КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ

автоматизації і інформаційних технологій

(факультет)

інформаційних технологій проектування та прикладної математики

(кафедра)

ПОЯСНЮВАЛЬНА ЗАПИСКА
ДО КВАЛІФІКАЦІЙНОЇ РОБОТИ

НА ЗДОБУТТЯ ОСВІТНЬОГО РІВНЯ «БАКАЛАВР»

на тему: «Методи створення та оптимізації вебінтерфейсів у сучасних інформаційних системах»

ГОВОРУНЕЦЬ СЕРГІЙ МИКОЛАЙОВИЧ
(прізвище, ім’я та по батькові студента повністю)

Київ 2026 р.

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

БУДІВНИЦТВА І АРХІТЕКТУРИ

автоматизації і інформаційних технологій

(факультет)

інформаційних технологій проектування та прикладної математики

(кафедра)

ЗАТВЕРДЖУЮ

Завідувач кафедри ІТППМ

д.т.н., професор Бородавка Є.В.

_____

„___” _____2026 року

ПОЯСНЮВАЛЬНА ЗАПИСКА
ДО КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА ЗДОБУТТЯ ОСВІТНЬОГО РІВНЯ «БАКАЛАВР»

на тему: «Методи створення та оптимізації вебінтерфейсів у сучасних інформаційних системах»

Виконав: студент 4-го курсу, групи ІСТ-22

Спеціальності: 126 «Інформаційні системи та технології»

(шифр і назва спеціальності)

Говорунець С.М.

(прізвище та ініціали)

Керівник д.т.н., професор Терентьєв О.О.

(прізвище та ініціали)

Рецензент

(прізвище та ініціали)

Київ, 2026 р.

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БУДІВНИЦТВА І АРХІТЕКТУРИ

автоматизації і інформаційних технологій

(факультет)

інформаційних технологій проектування та прикладної математики

(кафедра)

Освітній рівень: «бакалавр»

Спеціальність: 126 «Інформаційні системи та технології»

Освітня програма: «Інформаційні системи та технології»

1. Тема роботи: «Методи створення та оптимізації вебінтерфейсів у сучасних інформаційних системах» затверджена наказом ректора КНУБА № 444/52-15/13.6/26 від “08” квітня 2026 року.
2. Керівник роботи: Терентьєв Олександр Олександрович, д.т.н, професор кафедри ІТППМ.
3. Строк подання студентом роботи до захисту: червень 2026 року.
4. Зміст пояснювальної записки за розділами:

   - Р.1. Аналіз предметної області створення вебінтерфейсів.
   - Р.2. Аналіз та аудит існуючого вебінтерфейсу об'єкта дослідження.
   - Р.3. Проєктування та розробка оптимізованого вебінтерфейсу інформаційної системи.

5. Перелік демонстраційного матеріалу (інформаційні слайди):

   - С.1. Актуальність, мета та завдання дослідження.
   - С.2. Об’єкт, предмет дослідження та аналіз предметної області.
   - С.3. Результати технічного аудиту сайту ЧПФК в Google Lighthouse.
   - С.4. Проєктування інформаційної архітектури та UI/UX у Figma.
   - С.5. Програмна реалізація та впроваджені методи оптимізації.
   - С.6. Порівняльний аналіз технічних метрик та загальні висновки.

6. Календарний план виконання кваліфікаційної роботи бакалавра:

Види робіт та їх зміст	Дата виконання
Р. 1. Аналіз предметної області створення вебінтерфейсів	Лютий 2026 р.
Р. 2. Аналіз та аудит існуючого вебінтерфейсу об'єкта дослідження	Березень-Квітень 2026 р.
Р. 3. Проєктування та розробка оптимізованого вебінтерфейсу	Квітень-Травень 2026 р.
Остаточне оформлення роботи та перевірка нормоконтролю	Травень 2026 р.
Направлення роботи на рецензування	Червень 2026 р.
Попередній захист роботи на кафедрі	Червень 2026 р.

8. Дата видачі завдання: 12 січня 2026 р.

Керівник _____ Терентьєв О.О.
(підпис) (прізвище та ініціали)

Бакалавр _____ Говорунець С.М.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

«Методи створення та оптимізації вебінтерфейсів у сучасних інформаційних системах».

Кваліфікаційна робота бакалавра за спеціальністю 126 «Інформаційні системи та технології». Освітня програма: «Інформаційні системи та технології». Київський національний університет будівництва і архітектури. Київ, 2026.

Кваліфікаційна робота присвячена дослідженню сучасних методів створення та оптимізації вебінтерфейсів інформаційних систем. У роботі проведено аналіз теоретичних засад веброзробки, принципів адаптивного дизайну, доступності вебресурсів, пошукової оптимізації та підвищення продуктивності вебзастосунків.

Об'єктом дослідження є вебінтерфейс інформаційної системи закладу освіти.

Предметом дослідження є методи проєктування, розробки та оптимізації вебінтерфейсів з використанням сучасних технологій HTML5, CSS3 та JavaScript.

У практичній частині виконано аудит існуючого вебресурсу коледжу, проведено аналіз показників продуктивності, доступності та SEO за допомогою інструменту Lighthouse. На основі отриманих результатів розроблено новий адаптивний вебінтерфейс, створено дизайн-макети у Figma, реалізовано систему навігації, адаптивне меню, форму з клієнтською валідацією та комплекс заходів з оптимізації швидкодії.

За результатами роботи досягнуто покращення показників продуктивності, доступності та пошукової оптимізації вебресурсу, а також підвищено зручність використання сайту для різних категорій користувачів.

Ключові слова: вебінтерфейс, інформаційна система, веброзробка, адаптивний дизайн, HTML5, CSS3, JavaScript, Lighthouse, SEO, доступність, оптимізація продуктивності.

SUMMARY

"Methods of Creating and Optimizing Web Interfaces in Modern Information Systems".

Bachelor's qualification thesis in specialty 126 "Information Systems and Technologies". Educational program: "Information Systems and Technologies". Kyiv National University of Construction and Architecture. Kyiv, 2026.

The qualification thesis is devoted to the study of modern methods for creating and optimizing web interfaces in information systems. The paper analyzes the theoretical foundations of web development, responsive design principles, web accessibility, search engine optimization, and performance improvement techniques.

The object of research is the web interface of an educational institution information system.

The subject of research is the methods of designing, developing, and optimizing web interfaces using modern HTML5, CSS3, and JavaScript technologies.

The practical part includes an audit of the existing college website, analysis of performance, accessibility, and SEO indicators using the Lighthouse tool. Based on the obtained results, a new responsive web interface was designed and implemented. Figma prototypes, adaptive navigation, a client-side validation form, and performance optimization techniques were developed.

The results demonstrate significant improvements in performance, accessibility, search engine optimization, and overall user experience of the web resource.

Keywords: web interface, information system, web development, responsive design, HTML5, CSS3, JavaScript, Lighthouse, SEO, accessibility, performance optimization.

ЗМІСТ ДИПЛОМНОЇ РОБОТИ

ВСТУП (актуальність, мета, завдання, об'єкт та предмет дослідження)

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ СТВОРЕННЯ ВЕБІНТЕРФЕЙСІВ

- **1.1 Роль вебінтерфейсів у сучасних інформаційних системах**
  - Визначення вебінтерфейсу
  - Значення та критичність
  - Приклади застосування (банкінг, CRM, державні сервіси, соціальні мережі)
- **1.2 Основні принципи проєктування вебінтерфейсів**
  - UX та UI як основа взаємодії
  - Адаптивність та кросплатформеність
  - Доступність вебінтерфейсів
  - Human-centered design
  - Консистентність та візуальна ієрархія
- **1.3 Архітектурні підходи до створення вебінтерфейсів**
  - SPA vs MPA, MVC
  - Сучасні фреймворки та компонентний підхід
- **1.4 Методи оптимізації вебінтерфейсів**
  - 1.4.1 Оптимізація продуктивності (мінімізація ресурсів, lazy loading, кешування, інструмент Lighthouse)
  - 1.4.2 Оптимізація зручності використання (юзабіліті-тестування, A/B тестування, UX-метрики)
  - 1.4.3 Оптимізація доступності (стандарти WCAG, семантична верстка)
- **Висновки до розділу 1**

