Форма № 24

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій
(повне найменування інституту, факультету)

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)
бакалавр
(ступінь вищої освіти)

на тему РОЗРОБЛЕННЯ ВЕБСАЙТУ ДЛЯ ІНФОРМАЦІЙНОЇ ПІДТРИМКИ ВПРОВАДЖЕННЯ ДУАЛЬНОЇ ФОРМИ ОСВІТИ
WEBSITE DEVELOPMENT FOR INFORMATIONAL SUPPORT OF DUAL EDUCATION

Виконав: студент(ка) 4 курсу, групи КНТ-228
Спеціальності 122 Комп'ютерні науки
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Комп'ютерні науки

Перетятко В.В.
(прізвище та ініціали)
Керівник Каплієнко Т.І.
(прізвище та ініціали)
Рецензент Шитікова О.В.
(прізвище та ініціали)

2022

Форма № 25

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(повне найменування закладу вищої освіти)

Інститут, факультет ФКНТ
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
Спеціальність 122 Комп'ютерні науки
(код і найменування)
Освітня програма (спеціалізація) Комп'ютерні науки
(назва освітньої програми (спеціалізації)

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
_____С.О. Субботін
"____" _____2022 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Перетятко Валерії Валеріївни
(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Розроблення вебсайту для інформаційної підтримки впровадження дуальної форми освіти. Website development for informational support of dual education

керівник проєкту (роботи) Каплієнко Тетяна Ігорівна, к.т.н., доцент ,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від " 27 " квітня 2022 року № 102
2. Строк подання студентом проєкту (роботи) 07 червня 2022 року
3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Реферат. Перелік скорочень та умовних познак. Вступ. 1. Аналіз проблеми та постановка завдань роботи. 2. Засоби веброзробки. 3. Розробка сайту. 4. Експлуатація, тестування та експериментальне дослідження. Додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	Каплієнко Т.І., доцент		
Нормоконтроль	Камінська Ж.К., асистент		

7. Дата видачі завдання « 04 » квітня 2022 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій	2 тиждень	Розділ 2
	розробки.		
4	Розробка архітектури програми.	2 тиждень	Розділ 3
5	Розробка програми.	3-4 тижні	Розділ 3, 4
6	Тестування та експериментальне дослідження	5 тиждень	Розділ 4
	програмного забезпечення.		
7	Оформлення пояснювальної записки та документів	6 тиждень	Додатки
	до неї.		
8	Нормоконтроль та рецензування.	7 тиждень	
9	Захист роботи.	8 тиждень	

Студент(ка) _____ Перетятко В.В.
(підпис) (прізвище та ініціали)

Керівник проєкту (роботи) _____ Каплієнко Т.І.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до випускної кваліфікаційної роботи бакалавра: 95 с., 6 табл., 34 рис., 2 дод., 20 джерел.

ДУАЛЬНА ОСВІТА, МОВА PYTHON, ФРЕЙМВОРК DJANGO, MVC, MYSQL, ORM.

Об’єкт дослідження – інформаційна підтримка дуальної форми освіти у Національному університеті «Запорізька політехніка» (НУ «Запорізька політехніка»).

Предмет дослідження – розроблення вебсайту для інформаційної підтримки впровадження дуальної форми освіти у НУ «Запорізька політехніка».

Мета роботи – створення інформаційного сайту для поширення інформації про дуальну освіту.

Матеріали, методи та технічні засоби: основні інструменти для веброзробки HTML/CSS, фреймворк Bootstrap, мова програмування Python, фреймворк Django, реляційна база даних SQL та система управління базами даних MySQL, інтегроване середовище розробки Visual Studio Code та редактор Sublime Text.

Результати. Розроблено вебсайт, який надає можливість керувати моделями новин та даними, що містить сайт.

Висновки. Проведено аналіз існуючих аналогів та визначено основні завдання на роботу. Створено сайт для публікації та зберігання інформації про дуальну освіту.

Галузь використання. Проєкт може бути використано для цілей університету, а саме для розповсюдження інформації та новин університету, що стосуються дуальної освіти.

ABSTRACT

Explanatory note to the final qualifying work of the bachelor: 95 p., 6 tables, 34 figures, 2 appendixes, 20 sources.

DUAL EDUCATION, PYTHON LANGUAGE, DJANGO FRAMEWORK, MVC, MYSQL, ORM.

The object of the work is the informational support of dual education.

The subject of the work is website development for informational support of dual education.

The purpose of the work is creating website for sharing information about the dual education

Materials, methods and tools: basic HTML/CSS webdevelopment tools, Bootstrap framework, Python programming language, Django framework, SQL relational database and MySQL database management system, Visual Studio Code integrated development environment and Sublime Text text editor.

Results. A developed website that allows to control models of news and contained data in the site.

Conclusions. The analysis of existing analogues has been done and the main tasks for work were defined. A website has been created for publishing and storing information about dual education.

The field of use. The project can be used for the purposes of the university, namely to broadcast information and news of the university related to dual education.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

HTTP	– HyperText Transfer Protocol;
IP	– Internet Protocol;
MVC	– Model-view-controller;
ORM	– Object-relational Mapping;
TCP	– Transmission Control Protocol;
URL	– Uniform Resourse Locator;
WSGI	– Web Server Gateway Interface.
WWW	– World Wide Web;
ЗВО	– Заклад вищої освіти;
НУ «Запорізька політехніка»	– Національний університет «Запорізька політехніка»;
СУБД	– Система управління базами даних;
ТДАТУ	– Таврійський державний агротехнологічний університет імені Дмитра Моторного;
УкрДУЗТ	– Український державний університет залізничного транспорту;
УКД	– Університет Короля Данила;

ВСТУП

Для виховання творчих, конкурентоспроможних та сучасних фахівців першочерговим завданням навчальних закладів є не лише вирішення проблеми інтеграції науки та освіти, а й вирішення проблеми інтеграції з виробництвом та з конкретними суб'єктами господарювання. Тому одним із завдань університету є поширення інформації про переваги дуальної освіти.

Дуальна освіта – форма навчання, яка надає можливість студенту навчатися в закладі вищої освіти (ЗВО), вивчаючи теорію та одночасно працювати, закріплюючи отримані знання та опановуючи практичні навички на робочому місці.

Дуальна освіта означає «подвійна», це коли процес навчання складається близько із 50% практики. Така форма навчання успішно зарекомендувала себе і функціонує вже понад 50 років у Німеччині. Переваги такої системи навчання підготовки спеціалістів полягають у тому, що отримані теоретичні знання можна відразу застосовувати на практиці. В результаті такого навчання молодий спеціаліст має знати та вміти робити саме те, що від нього вимагає роботодавець.

Зазвичай кваліфікація спеціаліста, який отримав «класичну» вищу освіту, не відповідає сучасним вимогам бізнесу та роботодавців. Виникає проблема перенавчання випускників вже на підприємствах, а отже витрачаються додаткові ресурси. Тому зараз роботодавець зацікавлений у тому щоб фахівець після випуску зміг приступити до самостійної роботи якнайшвидше.

Оскільки експеримент по дуальній освіті в Україні впровадили наприкінці 2019 року, то склалася така ситуація, що на сьогодні студенти мало ознайомлені, а підприємства мають низький рівень поінформованості про таку форму навчання. Отже актуальність даної теми полягає в поширенні інформації про впровадження та можливості використання дуальної форми здобуття освіти у НУ «Запорізька політехніка», шляхом створення та апробації вебсайту, що надає інформаційну підтримку саме дуальній формі.

Об’єктом дослідження є інформаційна підтримка дуальної форми освіти у НУ «Запорізька політехніка».

Предметом дослідження є розроблення вебсайту для інформаційної підтримки впровадження дуальної форми освіти у НУ «Запорізька політехніка».

Метою роботи є аналіз існуючих методів та програмна розробка вебсайту для інформаційної підтримки впровадження і популяризації дуальної освіти, ознайомлення з такою формою здобуття освіти та її поширення для молодих спеціалістів та роботодавців, підвищення конкурентоздатності випускника з дипломом на сучасному ринку праці.

1 АНАЛІЗ ПРОБЛЕМИ ТА ПОСТАНОВКА ЗАВДАНЬ РОБОТИ

1.1 Аналіз предметної області

Професійний рівень підготовки кадрів впливає на економічну конкуренцію та якість життя населення. Саме такому рівню підготовки сприяє нова форма освіти – дуальна. Вона пропонує навчання в університеті і на робочому місці, щоб студенти одразу застосовували теорію на практиці.

Очевидно, що наука та освіта самі по собі не є факторами економічного розвитку. Наука породжує нові знання, а освіта готує здібних фахівців необхідних для економіки та створює сприятливі умови для покращення добробуту, при умові що вони реалізуються у процесі виробництва.

Сьогодні вирішенню проблеми інтеграції освіти, науки і виробництва, дуальної освіти присвячено чимало праць вітчизняних і зарубіжних вчених: Абашкіна Н.В. [1], Амеліна С.М. [2], Мілл У. [3], Грін Л., Карлюк С., Панфілов Ю. [4], Кримчак Л. Ю. [5], Бегма О. [6], Яковенко К. В. [7], Кулалаєва Н. В. [8], чиї праці розкривають значення, відношення та проблеми дуальної освіти.

Слід зазначити, що вітчизняними вченими не в повній мірі розглянуті теоретико-методологічні засади формування дуальної освітньої інноваційної моделі у такому вигляді в навчальних закладах та підготовці кадрів в умовах ринкової економіки, особливо у звичайних ЗВО країни. Тому найактуальнішим завданням організації сучасного навчального процесу є впровадження дуальної системи освіти як практичної професійної освіти. Запровадження цієї системи відкриває нові перспективи для підвищення ефективності професійної освіти.

Дуальна освіта вбачає розробку та підписання «Договору про виробниче навчання» між вступником та власником бізнесу або керівником закладу. У цьому договорі відображаються мета, спосіб, зміст, початок та тривалість навчання, навчальна діяльність поза підприємством, вид роботи після навчання, тривалість робочих днів, випробувальний термін, розмір та тривалість оплати праці, тривалість відпустки, а також умови розірвання контракту. Контракт повинен гарантувати систематичну та розумну підготовку вибраних

спеціальностей, захист студентів від протиправної поведінки з боку власників бізнесу [9]. Контроль, оцінка та присвоєння кваліфікації робітників для галузевого навчання здійснюються торгово-промисловими палатами, які реєструють галузевий контракт на навчання та визначають, чи є у компанії персонал та умови для забезпечення належного навчання.

Модель дуального навчання відображена на рисунку 1.1.

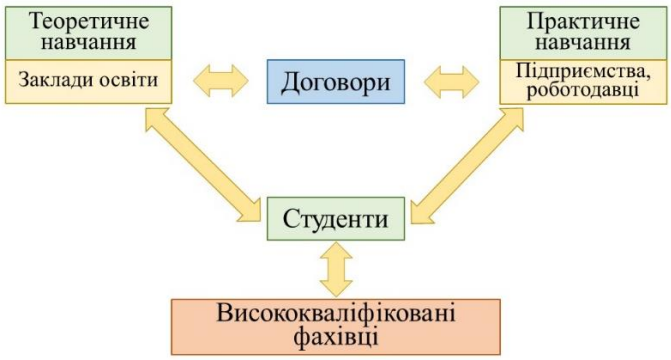


Рисунок 1.1 – Модель дуального навчання

У таблиці 1.1 наведено основні переваги щодо впровадження дуальної форми здобуття освіти [10].

Таблиця 1.1 – Переваги дуальної форми здобуття освіти

Для роботодавців	Для здобувачів освіти	Для закладів освіти
Підготовка кадрів для підприємства	Ознайомлення з роботою та її особливостями в компанії під керівництвом наставників	Коригування навчальної програми, використовуючи нові інформаційні технології
Запобігання витрат зайвих ресурсів та часу, який потрібен для адаптації молодого спеціаліста	Виконання роботи, застосовуючи реальні дані та можливість використовувати сучасне обладнання	Підвищення кваліфікації працівників закладів освіти
Залучення найбільш вмотивованих та талановитих спеціалістів до роботи на підприємстві	Ознайомлення з роботою різних відділів у результаті обміну	Підвищення іміджу закладу освіти та привернення уваги абітурієнтів
Сприяння у пошуку кваліфікованих спеціалістів у регіонах	Отримання заробітної плати вже при проходженні навчання	
	Користування попитом на ринку праці в регіоні	

Дуальна освіта допомагає запроваджувати нові робочі місця, економити гроші державної казни, видозмінювати ринок праці, а при необхідності, також державні послуги, допомагає впровадженню реформ в освіту і, головне, створює умови, мотивує молодь навчатися і працювати не найманими працівниками за кордоном, а затребуваним фахівцем у власній державі [11].

Дуальна форма здобуття освіти зараз існує у багатьох країнах Європейського Союзу. Їх досвід свідчить про ефективність даної форми освіти. Для України така форма подвійного навчання є відносно новою. На сьогодні дуальна форма – це майбутнє, співпраця закладу освіти з бізнесом, це успіх у розвитку професійного навчання та підвищення рівня економіки країни.

Враховуючи актуальність даної форми навчання, можна сказати, що основним джерелом інформації про можливість навчатися дуально є заклад освіти.

На сьогодні одним із найпоширеніших джерел інформації є інтернет. Інформаційні сайти надають змогу відвідувачам зручно та швидко отримати всю необхідну інформацію з того чи іншого питання. Отже зібрана та розміщена вичерпна інформація у онлайн-форматі, специфічна саме для НУ «Запорізька політехніка», полегшить пошук та ознайомлення відвідувачів.

Під час написання даної роботи, було проведено аналіз існуючих інформаційних сайтів закладів освіти щодо висвітлення та впровадження форми дуального навчання. Лише кілька закладів освіти мають окремі сторінки на сайтах та в соціальних мережах із вичерпною інформацією про цю форму навчання.

НУ «Запорізька політехніка» долучається до цієї форми освіти, тому виникає потреба у своєчасному та повному інформуванні вступників, студентів та роботодавців. Всі зацікавлені сторони повинні мати можливість дізнатися умови та деталі такої форми навчання та співпраці, а також іншу нормативну та супутню інформацію.

Зважаючи на всі перелічені факти, виникає необхідність у розробці вебсайту для інформаційної підтримки впровадження дуальної форми освіти у НУ «Запорізька політехніка».

Головні цілі сайту:

- ознайомлення з нормативною базою про дуальну форму освіти;
- ознайомлення з порядком зарахування здобувачів;
- висвітлення новин щодо дуальної освіти у вищі та країні;

– інформування роботодавців та партнерів;

– зворотній зв'язок.

Такий сайт може стати ефективним інструментом для залучення в університет нових вступників, партнерів та роботодавців. Але тільки професійна та своєчасна розробка та підтримка сайту інформаційного типу може забезпечити необхідний високий рівень, якість і надійність результату.

Загалом впровадження дуального навчання в систему підготовки комерційних кадрів та паралельне інформування суспільства про дану форму навчання, сприятиме переходу на якісно новий рівень підготовки висококваліфікованих та конкурентоспроможних спеціалістів.

1.2 Аналіз аналогів

Зі збільшенням онлайн-активності молоді та просуванням різноманітних гаджетів все більш актуальним стає розміщення інформації на сайтах і в соціальних мережах.

Вебсайт – це не тільки обличчя організації, а й тонкий маркетинговий інструмент, який може зацікавити цільову аудиторію, зміцнити імідж, залучити потенційних здобувачів і партнерів. Вебсайт університету може бути використаний в різних цілях: для реклами, інформування громадськості про університет, ознайомлення з новинами ЗВО, впізнаваності бренду, зв’язків із громадськістю, офіційних повідомлень. Шлях до успіху в цій сфері – це добре розроблений вебсайт.

Кожен повністю функціональний сайт є автоматизованою системою інформаційних технологій, призначеною для використання відвідувачами інтернету.

Під час написання даної роботи, виникла необхідність в порівнянні сайтів, присвячених дуальній освіті, різних ЗВО нашої держави. При виконанні даної задачі було виявлено, що реалізацій у вигляді окремих вебсайтів не

знайдено, але існують вебсторінки університетів з інформацією про дуальну освіту. Таким чином був проведений аналіз не вебсайтів, а існуючих вебсторінок.

Для аналізу були обрані вебсторінки з інформацією про дуальну освіту трьох університетів: Таврійський державний агротехнологічний університет імені Дмитра Моторного (ТДАТУ), Університет Короля Данила (УКД), Український державний університет залізничного транспорту (УкрДУЗТ).

Проведемо аналіз сайту ТДАТУ, на якому розташована інформаційна сторінка про дуальну освіту. Інтерфейс сайту зрозумілий, що дає можливість легко орієнтуватись серед інформації, можна швидко знайти вкладку про дуальну освіту. Відповідна сторінка сайту університету містить інформацію з переліком кафедр, що ведуть дуальне навчання, але відсутній наглядний матеріал (рисунки, таблиці, схеми). Конкретно за кожним посиланням можна переглянути досвід, звітність кафедри та договори з підприємствами.

Сторінка не містить форми зворотного зв’язку для запитань та уточнення інформації, що ускладнює розуміння щодо форми навчання. Сайт підтримує адаптивну верстку, тому має добре відображення на мобільних пристроях.

Сторінка сайту відображена на рисунку 1.2 [12].

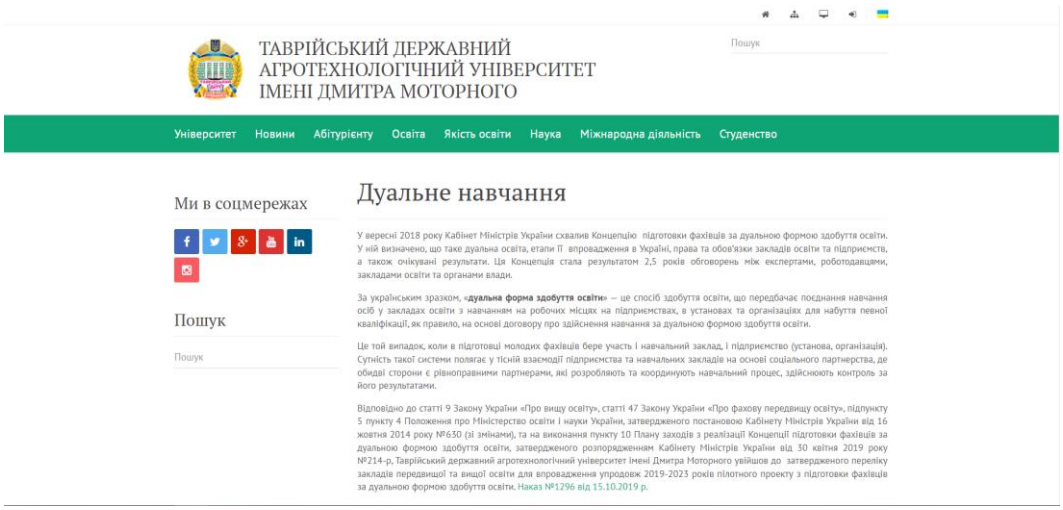


Рисунок 1.2 – Сторінка «Дуальне навчання» сайту ТДАТУ [12]

Наступним аналогом для порівняння є сторінка сайту УКД. Інтерфейс сайту зручний, дозволяє віднайти потрібні матеріали. Дизайн сайту привабливий та лаконічний, має власний стиль. Інформація на сторінці відображена у вигляді схем та ілюстрацій, що полегшує розуміння теми. Немає переліку кафедр або спеціальностей, за якими надається дуальна освіта, але є загальна форма зворотного зв'язку. Сторінка не містить законодавчої бази щодо даної форми навчання. Мобільна версія сайту загалом задовільна. Сторінка сайту відображена на рисунку 1.3 [13].

