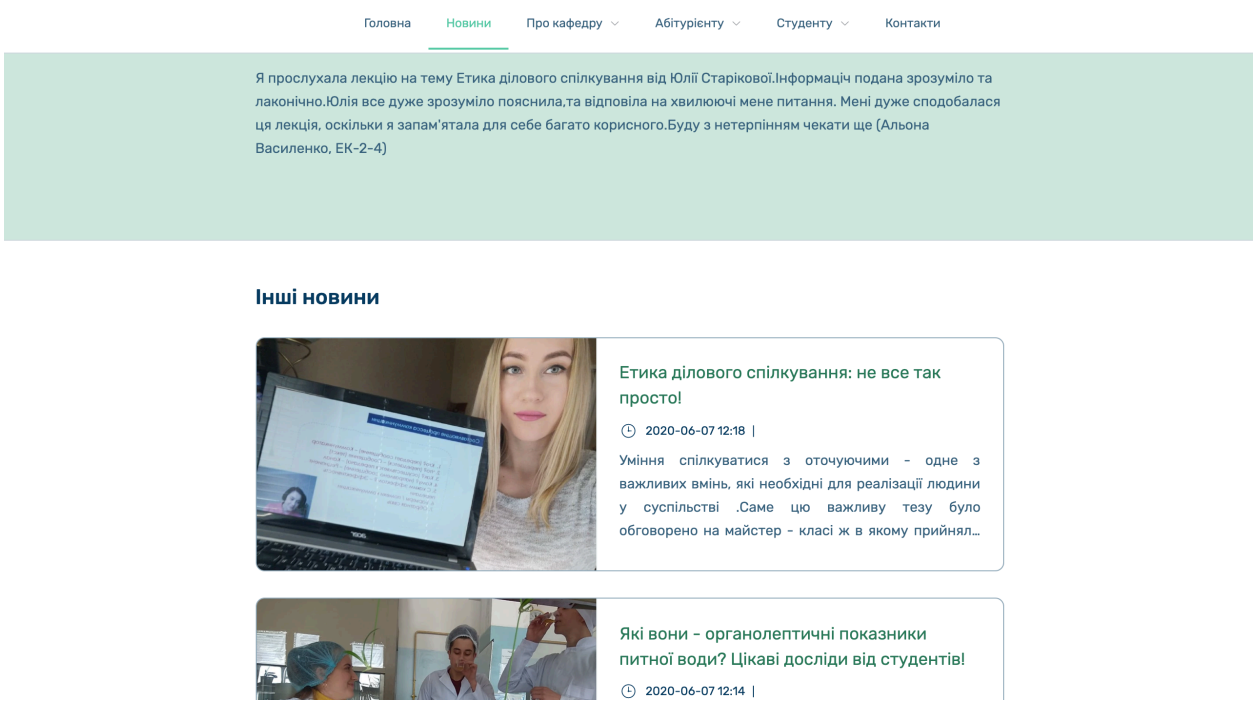

Рис. Г5. Пропозиції інших новин після статті

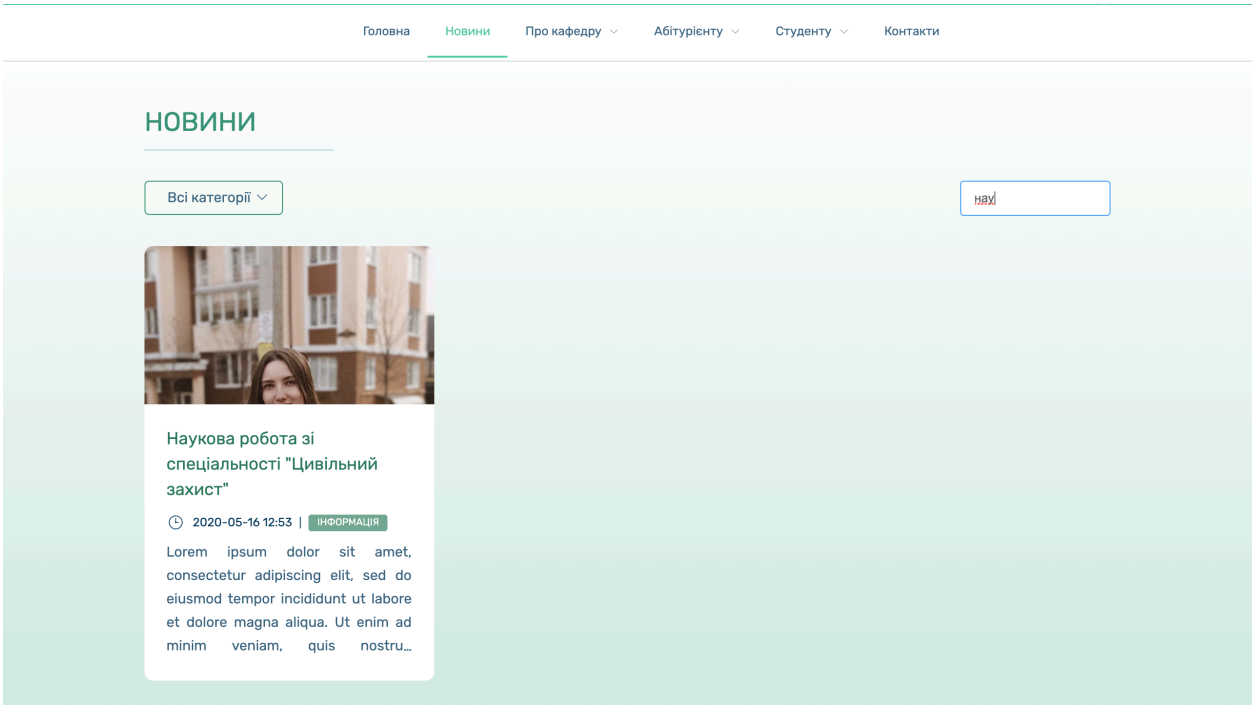

Рис. Г6. Приклад пошуку новин за фразою