РОЗДІЛ 2. АНАЛІЗ ТА АУДІТ ІСНУЮЧОГО ВЕБІНТЕРФЕЙСУ ОБ'ЄКТА ДОСЛІДЖЕННЯ

- **2.1 Загальна характеристика інформаційного ресурсу об'єкта дослідження** (Офіційний сайт Черкаського політехнічного фахового коледжу – polytechnic.ck.ua)

- **2.2 Користувацький UX-аналіз та сценарії взаємодії**
    - Портрети цільових груп користувачів (абітурієнти, студенти, викладачі)
    - Аналіз сценаріїв навігації та виявлені юзабіліті-проблеми
- **2.3 Поглиблений технічний аудит вебінтерфейсу за допомогою Google Lighthouse**
    - 2.3.1 Проблеми продуктивності сайту (Performance)
    - 2.3.2 Порушення стандартів інклюзивності та доступності (Accessibility)
    - 2.3.3 Аналіз найкращих практик розробки та безпеки (Best Practices)
    - 2.3.4 Критичні помилки пошукової оптимізації (SEO та компрометація коду)
- **2.4 Порівняльний аналіз технічного стану вебсайтів закладів освіти України** (КНУБА, КПІ, УКУ, ЧПФК)
- **2.5 Обґрунтування вибору технологічного стеку та інструментів розробки** (Вибір HTML5, CSS3, Vanilla JS, Figma)
- **Висновки до розділу 2**

РОЗДІЛ 3. ПРОЄКТУВАННЯ ТА РОЗРОБКА ОПТИМІЗОВАНОГО ВЕБІНТЕРФЕЙСУ ІНФОРМАЦІЙНОЇ СИСТЕМИ

- **3.1 Формулювання технічного завдання на розробку** (Вимоги до метрик Lighthouse, зображень, кросбраузерності)
- **3.2 Проєктування інформаційної архітектури та UI/UX дизайну в Figma**
    - 3.2.1 Інформаційна структура та карта сторінок сайту
    - 3.2.2 Побудова UX-каркасів сторінок (Wireframes)
    - 3.2.3 Візуальний дизайн (UI), фірмова колористика та типографіка
    - 3.2.4 Розробка адаптивних версій інтерфейсу під різні пристрої
- **3.3 Програмна реалізація вебінтерфейсу інформаційної системи**
    - 3.3.1 Архітектура та структура файлів проєкту
    - 3.3.2 Семантична верстка інтерфейсу засобами HTML5
    - 3.3.3 Побудова адаптивної системи стилів (CSS Grid, Flexbox, методологія BEM)

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ СТВОРЕННЯ ВЕБІНТЕРФЕЙСІВ

1.1 Роль вебінтерфейсів у сучасних інформаційних системах

На сучасному етапі розвитку інформаційних технологій інформаційні системи стали невід'ємною складовою більшості сфер діяльності людини. Стрімке зростання обсягів даних, розвиток цифрових технологій та поширення інтернет-сервісів спричинили активну цифровізацію суспільства. Сьогодні такі системи застосовуються у банківській сфері, електронній комерції, державному управлінні, освіті, медицині та інших галузях.

Інформаційні системи забезпечують обробку, зберігання та передачу даних і підтримують різноманітні бізнес-процеси. Водночас ефективність їх використання значною мірою залежить від зручності взаємодії користувача із системою. Саме тому ключову роль відіграють засоби взаємодії, зокрема вебінтерфейси.

Вебтехнології забезпечують доступ до функціональних можливостей систем через веббраузер без необхідності встановлення додаткового програмного забезпечення. Це робить вебінтерфейси універсальним інструментом взаємодії, що дозволяє працювати з системою незалежно від типу пристрою чи операційної системи за умови наявності підключення до мережі Інтернет.

Вебінтерфейс – це програмна оболонка, яка забезпечує взаємодію користувача з інформаційною системою через веббраузер. За його допомогою здійснюється введення даних, отримання результатів їх обробки та керування функціональними можливостями системи. Таким чином, вебінтерфейс виступає посередником між користувачем і внутрішньою логікою програмного продукту. Зручний і логічно організований інтерфейс дозволяє швидко виконувати необхідні дії, зменшує ймовірність помилок і скорочує час освоєння системи.

Таблиця 1.1

Основні структурні елементи вебінтерфейсу.

Компонент	Призначення
Навігаційне меню	Забезпечує перехід між розділами системи
Форми введення	Дозволяють користувачу вводити дані
Кнопки керування	Виконання дій

Таблиці	Відображення структурованої інформації
Візуальні елементи	Полегшують сприйняття інформації

Особливо важливу роль вебінтерфейси відіграють у системах із великою кількістю користувачів, зокрема в інтернет-банкінгу, CRM-платформах, інтернет-магазинах та соціальних мережах. Наприклад, система інтернет-банкінгу Privat24 надає можливість виконувати фінансові операції та переглядати стан рахунків через вебінтерфейс, забезпечуючи швидкий доступ до банківських послуг.

CRM-системи, такі як Bitrix24, Salesforce та HubSpot CRM, використовуються для управління взаємовідносинами з клієнтами, зберігання та аналізу даних, а також автоматизації бізнес-процесів. Вебінтерфейс у таких системах повинен забезпечувати ефективну роботу з великими обсягами інформації та швидкий доступ до необхідних даних.

У практичній діяльності було використано CRM-систему Bitrix24 для роботи з клієнтською базою та обробки заявок. Інтерфейс системи реалізовано у вигляді табличної структури, що містить інформацію про клієнтів: контактні дані, джерела залучення та поточний статус взаємодії. Така організація дозволяє оперативно змінювати статус клієнтів, додавати коментарі та швидко отримувати доступ до необхідної інформації.

Застосування подібного вебінтерфейсу забезпечує узгодженість дій між користувачами системи, безперервність роботи з клієнтами та рівномірний розподіл навантаження між співробітниками. Це, у свою чергу, підвищує ефективність обробки заявок та загальну продуктивність роботи персоналу. Отже, практичний досвід підтверджує, що якісно спроєктований вебінтерфейс є важливим фактором оптимізації бізнес-процесів.

Зручність, швидкість роботи та логічність побудови інтерфейсу безпосередньо впливають на вибір користувачем конкретного сервісу. Платформи, що забезпечують кращий користувацький досвід, отримують конкурентну перевагу та залучають ширшу аудиторію.

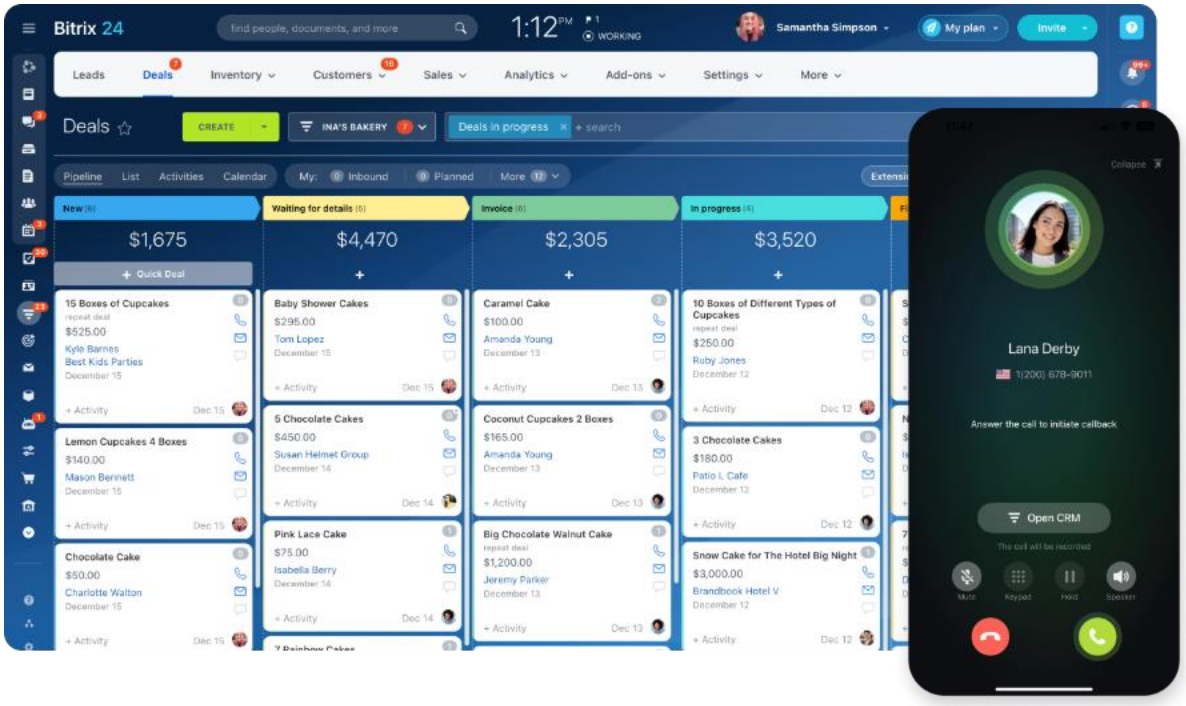


Рисунок 1.1 – Приклад вебінтерфейсу CRM-системи Bitrix24.