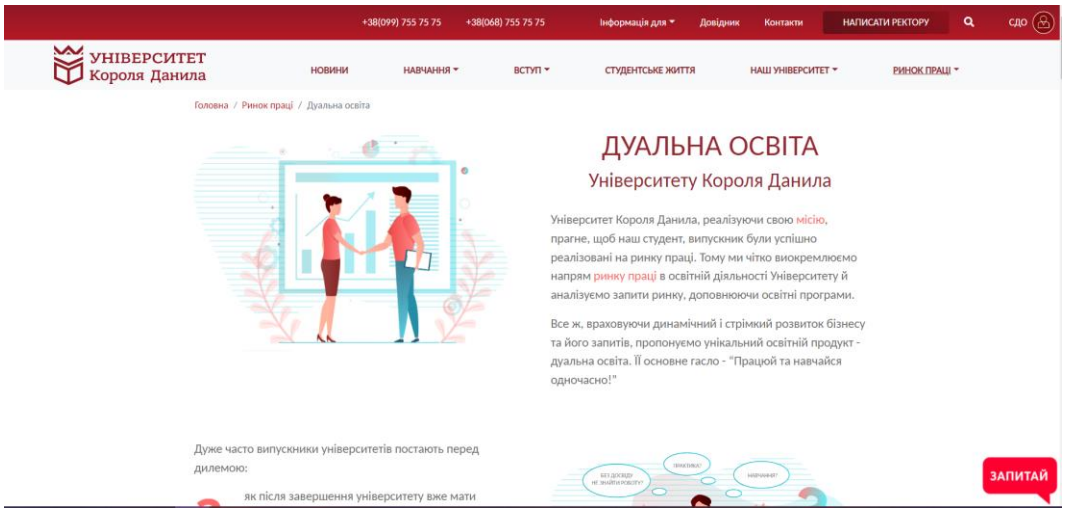


Рисунок 1.3 – Сторінка «Дуальна освіта» сайту УКД [13]

Розглянемо також сторінку УкрДУЗТ. Зручність інтерфейсу надає змогу ознайомитись з потрібною інформацією. Візуального відображення теми не надано. Немає списку спеціальностей, за якими впроваджено подвійне навчання. Сторінка має посилання на нормативні документи щодо надання дуальної освіти, але не містить форми зворотного зв’язку. Доступна мобільна версія сайту. Сторінка сайту відображена на рисунку 1.4 [14].

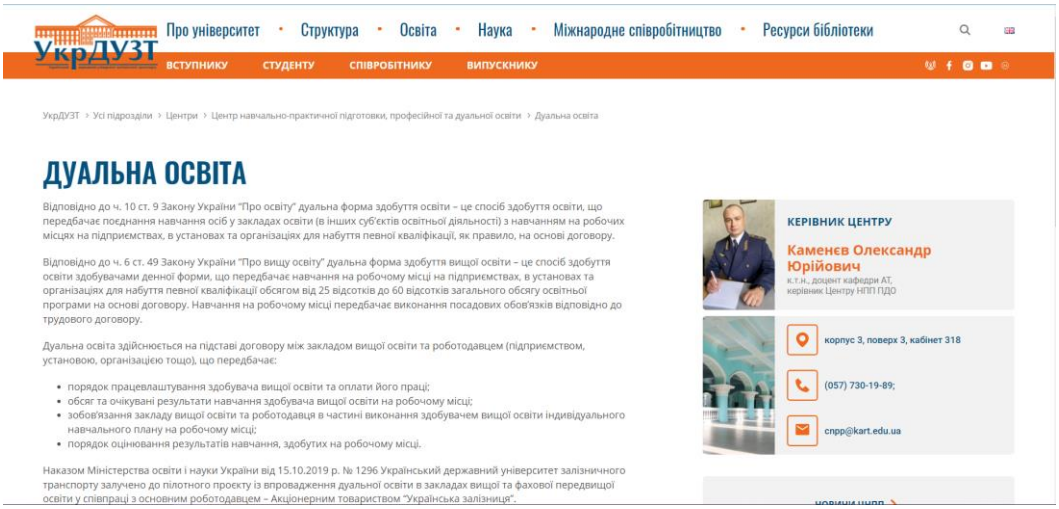


Рисунок 1.4 – Сторінка «Дуальна освіта» сайту УкрДУЗТ [14]

При розгляді наведених сторінок були виявлені переваги та недоліки, які зведені в порівняльну таблицю 1.2.

Таблиця 1.2 – Порівняльна характеристика аналогів

Критерії	ЗВО		
	ТДАТУ	УКД	Укр ДУЗТ
1	2	3	4
Зрозумілий інтерфейс	+	+	+
Загальна інформація про дуальну освіту	+	+/-	+
Наглядність подання інформації	-	+	-
Перелік спеціальностей	+	-	-

Продовження таблиці 1.2

1	2	3	4
Звітність про результати освіти	+	-	-
Нормативно-правова база	+	-	+
Форма зворотного зв'язку	-	+	-
Мобільна версія	+	+/-	+/-

Оскільки НУ «Запорізька політехніка» має велику кількість спеціальностей та співпрацює з багатьма підприємствами, які впровадили дуальну освіту, було прийнято рішення створити окремий вебсайт, щоб якісно та своєчасно донести інформацію студентам, роботодавцям для більш зручного перегляду.

1.3 Постановка задач розробки

Оскільки завданням роботи є розробка вебсайту, призначеного для інформаційної підтримки впровадження дуальної форми освіти, то можна коротко описати структуру сайту (наявні сторінки) наступним чином:

- головна;
- нормативна база (НУ «Запорізька політехніка» та Міністерство освіти і науки України);
- партнери (роботодавці);
- порядок зарахування здобувачів;
- новини;
- досвід інших ЗВО;

- зворотній зв’язок;
- контакти.

Слід зазначити, що в процесі розробки структура може змінюватися.

Під час роботи над створенням інформаційного сайту, необхідно буде виконати наступні завдання:

- зробити аналіз інформаційного забезпечення;
- створення організаційної та функціональної структури;
- зробити аналіз логічної структури проєкту;
- аналіз та створення структури бази даних;
- підбір шаблону;
- розробка форми зворотного зв’язку;
- проведення тестування та коригування помилок.

1.4 Висновок

В ході роботи було проаналізовано предметну область. Зроблено висновок, що в мережі немає аналогів у вигляді вебсайтів, але багато реалізацій вебсторінок. При проведенні аналізу аналогів виявлено основні переваги та недоліки інформаційних сторінок. На основі проведеного аналізу були намічені основні задачі розробки інформаційного сайту супроводження дуальної форми освіти в НУ «Запорізька політехніка».

2 ЗАСОБИ ВЕБРОЗРОБКИ

2.1 Базові інтернет-технології

На сьогодні інтернет є глобальною комп'ютерною мережею, що охоплює весь світ. Основне, чим виділяється мережа інтернет - це її найважливіші протоколи - Transmission Control Protocol (TCP) і Internet Protocol (IP).

Функціонування інтернету засноване на використанні протоколу зв’язку TCP/IP. Протокол складається з набору правил, що зобов’язані дотримуватися всі компанії, аби існувало забезпечення сумісності апаратних та програмних забезпечень. Протокол вирішує, як одна програма взаємодіє з іншою і як різні частини повного пакета контролюють передачу інформації. Розробка стандарту протоколу враховує можливі непередбачені обставини. Протокол містить правила обробки помилок.

Основні мережеві послуги інтернету:

– мережева система передачі інформації (FTP, Gopher);

– інформаційно-пошукові системи в мережі;

– послуги зв'язку (електронна пошта, IRC, Telnet, Usenet);

– інформаційна система мультимедії (World Wide Web).

World Wide Web (WWW) – найпопулярніший тип інформаційної служби інтернету. Це розподілена система, яка забезпечує доступ до взаємопов’язаних документів, що знаходяться на різних підключених до інтернету комп’ютерах. Сервіс базується на використанні архітектури клієнт-сервер.

Структура WWW має три головні частини:

– сервери, програми, що забезпечують адміністративні функції;

– домашня сторінка, розміщена на сервері та є окремою програмою, цій сторінці надається унікальна адреса – Uniform Resource Locator (URL);

– браузер, клієнтська частина програмного забезпечення WWW, що дає доступ користувачеві до необхідних ресурсів на сервері за його URL-адресою за допомогою графічного інтерфейсу.

На сьогодні для роботи в мережі існує безліч програм, але найпоширенішим залишається браузер або веббраузер. Слово має походження від англійських слів «web» – павутина (колишня назва мережі) і «browse» – переглядання, читання.

Однією з цілей проєкту WWW є забезпечення зручного способу доступу до документів, розміщених на віддалених комп'ютерах. Для передачі вебдокументів у стандартній формі гіпертексту було розроблено новий протокол передачі HyperText Transfer Protocol (HTTP). HTTP – простий протокол зв'язку, який враховує, що переданий документ містить гіпертекстові посилання, тобто містить інформацію про адреси, на які можуть надсилатися подальші запити. Цей протокол використовується для посилань на документи HTML, не дивлячись на URL-адреси. До складу URL-адреси входить три елементи: протокол доступу, ім'я серверної програми та шлях до документа (protocol://server.directory/filename).

Вся інформація, що зберігається в інтернеті, представлена у вигляді файлів на серверах з постійним підключенням до ліній зв'язку, які мають доступ до них. Усі файли є інтернет-ресурсами. Існують файли різного типу: текстового, гіпертекстового, електроного листа, графічного, звукового тощо. Усі файли зібрані у певну структуру, з якої їх можна отримати за допомогою протоколу доступу. Кожен інтернет-ресурс має свій протокол доступу. Наприклад, архіви файлів використовують протокол ftp, ресурси Gopher – gopher, ресурси WWW – http, групи новин – новини або nntp.

Мережа WWW утворює мільйон вебсерверів, розміщених по усьому світу. Вебсервер є програмою, що запускається на під'єднанному до мережі комп'ютеру і передає дані по протоколу HTTP.

Вебтехнологія – це поєднані методи та програмно-технічних засоби, які застосовують для оперативної роботи з вебресурсами у вебпросторі. Вебтехнології пов'язані з методами і засобами створення вебсторінок, що підтримують мультимедіа (текст, графіка, звук, анімація й відео). Проте в

сучасних умовах інтернет є потужним засобом комунікації, інтеграції, пошуку вебресурсів, надання різноманітних сервісів.

На сьогодні поняття вебтехнологій розширено. Це комплекс технічних, комунікаційних, програмних методів розв'язання завдань організації спільної діяльності користувачів із застосуванням мережі інтернет [16].

Вебсайт – це окремий вебресурс інтернету з зареєстрованим власником та унікальним ім'ям (домен), постійною адресою. Фізично цей ресурс встановлюється на інтернет-сервері, що підключений до мережі інтернет та може переглядатися з будь-якого веббраузера. До складу вебсайту входить головна й декілька підпорядкованих вебсторінок, також вебсторінки сайту мають єдину систему навігації, дизайн, загальну інформаційну спрямованість, що утворюють єдине ціле. Сторінки сайту пов'язані гіперпосиланнями та мають спільну тему, автора, компанію тощо [15, С.110].

Майже всі служби використовують технологію «клієнт-сервер», де кожна служба повинна мати сервер, а клієнти повинні використовувати спеціальне клієнтське програмне забезпечення для доступу до сервера.

Існує два типи комп'ютерів, що підключені до інтернету: сервери та клієнти. Комп'ютер, який надає послуги іншим комп'ютерам, називається сервером, а комп'ютер, який отримує послуги, називається клієнтом. Клієнт-серверне обчислення — процес при якому одна програма діє як клієнт, а інша — як сервер. Архітектура клієнт-сервер є основою розподілених обчислень [15, С.25].

2.1.1 Вебсервер

Поняття «вебсервер» може стосуватися як апаратної частини, так і програмного забезпечення. Або навіть до обох частин, що працюють разом.

З фізичної точки зору, «вебсервер» - це комп'ютер, який зберігає файли сайту (HTML-документи, CSS-стилі, JavaScript-файли, картинки та інші) і доставляє їх на пристрій кінцевого користувача (браузер і т.д.). Він підключений

до мережі інтернет і може бути доступний через доменне ім'я (наприклад mozilla.org).

З точки зору програмного забезпечення, вебсервер включає кілька компонентів, які контролюють доступ вебкористувачів до розміщених на сервері файлів, як мінімум - це HTTP-сервер. HTTP-сервер - це частина програмного забезпечення, яка розуміє URL-адреси (вебадреси) і HTTP (протокол, який браузер використовує для перегляду вебсторінок)[17].

Є декілька провідних вебсерверів - Apache, IIS, lighttpd, Sun Java System і Jigsaw . Крім цих вебсерверів, на ринку є інші вебсервери, але вони дуже дорогі. Основними з них є iPlanet Netscape, веблогіка Bea та IBM WebSphere**.**

2.1.2 Серверне та клієнтське вебпрограмування

Серверне вебпрограмування — це технологія, що дозволяє запускати на вебсервері програми, що мають можливість отримувати дані від відвідувачів сайтів, які підтримуються цим вебсервером, і в свою чергу видавати їм оброблені дані у вигляді вебсторінок або інших файлів.

Інформація про користувачів та дані сайту зберігаються у базі даних, до якої можна звернутися через сервер. Детальнішу схему взаємодії сервера (server-side) та браузера (client-side) можна переглянути на рисунку 2.1.2 [18].

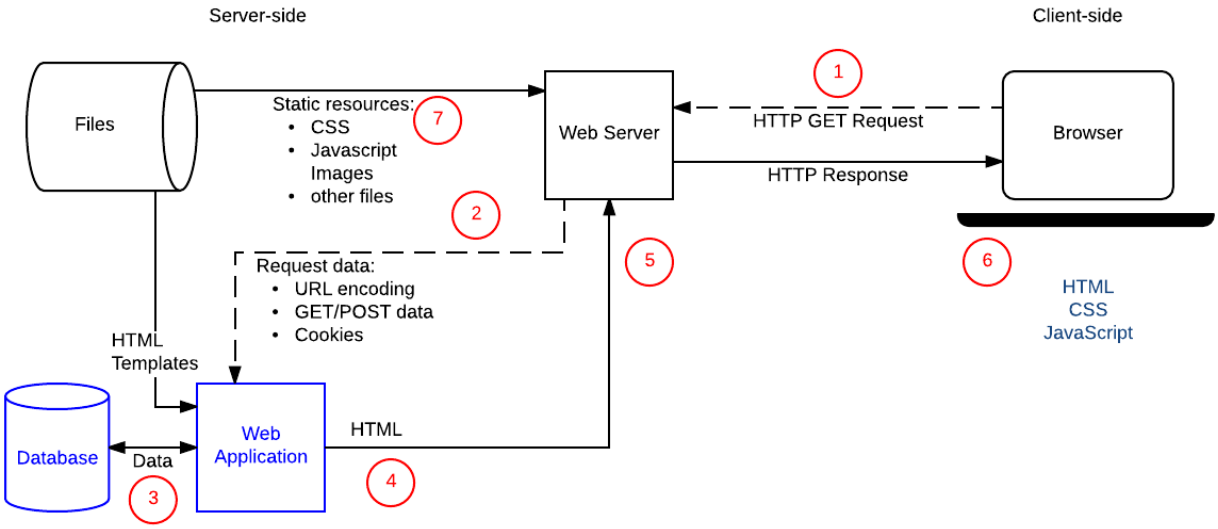


Рисунок 2.1.2 – Модель взаємодії сервера та браузера [18]

Код, який виконується в браузері, відомий як код клієнтської частини, перш за все пов'язаний з покращенням зовнішнього вигляду та поведінки відображуваної вебсторінки. Це стосується вибору і стилізації компонентів інтерфейсу користувача, створення макетів, навігації, перевірки форм. Програмування вебсайту на стороні сервера в основному включає вибір вмісту, який повертається браузеру у відповідь на запити. Код на стороні сервера обробляє такі завдання, як перевірка надісланих даних та запитів, використання баз даних для зберігання та вилучення даних, відправлення правильних даних клієнту за необхідності.

Програмування серверної частини дозволяє зберігати інформацію в базі даних і динамічно створювати та повертати HTML та інші типи файлів (наприклад, PDF, зображення тощо). Також є можливість просто повернути дані (JSON, XML, і т. д.) для відображення, використовуючи відповідний фреймворк клієнтської частини (це зменшує навантаження процесора на сервері та кількість даних, що передаються)[19].

2.2. Засоби розробки інтерфейсу

Для того щоб розробити коректно і надійно функціонуючий вебсайт необхідно проаналізувати та обрати правильні інструменти.

Для створення інтерфейсу було використано технології HTML + CSS.

Основою будь-якої вебсторінки є розмітка. Технології розмітки HTML визначають структуру і зміст сторінки. Зовнішній вигляд та стиль створюється за допомогою таблиць стилю CSS.

HTML (Hypertext Markup Language) — це мова, яка описує структуру сторінок документа, дозволяє робити форматування звичайного тексту в абзаци, списки, заголовки тощо, щоб створювати посилання на інші сторінки. У цій мові інструкції з форматування, так звані теги, вбудовуються у розділи документа, що містять інформацію. Теги передають браузерам вимоги до форматування і представлення інформації на екрані [20].

HTML має власний синтаксис, що відповідає стандартам W3C HTML 4.01 та W3C HTML 5.

HTML надає такі можливості:

– створення структури документу (заголовки, блоки абзаців, таблиці, списки, цитати);

– підключення до документу гіпертекстових посилань;

– створення інтерактивних форм;

– приєднання графічних зображень, звукових доріжок, відео до тексту.

CSS (Cascading Style Sheets) — це технологія, що описує зовнішній вигляд документа, за допомогою таких засобів, як HTML, XML і XHTML.

За допомогою CSS можна задати шрифти, кольори, параметри зображення сторінок та положення елементів сторінки. CSS відокремлює опис зовнішнього виду від логічної структури вебсторінки, яка реалізується за допомогою HTML. Цей поділ збільшує доступність документа і надає більш гнучке управління відображенням.

2.3 Аналіз мов програмування

Основним інструментом для створення вебсайтів є мова програмування, яка надає зручні функції для розробки методів зберігання необхідної інформації на сервері і реалізації клієнт-серверного обміну з користувачами.

Проведемо аналіз наступних найпоширеніших мов вебпрограмування: PHP, Python, Ruby.

PHP (hypertext preprocessor) – скриптова мова програмування загального призначення. PHP використовується для написання сценаріїв, які будуть виконуватись на стороні сервера. Ця мова програмування може виконувати різні операції з HTML-документами, перенаправляти на інші сторінки вебпрограми, керувати автентифікацією користувачів, обробляти файли, що завантажуються на сервер і надавати доступ до бази даних.

Основні переваги PHP:

– досить легкий в освоєнні;

– вільно поширюваний;

– дуже популярний, підтримується великою спільнотою програмістів;

– включає підтримку основних можливостей об'єктно-орієнтованого програмування;

– має велику кількість розширень та бібліотек;

– дозволяє припускатися помилок у коді, програма у коді якої є проблеми, буде виконуватися поки не дійде до ділянки з помилкою.

Основні недоліки PHP:

– неможливість асинхронної та багатопотокової роботи програми;

– низька якість коду;

– неузгодженість синтаксису функцій;

– вебзастосунки, які написані на PHP, часто мають проблеми з безпекою.

Python – це об’єктно-орієнтована мова програмування високого рівня. Синтаксис цієї мови простий та зручний, стандартні бібліотеки мають велику

кількість корисних функцій. Python використовується для рішення різних завдань, у тому числі для створення вебсайтів. За допомогою Python можливо динамічно генерувати HTML-код з боку серверу.

Python підтримує кілька парадигм програмування, зокрема об'єктно-орієнтоване, структурне, функціональне, імперативне та аспектно-орієнтоване. Серед основних архітектурних рис можна виділити такі: автоматичне керування пам'яттю, динамічна типізація, підтримка багатопоточних обчислень, повна інтроспекція, спосіб обробки виключень та зручні високорівневі структури даних. Код Python організовується в функції та класи, які можна об'єднати в модулі (які згодом можуть бути об'єднані у пакети).

Основні переваги Python:

– зручний і легко зрозумілий програмний код;

– зручне та швидке створення моделей системи;

– велика кількість документації, а також зростаюча спільнота програмістів, що використовує Python;

– об'єктно-орієнтована мова;

– безліч різноманітних бібліотек;

– різноманітність фреймворків для створення вебзастосунків.

Основні недоліки Python:

– низька швидкість, виконування коду рядково призводить до втрати часу;

– динамічна типізація, можна не оголошувати тип змінної, що іноді призводить до виникнення помилок.

Ruby – об'єктно-орієнтована мова програмування високого рівня. Мова має незалежну від операційної системи реалізацію багатопоточності, а також строгою динамічною типізацією. За допомогою Ruby можна швидко і просто програмувати, тому що немає необхідності оголошувати змінні та їх типізувати. Є хороша підтримка операцій з рядками та регулярними виразами.

Основні переваги Ruby:

– розширення можливостей мови за допомогою множини бібліотек;
– інтегрованість із різними системи управління базами даних (СУБД);
– простий та зручний інтерфейс;
– має засоби для роботи з масивами та рядками;
– наявність фреймворків для створення вебзастосунків.

Основні недоліки Ruby:

– складність у освоєнні;
– відсутність необхідної кількості документації;
– менш продуктивний, у порівнянні з Python та PHP.

При аналізі наведених мов були виявлені характеристики, які зведені в порівняльну таблицю 2.1.