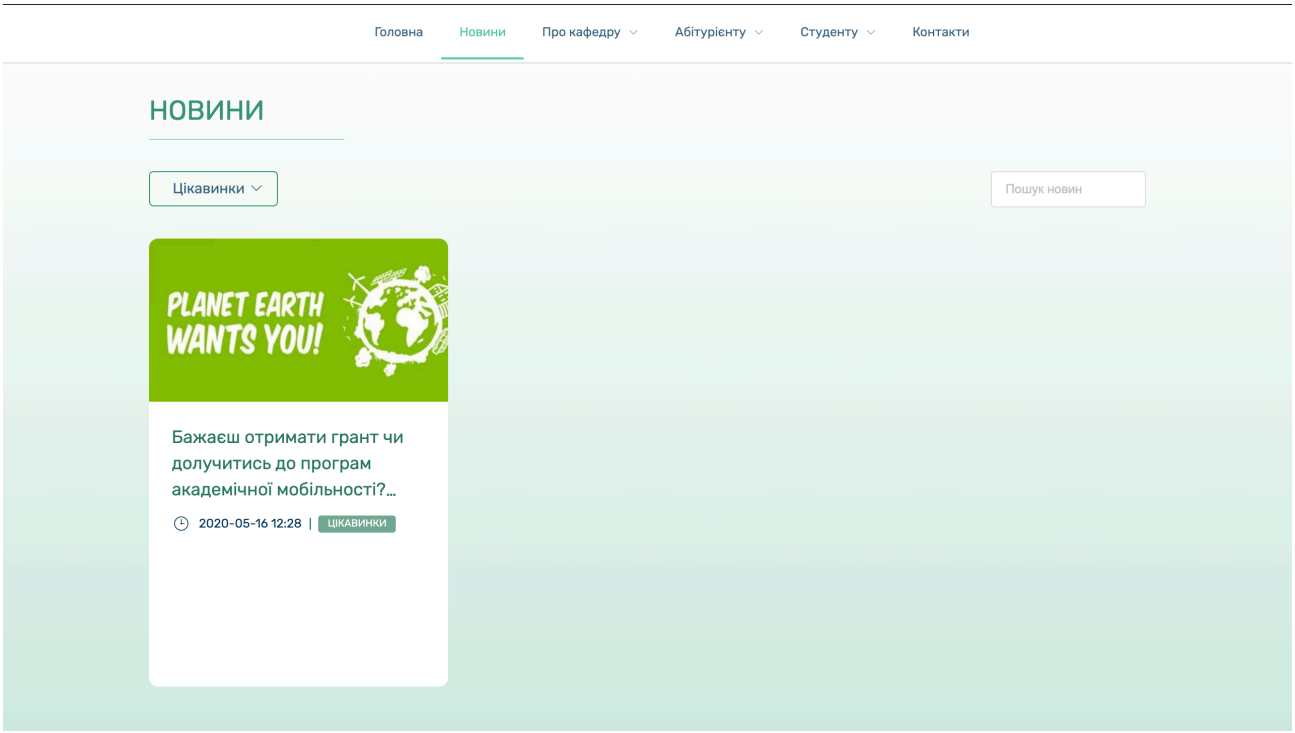

Рис. Г7. Приклад пошуку новин за категорією «Цікавинки»

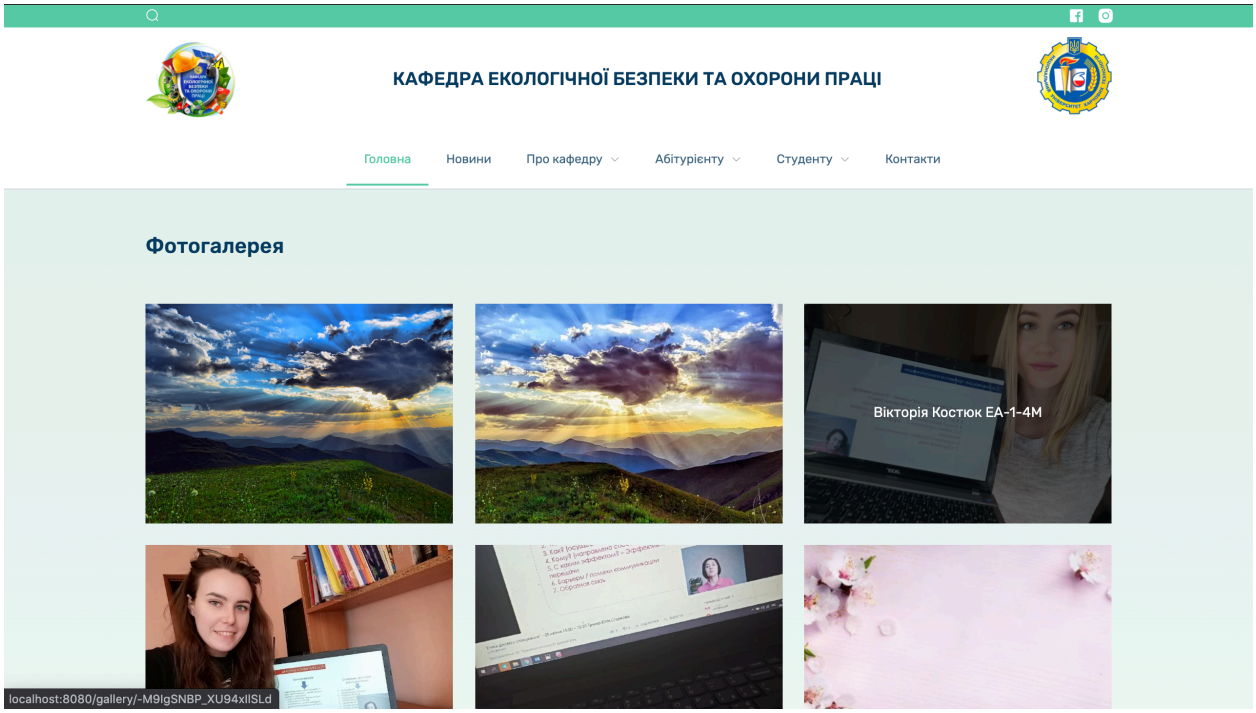

Рис. Г8. Сторінка «Фотогалерея»