Джерело: офіційний вебсайт сервісу Bitrix24

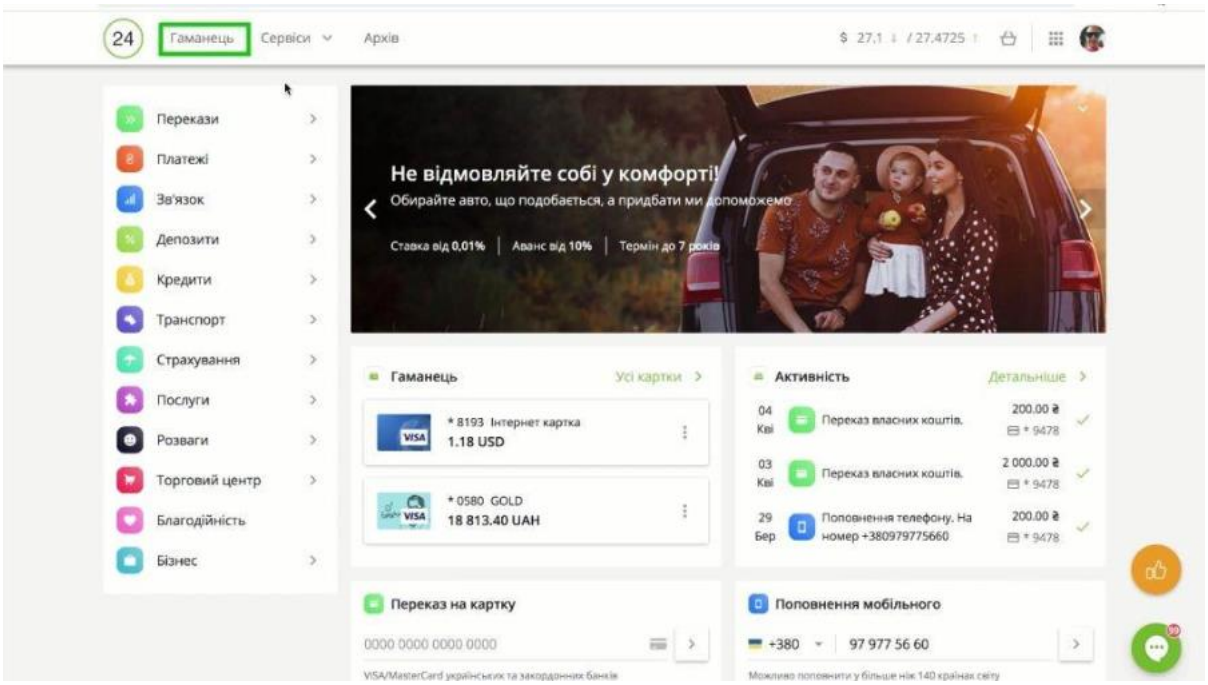


Рисунок 1.2 – Приклад вебінтерфейсу системи інтернет-банкінгу.

Джерело: офіційний вебсайт інтернет-банкінгу Privat24 (Десктоп-версія)

Окрім комерційних систем, вебінтерфейси відіграють важливу роль у сфері державного управління. У процесі цифровізації державні установи активно впроваджують електронні сервіси, що дозволяють громадянам отримувати адміністративні послуги дистанційно.

Такий підхід суттєво спрощує доступ до державних послуг, підвищує швидкість їх надання та зменшує навантаження на органи влади.

Прикладом сучасного державного цифрового сервісу є платформа Diia, що забезпечує доступ до електронних документів, реєстраційних процедур та інших адміністративних послуг. Її вебінтерфейс характеризується простотою, логічною структурою та орієнтацією на потреби користувача.

Водночас значна частина вебресурсів органів місцевого самоврядування все ще має застарілий інтерфейс. Такі системи часто відзначаються складною навігацією, надмірною кількістю інформації, відсутністю адаптивного дизайну та недостатньою узгодженістю елементів. Це ускладнює пошук необхідних даних, знижує ефективність використання ресурсів і негативно впливає на досвід користувача.

Рисунок 1.3 – Приклад сучасного державного вебінтерфейсу (Diia).

Джерело: UX Design Awards, кейс платформи Diia (2023)

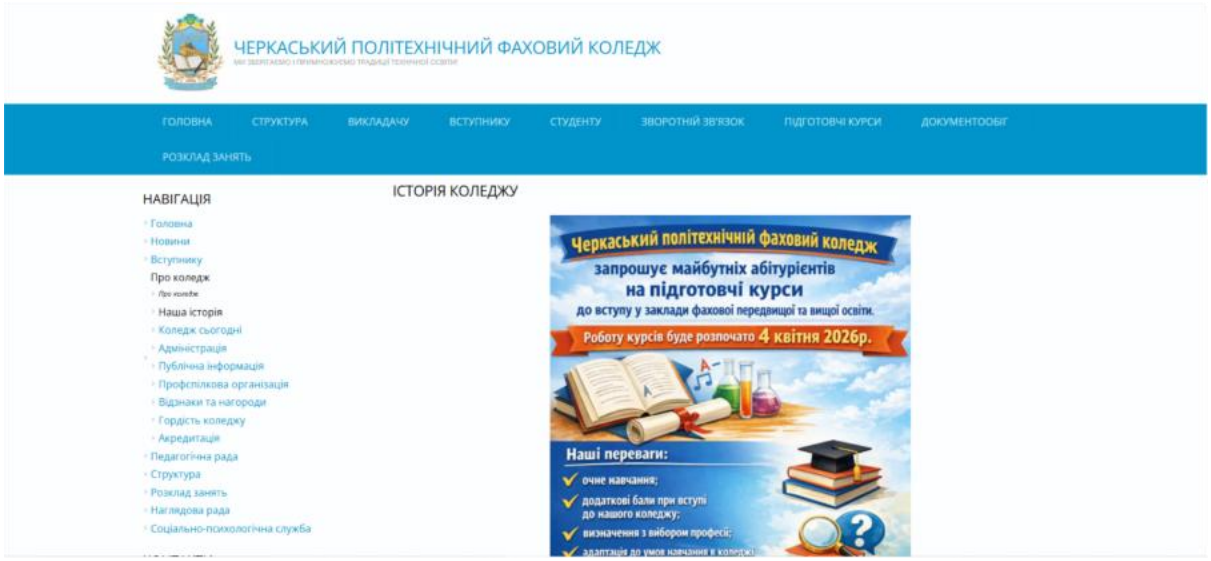


Рисунок 1.4 – Приклад застарілого вебінтерфейсу освітнього закладу.

Джерело: Офіційний вебсайт Черкаського політехнічного фахового коледжу.

Таблиця 1.2

Приклади використання вебінтерфейсів.

Сфера	**Приклад системи**	**Особливості**
Банківські системи	Інтернет-банкінг	Швидкий доступ до фінансових операцій
CRM системи	Управління клієнтами	Робота з великими обсягами даних
Державні сервіси	Електронні портали	Надання адміністративних послуг
Інтернет-магазини	Amazon, Rozetka	Простота покупки, швидкий пошук товарів
Відеосервіси	YouTube	Рекомендації контенту, зручна навігація
Аудіосервіси	Spotify	Персоналізація, швидкий доступ до медіа
Соціальні мережі	Instagram, Twitter	Інтуїтивність, швидка взаємодія, стрічка новин

Може скластися враження, що вебінтерфейс є обов’язковим компонентом будь-якої інформаційної системи, однак це не є універсальним правилом. Існують програмні системи, які можуть ефективно функціонувати без розвиненого користувацького

інтерфейсу, зокрема на рівні баз даних, серверної логіки або програмних інтерфейсів (API).

Проте саме наявність зручного та зрозумілого вебінтерфейсу визначає практичну цінність системи для кінцевого користувача та забезпечує ефективну взаємодію з її функціональними можливостями.