Таблиця 2.1 – Порівняльна характеристика мов вебпрограмування

Характеристика	PHP	Python	Ruby
Рік створення	1994	1991	1995
Динамічна типізація	+	+	+
Об’єктно-орієнтованість	+	+	+
Можливість компілювання	+	+	+
Наявність бібліотек та фреймворків	+	+	+
Парадигми програмування:	Мульти-парадигмальний	Мульти-парадигмальний	Мульти-парадигмальний

Таким чином, взявши до уваги проаналізовані особливості та переваги, мовою розробки було обрано Python. Вона проста у вивченні, а також має

гарну інтеграцію з СУБД. Недоліки цієї мови не є критичними оскільки швидкість роботи є задовільною для вебсайту.

2.4 Фреймворк Django та його MVC архітектура

Для розробки вебсайту було використано фреймворк Django. Цей фреймворк є безкоштовним і з відкритим доступом для створення вебзастосунків, написаний мовою Python, що використовує шаблон проєктування Model-view-controller (MVC).

Для двосторонньої взаємодії потрібні динамічні вебсторінки, так як сторінка HTML не вміє реагувати на запити користувача, бо вона є статичною. MVC – ключ до розуміння розробки інтерактивних і динамічних вебзастосунків.

MVC («модель - уявлення - контролер») – схема, що дозволяє застосовувати декілька шаблонів проєктування. У результаті чого модель застосунку, інтерфейс користувача і взаємодія з користувачем розділені на три окремих компоненти, що мають мінімальний вплив один на одного при модифікації. Тобто один блок несе в собі дані застосунку, другий визначає зовнішній вигляд, а третій контролює роботу застосунку.

Розглянемо компоненти MVC. Модель – компонент що відповідає за дані, визначає структуру застосунку, список задач та окремі завдання.

Уявлення несе у собі взаємодію з користувачем, вирішує зовнішній вигляд застосунку та яким чином його використовувати.

Уявлення вирішує такі завдання:

– приймання HTTP-запитів;
– реалізація бізнес-логіки, певну методами і властивостями;
– відправлення HTTP-відповідей на запити.

Уявлення отримує дані від моделі і надає доступ до цих даних у виді шаблонів (templates) або спочатку робить обробку даних, а потім надає до них доступ.

Контролер відповідає за зв’язок між «model» і «view». Його код визначає як сайт повинен реагувати на дії користувача. Цей компонент можна вважати мозком MVC.

Модель взаємодії між компонентами шаблону MVC зображена на рисунку 2.1.

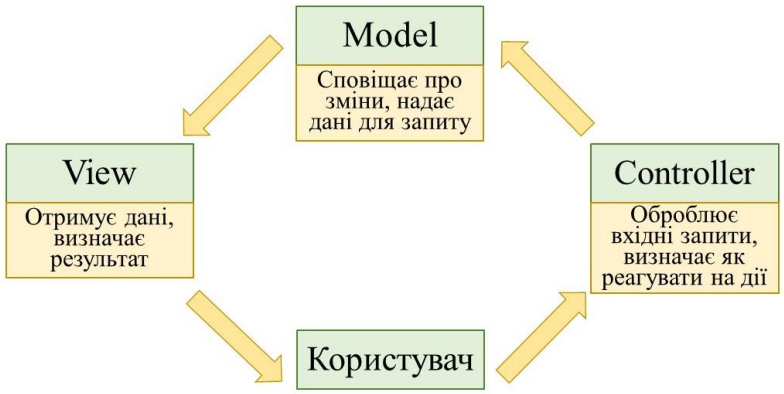


Рисунок 2.1 – Взаємодія між компонентами шаблону MVC

Django дотримується принципу «don’t repeat yourself» – не повторюй себе. Тобто при використанні Django немає потреби декілька разів переписувати той самий код, завдяки цьому скорочується час на створення сайтів. Фреймворк дозволяє створювати сайт із компонентів. Django підходить для розробки високонавантажених вебзастосунків.

Django був створений розробниками видання Lawrence-Journal World для публікування новин в інтернеті, а спеціалістами Едріаном Головатим і Саймоном Віллісоном створено та оприлюднено вебзастосунок. Завдяки цьому фреймворк пропонує ряд засобів, які допомагають у швидкій розробці вебсайтів інформаційного характеру.

Немає потреби у створенні контролерів та сторінок для адміністративної частини сайту, тому що Django має вбудований модуль для керування вмістом, який можна застосувати для будь-якого сайту, що зроблений на Django, і який дозволяє керувати одночасно багатьма сайтами на одному сервері. Модуль

адміністрації дозволяє створювати, вносити зміни та видаляти різні об’єкти, що містяться на сайті, а також має інтерфейс для управління користувачами.

Django підтримує бази даних PostgreSQL, MySQL, SQLite і Oracle.

Django надає чималу кількість вже готових базових речей, які існують майже в будь-якому сучасному вебзастосунку:

– сесії (достатньо під’єднати до застосунку потрібний модуль і при кожному запиті з’являється request.session, де можна зберігати різні дані);

– авторизація, реєстрація, генерування паролів, система прав на об’єкти моделі даних, розсилання повідомлень по email;

– кешування (щоб не робити запити до бази кожен раз, коли потрібні дані, що змінюються рідко, їх можна кешувати)

Django використовує успадкування шаблонів, що є потужним інструментом. Суть успадкування полягає у розробці основного батьківського та дочірніх шаблонів Головна сторінка складається із загальних розділів і містить опис перевизначених блоків в успадкованих шаблонах. Цей фреймворк спрощує рішення проблем із забезпеченням безпеки застосунку. Більшість вбудованих механізмів не дозволяють застосовувати небезпечні рішення.

У Django присутня реалізація шаблонів і власна мова розмітки. Шаблони представляють собою файли з HTML-кодом, за допомогою якого відображаються дані. Файли можуть мати статичний або динамічний вміст.

Враховуючи перелічені властивості, фреймворк Django є зручним та сучасним способом для реалізації вебзастосунку. Має велику кількість зручних функцій та інструментів. Містить вбудовані модулі та чималу кількість вже прописаних базових речей для створення вебсайту та керування ним.

2.5 База даних та СУБД

СУБД несе обов’язок забезпечувати реляційну модель для обробки даних. Ця модель передбачає особливий тип зв'язку між сутностями з будь-яких таблиць. Для того, щоб зберігати та обробляти дані, цей тип СУБД повинен мати

певну структуру (таблиці). Таблиця може містити різни типи даних, що записуються у стовпці. Кожен запис містить багато атрибутів (стовпців) та унікальний ключ, які зберігаються в одній таблиці - всі ці дані пов'язані один з одним, як описується в реляційній моделі.

Зробимо аналіз найбільш поширених СУБД: SQLite, MySQL, PostgreSQL.

SQLite – база даних, яка може бути зручно вбудована у застосунки. Порівняно з мережевими СУБД, вона пропонує доволі великий набір інструментів для роботи з нею, оскільки така система є файловою.. Замість портів та сокетів звернення відбуваються до файлів, що містять дані. Саме тому SQLite є швидкою та потужною завдяки технологіям обслуговуючих бібліотек.

Серед переваг цієї СУБД можна відмітити файлову структуру, де вся база даних – це один файл, тому її легко перенести на інші машини. SQLite також відрізняється високою якістю розробки та тестування..

Недоліком цієї системи є відсутність системи користувачів, що існує у більш великих СУБД. Зазвичай цю систему використовують у не надто великих застосунках.

MySQL – це розповсюджена повноцінна серверна СУБД. Система має великий функціонал, розповсюдженість та ефективно співпрацює з різними сайтами та вебзастосунками. SQL функціонал реалізований не в повній мірі, але MySQL надає широкий спектр інструментів для розробки застосунків.

MySQL досить легко встановлювати та працювати з нею, що є її перевагами. Додаткові програми пропонують легку роботу з базою даних. Різноманітний функціонал цієї системи забезпечує підтримку більшості функціоналу SQL. Ця СУБД має велику кількість функцій, що надають безпеку та підтримуються за замовчуванням. Серед властивостей MySQL є робота з великими обсягами даних та легке масштабування. Спрощена робота системи дозволяє збільшити продуктивність. Але присутні недоліки з надійністю, оскільки інші СУБД мають перевагу над деякими методами обробки даних MySQL (зв'язки, транзакції, аудити).

PostgreSQL – найбільш професійна СУБД. Вона розповсюджена і відповідає стандартам SQL. Підтримує об'єктно-орієнтований або реляційний підходи до баз даних. Postgre має велику продуктивність, завдяки потужним технологіям.

До переваг PostgreSQL можна віднести відкрите програмне забезпечення відповідно до стандарту SQL. СУБД має багато вбудованих функцій і доповнень, які дозволяють розробляти і керувати своїми даними. Розширення функціоналу можливо за рахунок збереження своїх процедур. Крім реляційності СУБД об’єктно-орієнтована з можливістю успадкування. Проте недоліком виступає значне уповільнення серверу при простих операціях читання. Порівняно з іншими СУБД не дуже розповсюджена і популярна, через що досить складно знайти хостинг з підтримкою цієї системи.

Завдяки проведеному аналізу СУБД, було надано перевагу MySQL, оскільки вона успішно співпрацює з вебсайтами. Містить багату кількість функцій та інструментів. Крім цього система легко встановлюється та налаштовується, а що найголовніше є системою, що масштабується.

2.6 Вибір середовища розробки

Для написання проєкту використовувався застосунок SublimeText. Для запуску проєкту та перевірки на працездатність на локальному комп’ютері використовувалась програма Visual Studio Code.

Sublime Text – багатоплатформний текстовий редактор, плагіни якого підтримують мову програмування Python. Набір інструментів програми полегшує редагування вихідного коду програми. Існує палітра команд для покращення кодування і, до того ж, її можна легко налаштувати. Sublime Text надає підтримку великої кількоста мов програмування та можливості підсвічування синтаксису. Надається можливість завантаження пакетів для підтримки додаткових мов. Sublime Text забезпечує швидкість і продуктивність.

Із застосуванням новітніх технологій Microsoft, було створено середовище розробки Visual Studio Code. Це середовище є текстовим редактором, що здатний підключати велику кількість плагінів. Він підтримує різні мови програмування та платформи для розробки, не лише Windows, а й Linux і MacOS.

Вважається «легким» та швидким редактором коду для кросплатформної розробки веб та хмарних застосунків. Дозволяє зручний перегляд сайту, одночасне редагування коду і миттєве відображення змін сайту.

Visual Studio Code – це спрощений редактор коду з підтримкою операцій розробки, таких як відладка, запуск програми і контроль версій.

2.7 Висновок

В цьому розділі були розглянуті базові інтернет-технології, визначені основні засоби для розробки вебсайту. Для створення інтерфейсу було обрано HTML, CSS. Проведено аналіз мов програмування та обрано Python. Серед фреймворків надано перевагу Django, досліджено його архітектуру. Системою для управління базою даних визначено MySQL. Обрано середовище розробки Visual Studio Code для створення та роботи з проєктом.

3 РОЗРОБКА САЙТУ

3.1 Структура сайту

Оскільки програмною платформою для створення сайту була обрана Django Framework, то структура повинна відповідати тим обмеженням, що накладає платформа. Філософія Django Framework має на увазі модульну архітектуру сайту. Після аналізу завдання, було виділено кілька модулів: модуль управління сторінками, модуль новин, модуль зберігання та обробки документів, модуль зворотного зв'язку. Ці модулі повинні вміти створювати, обробляти та зберігати дані. Модулі є дочірніми елементами головної програми, вони обробляють вхідні запити від відвідувачів сайту та розподіляють за модулями залежно від типу даних. Наочно структура представлена на рисунку 3.1.

Рисунок 3.1 - Схема структури сайту

3.2 Проєктування функціональних вимог до вебсайту

Після визначення зі структурою сайту, можна виділити наступні функціональні вимоги до розроблюваного вебсайту:

- авторизація облікових записів;
- перегляд створених новин;
- додавання, редагування та вилучення новин;
- перегляд детальної інформації конкретної новини;
- перегляд контактів університету;
- додавання, редагування та вилучення контактів;

– перегляд контактів користувачів;

– додавання, редагування та вилучення контактів користувачів;

– керування категоріями, тегами, профілями

– взаємодія з формою зворотного зв'язку.

Всі відвідувачі сайту були розділені на дві групи: адміністратори та користувачі.

Адміністраторам надаються розширені права роботи з системою.

На рисунку 3.2 наведено діаграму прецедентів для роботи адміністратора з системою.

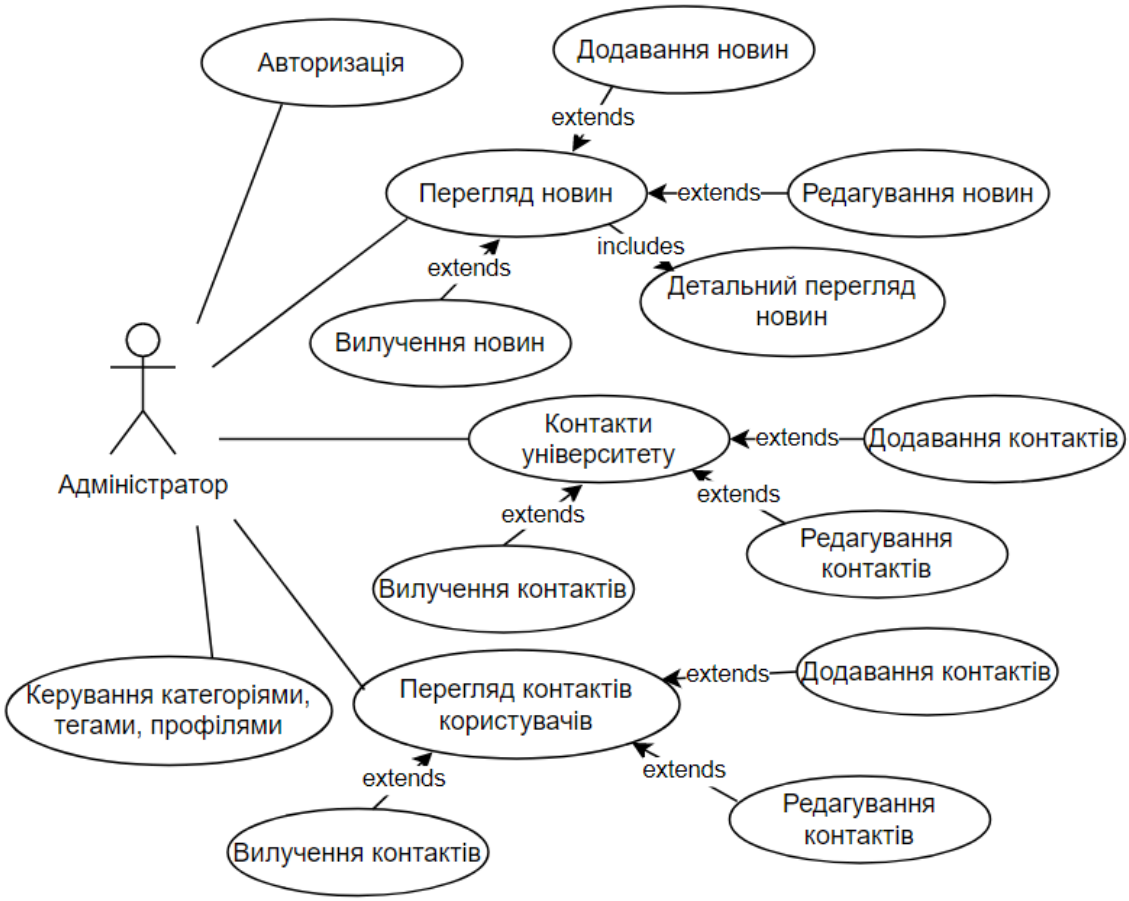


Рисунок 3.2 – Діаграма прецедента «Робота адміністратора з системою»

На рисунку 3.3 наведено діаграму прецедентів для роботи користувачів з системою.

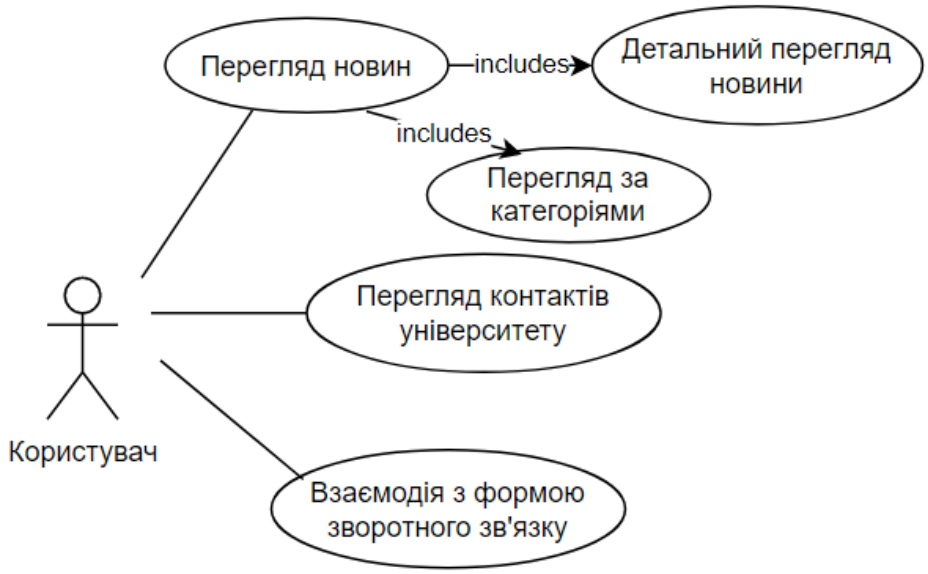


Рисунок 3.3 – Діаграма прецедента «Робота користувача з системою»

3.3 Django Models

Моделі забезпечують рівень абстракції бази даних.

Техніка, що була використана, називається Object-relational Mapping (ORM).

Більшість баз даних використовують SQL, проте кожна їх реалізує це по-своєму. Знання тонкощів SQL може бути досить проблематичним. ORM, з іншого боку, забезпечує просте зіставлення між об'єктом та базою даних. При цьому програмісту не потрібно знати тонкощів налаштування бази даних або вміти писати складні SQL запити.

В Django модель – і є цей об'єкт. При створенні моделі Django створює відповідну таблицю в базі даних без необхідності писати SQL вручну. Це продемонстровано на рисунку 3.4.

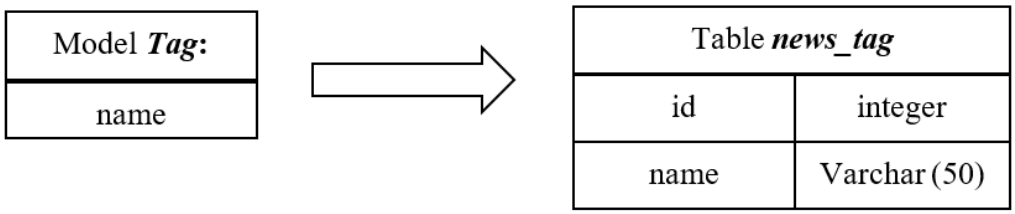


Рисунок 3.4 – Відповідність моделі та таблиці

3.3.1 Установка Django та створення проєкту

Django працює як із Python 2.7, так і з більш пізнішими версіями 3.x. У роботі було використано Python версії 3.10 – остання на момент написання роботи версія. Для встановлення всіх додаткових пакетів Python використовується PIP (акронім для «Pip Installs Packages»). PIP поставляється разом з Python 2.7.9 та пізніше, Python 3.4 та пізніше.

Першим кроком створюється активне віртуальне оточення за допомогою команди python -m venv venv, як зображено на рисунку 3.5.

```
PS F:\Diploma\2022\Site\django4dualis-main\dualis> python -m venv venv
PS F:\Diploma\2022\Site\django4dualis-main\dualis>
```

Рисунок 3.5 – Створення активного віртуального оточення.

Перебуваючи в активному віртуальному оточенні, наступним кроком буде встановлення Django. Ця процедура нічим не відрізняється від установки більшості інших пакетів у Python, інсталюється так само через утиліту PIP. Після завершення інсталяції можна створювати проєкт.

В результаті роботи команди створення проєкту (startproject dualis), буде створено директорію проєкту dualis всередині тієї директорії, що була до цього. Результат роботи команди startproject представлено на рисунку 3.6.