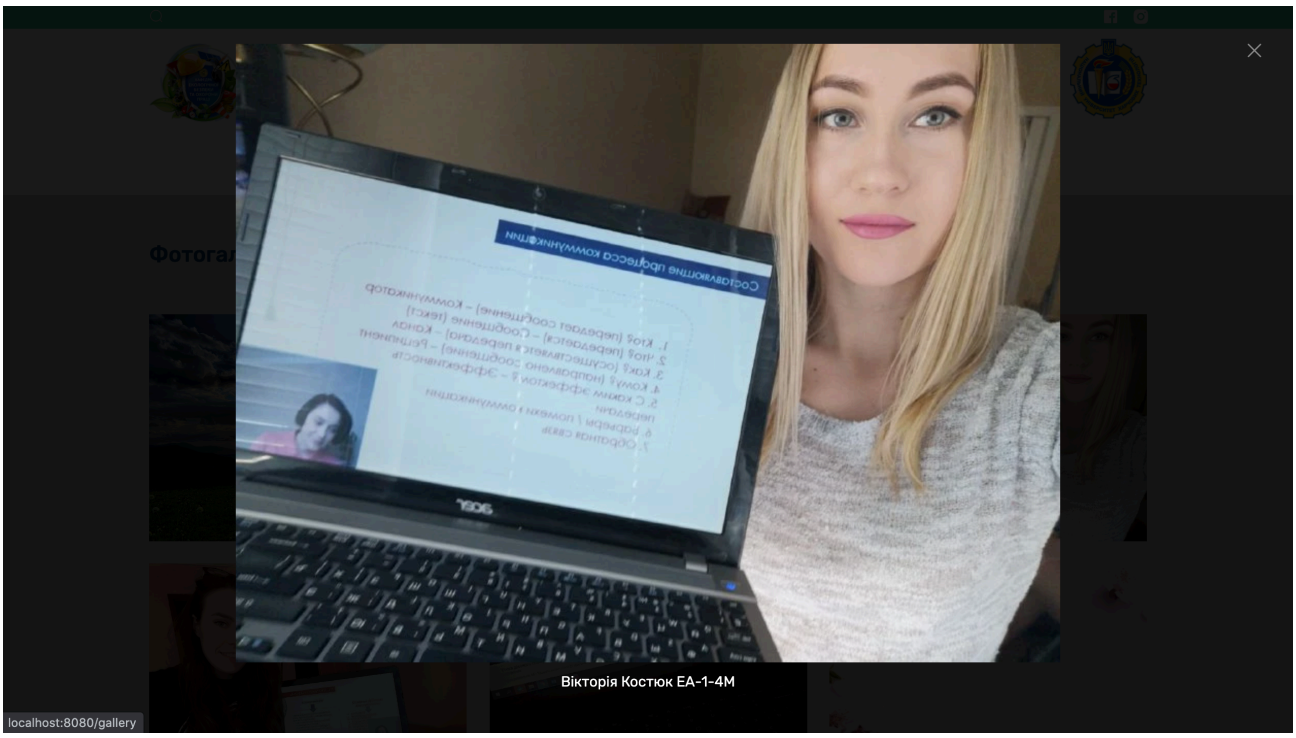

Рис. Г9. Приклад наближенної фотографії

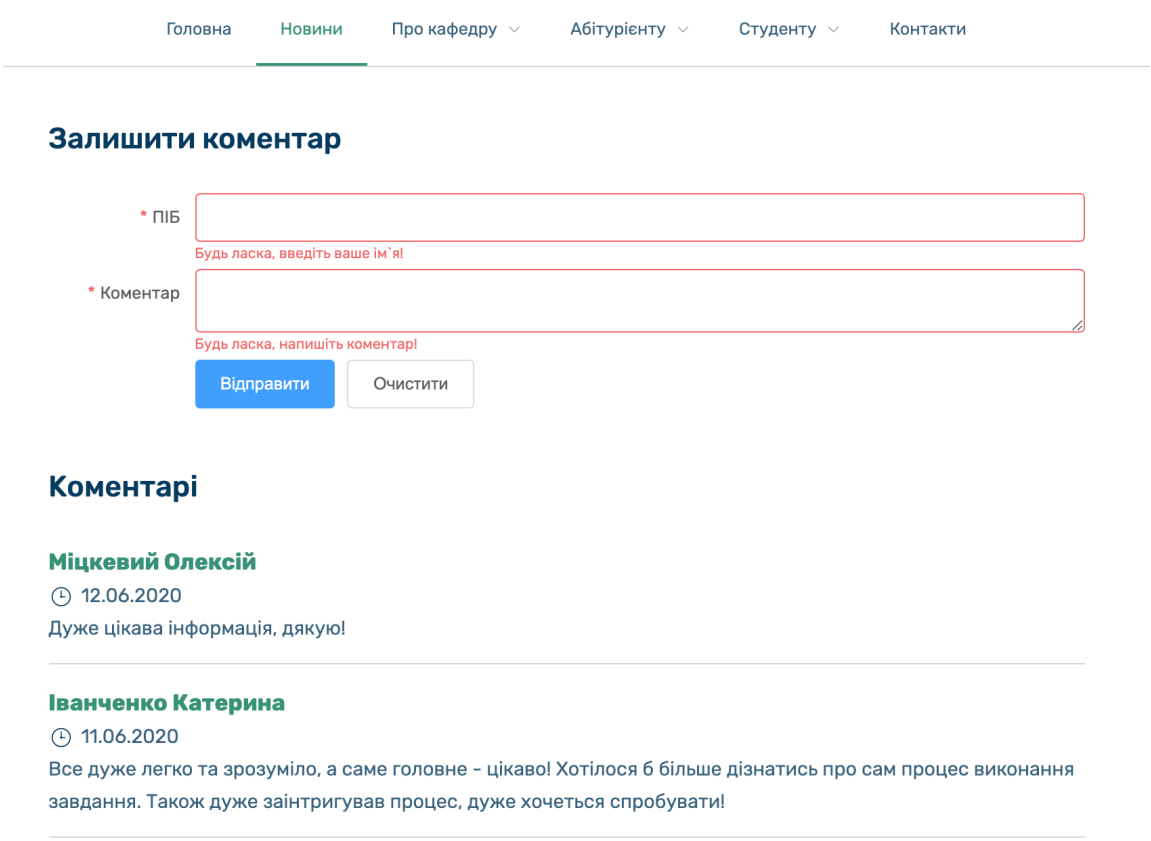

Рис. Г10. Коментарі: бов’язкові поля для заповнення.