Розвиток вебінтерфейсів значною мірою зумовлений конкурентною боротьбою між цифровими сервісами. У сучасних умовах користувач має широкий вибір альтернатив, тому розробники прагнуть створювати максимально зручні, інтуїтивно зрозумілі та ефективні інтерфейси з метою залучення й утримання аудиторії.

Відомий принцип, сформульований у праці «Don't Make Me Think» Стівена Круга, полягає в тому, що інтерфейс не повинен вимагати від користувача зайвих зусиль для розуміння: усі дії мають бути очевидними, передбачуваними та інтуїтивно зрозумілими.

Користувач взаємодіє з інформаційною системою для досягнення конкретної мети, а не заради самого процесу використання. Тому вебінтерфейс повинен забезпечувати просту, зрозумілу та ефективну взаємодію, мінімізуючи когнітивне навантаження та усуваючи потребу в додатковому навчанні.

У зв’язку з цим доцільним є розгляд основних принципів проєктування вебінтерфейсів, зокрема UX та UI, які визначають зручність і ефективність взаємодії користувача з системою.

1.2 Основні принципи проєктування вебінтерфейсів

Ключовими складовими сучасних вебінтерфейсів є UX (User Experience) та UI (User Interface), які визначають ефективність взаємодії користувача з системою.

1.2.1 UX та UI як основа проєктування вебінтерфейсів

UX охоплює загальний досвід користувача під час роботи із системою. Він включає зручність навігації, логіку побудови інтерфейсу, швидкість виконання дій, а також загальне сприйняття продукту. Його метою є забезпечення максимально простого та ефективного досягнення користувачем поставленої мети.

UI відповідає за візуальне оформлення інтерфейсу та спосіб подання інформації. До нього належать кольори, шрифти, кнопки, іконки, відступи та інші елементи, що формують зовнішній вигляд системи.

Відповідно до підходу Дональда Нормана, ефективний дизайн повинен враховувати природні моделі сприйняття користувача та мінімізувати необхідність додаткового навчання. Порушення звичних моделей взаємодії призводить до ускладнення використання навіть простих об’єктів.

Прикладом неінтуїтивного дизайну є віддзеркалене розташування елементів, що суперечить звичним когнітивним моделям користувача.

Рисунок 1.5 – Приклад неінтуїтивного дизайну(The Backward Clock).

Джерело: Norman D. The Design of Everyday Things (drawing by Eilee n Conway).

Як видно з рисунка, навіть знайомий об’єкт може стати складним у використанні через порушення очікуваної логіки взаємодії. Це підкреслює важливість врахування принципів UX при проєктуванні інтерфейсів.

Практичне значення UX/UI видно на прикладах інформаційних систем:

- У **інтернет-банкінгу** користувач очікує швидкого доступу до балансу та платежів; складна навігація або нечіткі кнопки свідчать про низьку якість UX/UI.
- У **CRM-системах** важлива ефективна робота з великими обсягами даних: групування, фільтри та логічна структура таблиць – це UX, а кольори, візуальна ієрархія та стиль таблиць – UI.

- У **соціальних мережах** UX проявляється у швидкості взаємодії та логіці стрічки новин, а UI – у візуальному оформленні контенту та кнопок.

Практичні підходи до UX/UI (аналіз сценаріїв користувача, побудова структури інтерфейсу, принципи візуального оформлення) були додатково опрацьовані автором під час курсу з веброзробки та UX/UI-дизайну (Genius Space; лектори: Кравчук К., Чебан І., Педорич Є.).

UX та UI взаємодоповнюють один одного: їх узгоджене застосування підвищує зручність, зменшує кількість помилок і забезпечує ефективну роботу користувача з інтерфейсом.

1.2.2 Адаптивність та кросплатформеність вебінтерфейсів

Адаптивність вебінтерфейсу гарантує, що система працює зручно на будь-якому пристрої: ПК, ноутбуку, планшеті чи смартфоні. Різні екрани мають різні розміри, роздільну здатність та способи введення, тому інтерфейс має підлаштовуватися під користувача.

Для цього застосовують гнучкі макети, адаптивні сітки та масштабовані елементи, а також CSS-технології, зокрема медіа-запити (media queries) і контрольні точки (breakpoints). Основні підходи – responsive design, коли сторінка змінює структуру залежно від розміру екрана, та mobile-first, коли дизайн спочатку роблять під мобільні пристрої, а потім розширюють для більших екранів.

Для прикладу інтернет-магазини роблять зручний перегляд товарів, пошук і оформлення замовлення на будь-якому пристрої.

У системах інтернет-банкінгу роблять ставку на мобільні додатки, що зумовлює акцент на швидкому виконанні фінансових операцій; дрібні кнопки або складна навігація на мобільному можуть спричиняти помилки.

Кросплатформеність гарантує однаковий користувацький досвід у різних браузерах і операційних системах, забезпечуючи стабільність роботи та мінімізуючи несподівані помилки.

1.2.3 Доступність вебінтерфейсів

Accessibility дозволяє користувачам із фізичними або когнітивними особливостями ефективно взаємодіяти з інформаційною системою. Це стосується, наприклад, людей із порушеннями зору, слуху або моторики, а також тих, хто працює в обмежених умовах, як низька швидкість інтернету або маленькі екрани мобільних пристроїв. Основою забезпечення доступності є рекомендації WCAG, які визначають принципи побудови доступного вебконтенту.

Практична реалізація доступності передбачає використання різних інструментів залежно від потреб користувачів:

- **Для користувачів із порушеннями зору**: екрани-ридери, збільшувальні програми, висококонтрастні теми, голосове керування
(наприклад, VoiceOver на iPhone, TalkBack на Android).
- **Для людей із порушеннями слуху**: субтитри у відео, візуальні сигнали замість звукових сповіщень.
- **Для користувачів із проблемами моторики**: управління через клавіатуру, спрощені форми, великі кнопки, голосове введення.
- **Для людей із когнітивними особливостями**: простий текст, зрозумілі інструкції, послідовна навігація.

Застосування принципів доступності робить вебінтерфейс універсальнішим, підвищує ефективність використання системи та забезпечує соціальну інклюзію.

1.2.4 Орієнтація на користувача (Human-Centered Design)

Human-Centered Design (HCD) – підхід
до проєктування вебінтерфейсів, який ставить користувача в центр уваги. Замість того, щоб зосереджуватись лише на технічних можливостях системи, HCD аналізує потреби, поведінку та сценарії взаємодії реальних користувачів.

Ключові моменти підходу:

- **Визначення цільової аудиторії**. Розуміння, хто користується системою і які завдання вони виконують, допомагає сформувати вимоги до інтерфейсу ще до розробки.
- **Моделювання сценаріїв використання**. Опис типових дій користувачів дозволяє структурувати інтерфейс і логіку взаємодії.
- **Ітеративне вдосконалення**. Прототипи тестуються на реальних користувачах; результати аналізуються, а інтерфейс змінюється для покращення зручності.
- **Командна робота**. HCD передбачає координацію між дизайнерами, розробниками, аналітиками та іншими фахівцями, щоб врахувати і технічні, і користувацькі вимоги.

На практиці це означає, що ще до етапу оптимізації UX/UI (тестуванням, A/B-тестами, метриками) визначаються реальні потреби користувачів та сценарії їх взаємодії з системою. Наприклад:

- У **банківських додатках** мобільний інтерфейс роблять максимально простим для швидких фінансових операцій.
- У **CRM-системах** на десктопі відображають великі обсяги даних, а на мобільному лише ключові функції.

Таким чином, HCD стає основою для подальшої оптимізації вебінтерфейсів, бо забезпечує реальну відповідність системи потребам користувача і спрощує процес покращення UX/UI.

1.2.5 Консистентність та візуальна ієрархія

Консистентність і візуальна ієрархія відіграють ключову роль у зручності вебінтерфейсів, допомагаючи користувачам швидко орієнтуватися та виконувати основні дії.

Консистентність забезпечується узгодженим використанням кольорів, шрифтів, кнопок та розташування елементів. Це дозволяє передбачати поведінку системи і ефективно працювати з великими обсягами даних, як у CRM-системах (рисунок 1.1).