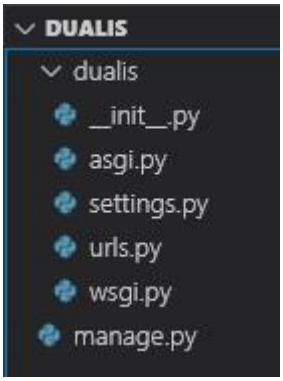


Рисунок 3.6 – Створення директорії проєкту dualis

Розглянемо докладніше призначення даних файлів:

– manage.py – утиліта для командного рядка, що дозволяє взаємодіяти з проєктом багатьма способами;

– dualis/__init__.py – порожній файл, який вказує Python, що ця директорія dualis є модулем Python;

– dualis/settings.py – конфігураційний файл для цього проєкту. У ньому визначаються база даних, параметри підключення до неї, підключені до проєкту програми та багато іншого;

– dualis/urls.py – певні URL-адреси для даного проєкту, в певному сенсі «зміст» майбутнього сайту;

– dualis/wsgi.py – вхідна точка для Web Server Gateway Interface (WSGI)-сумісних вебсерверів для виконання проєкту.

3.4 Створення моделей

Після завантаження сайту, користувач потрапить (побачить перед собою) на головну сторінку сайту. На даній сторінці будуть публікуватися новини. Розглянемо на прикладі цієї сторінки процес створення моделей.

Для цього звернемося до файлу models.py, який Django автоматично створив під час ініціалізації програми dualis. Створюємо моделі даних для новин. Створимо три такі моделі даних:

– Profile – зберігає інформацію про користувачів новини;

– Tag – містить дані про категорії, за якими групуються записи новини;

– Post – використовується для зберігання контенту та метаданих про кожен пост новини.

Розглянемо кожну модель більш детально.

Модель Profile , частина коду якої представлена на рисунку 3.7.

```
26  class Profile(models.Model):
27      class Meta:
28          verbose_name = 'Профіль'
29          verbose_name_plural = 'Профілі'
30
31
32      user = models.OneToOneField(
33          settings.AUTH_USER_MODEL,
34          on_delete=models.PROTECT,
35      )
36      website = models.URLField(blank=True)
37      bio = models.CharField(max_length=240, blank=True)
38
39      def __str__(self):
40          return self.user.get_username()
41
```

Рисунок 3.7 – Код моделі Profile

Модель профілю містить кілька полів:

– user - зв'язок з користувачем Django;

– website - опціональний URL, за яким можна дізнатися більше про користувача;

– bio - опціональний невеликий опис («про себе»).

Використаний в даній моделі метод __str__ зробить зручнішим відображення профілів на панелі адміністратора.

Наступна модель Tag, частину коду даної моделі можна побачити на рисунку 3.8.

```python
class Tag(models.Model):
    class Meta:
        verbose_name = 'Тег'
        verbose_name_plural = 'Теги'


    name = models.CharField(max_length=50, unique=True)

    def __str__(self):
        return self.name
```

Рисунок 3.8 – Код моделі Tag

У моделі Tag буде єдине поле name – коротке ім'я тега.

Далі розглянемо модель Post, частина коду даної моделі представлена на рисунку 3.9.

```python
class Post(models.Model):
    class Meta:
        verbose_name = 'Новину'
        verbose_name_plural = 'Новини'
        ordering = ["-publish_date"]

    title = models.CharField(max_length=255, unique=True, verbose_name = 'Заголовок')
    subtitle = models.CharField(max_length=255, blank=True, verbose_name = 'Підзаголовок')
    slug = models.SlugField(max_length=255, unique=True, verbose_name = 'Слаг')
    body = models.TextField(blank=True, verbose_name = 'Контент')
    photo = models.ImageField(upload_to='photos/%Y/%m/%d/', verbose_name = 'Фото')
    meta_description = models.CharField(max_length=150, blank=True, verbose_name = 'Мета опис')
    date_created = models.DateTimeField(auto_now_add=True, verbose_name = 'Дата створення')
    date_modified = models.DateTimeField(auto_now=True, verbose_name = 'Оновлено')
    publish_date = models.DateTimeField(blank=True, null=True, verbose_name = 'Дата публікування')
    published = models.BooleanField(default=False, verbose_name = 'Опубліковано')
    category = models.ForeignKey('Category', on_delete=models.PROTECT, null=True, verbose_name = 'Категорі

    def get_absolute_url(self):
        return reverse('view_news', kwargs={"news_id": self.pk})

    def __str__(self):
        return self.title

    author = models.ForeignKey(Profile, on_delete=models.PROTECT)
    tags = models.ManyToManyField(Tag, blank=True)
```

Рисунок 3.9 – Код моделі Post

Модель Post – найскладніша, містить багато полів:

– title – заголовок новини;

– subtitle – підзаголовок новини;

– slug – деяка мітка, яка містить у собі лише літери, числа, підкреслення та дефіс. Використовується для створення URL;

– body – контент новини;

– photo – опціональне зображення новини;

– meta_description - мета опис;
– date_- дата створення;
– date_modified – дата оновлення;
– publish_date - дата публікування;
– published – опубліковано;
– category – категорія.

На даному етапі вже використовується клас Meta. Він вбудовується всередину створених напередодні класів, може бути використаний у будь-якій моделі для вказівки різних опцій, притаманних конкретної моделі. В даному випадку в класі Category використовуємо опцію ordering: вказуємо Django, як саме хочемо, щоб відбувалося сортування – за датою публікації, якщо явно не вказано інакше (за допомогою order_by()) та опцію verbose_name: вказівку імені для відображення.

Аргумент on_delete=models.PROTECT для поля author гарантує, що при видаленні постів автор не буде випадково видалений.

Кожен тег може бути пов'язаний з багатьма повідомленнями, тому для поля tags використовується ставлення ManyToManyField.

І нарешті, класи містять у собі метод __str__(). Він використовується для того, щоб зробити відображення об'єктів цього класу, а саме використовувати поле імені цих класів при відображенні.

3.5 Розробка схеми бази даних

Django автоматично створює відповідну таблицю в базі даних без необхідності писати SQL вручну.

Усі створені моделі зі своїми властивостями наглядно представлені на рисунку 3.10.

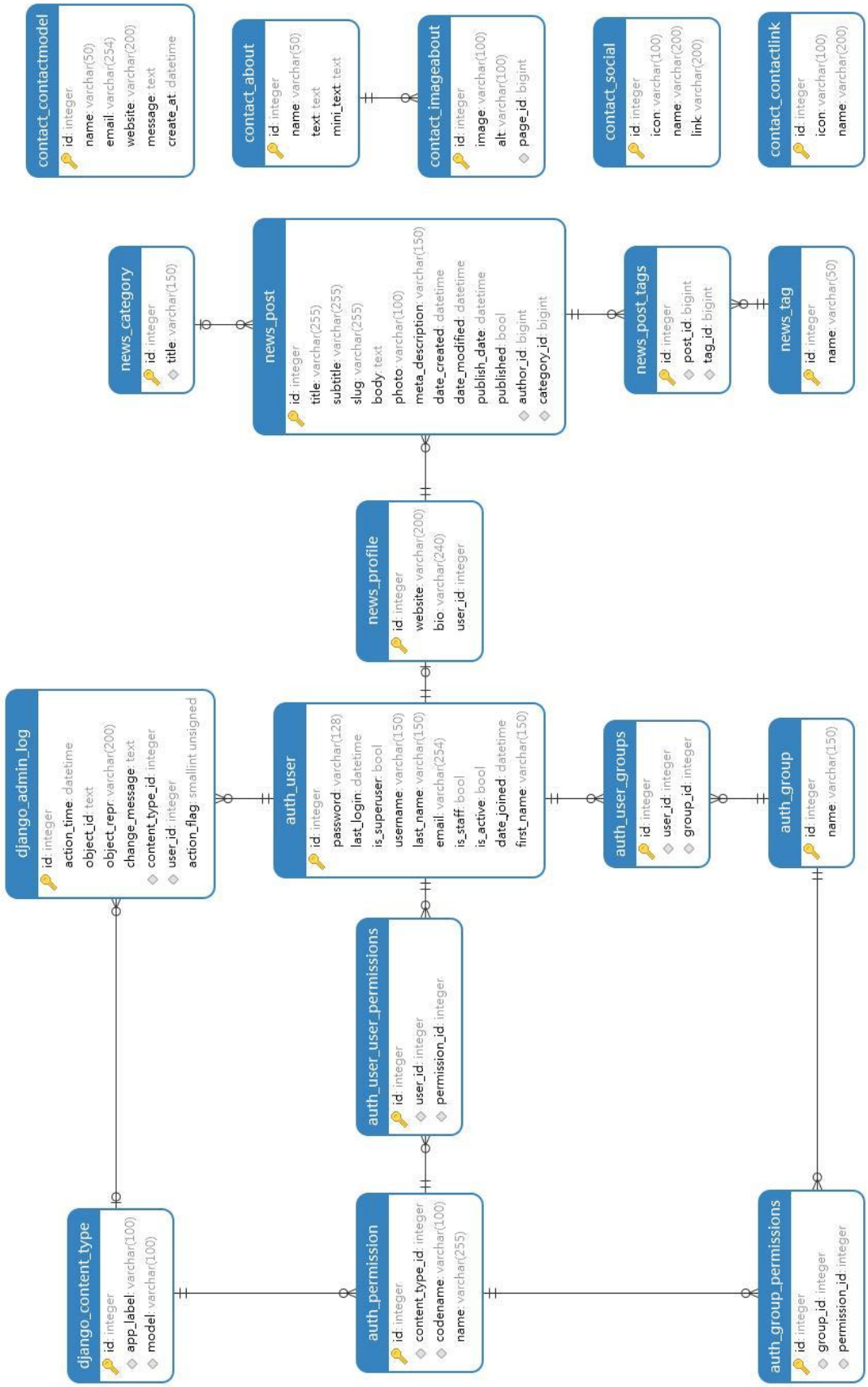


Рисунок 3.10 – Схема бази даних

Усі описані характеристики сутностей наведені в таблицях 3.1 – 3.3.

Таблиця 3.1 – Характеристики полів news_post

Поле	Тип поля	Характеристика
title	varchar (255)	Заголовок
subtitle	varchar (255)	Підзаголовок
slug	varchar (255)	Слаг
body	text	Контент
photo	varchar (100)	Фото
meta_description	varchar (150)	Опис
date_created	datetime	Дата створення
date_modified	datetime	Оновлено
publish_date	datetime	Дата публікування
published	bool	Опубліковано

Таблиця 3.2 – Характеристики полів news_profile.

Поле	Тип поля	Характеристика
website	varchar (200)	URL користувача
bio	varchar (240)	Невеличкий опис
user_id	integer	id користувача

Таблиця 3.3 – Характеристики полів auth_user.

Поле	Тип поля	Характеристика
password	varchar(128)	Пароль
last_login	datetime	Останній вхід в аккаунт(профіль)
is_superuser	bool	Перевірка чи є користувач адміністратором
username	varchar(150)	Нік користувача
last_name	varchar(150)	Фамілія користувача
email	varchar(254)	Електронна пошта
is_staff	bool	Чи є адміном
is_active	bool	Перебуває онлайн(у мережі)
date_joined	datetime	Дата реєстрації
first_name	varchar(150)	Ім’я користувача

Сутність «news_profile» містить унікальні поля «website», «user_id», які відповідають за адрес користувача, його ідентифікаційний номер та не можуть дублюватися у базі даних.

3.6 Налаштування інтерфейсу для роботи з базою даних

Як вже зазначалося до цього, Django забезпечує вбудований інтерфейс для адміністраторів сайту. Залишається його лише налаштувати, щоб бачити саме ту інформацію, яку потрібно. Опишемо налаштування на прикладі сторінки «Новини». Для цього звернемося до файлу Dualis\news\admin.py. Вказаний файл відповідає за налаштування інтерфейсу адміністрування, додавання та видалення новин.

Для початку необхідно вказати з якими моделями будемо працювати. За підключення необхідних моделей відповідає команда from news.models import Profile, Post, Tag, Category. На рисунку 3.11 представлено код підключення моделей.

Рисунок 3.11 – Програмний код підключення моделей

У коді @admin.register показує Django, яким саме моделям необхідно забезпечити інтерфейс. Також вказуються деякі додаткові параметри для відображення:

– list_display: які з полів таблиці виводити;

– list_filter: створює фільтри за вказаними параметрами. Фільтри змінюються в залежності від типу поля;

– list_editable: вказує, які поля можна буде редагувати під час відображення списком. Це дозволить редагувати безліч рядків за один раз;

– prepopulated_fields: поля з цим атрибутом отримуватимуть значення автоматично на основі інших полів. В даному випадку – slug успадковує значення "title" та "subtitle". Налаштування параметрів відображення наведені на рисунку 3.12.

admin.py

```
list_display = (
    "id",
    "title",
    "category",
    "subtitle",
    "slug",
    "publish_date",
    "published",
)
   list_filter = (
    "published",
    "publish_date",
    "category",
)
list_editable = (
    "title",
    "subtitle",
    "slug",
    "publish_date",
    "published",
)
search_fields = (
    "title",
    "subtitle",
```

Рисунок 3.12 – Налаштування параметрів відображення.

Результат налаштування властивостей відображення у панелі адміністрації Django наведено на рисунку 3.13.

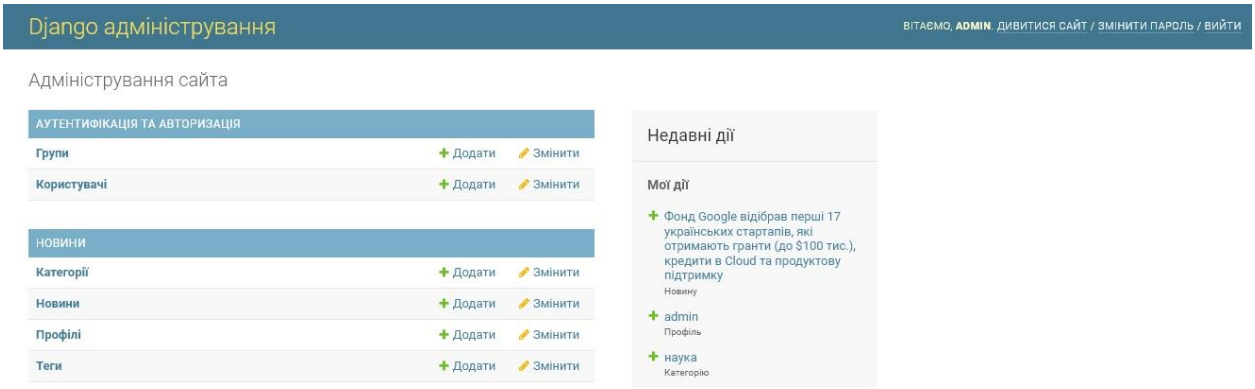


Рисунок 3.13 – Результат налаштування

3.7 Висновок

В ході роботи було проаналізовано та створено структуру для зручного розуміння сайту. Описано створення основних моделей сайту, їх призначення та функції з наведеним кодом та рисунками на прикладі сторінки «Новини». Наведено розроблену схему бази даних та налаштування інтерфейсу, що

дозволяє працювати з її даними. Всі інші моделі проєкту розробляються за аналогією описаного прикладу в цьому розділі.

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ

4.1 Тестування вебсайту

Для перевірки коректності роботи розробленої системи необхідно провести її тестування за різними параметрами. По перше, необхідно провести тестування системи на її коректне відображення різними пристроями та браузерами. По-друге, необхідно перевірити коректність роботи частини адміністрування системи (на прикладі додавання новин).

Тестування на коректність відображення розробленого продукту проводиться в різних браузерах. Під коректним відображенням розуміється правильне відображення інформації на сторінці. Використання стандартизованих технологій розробки вебінтерфейсу передбачає, що не повинно виникнути проблем і помилок коректності відображення. Процес тестування підтвердив це припущення. Інформаційний сайт дійсно коректно відображається у найпопулярніших браузерах. На рисунках 4.1 та 4.2 представлено відображення сайту у різних браузерах.

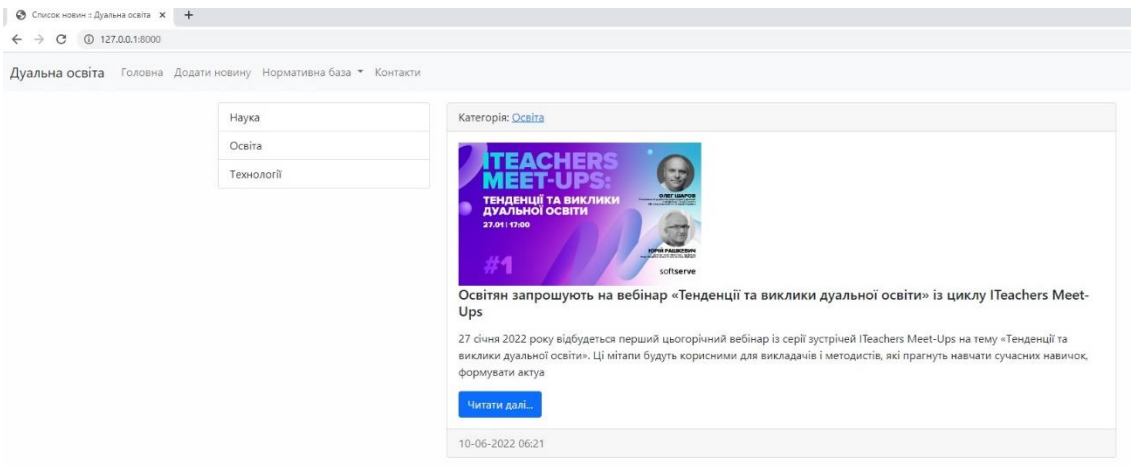


Рисунок 4.1 – Відображення сторінки новин у браузері Google Chrome

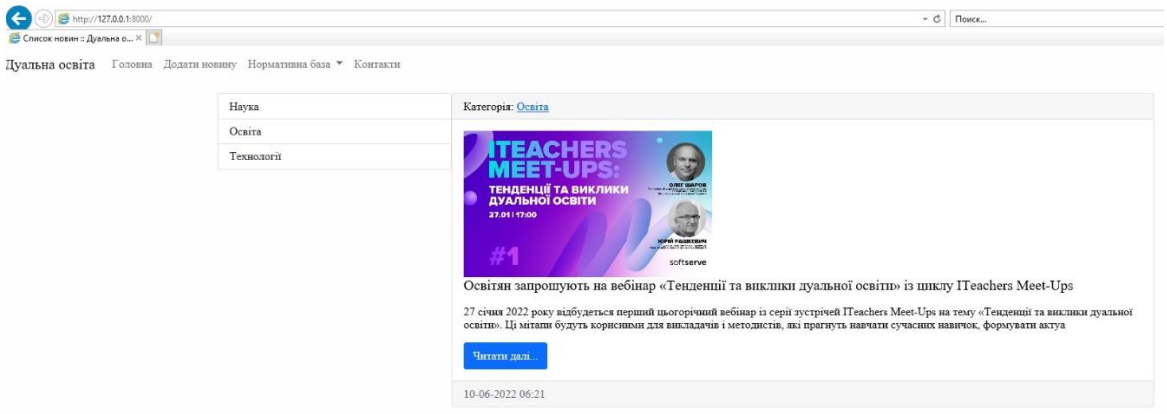


Рисунок 4.2 – Відображення сторінки новин у браузері InternetExplorer

Другим етапом тестування є перевірка створеного продукту на коректність роботи. Необхідно, щоб у процесі роботи із системою не виникало різних помилок. Цей тест був проведений шляхом додавання та видалення новин, а також роботи з доступом користувачів з правами адміністратора до панелі адміністрування.

Перш за все, щоб додати чи видалити новину, потрібно увійти з правами адміністратора до адміністративної частини сайту. Для цього потрібно ввести логін та пароль зареєстрованого користувача з правами адміністратора. Процес входу показаний на рисунку 4.3. На рисунку 4.4 продемонстровано розділ адміністрування сайту.