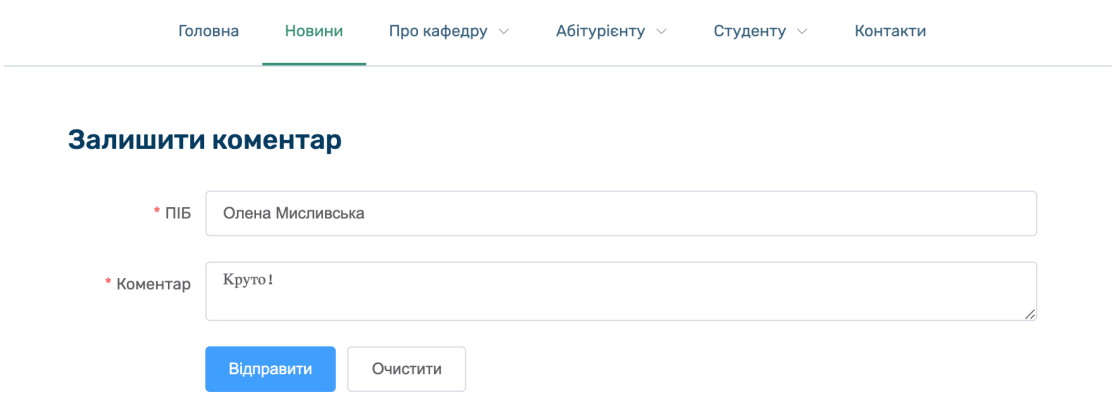

Рис. Г11. Коментарі: приклад заповнення полів

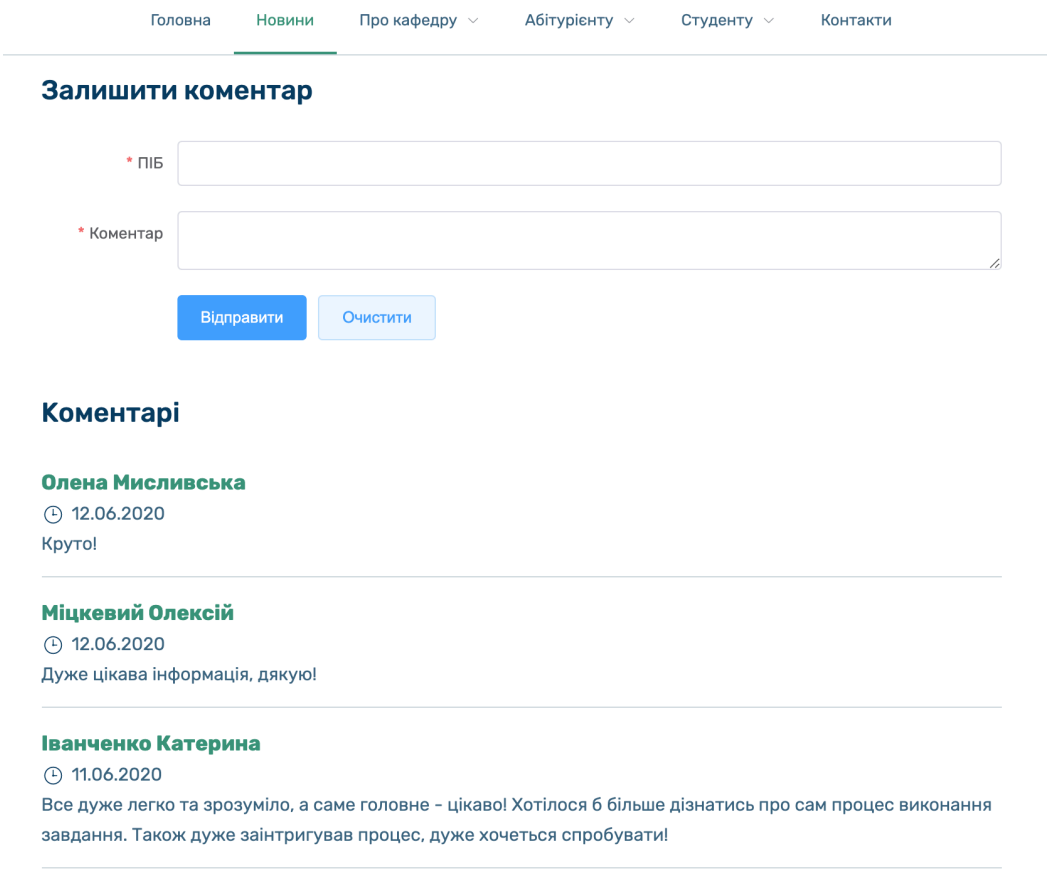

Рис. Г12. Коментарі: приклад відправленого коментарю

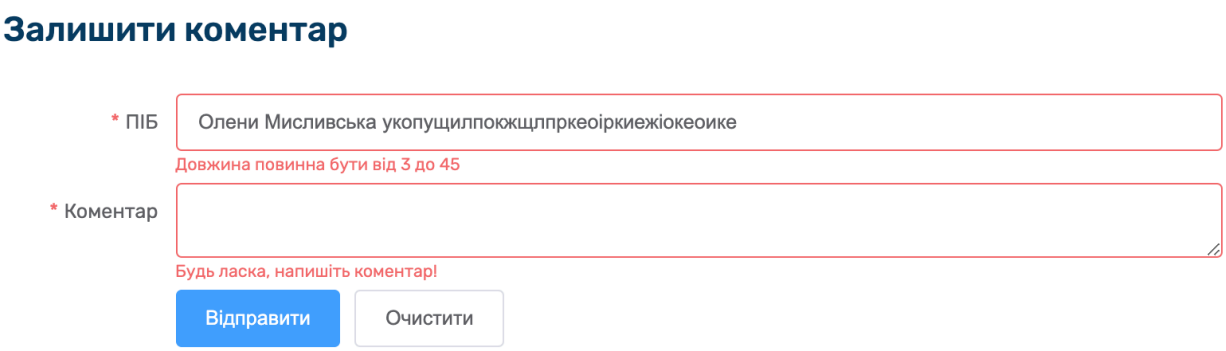

Рис. Г13. Коментарі: приклад невірного вводу поля «ПІБ»

Додаток Д. Фрагменти коду веб-сайту

1Д. Код компоненту ContactHeader.vue

```
<template>
   <div class="contacts">
      <div class="kektainer">
            <div class="search">
               <div class="el-icon-search"></div>
               <input type="text"
               class="search-input"
               placeholder="Пошук по сайту..."
               v-model="inputValue">
            </div>
            <div class="social">
               <a href="https://www.facebook.com/groups/ecologynuft/?source_id=278120335986339"
               target="_blank"
               style="margin-right: 13px;">
                  <img  src="../assets/icons/icons8-facebook.svg" alt="facebook">
               </a>

               <a href="https://www.instagram.com/ecology_nuft/"
               target="_blank"
               style="margin-right: -3px;">
                  <img src="../assets/icons/icons8-instagram.svg" alt="instagram">
               </a>
            </div>
      </div>
   </div>
</template>

<script>
export default {
   data() {
      return {
         inputValue: ''
      }
   }
}
</script>

<style scoped>
   img {
      display: block;
      width: 20px;
      transition: all .1s linear;
   }

   img:hover {
      transform: scale(1.3);
   }

   .kektainer {
      display: flex;
      height: 100%;
      align-items: center;
   }

   .contacts {
      height: 25px;
      background-color: #58caa8; /* ; 399479*/
```

```
    transition: height .2s linear;

    position: -webkit-sticky;
    position: sticky;
    top: 0;

    z-index: 1000;
  }

  .search {
    display: flex;
    align-items: center;
  }

  .search-input {
    width: 220px;
    height: 18px;
    margin-left: 15px;
    outline: none;
    color: #0a3d62;
    border-radius: 3px;
    font-family: inherit;