Візуальна ієрархія визначає порядок сприйняття інформації та допомагає зосередитися на ключових елементах. Вона формується через розмір, контраст, колір і розташування елементів. Прикладом є інтерфейси інтернет-банкінгу (рисунок 1.2), де основні функції винесені у зони швидкого доступу, або державні цифрові сервіси (рисунок 1.3), де інтерфейс інтуїтивно зрозумілий навіть для новачків.

Недотримання цих принципів перевантажує інтерфейс і ускладнює роботу користувача. Використання консистентності та візуальної ієрархії зменшує когнітивне навантаження, робить систему передбачуваною і підвищує ефективність взаємодії.

1.3 Архітектурні підходи до створення вебінтерфейсів

При проєктуванні вебінтерфейсів важливо обрати архітектурний підхід, який визначає організацію обміну даними між клієнтом і сервером, а також взаємодію елементів інтерфейсу. Найпоширенішими є SPA (Single Page Application), MPA (Multi Page Application) та патерн MVC (Model-View-Controller).

SPA vs MPA

SPA (Single Page Application) передбачає завантаження інтерфейсу один раз, після чого оновлення контенту відбувається динамічно без

перезавантаження сторінки. Це забезпечує швидку та плавну взаємодію, що особливо важливо для інтерактивних систем, таких як CRM або інтернет-банкінг.

MPA (Multi Page Application) реалізує класичну модель, де кожна сторінка завантажується окремо при переході між розділами. Такий підхід доцільний для інформаційних сайтів, порталів та проєктів із високими вимогами до SEO.

Таблиця 1.3

Переваги і недоліки SPA, MPA.

Характеристика	**SPA (Single Page Application)**	**MPA (Multi Page Application)**
Принцип роботи	Завантаження однієї сторінки, динамічне оновлення контенту без перезавантаження	Кожна сторінка завантажується окремо при переході
Швидкість взаємодії	Висока, переходи між розділами відбуваються плавно	Повільніша, оскільки кожен перехід потребує завантаження сторінки
Інтерактивність	Підтримує складні інтерактивні елементи та оновлення в реальному часі	Менш інтерактивна, підходить для статичного контенту
SEO	Складніше оптимізувати	Краще для SEO, кожна сторінка має власний URL та метадані
Технології	React, Angular, Vue	Серверні технології, HTML-рендеринг
Сфера застосування	CRM-системи, динамічні вебсервіси, інтернет-банкінг	Блоги, інформаційні сайти, портфоліо, державні портали
Переваги	Швидка та плавна взаємодія, реальний час, гнучка інтеграція компонентів	Простота реалізації, краща SEO, менша залежність від JavaScript
Недоліки	Складність SEO, довше початкове завантаження	Повільніша взаємодія, більші витрати на підтримку навігації та стилів

MVC (Model-View-Controller). Патерн MVC передбачає поділ системи на три складові:

- **Model (Модель)** – відповідає за дані та бізнес-логіку системи.
- **View (Вид)** – забезпечує відображення даних користувачу.
- **Controller (Контролер)** – обробляє дії користувача та координує взаємодію між моделлю і видом.

Такий поділ спрощує підтримку та масштабування системи. Наприклад, у CRM-системі модель зберігає дані про клієнтів, вид відображає таблиці, а контролер обробляє фільтрацію та додавання записів. MVC також полегшує командну роботу, оскільки розмежовує відповідальність між учасниками розробки.

1.3.1 Компонентний підхід та сучасні фреймворки

Компонентний підхід передбачає поділ інтерфейсу на незалежні та повторно використовувані частини – компоненти. Кожен компонент відповідає за окремий елемент або функцію, що спрощує розробку, підтримку та масштабування вебдодатків.

Сучасні вебфреймворки та бібліотеки активно використовують компонентний підхід, що спрощує розробку SPA та MPA. Наприклад:

- **React** – дозволяє створювати інтерфейси з повторно використовуваних компонентів, підтримує управління станом через хуки та контекст, добре інтегрується з іншими бібліотеками.
- **Vue.js** – пропонує гнучкий підхід до компонентів, включно з шаблонами, стилями та логікою, що забезпечує швидкий розвиток додатків.
- **Angular** – фреймворк для створення масштабованих вебдодатків, забезпечує повноцінну архітектуру з модульністю та інжекцією залежностей.

Таблиця 1.4

Компонентний підхід у сучасних фреймворках.

Принцип компонентного підходу	**Проблеми**	**Рішення**

Розділення інтерфейсу на незалежні, повторно використовувані частини (компоненти)	Складна структура при великій кількості компонентів	Створювати чітку ієрархію компонентів, використовувати централізоване управління станом (Redux, Vuex, Pinia)
Ізольоване управління станом та логікою	Повільне завантаження SPA через велику кількість компонентів	Впроваджувати відкладене завантаження компонентів (lazy loading), оптимізувати ресурси
Повторне використання коду	Залежності між компонентами ускладнюють підтримку	Проектувати компоненти максимально ізольованими, взаємодія через чіткі інтерфейси (props, events)
Масштабованість та зручність тестування	Потенційне дублювання коду або стилів	Використовувати спільні бібліотеки компонентів, стилів та шаблонів

Компонентний підхід дозволяє повторно використовувати код, ізолювати логіку та підвищити масштабованість системи. Водночас при великій кількості компонентів важливо дотримуватись чіткої структури та організації стану.

Компонентна архітектура у поєднанні з сучасними фреймворками забезпечує гнучкість, зменшує кількість помилок і підвищує ефективність командної розробки.

1.4 Методи оптимізації вебінтерфейсів

Створення ефективного вебінтерфейсу передбачає не лише продуману структуру та дизайн, але й високу швидкість роботи та миттєвий відгук системи. Оптимізація дозволяє зменшити час завантаження, забезпечити плавну взаємодію та підвищити зручність використання, що безпосередньо впливає на ефективність роботи користувача.

1.4.1 Оптимізація продуктивності

Продуктивність вебінтерфейсу визначає, наскільки швидко користувач може виконувати дії та отримувати результат. навіть затримки у частки секунди можуть ускладнювати взаємодію та знижувати ефективність роботи. Основні методи оптимізації:

- **Мінімізація ресурсів** – зменшення розміру CSS, JavaScript та зображень за рахунок стиснення, об’єднання файлів та видалення зайвого коду.
- **Lazy loading (відкладене завантаження)** – завантаження ресурсів лише в момент потреби (наприклад, при прокрутці сторінки).
- **Кешування** – зберігання часто використовуваних ресурсів на стороні користувача або на сервері, що дозволяє повторно використовувати їх без повторного завантаження.
- **Lighthouse** – автоматизований інструмент для оцінки продуктивності, доступності та SEO вебсайтів з рекомендаціями щодо покращення.

Застосування цих методів скорочує час завантаження та підвищує швидкість роботи інтерфейсу. У системах на кшталт інтернет-банкінгу або CRM це критично, оскільки користувач очікує миттєвої реакції.

Таблиця 1.5

Проблеми та рішення в оптимізації вебінтерфейсів.

Метод	Проблема	Рішення
Мінімізація ресурсів	Повільне завантаження сторінки через великі CSS/JS/зображення	Стиснення, об’єднання та видалення зайвого коду, зменшення розміру файлів
Lazy loading	Довге очікування завантаження всіх елементів відразу	Відкладене завантаження ресурсів, що не потрібні користувачу одразу
Кешування	Повторне завантаження тих самих ресурсів збільшує трафік і час відгуку	Збереження часто використовуваних ресурсів на стороні користувача або сервері
Lighthouse	Неможливо легко оцінити продуктивність і знайти вузькі місця	Автоматизований аналіз продуктивності, доступності та SEO з рекомендаціями щодо оптимізації

1.4.2 Оптимізація зручності використання

Швидка робота інтерфейсу не гарантує його ефективності, якщо інтерфейс складний у використанні. Тому важливо оцінювати реальну взаємодію користувача із системою. Для оцінки та покращення зручності застосовують методи тестування на живих користувачах, що дозволяють

об’єктивно визначити, які елементи інтерфейсу працюють добре, а які потребують доопрацювання.

- **Юзабіліті-тестування** – спостереження за виконанням конкретних завдань, виявляють помилки та складні для сприйняття елементи.
- **A/B тестування** – порівнює два варіанти сторінки або інтерфейсу, щоб визначити, який забезпечує кращий досвід користувача та вищу ефективність завершення дій.
- **UX-метрики** – кількісна оцінка взаємодії (час виконання, кількість помилок, коефіцієнт успіху, задоволеність).