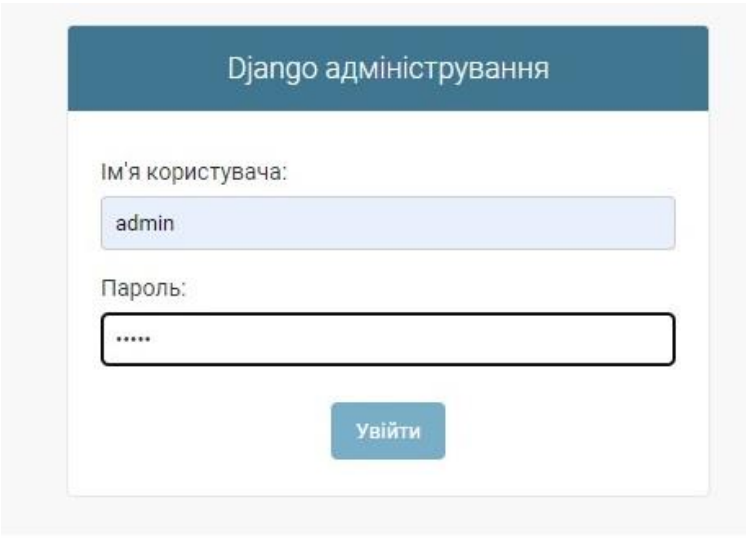


Рисунок 4.3 – Вхід користувача адміністратора

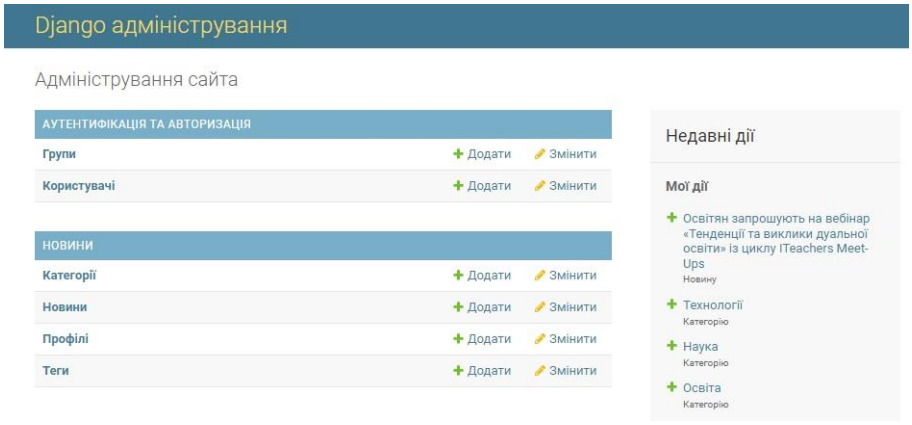


Рисунок 4.4 – Розділ адміністрування сайту

Після виконання входу, адміністратор має можливість керувати користувачами (які були створені за допомогою команди *python manage.py createsuperuser* під час розгортання проєкту на хостингу):

– додавати користувача;
– видаляти користувача;
– створювати групи для користувачів.

Можливість керування зображена на рисунку 4.5.

Рисунок 4.5 – Керування користувачами

Перевіримо працездатність процесу додавання новини. Для цього перейдемо в розділ адміністрування новин, що наведений на рисунку 4.6.

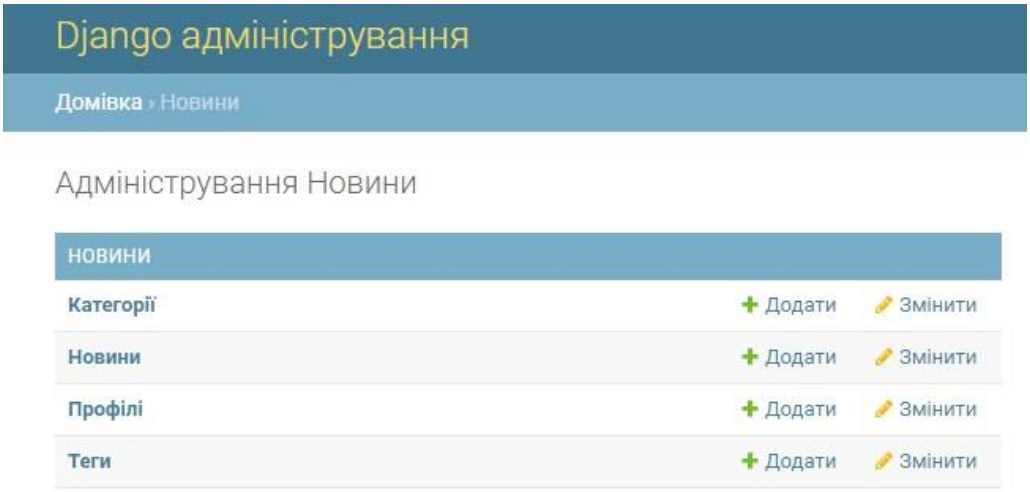


Рисунок 4.6 – Розділ адміністрування новин

Даний розділ дозволяє керувати наступними полями:

– Категорія – всі новини групуються за категоріями (наприклад, освіта, технології і .т.д.);

– Новини – безпосереднє керування статтями (додавання, видалення);

– Профілі – поле, яке вказує на автора, який додав новину на сайт. Дане поле також має можливість додавання та видалення профілів авторів;

– Тег - асоційоване ключове слово, що стосується частини інформації. Такі метадані допомагають описати ці частини інформації та швидко знаходити їх через пошуковий запит.

Після переходу на поле «Новини» можна побачити список вже доданих та опублікованих матеріалів, які наведені на рисунку 4.7.

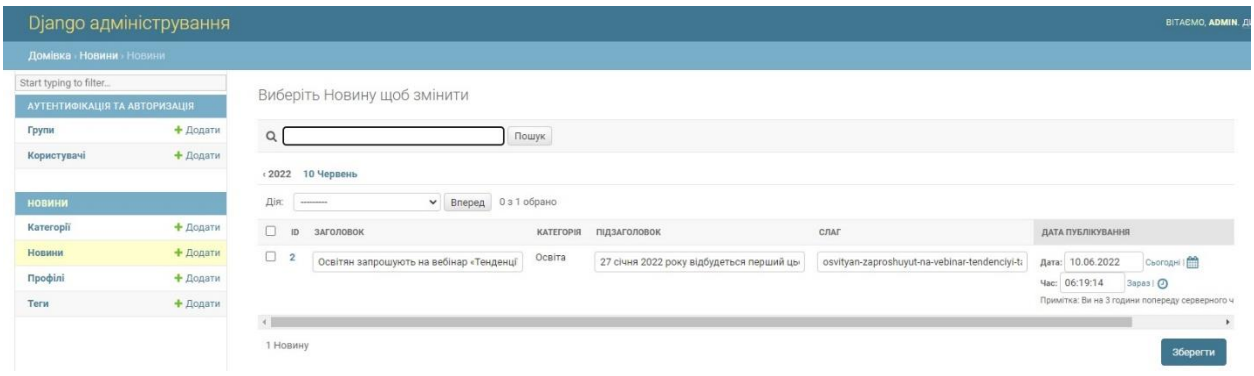


Рисунок 4.7 – Список вже доданих та опублікованих матеріалів

Розглянемо процедуру додавання новини. Для цього потрібно використати форму додавання новини, що на рисунку 4.8 та заповнити її поля, як показано на рисунку 4.9.

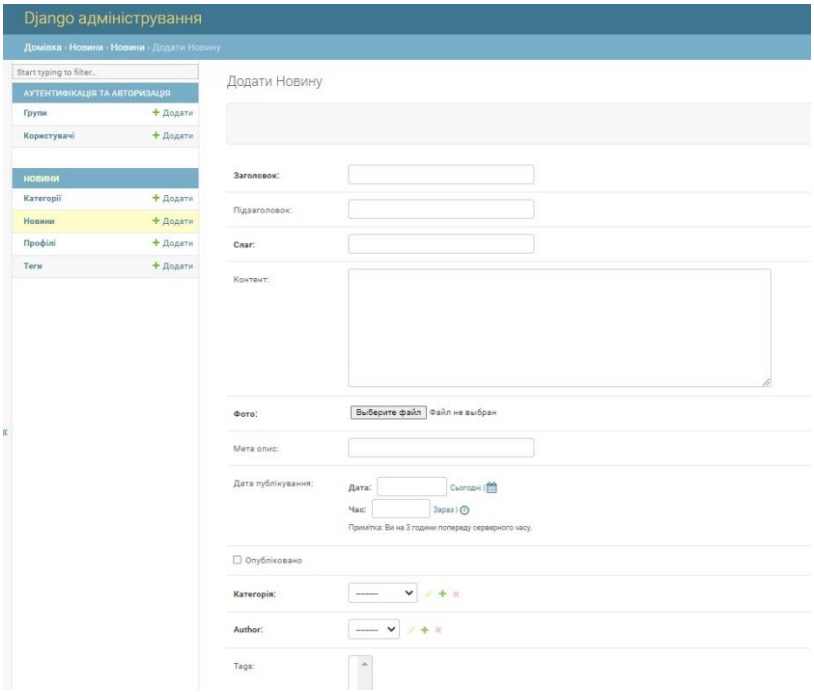


Рисунок 4.8 – Форма додавання новини

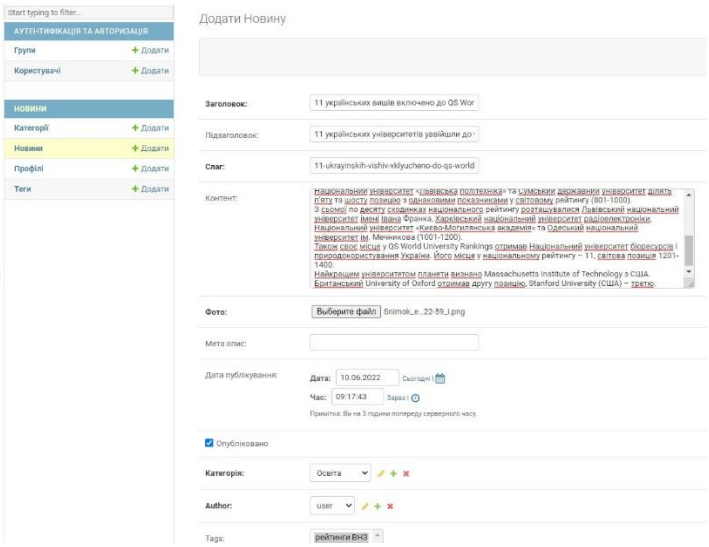


Рисунок 4.9 – Заповнена форма додавання новини

Після збереження заповненої форми, новина публікується на сторінці і її можна переглянути. На рисунку 4.10 можна побачити готову публікацію новини.

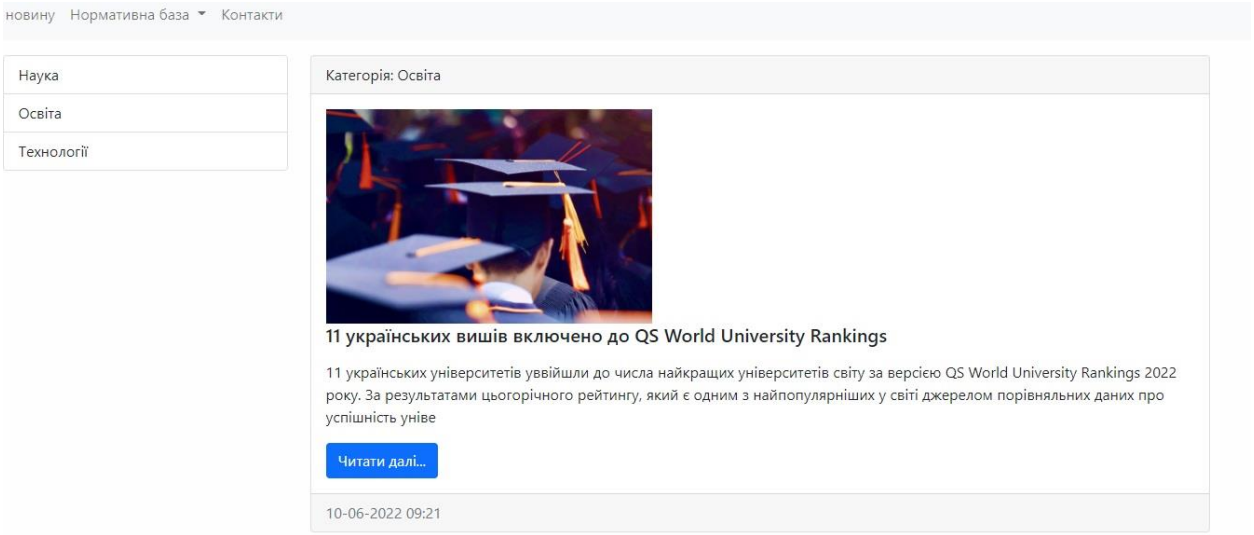
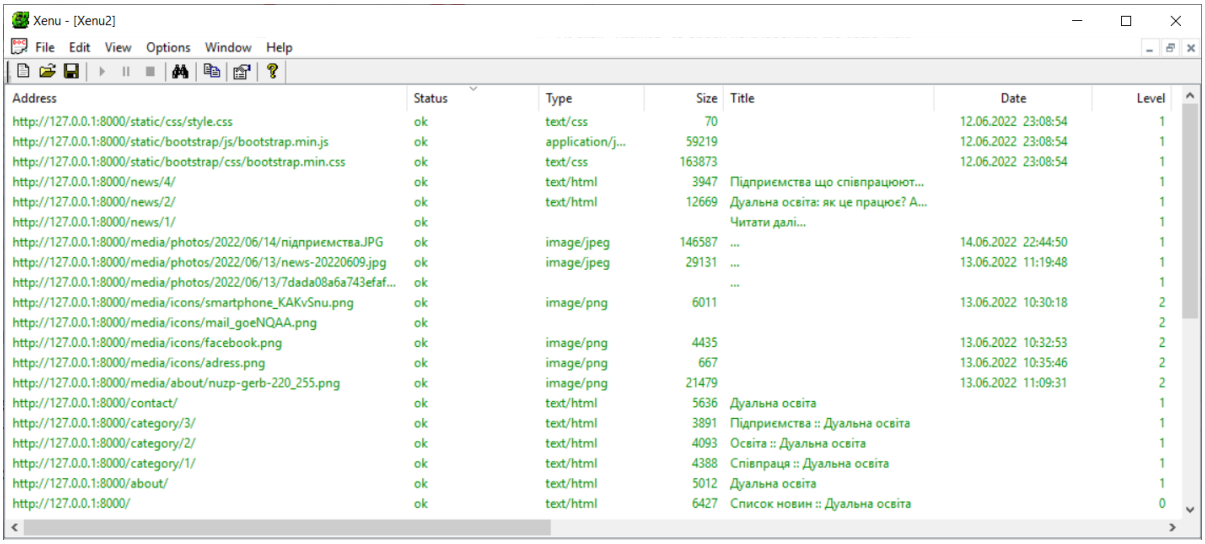


Рисунок 4.10 – Опублікована новина

4.1.1 Тестування посилань вебсайту

Після внесення змін у сайт, таких як видалення новин, категорій, зміна зображень, існує ризик залишити «порожні» посилання на неіснуючі елементи, яке залишилося після редагувань. За допомогою застосунку «Xenu» можна провести аналіз усіх існуючих посилань сайту, як зображено на рисунку 4.11.



Address	Status	Type	Size	Title	Date	Level
http://127.0.0.1:8000/static/css/style.css	ok	text/css	70		12.06.2022 23:08:54	1
http://127.0.0.1:8000/static/bootstrap/js/bootstrap.min.js	ok	application/j...	59219		12.06.2022 23:08:54	1
http://127.0.0.1:8000/static/bootstrap/css/bootstrap.min.css	ok	text/css	163873		12.06.2022 23:08:54	1
http://127.0.0.1:8000/news/4/	ok	text/html	3947	Підприємства що співпрацюют...		1
http://127.0.0.1:8000/news/2/	ok	text/html	12669	Дуальна освіта: як це працює? А...		1
http://127.0.0.1:8000/news/1/	ok			Читати далі...		1
http://127.0.0.1:8000/media/photos/2022/06/14/підприємства.JPG	ok	image/jpeg	146587	...	14.06.2022 22:44:50	1
http://127.0.0.1:8000/media/photos/2022/06/13/news-20220609.jpg	ok	image/jpeg	29131	...	13.06.2022 11:19:48	1
http://127.0.0.1:8000/media/photos/2022/06/13/7dada08a6a743efaf...	ok			...		1
http://127.0.0.1:8000/media/icons/smartphone_KAKvSnu.png	ok	image/png	6011		13.06.2022 10:30:18	2
http://127.0.0.1:8000/media/icons/mail_goeNQAA.png	ok					2
http://127.0.0.1:8000/media/icons/facebook.png	ok	image/png	4435		13.06.2022 10:32:53	2
http://127.0.0.1:8000/media/icons/adress.png	ok	image/png	667		13.06.2022 10:35:46	2
http://127.0.0.1:8000/media/about/nuzp-gerb-220_255.png	ok	image/png	21479		13.06.2022 11:09:31	2
http://127.0.0.1:8000/contact/	ok	text/html	5636	Дуальна освіта		1
http://127.0.0.1:8000/category/3/	ok	text/html	3891	Підприємства :: Дуальна освіта		1
http://127.0.0.1:8000/category/2/	ok	text/html	4093	Освіта :: Дуальна освіта		1
http://127.0.0.1:8000/category/1/	ok	text/html	4388	Співпраця :: Дуальна освіта		1
http://127.0.0.1:8000/about/	ok	text/html	5012	Дуальна освіта		1
http://127.0.0.1:8000/	ok	text/html	6427	Список новин :: Дуальна освіта		0

Рисунок 4.11 –Перевірка посилань

4.2 Валідація форми

Для правильності введення даних користувача використовується валідація форми, аби уникнути помилок при заповнені полів.

Django автоматично підключає валідацію форми, перевіряє поля.

При неправильному введені даних у формі зворотного зв'язку з'являється повідомлення. Як наведено на рисунках 4.12 та 4.13.

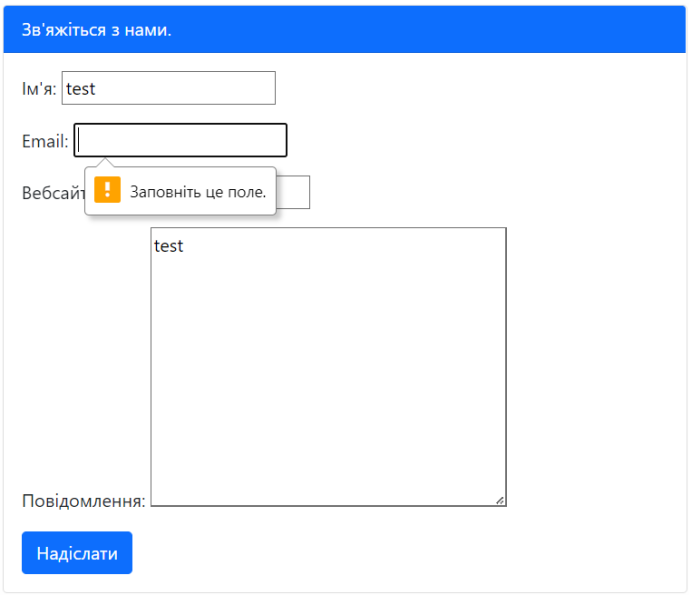


Рисунок 4.12 – Обов’язкова вимога заповнити поле «Email»

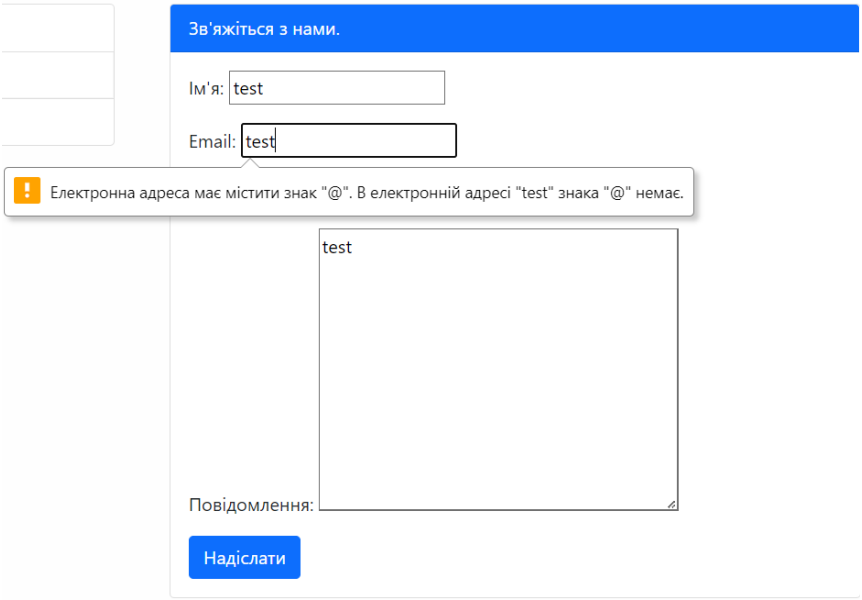


Рисунок 4.13 – Повідомлення про некоректно введену пошту

4.3 Адаптивність вебсайту

Аби впевнитися що сайт відображається коректно на пристроях з різною роздільною здатністю екрану, використаємо консоль розробника(Ctrl+Shift+I), що присутня у браузерах. За її допомогою можна змінювати ширину екрану, підлаштовуватися під інші пристрої. На рисунку 4.14 наведено приклад відображення сайту при ширині екрану 768 пікселів, тобто розширення планшетів.