    font-size: 12px;
    padding: 3px 8px;
    border: 0;

    opacity: 0;
    visibility: hidden;
    transition: .25s opacity, .25s visibility;
  }

  .contacts:hover {
    height: 40px;
  }

  .search:hover .search-input{
    opacity: 1;
    visibility: visible;
  }

  .social {
    display: flex;
    margin-left: auto;
    align-items: center;
  }

  .el-icon-search {
    color: #fff;
  }
</style>
```

2Д. Код компоненту Header.vue:

```
<template>
  <div style="background-color: white;">
    <div class="kektainer" >
      <div class="header-inner">

        <!-- <div  class="mobile-header-visible header-display-none line-right ">
```

```
            <a href="#" target="_blank">
              <img class="footer-icon" src="@/assets/icons/menu-wight.svg" alt="">
            </a>
          </div> -->

          <div  class="">
            <a href="#">
              <img class="logo-eco" src="@/assets/logo-eco.png">
            </a>
          </div>

          <div class="header-name mobile-display-none">
            <p class="header-name-style">Кафедра екологічної безпеки та охорони праці</p>
            <p class="header-faculty">БТЕК</p>

          </div>

          <div  class="mobile-display-none">
            <a href="http://nuft.edu.ua/" target="_blank">
              <img src="@/assets/logo-nuft.jpg" style="width: 145px;">
            </a>
          </div>
        </div>
        <div class="block-line"></div>
      </div>
    </div>
</template>

<script>
export default {

}
</script>

<style scoped>
  .block-line {
    width: 95%;
    margin: 0 auto;
  }

  .header-inner {
    display: flex;
    justify-content: space-between;
    align-items: center;

    height: 235px; /* 185px */
    color: #0a3d62;
  }

  .header-name {
    text-align: center;
    line-height: 2;
  }

  .header-name-style {
    font-weight: 500;
    text-transform: uppercase;
    font-size: 1.5em; /* 25px */
  }

  .header-faculty {
```

```
    text-transform: uppercase;
    font-size: 1.25em; /* 20px */
  }

  .logo-eco {
    display: block;
    width: 170px;
  }
</style>
```