Ці методи дозволяють виявляти слабкі місця інтерфейсу та приймати обґрунтовані рішення щодо його покращення.

Таблиця 1.6

Методи оптимізації зручності використання вебінтерфейсів.

Метод	Проблема	Рішення / Принцип роботи
Юзабіліті-тестування	Невідомо, наскільки зручно користувачам виконувати завдання	Перевірка інтерфейсу на реальних користувачах; спостереження за поведінкою, аналіз помилок та труднощів; дозволяє виявити вузькі місця
A/B тестування	Неясно, який дизайн/UX краще впливає на ефективність	Створюють два варіанти сторінки (старий і новий UX) і показують їх різним групам людей (~50/50); порівнюють показники завершення завдань, конверсії або інші метрики
UX-метрики	Складно кількісно оцінити зручність інтерфейсу	Використовують показники: час на виконання завдання, кількість помилок, коефіцієнт успіху, задоволеність користувача; допомагає приймати об’єктивні рішення

1.4.3 Оптимізація доступності

Доступність вебінтерфейсу означає, що система може бути зручною та зрозумілою для всіх користувачів, включно з людьми з обмеженими можливостями.

Основні підходи до оптимізації доступності:

- **WCAG (Web Content Accessibility Guidelines)** – міжнародні стандарти, які визначають правила організації контенту, кольорової схеми, навігації та інтерактивних елементів. WCAG орієнтовані на чотири основні принципи:
    1. **Сприйнятність (Perceivable)** – інформація має бути сприйнятною для всіх користувачів (контраст тексту та фону, масштабованість шрифтів, альтернативні тексти для зображень).
    2. **Керованість (Operable)** – користувачі повинні мати змогу взаємодіяти з інтерфейсом через клавіатуру, сенсорні пристрої або інші допоміжні технології.
    3. **Зрозумілість (Understandable)** – інтерфейс має бути логічним і передбачуваним: чіткі підказки, інтуїтивна навігація, зрозумілі повідомлення про помилки.
    4. **Надійність (Robust)** – контент і взаємодія повинні коректно працювати з різними браузерами.
- **Семантична верстка** – правильне використання HTML-елементів (<header>, <nav>, <main>, <footer>, <button>, <form> тощо) дозволяє правильно інтерпретувати структуру сторінки, полегшуючи навігацію та взаємодію.

Дотримання цих принципів підвищує зручність, розширює аудиторію користувачів і покращує сумісність із різними пристроями.

Таблиця 1.7

Методи оптимізації доступності вебінтерфейсів.

Метод	**Проблема**	**Рішення / Принцип роботи**
WCAG	Вебінтерфейс недоступний для людей з порушеннями зору, слуху або моторики	Використання стандартів WCAG для контрастності, альтернативних текстів, клавіатурної навігації та зрозумілої структури контенту
Семантична верстка	Допоміжні технології не можуть правильно інтерпретувати сторінку	Використання правильних HTML-елементів і структурних блоків, щоб екранні читалки та інші

		технології працювали коректно

Оптимізація продуктивності, зручності та доступності формує комплексний підхід до створення ефективних вебінтерфейсів. На практиці ці аспекти часто реалізуються окремо, що зумовлює необхідність узагальнення підходів і подальшої оптимізації.

Постановка задачі

Сучасні вебінтерфейси характеризуються високою складністю, інтеграцією великої кількості модулів та динамічним розвитком функціоналу. користувачі очікують швидкого та інтуїтивного виконання дій, таких як купівля товару або бронювання квитка

Водночас розробники стикаються з проблемами:

- Довгим завантаженням сторінок та ресурсів, що знижує продуктивність;
- Складністю створення зрозумілого та інтуїтивного інтерфейсу для різних груп користувачів;
- Низькою доступністю для людей із обмеженими можливостями;
- Відсутність єдиної системи оцінки UX та продуктивності інтерфейсу.
- Мета дослідження – розробити системний підхід до створення та оптимізації вебінтерфейсів, який забезпечує:
• швидкий та продуктивний доступ до функцій вебдодатка;
• максимальний комфорт і зрозумілість для користувача;
• відповідність стандартам WCAG та високу доступність;
- системну оцінку та вдосконалення інтерфейсу на основі UX-тестів, A/B тестування та метрик.швидкий та продуктивний доступ до функцій вебдодатка;

Вхідними даними для аналізу є:

- структура вебінтерфейсу та інтегровані модулі;
- поведінка користувачів під час взаємодії з додатком;
- показники продуктивності та стабільності роботи сторінок.

Вихідними результатами очікується:

- комплексна оцінка ефективності та зручності вебінтерфейсу;

- рекомендації щодо покращення навігації, дизайну та швидкодії;
- впровадження рішень, що дозволяють користувачу швидко виконувати потрібні дії з мінімальними зусиллями;
- підвищення загальної якості та конкурентоспроможності вебдодатка.

РОЗДІЛ 2. АНАЛІЗ ІСНУЮЧОГО ВЕБІНТЕРФЕЙСУ ТА ВИБІР МЕТОДІВ ОПТИМІЗАЦІЇ

2.1 Вибір об'єкта дослідження та характеристика цільової аудиторії

Для практичного дослідження методів оптимізації вебінтерфейсів обрано офіційний вебсайт Черкаського політехнічного фахового коледжу (polytechnic.ck.ua) – типовий представник застарілих освітніх вебінтерфейсів України, згаданий у розділі 1 як приклад неоптимального рішення (рисунок 1.4). Сайт охоплює три основні групи користувачів: абітурієнти та батьки (пошук спеціальностей, умов вступу, контактів), студенти (розклад, новини, матеріали) та викладачі й адміністрація (публікація документів та оголошень). Різноманітність аудиторії визначає підвищені вимоги до зручності та доступності інтерфейсу.

2.2 Порівняльний аналіз вебінтерфейсів освітніх закладів

З метою підтвердження актуальності обраної теми та обґрунтування вибору об'єкта дослідження проведено порівняльний аудит вебінтерфейсів чотирьох українських освітніх закладів за допомогою Google Lighthouse. До аналізу залучено сайти різного рівня та типу: Черкаський політехнічний фаховий коледж (polytechnic.ck.ua), Київський національний університет будівництва і архітектури (knuba.edu.ua), Київський політехнічний інститут (kpi.ua) та Український католицький університет (ucu.edu.ua). Результати зведено у таблицю 2.1.

Примітка: показники Performance можуть варіюватися між запусками залежно від умов мережі та завантаженості сервера. Для порівняння зафіксовано результати одного сеансу вимірювання.

Таблиця 2.1

Порівняльний аналіз різних українських освітніх сайтів.

Сайт	Performance	Accessibility	Best Practices	SEO
polytechnic.ck.ua	54	75	62	Помилка
knuba.edu.ua	68	84	54	85
kpi.ua	87	93	69	Помилка
ucu.edu.ua	44	74	54	92