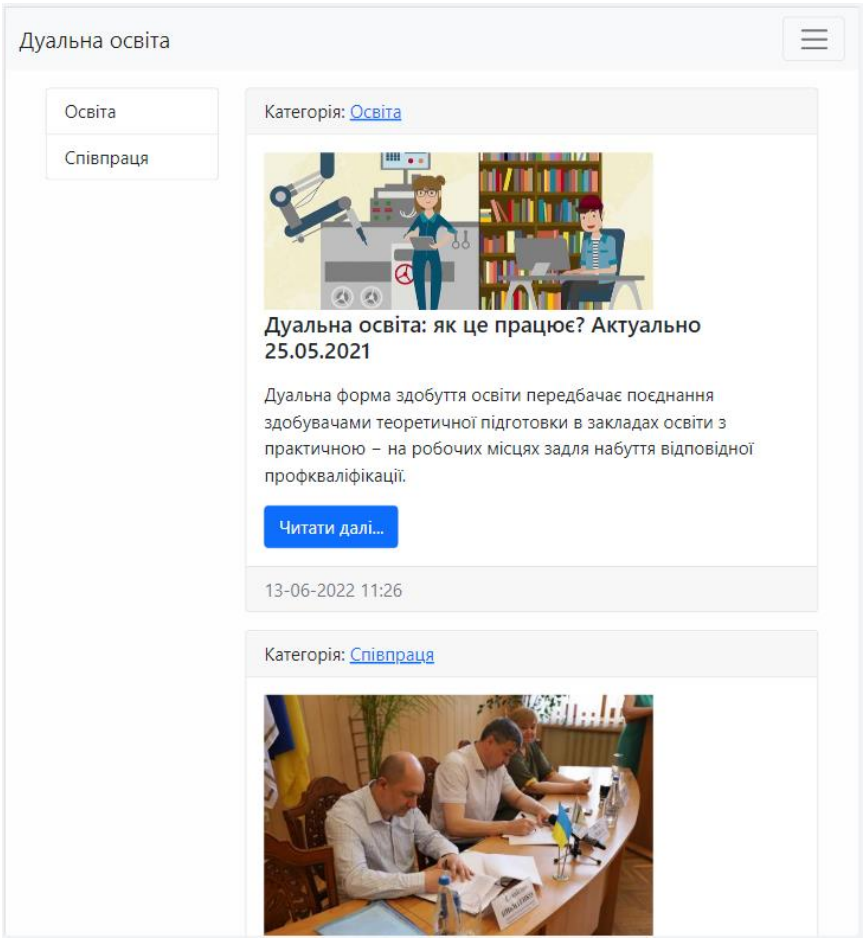


Рисунок 4.14 – Відображення сайту на планшеті

Наступним прикладом наведено розширення екрану розміром у 425 пікселів у ширину для смартфонів, який можна подивитися на рисунку 4.15.

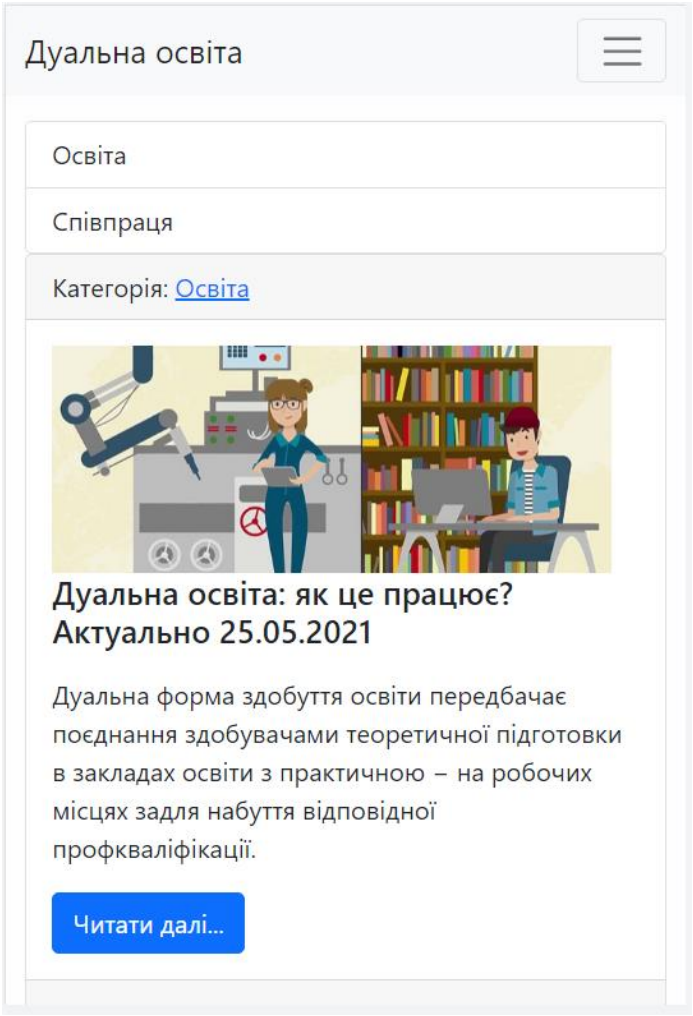


Рисунок 4.15 – Відображення сайту на смартфоні

4.4 Висновок

В даному розділі було наведено взаємодію користувача з системою. Зроблено тестування основних функцій вебсайту. Система та всі розроблені на даний момент функції адміністратора показали, що працюють і відповідають всім вимогам. Перевірено, що система попереджує користувача про помилкове заповнення форми. Проведено перевірку щодо відображення сайту на різних пристроях з різною роздільною здатністю екрану. Протестовано посилання сайту на пусті директорії.

Отже, на підставі проведених тестувань, перевірено що робота вебсайту коректна та відповідає затребуваним вимогам.

ВИСНОВКИ

Під час виконання бакалаврської роботи було розроблено вебсайт «Дуальна освіта» для НУ «Запорізька політехніка».

Була виявлена така проблема, що дуальна освіта досить нещодавно запроваджена, через що має низький рівень поінформованості серед студентів та роботодавців. Тож виникає потреба у групуванні інформації в одному місці.

Актуальність даного питання викликає необхідність у розробці вебсайту для інформаційної підтримки впровадження дуальної форми освіти у НУ «Запорізька політехніка».

Для вирішення проблеми було створено вебсайт. Це допомогло розв’язати такі питання як: популяризація дуальної освіти серед молодих спеціалістів та роботодавців, поява окремого сайту дуальної освіти в університеті для інформування.

Вебсайт можна використовувати у цілях університету, а саме для розповсюдження інформації та новин університету, що стосуються дуальної освіти.

Проаналізовано існуючі та поширені мови веброзробки, такі як PHP, Python, Ruby. Проведено порівняння популярних СУБД: SQLite, MySQL, PostgreSQL. Було обрано зручне середовище розробки Visual Studio Code та редактор SublimeText.

Розроблені основні модулі для публікації новин та зберігання інформації про новини, таким чином Profile – зберігає інформацію про користувачів новини; Tag – містить дані про категорії, за якими групуються записи новини; Post – використовується для зберігання контенту та метаданих про кожен пост новини.

Створено схему бази даних та налаштовано інтерфейс, що дозволяє працювати з її даними.

Протестовано основні функції вебсайту: система та функції адміністратора, валідація заповнення форми, відображення сайту на різних пристроях з різною роздільною здатністю екрану.

У ході виконання бакалаврської роботи було застосовано набуті навички проєктування та розробки сайту за допомогою фреймворку Django.

Поставлене завдання до проєкту було виконано, вебсайт працює коректно та відповідає затребуваним до нього вимогам.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Абашкіна, Н.В. Принципи розвитку професійної освіти в Німеччині: монографія. Київ : Вища школа, 1998. 207 с.

2. Амеліна, С.М. Особливості дуальної системи вищої професійної освіти у навчальних закладах Німеччини. Зб. наук. праць “Проблеми трудової і професійної підготовки”. 2010. Вип. 15. С.107–112.

3. Мілл, У. Дуальне навчання: досвід Німеччини. Олена Давліканова, Ніна Світайло (ред. та допов.). Маркетинг в Україні. 2016. № 6. С. 53–62.

4. Gren, L., Panfilov, Y., Karlyuk, S. Directions in dual form of training introduction at National Technical University «Kharkiv Polytechnic Institute»: state-managerial aspect. Теорія і практика управління соціальними системами: філософія, психологія, педагогіка, соціологія. 2019. № 1. С. 66–80.

5. Кримчак, Л. Ю. Система дуальної освіти як умова якісної підготовки конкурентоспроможних професіоналів до ринку праці в Україні [Електрон. ресурс]/ Л. Ю. Кримчак. – 2019. – Режим доступу до ресурсу: http://www.innovpedagogy.od.ua/archives/2019/11/part_2/20.pdf (дата звернення 20.03.2021).

6. Бегма, О. На виробництво – вже підготовленим. Сільські вісті. 15. [Електрон. ресурс]/ О. Бегма. – 2018 – Режим доступу до ресурсу: http://www.silskivisti.kiev.ua/ 19562/index.php?n=38199 (дата звернення 20.03.2021).

7. Яковенко, К. В. Реалізація дуальної системи освіти в підготовці майбутніх фахівців з інформаційних технологій [Електрон. ресурс]/ К. В. Яковенко. – 2016 – Режим доступу до ресурсу: http://repository.kpi.kharkov.ua/bitstream/KhPIPress/25362/1/Elita_2016_45_1 (дата звернення 20.03.2021).

8. Кулалаєва, Н. В. SWOT-аналіз упровадження елементів дуальної форми навчання в професійну підготовку майбутніх кваліфікованих робітників. Науковий вісник Інституту професійно-технічної освіти НАПН України.

Професійна педагогіка : зб. наук. праць. 2018. Вип. 15. С. 51–59.

9. Batysheva, S.Ja. (1997) Professyonaljnaja pedaghoghyka [Professional pedagogy]: Uchebnyk dlja studentov, obuchajushhykhsja po pedaghoghycheskym specyaljnostjam y napravlenyjam. Moscow: Assocyacyja «Professyonaljnoe obrazovanye», 512 p.

10. Давліканова, О. Б. Аналітичний звіт за результатами першого року проведення експерименту із запровадження пілотного проєкту у закладах фахової передвищої та вищої освіти з підготовки фахівців за дуальною формою здобуття освіти [Електрон. ресурс]/ О. Б. Давліканова, Т. Д. Іщенко, А. Б. Чайковська. – 2020. – Режим доступу до ресурсу: http://library.fes.de/pdf-files/bueros/ukraine/16469.pdf. С.47.

11. Дуальна освіта: маленькі кроки до вирішення глобальних проблем [Електрон. ресурс]. – Режим доступу до ресурсу: https://lb.ua/blog/kostiantyn_shvabii/489960_dualna_osvita_malenki_kroki.html.

12. Вебсторінка «Дуальне навчання»: [Електрон. ресурс]. – Режим доступу до ресурсу: http://www.tsatu.edu.ua/dualne-navchannja/.

13. Вебсторінка «Дуальна освіта»: [Електрон. ресурс]. – Режим доступу до ресурсу: https://ukd.edu.ua/dualna-osvita.

14. Вебсторінка «Дуальна освіта»: [Електрон. ресурс]. – Режим доступу до ресурсу: https://kart.edu.ua/unit/cpp/dualna-osvita.

15. Щедріна, О. І. Інтернет-технології в бізнесі : навч. посіб. / О. І. Щедріна, М. М. Агутін. — К. : КНЕУ, 2012. — 303 с.

16. Базові поняття і терміни веб-технологій / [А. В. Кільченко, О. І. Поповський, О- р В. Тебенко, О-й. В. Тебенко, Н. М. Матросова]; Упорядник: Кільченко, А. В. – К. : ІІТЗН НАПН України, 2014. – 49 с.

17. What is a web server? [Electronic resource]. – Access mode: https://developer.mozilla.org/en-US/docs/Learn/Common_questions/What_is_a_web_server.

18. Модель взаємодії сервера та браузера [Електрон. ресурс]. – Режим доступу до ресурсу: https://developer.mozilla.org/en-US/docs/Learn/Server-side/First_steps/Introduction/web_application_with_html_and_steps.png.

19. Introduction to the server side [Electronic resource]. – Access mode: https://developer.mozilla.org/en-US/docs/Learn/Server-side/First_steps/Introduction.

20. Основні поняття мови HTML та структура документів [Електрон. ресурс]. – Режим доступу до ресурсу: https://sites.google.com/site/vivcaemowebdizajndistancijno/html/lekcia-3-osnovni-ponatta-movi-html-ta-struktura-dokumentiv

ДОДАТОК А

Текст програми

А.1 Текст файлу contact/admin.py

```
from django.contrib import admin
from .models import ContactModel, ContactLink, About, Social, ImageAbout


class ImageAboutInline(admin.StackedInline):
    model = ImageAbout
    extra = 1


@admin.register(ContactModel)
class ContactModelAdmin(admin.ModelAdmin):
    list_display = ["id", "name", "email", "create_at"]
    list_display_links = ("name",)


@admin.register(About)
class AboutAdmin(admin.ModelAdmin):
    inlines = [ImageAboutInline]


admin.site.register(ContactLink)
admin.site.register(Social)
```

А.2 Текст файлу contact/models.py

```
from ckeditor.fields import RichTextField
from django.db import models

class ContactModel(models.Model):
    class Meta:
        verbose_name = 'Контакт'
        verbose_name_plural = 'Контакти'

    name = models.CharField(max_length=50, verbose_name = "Ім'я")
    email = models.EmailField()
   website = models.URLField(blank=True, null=True, verbose_name = 'Вебсайт')
        message = models.TextField(max_length=5000, verbose_name =
'Повідомлення')
    create_at = models.DateTimeField(auto_now_add=True)

    def __str__(self):
        return f'{self.name} - {self.email}'

class ContactLink(models.Model):
    class Meta:
```

```
            verbose_name = 'Контакт лінк'
            verbose_name_plural = 'Контакти лінків'

        icon = models.FileField(upload_to="icons/")
        name = models.CharField(max_length=200)

        def __str__(self):
            return self.name

    class About(models.Model):
        class Meta:
            verbose_name = 'Про нас'
            verbose_name_plural = 'Про нас'

        name = models.CharField(max_length=50, default='')
        text = RichTextField()
        mini_text = RichTextField()

        def get_first_image(self):
            item = self.about_images.first()
            return item.image.url

        def get_images(self):
            return self.about_images.order_by('id')[1:]


    class ImageAbout(models.Model):
        class Meta:
            verbose_name = 'Зображення сторінки про нас'
            verbose_name_plural = 'Зображення сторінки про нас'
        """ Класс модели изображений страницы о нас """
        image = models.ImageField(upload_to="about/")
              page    =    models.ForeignKey(About,    on_delete=models.CASCADE,
related_name="about_images")
        alt = models.CharField(max_length=100)


    class Social(models.Model):
        class Meta:
            verbose_name = 'Соц. мережі про нас'
            verbose_name_plural = 'Соц. мережі про нас'
        """ Класс модели соц. сетей страницы о нас """
        icon = models.FileField(upload_to="icons/")
        name = models.CharField(max_length=200)
        link = models.URLField()
```

А.3 Текст файлу contact/views.py

```
        from django.shortcuts import render
        from django.views import View
        from django.views.generic import CreateView

        from .models import ContactLink, About
        from .forms import ContactForm
        class ContactView(View):
            def get(self, request):
                contacts = ContactLink.objects.all()
                form = ContactForm()
                 return render(request, 'contact/index.html', {"contacts": contacts,
"form": form})


        class CreateContact(CreateView):
            form_class = ContactForm
            success_url = '/'


        class AboutView(View):
            def get(self, request):
                about = About.objects.last()
                return render(request, 'contact/about.html', {"about": about})
```

A.4 Текст файлу dualis/settings.py

```
        from pathlib import Path
        import os

        BASE_DIR = Path(__file__).resolve().parent.parent


        SECRET_KEY       =       'django-insecure-sd$r9gm9y^b-u1c%w0fu2$5yq%f1f4yf7-
$uur_p4+r%@o8+u7'

        DEBUG = True

        ALLOWED_HOSTS = []


        # Application definition

        INSTALLED_APPS = [
            'django.contrib.admin',
            'django.contrib.auth',
            'django.contrib.contenttypes',
            'django.contrib.sessions',
            'django.contrib.messages',
            'django.contrib.staticfiles',
```

```
    'ckeditor',

    'news.apps.NewsConfig',
    'contact.apps.ContactConfig',
]

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]

ROOT_URLCONF = 'dualis.urls'

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(BASE_DIR, 'templates')],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]

WSGI_APPLICATION = 'dualis.wsgi.application'


# Database
# https://docs.djangoproject.com/en/4.0/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': BASE_DIR / 'db.sqlite3',
    }
}


# Password validation
```

```
        #         https://docs.djangoproject.com/en/4.0/ref/settings/#auth-password-
validators

        AUTH_PASSWORD_VALIDATORS = [
            {
                                                                        'NAME':
'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
            },
            {
                                                                        'NAME':
'django.contrib.auth.password_validation.MinimumLengthValidator',
            },
            {
                                                                        'NAME':
'django.contrib.auth.password_validation.CommonPasswordValidator',
            },
            {
                                                                        'NAME':
'django.contrib.auth.password_validation.NumericPasswordValidator',
            },
        ]


        # Internationalization
        # https://docs.djangoproject.com/en/4.0/topics/i18n/

        LANGUAGE_CODE = 'uk'

        TIME_ZONE = 'Europe/Kiev'

        USE_I18N = True

        USE_TZ = True


        # Static files (CSS, JavaScript, Images)
        # https://docs.djangoproject.com/en/4.0/howto/static-files/

        STATIC_URL = '/static/'
        STATIC_ROOT = os.path.join(BASE_DIR, 'static')
        STATICFILES_DIRS = [
            os.path.join(BASE_DIR, 'dualis/static'),
        ]

        MEDIA_ROOT = os.path.join(BASE_DIR, 'media')
        MEDIA_URL = '/media/'
        CKEDITOR_UPLOAD_PATH = "uploads/"
        # Default primary key field type
        # https://docs.djangoproject.com/en/4.0/ref/settings/#default-auto-field
```

```
DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'
```

A.5 Текст файлу dualis/urls.py

```
from django.conf import settings
from django.conf.urls.static import static
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path('admin/', admin.site.urls),
    path('ckeditor/', include('ckeditor_uploader.urls')),
    path('', include('news.urls')),
    path('', include('contact.urls')),

]

if settings.DEBUG:
                        urlpatterns        +=        static(settings.MEDIA_URL,
document_root=settings.MEDIA_ROOT)
```

A.6 Текст файлу news/admin.py

```
from django.contrib import admin
from news.models import Profile, Post, Tag, Category


@admin.register(Profile)
class ProfileAdmin(admin.ModelAdmin):
    model = Profile


@admin.register(Tag)
class TagAdmin(admin.ModelAdmin):
    model = Tag


@admin.register(Post)
class PostAdmin(admin.ModelAdmin):
    model = Post

    list_display = (
        "id",
        "title",
        "category",
        "subtitle",
```

```
        "slug",
        "publish_date",
        "published",
    )
    list_filter = (
        "published",
        "publish_date",
        "category",
    )
    list_editable = (
        "title",
        "subtitle",
        "slug",
        "publish_date",
        "published",
    )
    search_fields = (
        "title",
        "subtitle",
        "slug",
        "body",
    )
    prepopulated_fields = {
        "slug": (
            "title",
            "subtitle",
        )
    }
    date_hierarchy = "publish_date"
    save_on_top = True

@admin.register(Category)
class CategoryAdmin(admin.ModelAdmin):
    model = Category
    list_display = (
        "id",
        "title"
    )
    list_display_link = (
         "id",
         "title"
    )
    search_fields = (
        "title",
    )
```

А.7 Текст файлу news/forms.py

```
from turtle import title
from django import forms
from .models import Category, Profile

class NewsForm(forms.Form):
          title    =    forms.CharField(max_length=255,    label='Заголовок',
widget=forms.TextInput(attrs={"class":"form-control"}))
        subtitle   =   forms.CharField(max_length=255,   label='Підзаголовок',
required=False, widget=forms.TextInput(attrs={"class":"form-control"}))
            body    =    forms.CharField(label='Контент',    required=False,
widget=forms.Textarea(attrs={"class":"form-control", "rows":"5"}))
      published = forms.BooleanField(label='Опубліковано', initial=False)
         category   =   forms.ModelChoiceField(queryset=Category.objects.all(),
label='Категорія',                empty_label='Оберіть                категорію',
widget=forms.Select(attrs={"class":"form-control"}))
          author    =    forms.ModelChoiceField(queryset=Profile.objects.all(),
label='Автор',                    empty_label='Оберіть                    Автора',
widget=forms.Select(attrs={"class":"form-control"}))
```