3Д. Код компоненту Menu.vue

```
<template>
 <div class="menu-container">
   <div class="menu-field">
     <el-menu :default-active="activeIndex" class="el-menu-demo" mode="horizontal"
@select="handleSelect" active-text-color="#58caa8" text-color="#3c6382">
       <el-menu-item index="1" :to="{ path: '/' }">Головна</el-menu-item>
       <el-menu-item index="2" :to="{ path: '/news' }">Новини</el-menu-item>
       <el-submenu index="3">
         <template slot="title">Про кафедру</template>
         <el-menu-item index="3-1">Колектив кафедри</el-menu-item>
         <el-menu-item index="3-2">Відзнаки та професійні досягнення</el-menu-item>
         <el-submenu index="3-3">
           <template slot="title">Життя кафедри</template>
           <el-menu-item index="3-3-1">Навчальний процес</el-menu-item>
           <el-menu-item index="3-3-2">Профорієнтаційна робота</el-menu-item>
           <el-menu-item index="3-3-3">Події кафедри</el-menu-item>
           <el-menu-item index="3-3-4">Практична підготовка</el-menu-item>
         </el-submenu>
       </el-submenu>
       <el-submenu index="4">
         <template slot="title">Абітурієнту</template>
         <el-menu-item index="4-1">Хочу бути бакалавром</el-menu-item>
         <el-menu-item index="4-2">Хочу бути магістром</el-menu-item>
           <el-submenu index="4-3">
             <template slot="title">Подача документів</template>
             <el-menu-item index="4-3-1">Випускнікам шкіл</el-menu-item>
             <el-menu-item index="4-3-2">Студентам НУХТ</el-menu-item>
             <el-menu-item index="4-3-3">Випускнікам коледжів та технікумів</el-menu-item>
             <el-menu-item index="4-3-4">Бакалаврам</el-menu-item>
             <el-menu-item index="4-3-5">Для отримання другої вищої освіти</el-menu-item>
             <el-menu-item index="4-3-6">Іноземним громадянам</el-menu-item>
           </el-submenu>
        </el-submenu>
       <el-submenu index="5">
         <template slot="title">Студенту</template>
         <el-menu-item index="5-1">Розклад занять</el-menu-item>
         <el-menu-item index="5-2">Курсове проектування</el-menu-item>
         <el-menu-item index="5-3">Дипломне проектування</el-menu-item>
         <el-menu-item index="5-4"><a href="/gallery">Фотогалерея</a></el-menu-item>
        </el-submenu>
       <el-menu-item index="6">Контакти</el-menu-item>

     </el-menu>
     <div class="block-line"></div>
   </div>
 </div>
</template>
```

```
<script>
  export default {
    data() {
      return {
        activeIndex: '1',
        activeIndex2: '1'
      };
    },
    methods: {
        handleSelect(key, keyPath) {
        console.log(key, keyPath);
      }
    }
  }
</script>

<style scoped>
  .block-line {
    margin-top: 3px;
  }

  .menu-field {
    background-color: #fff;
  }

  .el-menu.el-menu--horizontal {
    border-bottom: none;
  }

  .el-menu {
    padding-left: 310px;
  }

  .menu-container {
    position: -webkit-sticky;
    position: sticky;
    top: 0;
    z-index: 1000;
  }

</style>
```

4Д. Код компоненту Footer.vue

```
<template>
   <div>
      <footer class="footer">
         <div class="block-line"></div>
         <div class="kektainer">
            <div class="footer-inner">
               <div class="footer-item">
                  <a href="http://nuft.edu.ua/" target="_blank" class="link-hidden">
                     <img class="logo-eco" src="../assets/logo-nuft-footer.svg">
                  </a>
               </div>

               <div class="footer-item">
                     <div class="footer-contacts" style="margin-top: 23px;">
```

```
                    <div class="footer-item">
                        <span class="underline dark-blue">Адреса </span>
                        <a href="https://www.google.com/maps/place/
%D0%9D%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D1
%8B%D0%B9+
%D1%83%D0%BD%D0%B8%D0%B2%D0%B5%D1%80%D1%81%D0%B8%D1%82%D0%B5%D1%8
2+%D0%BF%D0%B8%D1%89%D0%B5%D0%B2%D1%8B%D1%85+
%D1%82%D0%B5%D1%85%D0%BD%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D0%B9/
@50.4388059,30.5094419,17z/data=!4m5!3m4!1s0x40d4cefbb3f8407b:0x5bf2348764ce2c0f!8m2!
3d50.4392434!4d30.5098714" target="_blank">Україна, 01601, м. Київ-33, <br>вул. Володимирська 68</
a>
                    </div>

                    <div class="footer-item">
                        <span class="underline dark-blue">Телефон </span>
                        <a href="tel:(067)0324531">(044)287-93-21</a>
                    </div>

                    <div class="footer-item">
                        <span class="underline dark-blue">E-mail </span>
                        <a href="mailto:ecologynuft@gmail.com">ecologynuft@gmail.com</a>
                    </div>
                </div>
            </div>

            <div class="footer-item flex-column">
                <div class="social">
                    <a href="https://www.facebook.com/groups/ecologynuft/?source_id=278120335986339"
target="_blank" style="margin-right: 13px;">
                        <img class="social-item" src="../assets/icons/icons8-facebook.svg" alt="facebook">
                    </a>

                    <a href="https://www.instagram.com/ecology_nuft/" target="_blank" style="margin-right:
-3px;">
                        <img class="social-item" src="../assets/icons/icons8-instagram.svg" alt="instagram">
                    </a>
                </div>
                <div class="block-line block-line--footer"></div>
                <div class="search">
                    <!-- <div class="el-icon-search"></div> -->
                    <input type="text" class="search-input" placeholder="Пошук по сайту..." v-
model="inputValue">
                 </div>
            </div>
        </div>
        <p class="footer-mark" style="text-transform: uppercase;">© 2020 Кафедра екологічної безпеки
та охорони праці</p>
      </div>
    </footer>
  </div>
</template>

<script>
export default {

}
</script>

<style scoped>
  .kektainer {
    position: relative;
```

```
}

.logo-eco {
   width: 120px;
}

.social-item {
   display: block;
   width: 40px;
   transition: all .1s linear;
}

.social-item:hover {
   /* width: 45px; */
   transform: scale(1.2);
}

.social {
   display: flex;
   align-items: center;
}

.footer {
   background-color: #a9d4c3;
   flex: 0 0 auto;
}

.footer-inner {
   height: 270px;
   width: 100%;
   display: flex;
   align-items: center;
   justify-content: space-between;
}

.footer a {
   color: #fff;
   transition: color .2s linear;
}

.footer a:hover {
   color: #0a3d62;
}

.footer-contacts {
   width: 45vw;
   display: flex;
   justify-content: space-between;
   align-items: flex-start;
}

.footer-icon {
   width: 35px;
   color: #ffc15c;
}

.footer-mark {
   position: absolute;
   bottom: 10px;
   left: 36%;
```

```
    font-size: 0.6em;
    color: rgba(10, 61, 98, 0.8);
    font-weight: 500;
  }

  .dark-blue {
    color: #0a3d62;
    font-weight: 500;
    font-size: 17px;
  }

  .underline::after {
    content: '';
    display: block;
    height: 1px;
    width: 100%;

    background-color: rgba(10, 61, 98, 0.4);
    margin: 10px 0 15px 0;
  }

  .flex-column {
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: space-between;
  }

  .search-input {
    width: 220px;
    height: 28px;
    margin-left: 15px;
    outline: none;
    color: #0a3d62;
    border-radius: 3px;
    font-family: inherit;
    font-size: 14px;
    padding: 3px 8px;
    border: 0;
  }

  .block-line--footer {
    width: 150px;
    margin: 15px 0 20px;
    background-color: rgba(10, 61, 98, 0.4);
  }

</style>
```