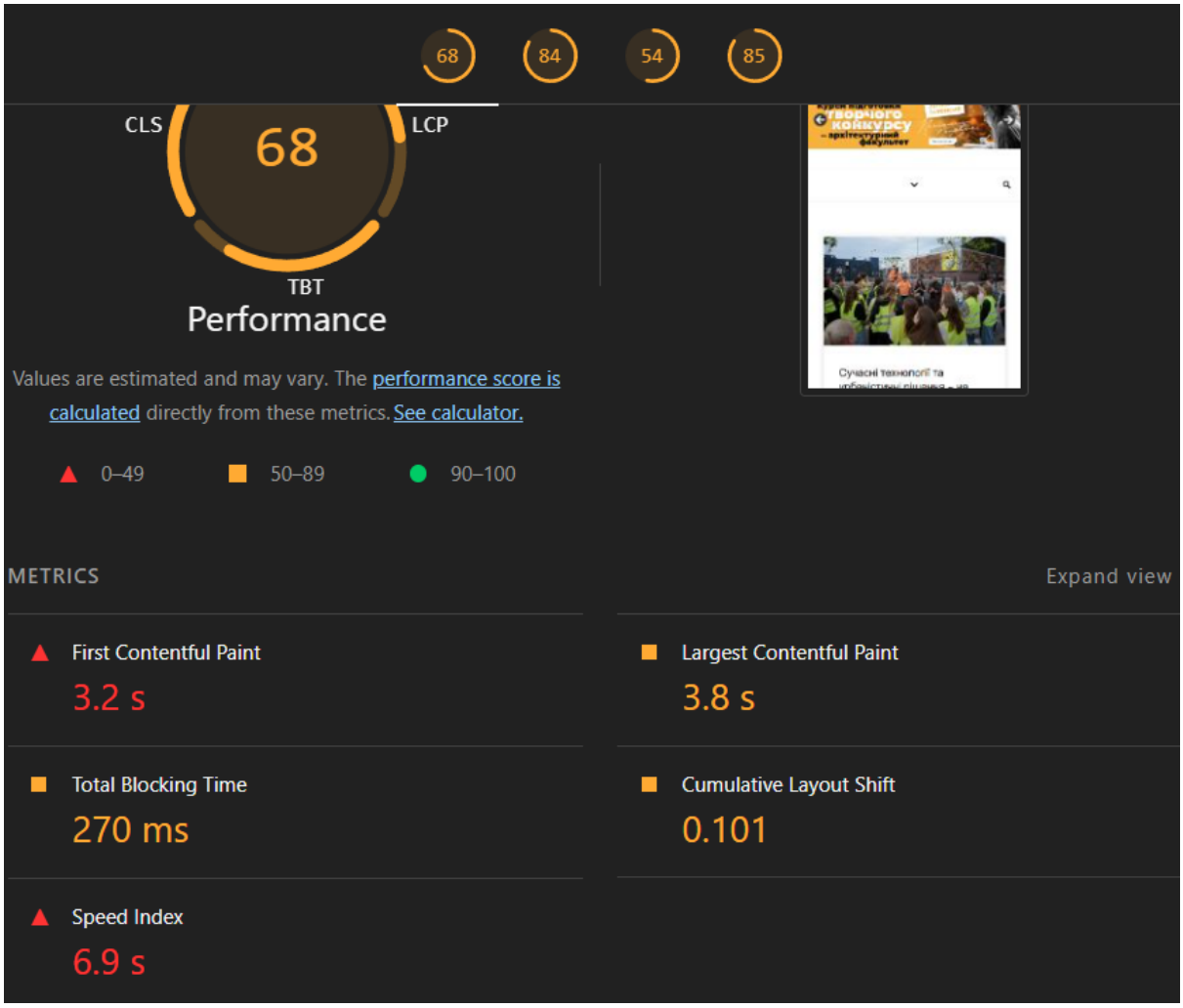


Рисунок 2.1 – Результати аудиту Google Lighthouse для сайту **knuba.edu.ua.**

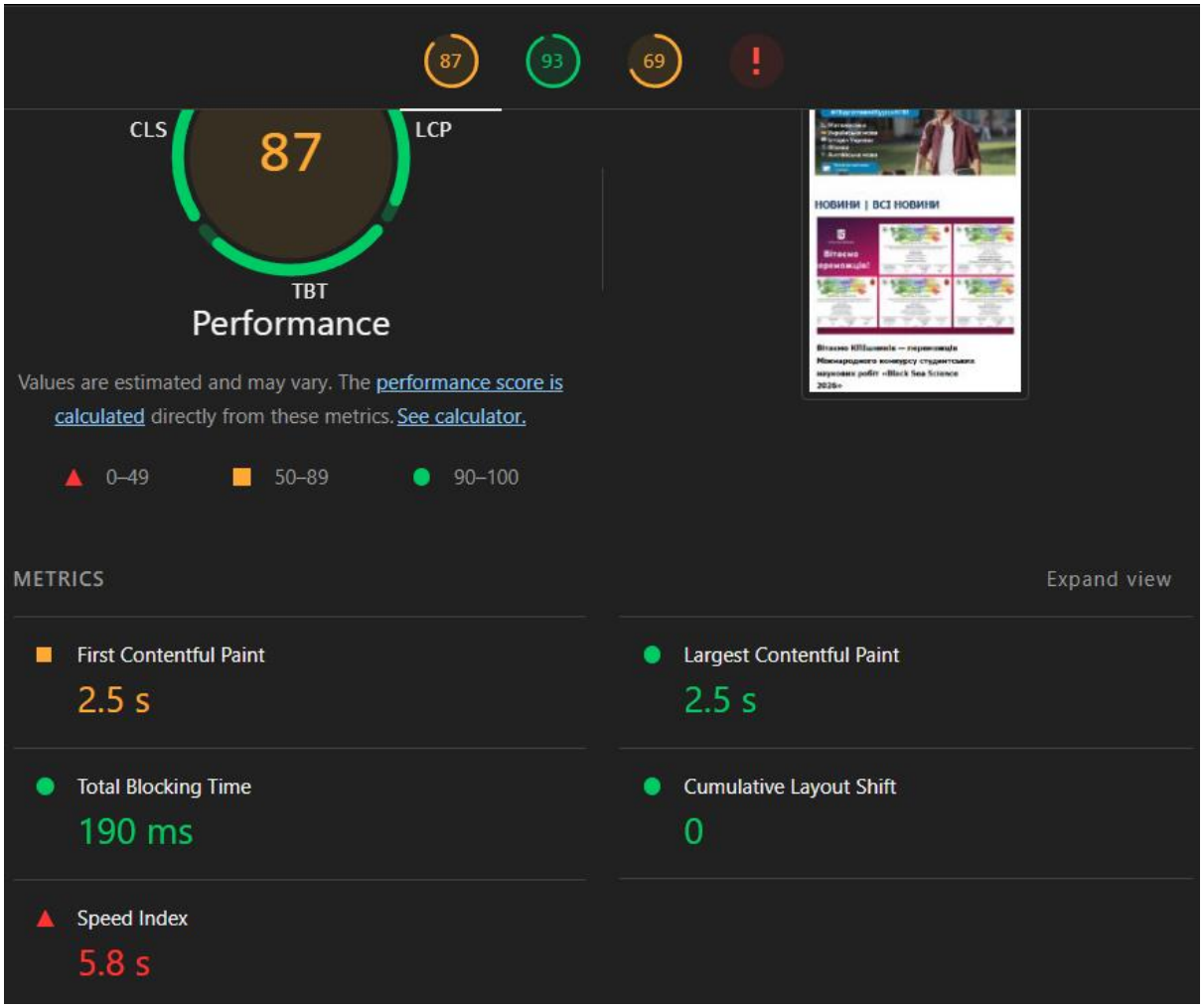


Рисунок 2.2 – Результати аудиту Google Lighthouse для сайту **kpi.ua.**

Аналіз результатів виявляє характерну закономірність: жоден із досліджуваних сайтів не досягає рекомендованого показника продуктивності у 90 балів. Найвищий результат Performance зафіксовано у КПІ (87), КНУБА в середньому має (68), УКУ (44), тоді як polytechnic.ck.ua показує не найгірші, але все одно незадовільні результати – 54 відповідно. Основною причиною низьких показників продуктивності в усіх випадках є перевантаженість головних сторінок важкими зображеннями, сторонніми скриптами та відсутність оптимізації ресурсів.

У категорії Accessibility КПІ виділяється найкращим результатом – 93 бали, що свідчить про увагу до доступності контенту. Натомість polytechnic.ck.ua з показником 75 та УКУ з 74 балами мають суттєві прогалини у цій категорії.

Найбільш контрастна картина спостерігається у SEO: УКУ демонструє відмінний результат 92 бали, КНУБА – 85, тоді як polytechnic.ck.ua та kpi.ua

отримали статус помилки, що означає наявність блокуючих проблем які унеможливлюють коректне індексування пошуковими системами.

Таким чином, проблеми з продуктивністю є системними для українських освітніх вебресурсів. Водночас приклад УКУ підтверджує що якісна SEO-оптимізація є цілком досяжною. Сайт polytechnic.ck.ua обраний як основний об'єкт дослідження оскільки має проблеми одночасно в усіх чотирьох категоріях – зокрема критичні помилки SEO, незадовільну продуктивність, порушення доступності та застарілий UX – що забезпечує найширше поле для демонстрації методів оптимізації.

2.3 Поглиблений технічний аудит вебінтерфейсу за допомогою Google Lighthouse

З метою кількісної оцінки технічного стану вебінтерфейсу проведено автоматизований аудит інструментом Google Lighthouse у браузері Chrome для головної сторінки сайту (https://polytechnic.ck.ua/). Результати зведено у таблицю 2.2.

Таблиця 2.2

Результати аудиту Google Lighthouse для сайту ЧПФК.