A.8 Текст файлу news/models.py

```
from tabnanny import verbose
from turtle import title
from unicodedata import category
from django.conf import settings
from django.db import models
from django.urls import reverse

class Profile(models.Model):
    class Meta:
        verbose_name = 'Профіль'
        verbose_name_plural = 'Профілі'


    user = models.OneToOneField(
        settings.AUTH_USER_MODEL,
        on_delete=models.PROTECT,
    )
    website = models.URLField(blank=True)
    bio = models.CharField(max_length=240, blank=True)

    def __str__(self):
        return self.user.get_username()

class Tag(models.Model):
    class Meta:
        verbose_name = 'Тег'
```

```
            verbose_name_plural = 'Теги'


        name = models.CharField(max_length=50, unique=True)

        def __str__(self):
            return self.name

    class Post(models.Model):
        class Meta:
            verbose_name = 'Новину'
            verbose_name_plural = 'Новини'
            ordering = ["-publish_date"]

         title = models.CharField(max_length=255, unique=True, verbose_name =
'Заголовок')
         subtitle = models.CharField(max_length=255, blank=True, verbose_name =
'Підзаголовок')
          slug = models.SlugField(max_length=255, unique=True, verbose_name =
'Слаг')
        body = models.TextField(blank=True, verbose_name = 'Контент')
         photo = models.ImageField(upload_to='photos/%Y/%m/%d/', verbose_name =
'Фото')
            meta_description = models.CharField(max_length=150, blank=True,
verbose_name = 'Мета опис')
         date_created = models.DateTimeField(auto_now_add=True, verbose_name =
'Дата створення')
          date_modified = models.DateTimeField(auto_now=True, verbose_name =
'Оновлено')
         publish_date = models.DateTimeField(blank=True, null=True, verbose_name
= 'Дата публікування')
            published = models.BooleanField(default=False, verbose_name =
'Опубліковано')
          category = models.ForeignKey('Category', on_delete=models.PROTECT,
null=True, verbose_name = 'Категорія')

        def get_absolute_url(self):
            return reverse('view_news', kwargs={"news_id": self.pk})

        def __str__(self):
            return self.title

        author = models.ForeignKey(Profile, on_delete=models.PROTECT)
        tags = models.ManyToManyField(Tag, blank=True)

    class Category(models.Model):
        class Meta:
            verbose_name = 'Категорію'
            verbose_name_plural = 'Категорії'
            ordering = ["title"]
```

```
                title    =    models.CharField(max_length=150,    db_index=True,
verbose_name='Заголовок категорії')

            def get_absolute_url(self):
                return reverse('category', kwargs={"category_id": self.pk})

            def __str__(self):
                return self.title
```

А.9 Текст файлу news/urls.py

```
from django.urls import path
from .views import *

urlpatterns = [
    path('', index, name='home'),
    path('category/<int:category_id>/', get_category, name='category'),
    path('news/<int:news_id>/', view_news, name='view_news'),
    path('news/add-news/', add_news, name='add_news'),
]
```

А.10 Текст файлу news/views.py

```
from turtle import title
from unicodedata import category
from django.shortcuts import render, get_object_or_404, redirect
from .models import Post, Category
from .forms import NewsForm


def index(request):
    post = Post.objects.all()

    context = {'news': post, 'title': 'Список новин', }
    return render(request, 'news/index.html', context)

def get_category(request, category_id):
    post = Post.objects.filter(category_id=category_id)

    category = Category.objects.get(pk=category_id)
    context = {'news': post, 'category': category, }
    return render(request, 'news/category.html', context)

def view_news(request, news_id):
    news_item = get_object_or_404(Post, pk=news_id)
```

```
        context = {'news_item': news_item, }
        return render(request, 'news/view_news.html', context)

    def add_news(request):
        if request.method == 'POST':
            form = NewsForm(request.POST)
            if form.is_valid():
                news = Post.objects.create(**form.cleaned_data)
                return redirect(news)
        else:
            form = NewsForm()
        return render(request, 'news/add_news.html', {'form': form})
```

A.11 Текст файлу news/templates/category.html

```
{% extends 'base.html' %}

{% block title %}

{{ category.title }} :: {{ block.super }}

{% endblock %}

{% block sidebar %}
{% include 'inc/_sidebar.html' %}
{% endblock %}

{% block content %}

{% for item in news %}
<div class="card  mb-3">
    <div class="card-header">
        Категорія: {{ item.category }}
    </div>
    <div class="card-body">

        <div class="media">
            {% if item.photo %}
            <img src="{{ item.photo.url }}" alt="..." width="350" class="mr-
3">
            {% else %}
           <img src="https://picsum.photos/id/1060/350/235/?blur=2" alt="..."
class="mr-3">
            {% endif%}
            <div class="media-body">
                <h5 class="card-title">{{ item.title }}</h5>
                <p class="card-text">{{ item.subtitle|linebreaks }}</p>
```

```
                        <a href="{{ item.get_absolute_url }}" class="btn btn-
primary">Читати далі...</a>
                    </div>
                </div>

            </div>
            <div class="card-footer text-muted">
                {{ item.date_created|date:"d-m-Y H:i" }}
            </div>
        </div>
        {% endfor %}

        {% endblock %}
```

A.12 Текст файлу news/templates/view_news.html

```
        {% extends 'base.html' %}
        {% block title %}
        {{ news_item.title }} :: {{ block.super }}
        {% endblock %}

        {% block sidebar %}

        {% include 'inc/_sidebar.html' %}

        {% endblock %}

        {% block content %}

        <div class="card  mb-3">
            <div class="card-header">
                 Категорія: <a href="{{ news_item.category.get_absolute_url }}">{{
news_item.category }}</a>
            </div>
            <div class="card-body">

                <div class="media">
                    {% if news_item.photo %}
                     <img src="{{ news_item.photo.url }}" height="auto" width="100%"
max-width="750" alt="..." class="mr-3">
                    {% else %}
                   <img src="https://picsum.photos/id/1060/750/235/?blur=2" alt="..."
class="mr-3">
                    {% endif%}
                    <div class="media-body">
                        <h5 class="card-title">{{ news_item.title }}</h5>
                        <p class="card-text">{{ news_item.body|linebreaks }}</p>
```

```
            </div>
        </div>

    </div>
    <div class="card-footer text-muted">
        {{ news_item.date_created|date:"d-m-Y H:i" }}
    </div>
</div>

{% endblock %}
```

А.13 Текст файлу templates/base.html

```
{% load static %}
<!doctype html>
<html lang="en">

<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="{% static 'bootstrap/css/bootstrap.min.css'
%}">
    <link rel="stylesheet" href="{% static 'css/style.css' %} ">
    <title>{% block title %}Дуальна освіта{% endblock %}</title>
</head>

<body>
    {% include 'inc/_nav.html' %}
    <div class="container mt-3">
        <div class="row">
            <div class="col-md-3">
                {% block sidebar %}SIDEBAR{% endblock%}
            </div>
            <div class="col-md-9">
                {% block content %}CONTENT{% endblock%}
            </div>

        </div>
    </div>
    <script src="{% static 'bootstrap/js/bootstrap.min.js' %}"></script>
</body>

</html>
```

А.14 Текст файлу templates/_nav.html

```
<nav class="navbar navbar-expand-lg navbar-light bg-light">
    <div class="container-fluid">
        <a class="navbar-brand" href="{% url 'home' %}">Дуальна освіта</a>
                <button   class="navbar-toggler"   type="button"   data-bs-
toggle="collapse"
                                data-bs-target="#navbarSupportedContent"   aria-
controls="navbarSupportedContent" aria-expanded="false"
                aria-label="Toggle navigation">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarSupportedContent">
                <ul class="navbar-nav me-auto mb-2 mb-lg-0">
                    <li class="nav-item"> <a class="nav-link" href="{% url 'home'
%}">Головна</a></li>
                    <li class="nav-item dropdown">
                                <a  class="nav-link  dropdown-toggle"  href="#"
id="navbarDropdown" role="button"
                            data-bs-toggle="dropdown" aria-expanded="false">
                            Нормативна база
                        </a>
                        <ul class="dropdown-menu" aria-labelledby="navbarDropdown">
                            <li><a class="dropdown-item" href="/">НУ “Запорізька
політехніка”</a></li>
                              <li><a class="dropdown-item" href="/">Міністерство
освіти і науки</a></li>
                            <li>
                                <hr class="dropdown-divider">
                            </li>
                              <li><a class="dropdown-item" href="/">Досвід інших
ЗВО</a></li>
                        </ul>
                    </li>

                        <li class="nav-item"> <a class="nav-link" href="{% url
'contact' %}">Контакти</a></li>
                    <li class="nav-item"> <a class="nav-link" href="{% url 'about'
%}">Про нас</a></li>
                </ul>

            </div>
        </div>
    </nav>
```

А.15 Текст файлу бази даних dualis/db.sqlite3

```
CREATE TABLE "main"."auth_group" (
```

```
  "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
  "name" varchar(150) NOT NULL,
  UNIQUE ("name" ASC)
);

CREATE TABLE "main"."auth_group_permissions" (
  "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
  "group_id" integer NOT NULL,
  "permission_id" integer NOT NULL,
  FOREIGN KEY ("group_id") REFERENCES "auth_group" ("id") ON DELETE NO ACTION ON UPDATE NO ACTION DEFERRABLE INITIALLY DEFERRED,
  FOREIGN KEY ("permission_id") REFERENCES "auth_permission" ("id") ON DELETE NO ACTION ON UPDATE NO ACTION DEFERRABLE INITIALLY DEFERRED
);

CREATE INDEX "main"."auth_group_permissions_group_id_b120cbf9"
ON "auth_group_permissions" (
  "group_id" ASC
);

CREATE UNIQUE INDEX "main"."auth_group_permissions_group_id_permission_id_0cd325b0_uniq"
ON "auth_group_permissions" (
  "group_id" ASC,
  "permission_id" ASC
);

CREATE INDEX "main"."auth_group_permissions_permission_id_84c5c92e"
ON "auth_group_permissions" (
  "permission_id" ASC
);

CREATE TABLE "main"."auth_permission" (
  "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
  "content_type_id" integer NOT NULL,
```

```
  "codename" varchar(100) NOT NULL,
  "name" varchar(255) NOT NULL,
  FOREIGN KEY ("content_type_id") REFERENCES "django_content_type" ("id") ON DELETE NO ACTION ON UPDATE NO ACTION DEFERRABLE INITIALLY DEFERRED
);

INSERT INTO "main"."sqlite_sequence" (name, seq) VALUES ('auth_permission', '60');

CREATE INDEX "main"."auth_permission_content_type_id_2f476e4b"
ON "auth_permission" (
  "content_type_id" ASC
);

CREATE UNIQUE INDEX "main"."auth_permission_content_type_id_codename_01ab375a_uniq"
ON "auth_permission" (
  "content_type_id" ASC,
  "codename" ASC
);

CREATE TABLE "main"."auth_user" (
  "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
  "password" varchar(128) NOT NULL,
  "last_login" datetime,
  "is_superuser" bool NOT NULL,
  "username" varchar(150) NOT NULL,
  "last_name" varchar(150) NOT NULL,
  "email" varchar(254) NOT NULL,
  "is_staff" bool NOT NULL,
  "is_active" bool NOT NULL,
  "date_joined" datetime NOT NULL,
  "first_name" varchar(150) NOT NULL,
  UNIQUE ("username" ASC)
);
```

```
INSERT INTO "main"."sqlite_sequence" (name, seq) VALUES ('auth_user', '2');

CREATE TABLE "main"."auth_user_groups" (
  "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
  "user_id" integer NOT NULL,
  "group_id" integer NOT NULL,
  FOREIGN KEY ("user_id") REFERENCES "auth_user" ("id") ON DELETE NO ACTION ON UPDATE NO ACTION DEFERRABLE INITIALLY DEFERRED,
  FOREIGN KEY ("group_id") REFERENCES "auth_group" ("id") ON DELETE NO ACTION ON UPDATE NO ACTION DEFERRABLE INITIALLY DEFERRED
);

CREATE INDEX "main"."auth_user_groups_group_id_97559544"
ON "auth_user_groups" (
  "group_id" ASC
);

CREATE INDEX "main"."auth_user_groups_user_id_6a12ed8b"
ON "auth_user_groups" (
  "user_id" ASC
);

CREATE UNIQUE INDEX "main"."auth_user_groups_user_id_group_id_94350c0c_uniq"
ON "auth_user_groups" (
  "user_id" ASC,
  "group_id" ASC
);

CREATE TABLE "main"."auth_user_user_permissions" (
  "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
  "user_id" integer NOT NULL,
  "permission_id" integer NOT NULL,
  FOREIGN KEY ("user_id") REFERENCES "auth_user" ("id") ON DELETE NO ACTION ON UPDATE NO ACTION DEFERRABLE INITIALLY DEFERRED,
```

```
    FOREIGN KEY ("permission_id") REFERENCES "auth_permission" ("id") ON DELETE NO ACTION ON UPDATE NO ACTION DEFERRABLE INITIALLY DEFERRED
);

CREATE INDEX "main"."auth_user_user_permissions_permission_id_1fbb5f2c"
ON "auth_user_user_permissions" (
  "permission_id" ASC
);

CREATE INDEX "main"."auth_user_user_permissions_user_id_a95ead1b"
ON "auth_user_user_permissions" (
  "user_id" ASC
);

CREATE UNIQUE INDEX "main"."auth_user_user_permissions_user_id_permission_id_14a6b632_uniq"
ON "auth_user_user_permissions" (
  "user_id" ASC,
  "permission_id" ASC
);

CREATE TABLE "main"."contact_about" (
  "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
  "name" varchar(50) NOT NULL,
  "text" text NOT NULL,
  "mini_text" text NOT NULL
);

INSERT INTO "main"."sqlite_sequence" (name, seq) VALUES ('contact_about', '2');

CREATE TABLE "main"."contact_contactlink" (
  "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
  "icon" varchar(100) NOT NULL,
  "name" varchar(200) NOT NULL
);
```

```
INSERT INTO "main"."sqlite_sequence" (name, seq) VALUES ('contact_contactlink', '5');

CREATE TABLE "main"."contact_contactmodel" (
  "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
  "name" varchar(50) NOT NULL,
  "email" varchar(254) NOT NULL,
  "website" varchar(200),
  "message" text NOT NULL,
  "create_at" datetime NOT NULL
);

INSERT INTO "main"."sqlite_sequence" (name, seq) VALUES ('contact_contactmodel', '3');

CREATE TABLE "main"."contact_imageabout" (
  "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
  "image" varchar(100) NOT NULL,
  "alt" varchar(100) NOT NULL,
  "page_id" bigint NOT NULL,
  FOREIGN KEY ("page_id") REFERENCES "contact_about" ("id") ON DELETE NO ACTION ON UPDATE NO ACTION DEFERRABLE INITIALLY DEFERRED
);

INSERT INTO "main"."sqlite_sequence" (name, seq) VALUES ('contact_imageabout', '2');

CREATE INDEX "main"."contact_imageabout_page_id_145491fc"
ON "contact_imageabout" (
  "page_id" ASC
);

CREATE TABLE "main"."contact_social" (
  "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
  "icon" varchar(100) NOT NULL,
  "name" varchar(200) NOT NULL,
```

```
  "link" varchar(200) NOT NULL
);

INSERT INTO "main"."sqlite_sequence" (name, seq) VALUES ('contact_social', '1');

CREATE TABLE "main"."django_admin_log" (
  "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
  "action_time" datetime NOT NULL,
  "object_id" text,
  "object_repr" varchar(200) NOT NULL,
  "change_message" text NOT NULL,
  "content_type_id" integer,
  "user_id" integer NOT NULL,
  "action_flag" smallint unsigned NOT NULL,
  FOREIGN KEY ("content_type_id") REFERENCES "django_content_type" ("id") ON DELETE NO ACTION ON UPDATE NO ACTION DEFERRABLE INITIALLY DEFERRED,
  FOREIGN KEY ("user_id") REFERENCES "auth_user" ("id") ON DELETE NO ACTION ON UPDATE NO ACTION DEFERRABLE INITIALLY DEFERRED,
  ("action_flag" >= 0)
);

INSERT INTO "main"."sqlite_sequence" (name, seq) VALUES ('django_admin_log', '30');

CREATE INDEX "main"."django_admin_log_content_type_id_c4bce8eb"
ON "django_admin_log" (
  "content_type_id" ASC
);

CREATE INDEX "main"."django_admin_log_user_id_c564eba6"
ON "django_admin_log" (
  "user_id" ASC
);

CREATE TABLE "main"."django_content_type" (
  "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
```

```
  "app_label" varchar(100) NOT NULL,
  "model" varchar(100) NOT NULL
);

INSERT INTO "main"."sqlite_sequence" (name, seq) VALUES ('django_content_type', '15');

CREATE UNIQUE INDEX "main"."django_content_type_app_label_model_76bd3d3b_uniq"
ON "django_content_type" (
  "app_label" ASC,
  "model" ASC
);

CREATE TABLE "main"."news_category" (
  "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
  "title" varchar(150) NOT NULL
);

INSERT INTO "main"."sqlite_sequence" (name, seq) VALUES ('news_category', '3');

CREATE INDEX "main"."news_category_title_175a164c"
ON "news_category" (
  "title" ASC
);

CREATE TABLE "main"."news_post" (
  "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
  "title" varchar(255) NOT NULL,
  "subtitle" varchar(255) NOT NULL,
  "slug" varchar(255) NOT NULL,
  "body" text NOT NULL,
  "photo" varchar(100) NOT NULL,
  "meta_description" varchar(150) NOT NULL,
  "date_created" datetime NOT NULL,
```

```
  "date_modified" datetime NOT NULL,
  "publish_date" datetime,
  "published" bool NOT NULL,
  "author_id" bigint NOT NULL,
  "category_id" bigint,
  FOREIGN KEY ("author_id") REFERENCES "news_profile" ("id") ON DELETE NO ACTION ON UPDATE NO ACTION DEFERRABLE INITIALLY DEFERRED,
  FOREIGN KEY ("category_id") REFERENCES "news_category" ("id") ON DELETE NO ACTION ON UPDATE NO ACTION DEFERRABLE INITIALLY DEFERRED,
  UNIQUE ("title" ASC),
  UNIQUE ("slug" ASC)
);

INSERT INTO "main"."sqlite_sequence" (name, seq) VALUES ('news_post', '5');

CREATE INDEX "main"."news_post_author_id_d49395b0"
ON "news_post" (
  "author_id" ASC
);

CREATE INDEX "main"."news_post_category_id_efb3abe5"
ON "news_post" (
  "category_id" ASC
);

CREATE TABLE "main"."news_post_tags" (
  "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
  "post_id" bigint NOT NULL,
  "tag_id" bigint NOT NULL,
  FOREIGN KEY ("post_id") REFERENCES "news_post" ("id") ON DELETE NO ACTION ON UPDATE NO ACTION DEFERRABLE INITIALLY DEFERRED,
  FOREIGN KEY ("tag_id") REFERENCES "news_tag" ("id") ON DELETE NO ACTION ON UPDATE NO ACTION DEFERRABLE INITIALLY DEFERRED
);
```

```
INSERT INTO "main"."sqlite_sequence" (name, seq) VALUES ('news_post_tags', '7');

CREATE INDEX "main"."news_post_tags_post_id_4edd83c1"
ON "news_post_tags" (
  "post_id" ASC
);

CREATE UNIQUE INDEX "main"."news_post_tags_post_id_tag_id_a1986387_uniq"
ON "news_post_tags" (
  "post_id" ASC,
  "tag_id" ASC
);

CREATE INDEX "main"."news_post_tags_tag_id_f6adfa19"
ON "news_post_tags" (
  "tag_id" ASC
);

CREATE TABLE "main"."news_profile" (
  "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
  "website" varchar(200) NOT NULL,
  "bio" varchar(240) NOT NULL,
  "user_id" integer NOT NULL,
  FOREIGN KEY ("user_id") REFERENCES "auth_user" ("id") ON DELETE NO ACTION ON UPDATE NO ACTION DEFERRABLE INITIALLY DEFERRED,
  UNIQUE ("user_id" ASC)
);
INSERT INTO "main"."sqlite_sequence" (name, seq) VALUES ('news_profile', '1');
CREATE TABLE "main"."news_tag" (
  "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
  "name" varchar(50) NOT NULL,
  UNIQUE ("name" ASC)
);
INSERT INTO "main"."sqlite_sequence" (name, seq) VALUES ('news_tag', '4');
```