5Д. Код компоненту Article.vue:

```
<template>
  <div class="article kektainer">
    <div><h2>Новини</h2></div>
    <div style="display: flex; justify-content: space-between;">
      <NewsCategory v-if="isNewsCategory"
            @select="selectedCategory"/>
      <div>
```

```
            <el-input placeholder="Пошук новин" v-model="search"></el-input>
          </div>
        </div>

        <div class="article-field" v-loading="loading">
          <div class="article-item" v-for="article in selectByCategory" :key="article.id">
            <a :href="`/news/${article.id}`" class="link">
              <div class="inner">

                <div class="picture">
                  <div class="picture-inner"  :style="{backgroundImage: `url('${article.photo}')`}"></div>
                </div>

                <div class="container">
                  <h3 class="name text-hidden text-hidden--3">{{ article.title }}</h3>

                  <date-category :articleDate="article.createDate" :category="article.category"></date-
category>

                  <div class="description text-hidden">
                    <p>{{ article.description }}</p>
                  </div>
                </div>

              </div>
            </a>
          </div>
          <div v-if="!loading && !selectByCategory.length"><h3>Нічого не знайдено</h3></div>
        </div>
    </div>

</template>

<script>
import DateCategory from './DateCategory.vue'
import NewsCategory from './NewsCategory.vue'

import moment from 'moment';
import _ from 'lodash';

export default {
    data() {
        return {
            articles: [],
            selectByCategory: [],
            loading: true,
            search: '',
            selected: ''
        }
    },

    props: {
        isNewsCategory: {
            type: Boolean,
            default: true
        },

        itemsOnPageImp: {
            type: Number
        }
    },
```

```
  components: {
    DateCategory,
    NewsCategory
  },

  methods: {
    transformNews(news) {
      return _.map(news, item => ({
        photo: !_.isUndefined(item.photo) ? item.photo.url : 'https://www.ticketpro.by/storage/img/no-
image.png',
        title: item.title,
        category: !_.isUndefined(item.category.title) ? item.category.title : 'Без категорії',
        description: item.description,
        id: item.id,
        createDate: moment(item.created).format("YYYY-MM-DD HH:mm")
      }))
    },

    async loadNews({startAt = 0, itemsOnPage = 6, q = ''}) {
      this.loading = true;
      const resp = await fetch(`https://us-central1-nuft-kebop.cloudfunctions.net/news?startAt=${startAt}
&itemsOnPage=${itemsOnPage}&q=${q}`);
      const result = await resp.json();
      this.loading = false;
      return this.transformNews(result.data);
    },


    selectedCategory(option) {
      this.selected = option.categoryType;
      this.selectByCategory = [];
      let vm = this;
      this.articles.map((item) => {
        if(item.category === option.categoryType) {
          vm.selectByCategory.push(item)
        }
      });
      if (this.selected === 'all_categories') {
        this.selectByCategory = this.articles;
      }
    }
  },

  async mounted() {
    this.articles = await this.loadNews({q: this.search, itemsOnPage: this.itemsOnPageImp});
    this.selectByCategory = this.articles;
  },

  computed: {
    filterArticles() {
      if(this.selectByCategory.length) {
        return this.selectByCategory
      }
      else {
        return this.articles
      }
    }
  },

  watch: {
```

```
        async search(searchValue) {
            this.selectByCategory = await this.loadNews({q: searchValue});
        }
    }

}
</script>

<style>
    .el-loading-mask {
        background-color: transparent !important;
    }
</style>

<style scoped>

    h2 {
        margin-bottom: 35px;
        font-size: 30px;
        font-weight: normal;
        color: rgb(57, 148, 121);
        padding: 0 15px 5px 0;
        border-bottom: 1px solid rgba(57, 148, 121, 0.4);
        width: 200px;
        text-transform: uppercase;
    }

    .article {
        margin: 0 auto;
        display: flex;
        flex-direction: column;
    }

    .article-field {
        display: flex;
        justify-content: space-between;
        min-height: 495px;
        flex-wrap: wrap;
    }

    .text-hidden{
        display: -webkit-box;
        -webkit-box-orient: vertical;
        overflow: hidden;
        -webkit-line-clamp: 5;
    }

    .text-hidden--3{
        -webkit-line-clamp: 3;
    }

    .block {
        display: flex;
        justify-content: space-between;
        flex-wrap: wrap;
        /* width: 100%; */
    }

    .article-item {
        width: 330px; /*  */
        height: 495px; /* 0px */
```

```
    border-radius: 10px;

    margin-bottom: 30px;

    background-color: #fff;
    overflow: hidden;
}

.inner {
    width: 330px; /* -80px */
    height: 495px; /* 0px */
}

.container {
    width: 280px; /* -60px */
    margin: 25px auto 15px;
}

.picture {
    display: block;
    width: 330px;
    height: 180px;

    overflow: hidden;
}

.picture-inner {
    width: inherit;
    height: inherit;
    background-size: cover;

    transition: transform .2s;
}

.article-item:hover .picture-inner{
    transform: scale(1.2);
}

.name {
    font-size: 1.150em; /* 20px */
    font-weight: 400;
    color: #29795a;

    margin-bottom: 12px; /* -15px */
    line-height: 1.5;
}

.description {
    font-size: 1rem; /* 14px */
    color: #3c6382;
    font-weight: 300;
    line-height: 1.7;
    text-align: justify;
    word-wrap: break-word;

    margin-bottom: 25px;
}

.link {
    display: block;
    color: #29795a;
```

```
    font-size: 1.063em;
  }

  .description-button {
    width: 135px;
    padding: 8px 0;
    border: 2px solid #0a3d62;
    color: #0a3d62;
    font-size: 12px;
    margin: 0 auto;
    text-align: center;
    border-radius: 4px;
    font-weight: 500;
  }

  .description-button:hover {
    border-color: #29795a;
    background-color: rgba(41, 121, 90, 0.1);
  }

</style>
```

6Д. Код компоненту GaleryPhoto.vue

```
<template>
<div>
  <div class="kektainer">
      <div class="gallery-item" v-for="photo in photos" :key="photo.id">
        <router-link :to="`/gallery/${photo.id}`" class="link">

        <div class="gallery-item-inner" :style="{backgroundImage: `url('${photo.picture}')`}">
          <div class="gallery-description">
            <div class="description-inner">{{photo.description}}</div>
          </div>
        </div>

        </router-link>
      </div>
    <div v-if="!loading && !photos.length"><h3>Нічого не знайдено</h3></div>
  </div>
  <div v-if="loading" style="height: 200px;" v-loading="loading"></div>
  <intersect @enter="load"><div></div></intersect>
  </div>
</template>

<script>
import _ from 'lodash';

import Intersect from 'vue-intersect'

export default {

  components: {
    Intersect
  },

  data() {
    return {
```

```
            photos: [],
            loading: true,
            start: 0,
            itemsPerPage: 12,
            isEndReached: false,
        }
    },

    methods: {
        transformPhotos(news) {
            return _.map(news, item => ({
                picture: item.url,
                description: item.description,
                id: item.id,
            }))
        },

        async loadPhotos({startAt = 0, itemsOnPage = this.itemsPerPage}) {
            this.loading = true;

            await new Promise(resolve => setTimeout(() => {resolve()}, 2000))

            const resp = await fetch(`https://us-central1-nuft-kebop.cloudfunctions.net/gallery
                                ?start=${startAt}&itemOnPage=${itemsOnPage}`);

            const result = await resp.json();
            this.loading = false;

            if(!result.success)
                this.isEndReached = true;
            else
                this.isEndReached = false;

            return this.transformPhotos(result.data);
        },

        async load() {
            if(this.isEndReached) return;

            this.start += this.itemsPerPage;
            const data = await this.loadPhotos({startAt: this.start});
            this.photos = [...this.photos, ...data];
        },
    },

    async mounted() {
        this.photos = await this.loadPhotos({startAt: this.start});
    }
}
</script>

<style scoped>
    .kektainer {
        display: flex;
        justify-content: space-between;
        flex-wrap: wrap;
        width: inherit;
    }

    .gallery-item {
        margin-bottom: 25px;
```

```
    width: 350px;
    height: 250px;
  }

  .gallery-item-inner {
    height: 250px;
    background-position: center;
    background-size: cover;
  }

  .gallery-description {
    display: flex;
    justify-content: center;
    align-items: center;

    height: inherit;
    opacity: 0;
    background-color: rgba(0, 0, 0, 0.6);
    transition: all .3s linear
  }

  .description-inner {
    padding: 15px 25px;
    color: #fff;
  }

  .gallery-description:hover {
    opacity: 1;
  }

</style>
```

7Д. Код файлу router.js

```
import Vue from 'vue'
import Router from 'vue-router'

import MainContent from '@/components/Main'
import NewsPage from '@/components/NewsPage'
import ArticlePage from '@/components/ArticlePage'
import GalleryPage from '@/components/GalleryPage'
import GalleryBigPhoto from '@/components/gallery/GalleryBigPhoto'



Vue.use(Router);

export default new Router ({
  mode: 'history',
  routes: [
    {
      path: '/',
      name: 'main',
      component: MainContent
    },

    {
      path: '/news/:id_article',
      name: 'news_article',
```

```
      component: ArticlePage
    },

    {
      path: '/news',
      name: 'news',
      component: NewsPage
    },

    {
      path: '/gallery',
      name: 'gallery',
      component: GalleryPage,
      children:
      [{
        path: '/gallery/:id_photo',
        name: 'galleryPhoto',
        component: GalleryBigPhoto,
      }]
    }

  ]
})
```