ДОДАТОК Б

Слайди презентації

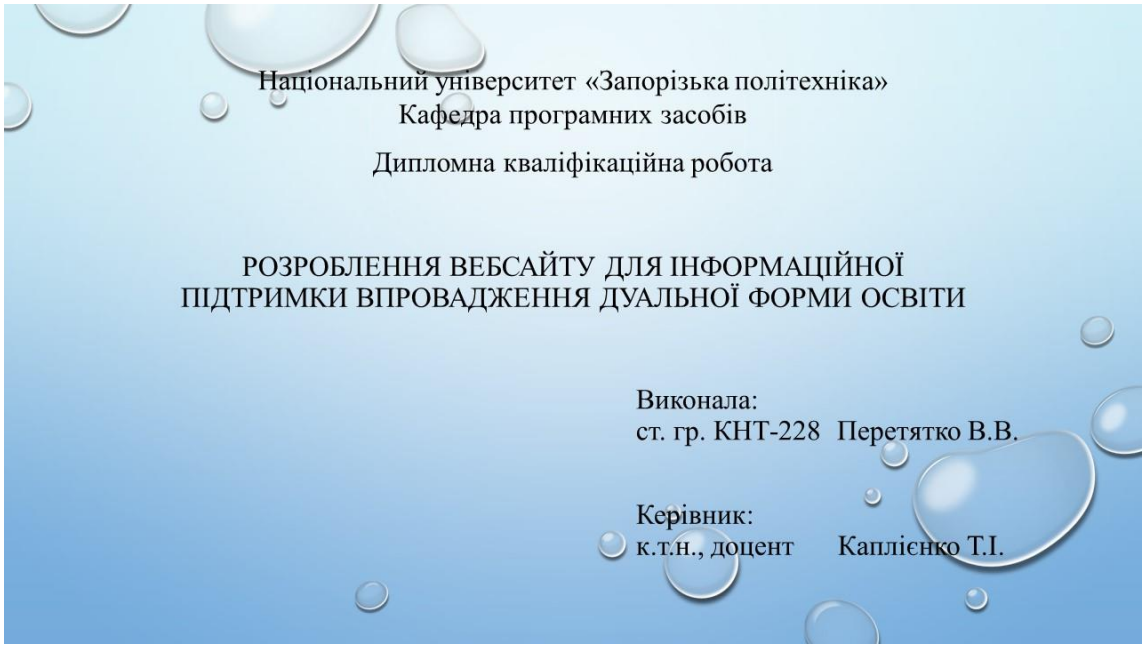


Рисунок Б.1 – Слайд «Титульний лист»

Рисунок Б.2 – Слайд «Об’єкт, предмет та мета дослідження»

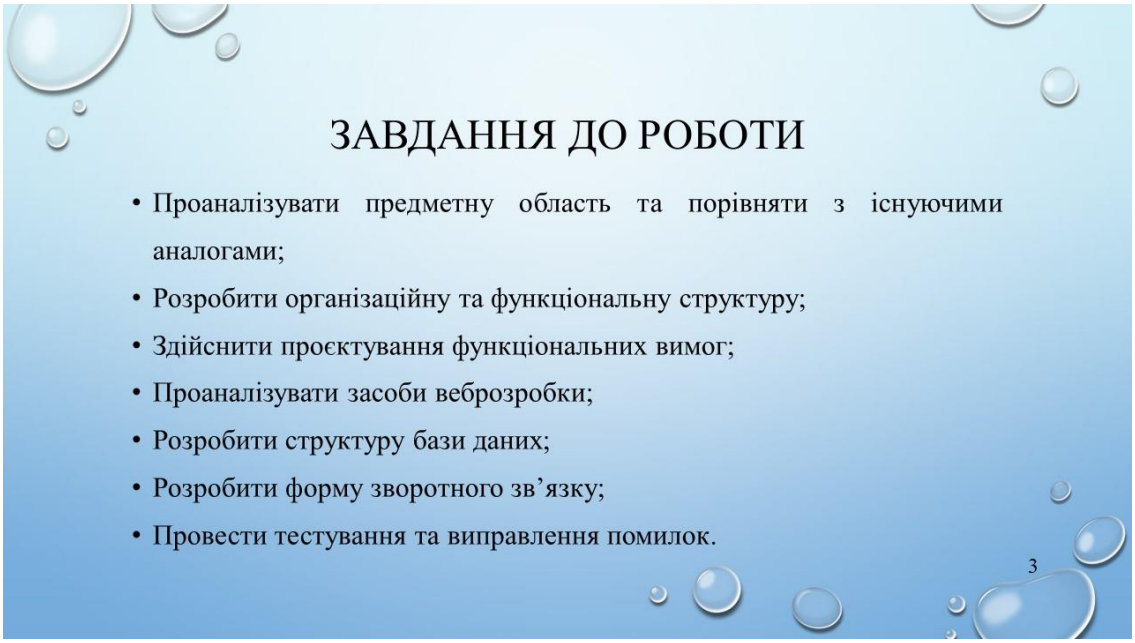
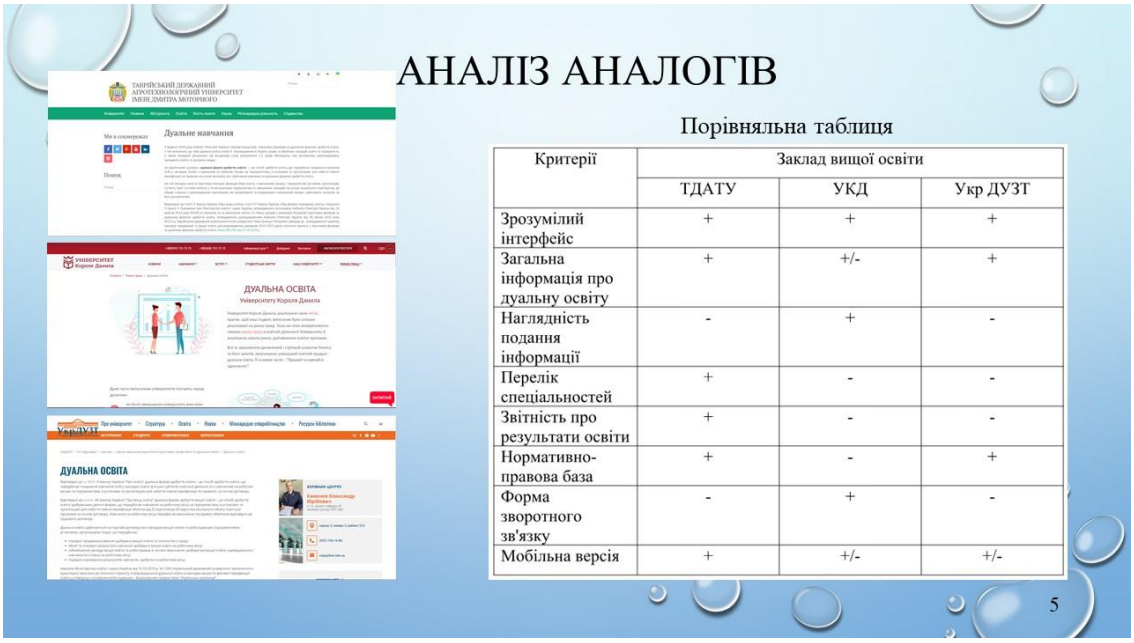


Рисунок Б.3 – Слайд «Завдання до роботи»

Рисунок Б.4 – Слайд «Аналіз предметної області»



Критерії	Заклад вищої освіти		
	ТДАТУ	УКД	Укр ДУЗТ
Зрозумілий інтерфейс	+	+	+
Загальна інформація про дуальну освіту	+	+/-	+
Наглядність подання інформації	-	+	-
Перелік спеціальностей	+	-	-
Звітність про результати освіти	+	-	-
Нормативно-правова база	+	-	+
Форма зворотного зв'язку	-	+	-
Мобільна версія	+	+/-	+/-

Рисунок Б.5 – Слайд «Аналіз аналогів»

Рисунок Б.6 – Слайд «Структура сайту»

Рисунок Б.7 – Слайд «Діаграма прецедентів»

Характеристика	PHP	Python	Ruby
Рік створення	1994	1991	1995
Динамічна типізація	+	+	+
Об'єктно-орієнтованість	+	+	+
Можливість компілювання	+	+	+
Наявність бібліотек та фреймворків	+	+	+
Парадигми програмування:	Мульти-парадигмальний	Мульти-парадигмальний	Мульти-парадигмальний

Рисунок Б.8 – Слайд «Вибір мови програмування»

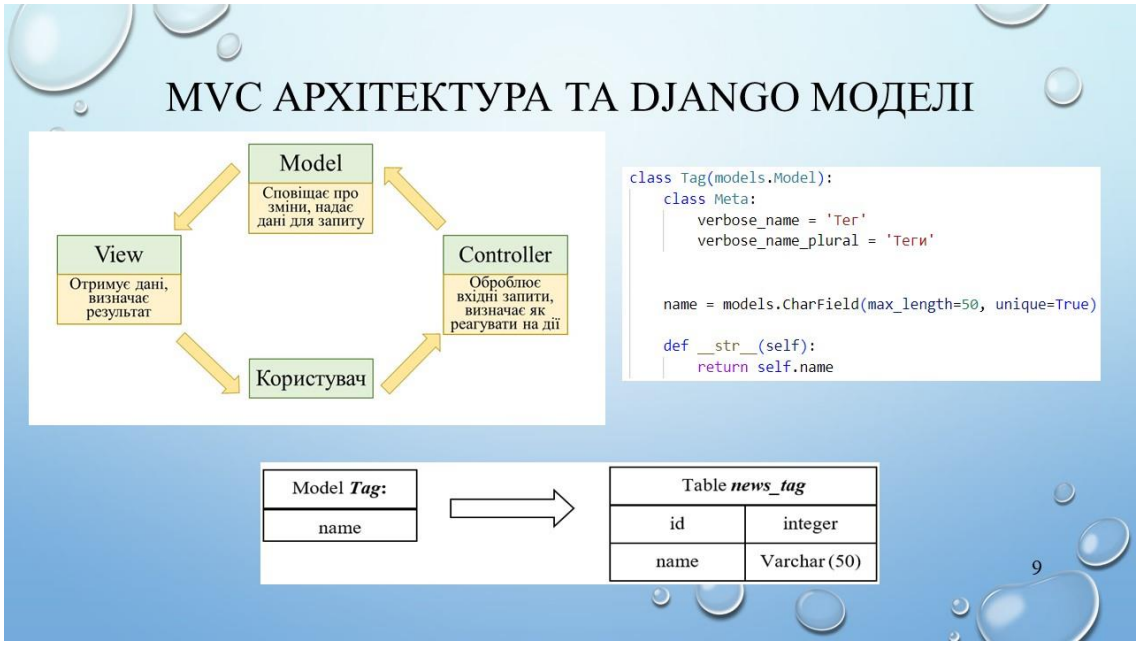


Рисунок Б.9 – Слайд «Django архітектура та моделі»

Рисунок Б.10 – Слайд «Схема бази даних»

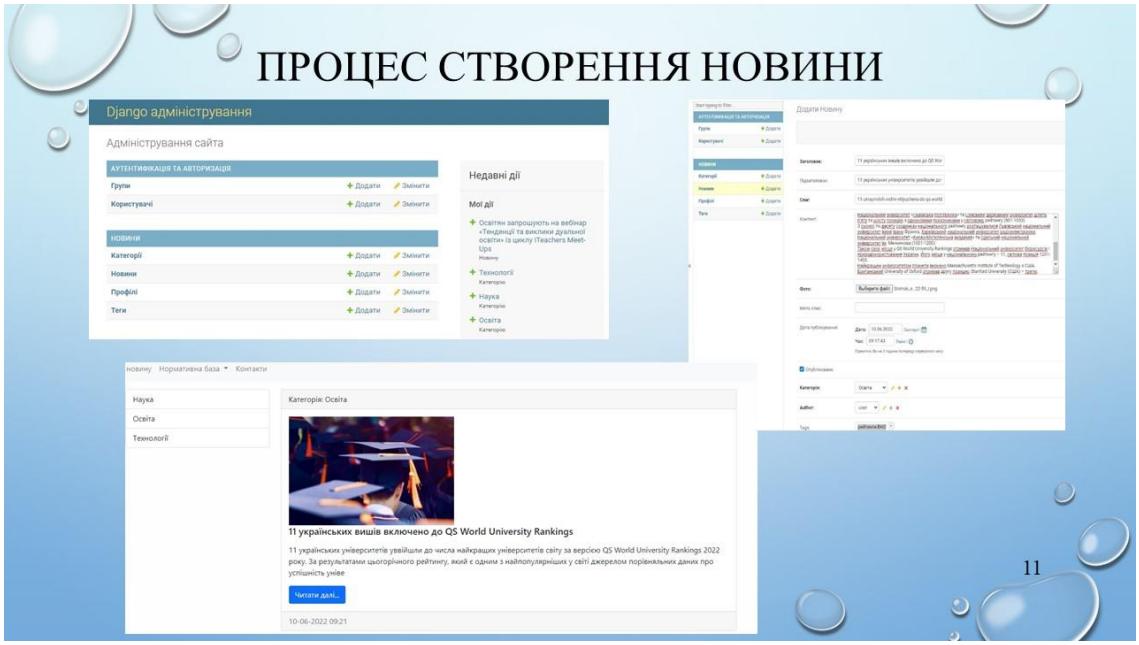


Рисунок Б.11 – Слайд «Процес створення новини»

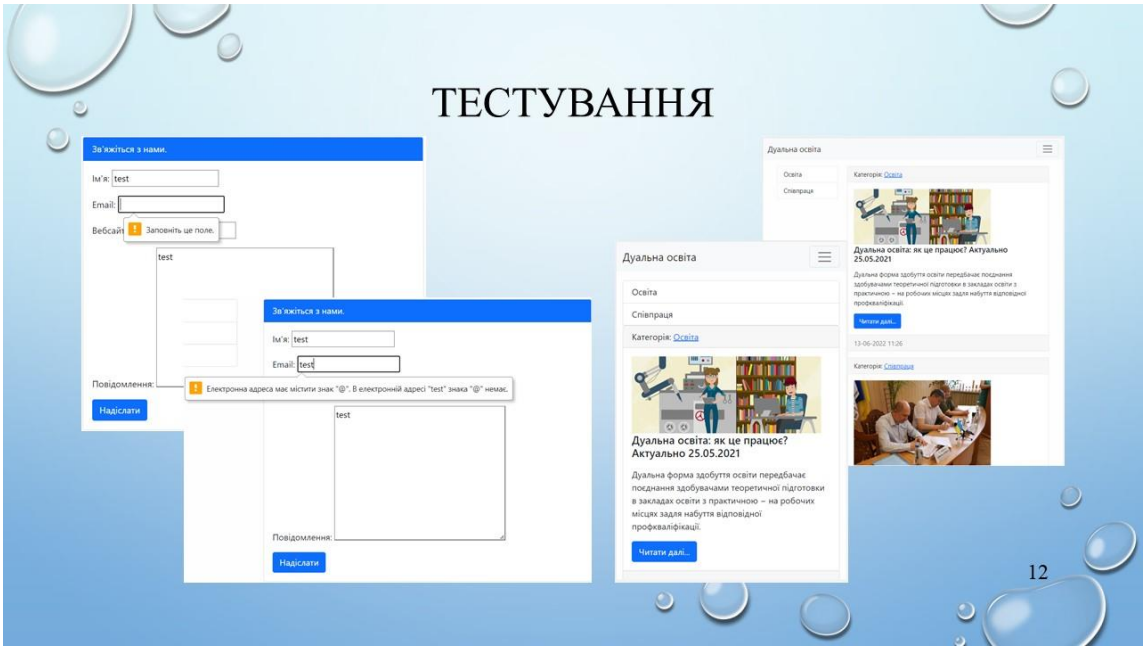


Рисунок Б.12 – Слайд «Тестування»

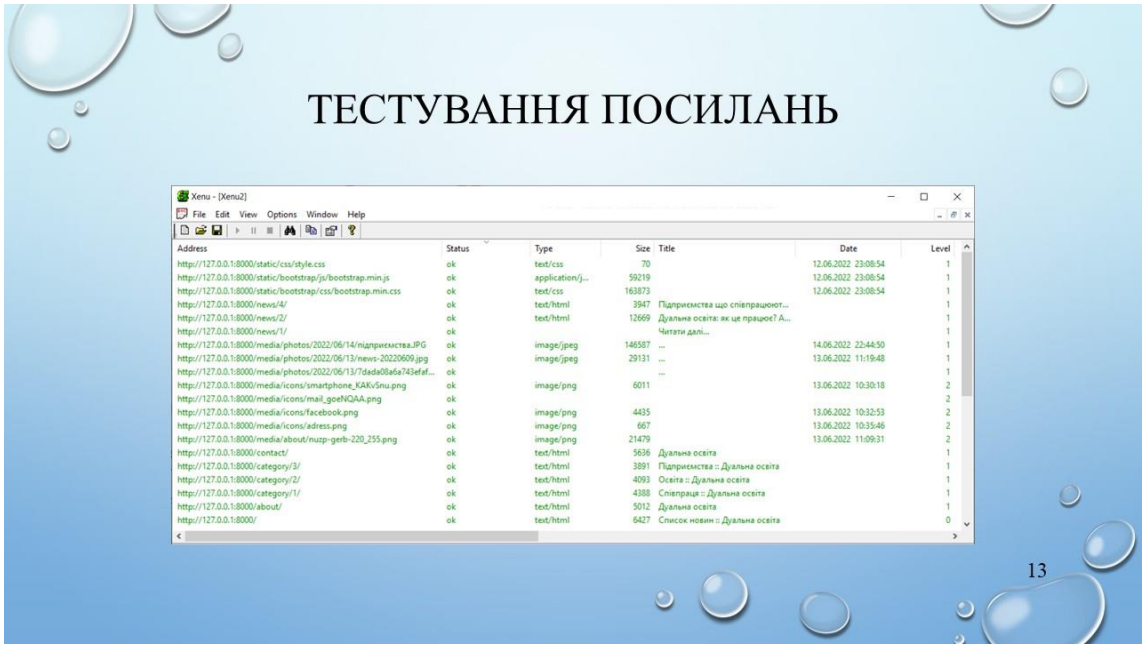


Рисунок Б.13 – Слайд «Тестування посилань»

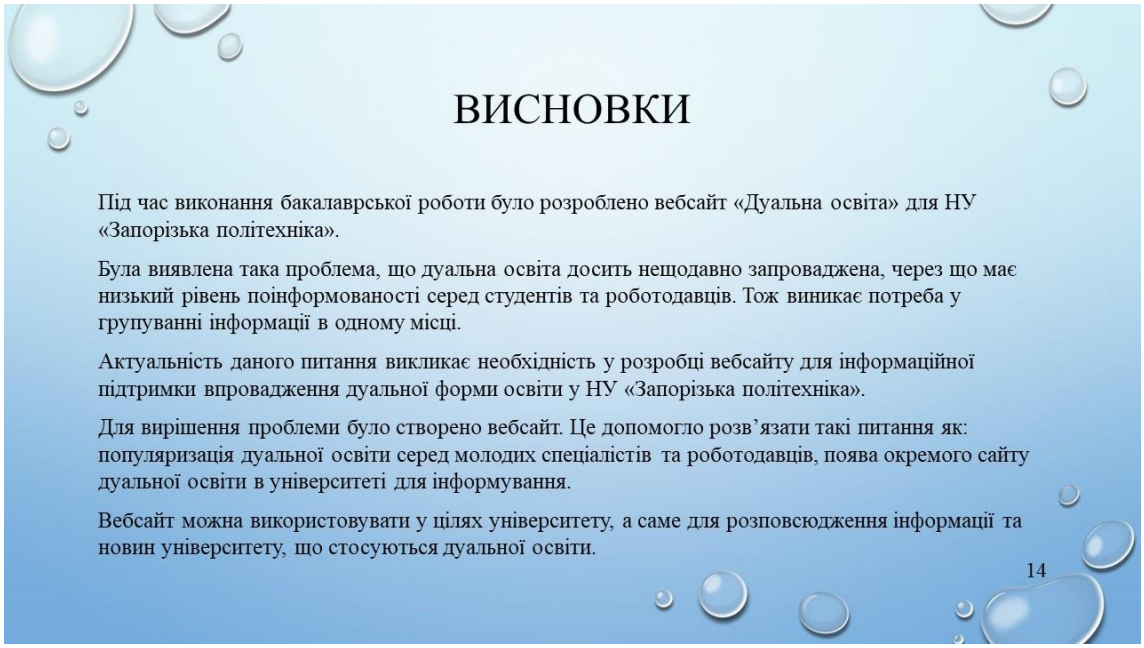


Рисунок Б.14 – Слайд «Висновки»