Категорія	Бал	Оцінка	Ключові метрики
Performance	54	Незадовільно	FCP: 12.9s, LCP: 12.9s, SI: 12.9s
Accessibility	75	Задовільно	Відсутні alt-атрибути, низький контраст
Best Practices	62	Незадовільно	2 HTTP-запити, відсутній DOCTYPE
SEO	Помилка	Критично	Немає meta description, помилки canonical

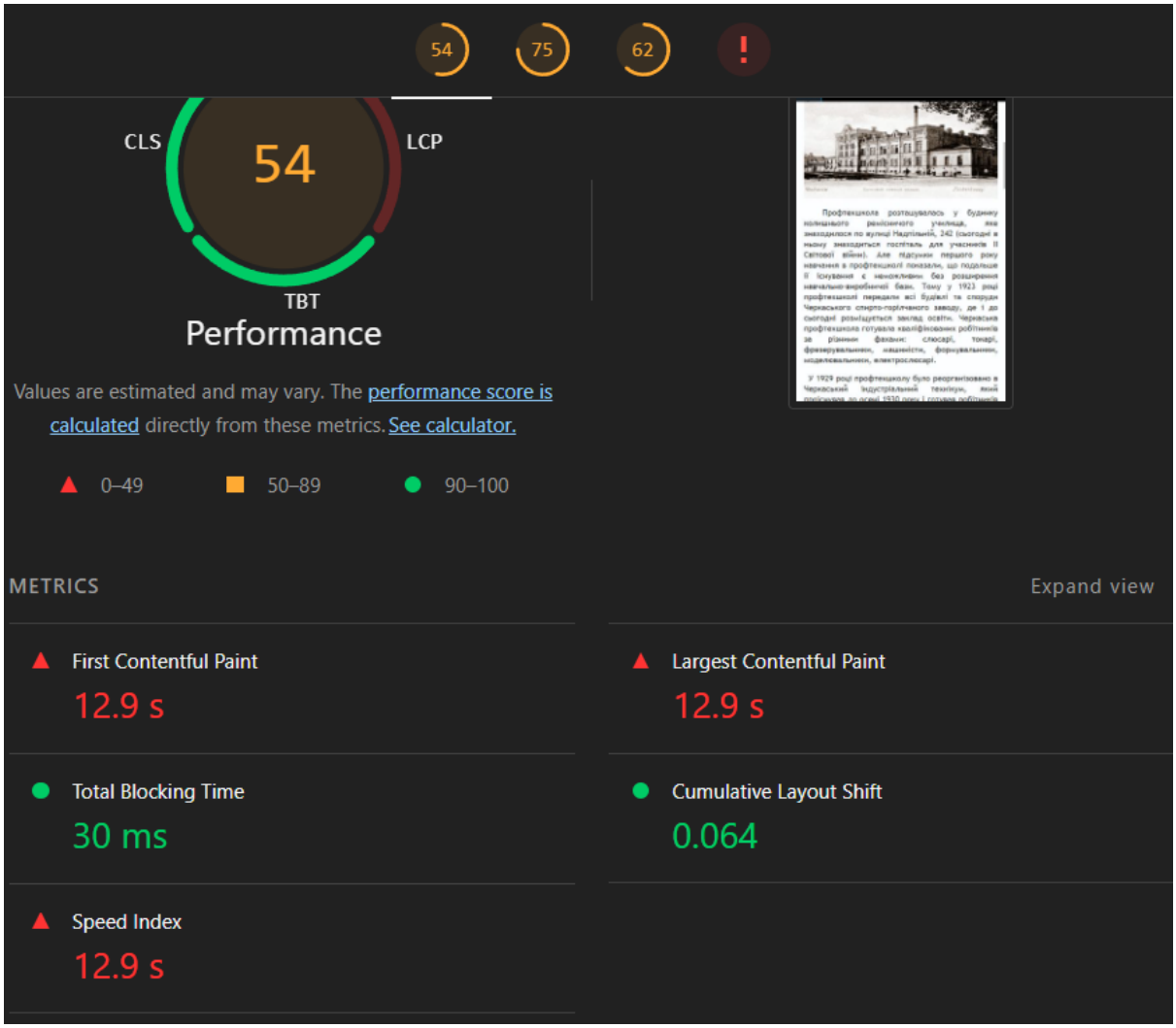


Рисунок 2.3 – Результати аудиту Google Lighthouse для сайту **polytechnic.ck.ua**.

Загальна картина результатів аудиту є незадовільною: жодна з категорій не досягла рекомендованого показника у 90 балів і вище. Особливо критичною є ситуація з SEO, де Lighthouse зафіксував блокуючі помилки, що унеможливлюють нормальне індексування сайту пошуковими системами. Нижче наведено детальний аналіз виявлених проблем по кожній категорії.

2.3.1 Проблеми продуктивності (Performance: 54)

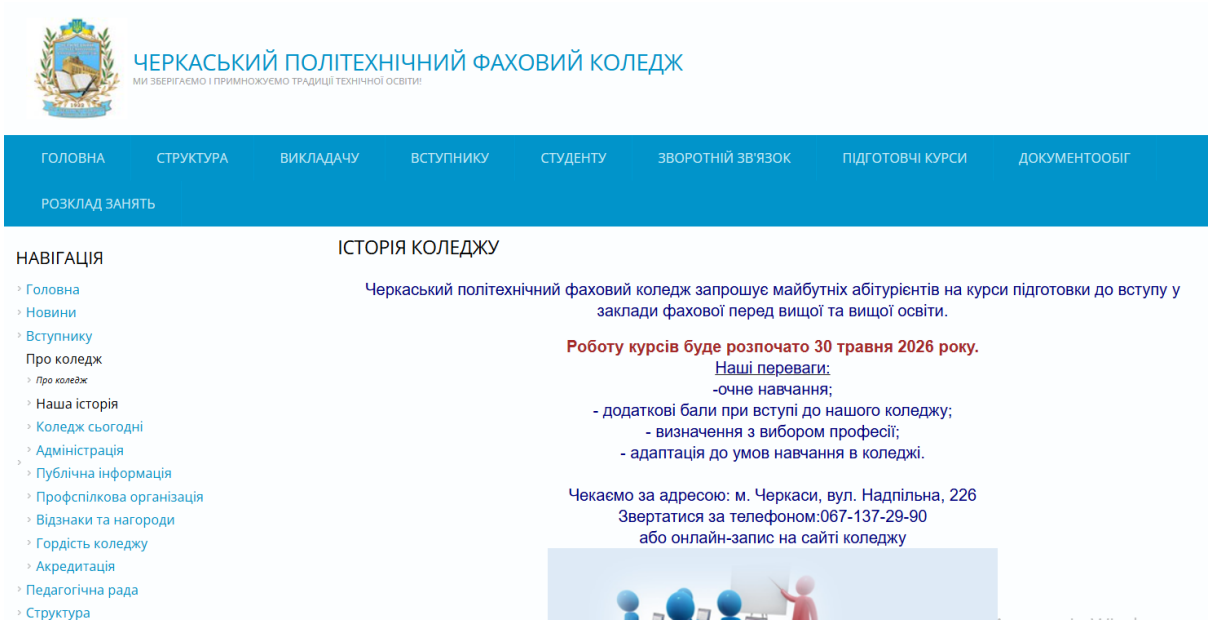


Рисунок 2.4 – Головна сторінка сайту polytechnic.ck.ua.

Показник продуктивності 54 бали свідчить про суттєві проблеми зі швидкістю завантаження. За шкалою Lighthouse значення 50–89 відноситься до категорії "потребує покращення". Метрики підтверджують критичний стан: FCP становить 12.9 с при нормі до 1.8 с, LCP – 12.9 с при нормі до 2.5 с, Speed Index – 12.9 с. Користувач бачить порожню сторінку майже 13 секунд, що є неприйнятним для сучасного вебресурсу.

Основними факторами, що негативно впливають на продуктивність, є:

великий розмір зображень без стиснення та оптимізації – зображення завантажуються у повному розмірі без адаптації під розмір екрану;

відсутність технік відкладеного завантаження (lazy loading) – усі ресурси завантажуються одночасно при відкритті сторінки;

не мінімізовані CSS та JavaScript файли – наявність зайвого коду та пробілів збільшує розмір файлів;

завантаження зовнішніх ресурсів з інших доменів без кешування, що збільшує час відповіді.

2.3.2 Проблеми доступності (Accessibility: 75)

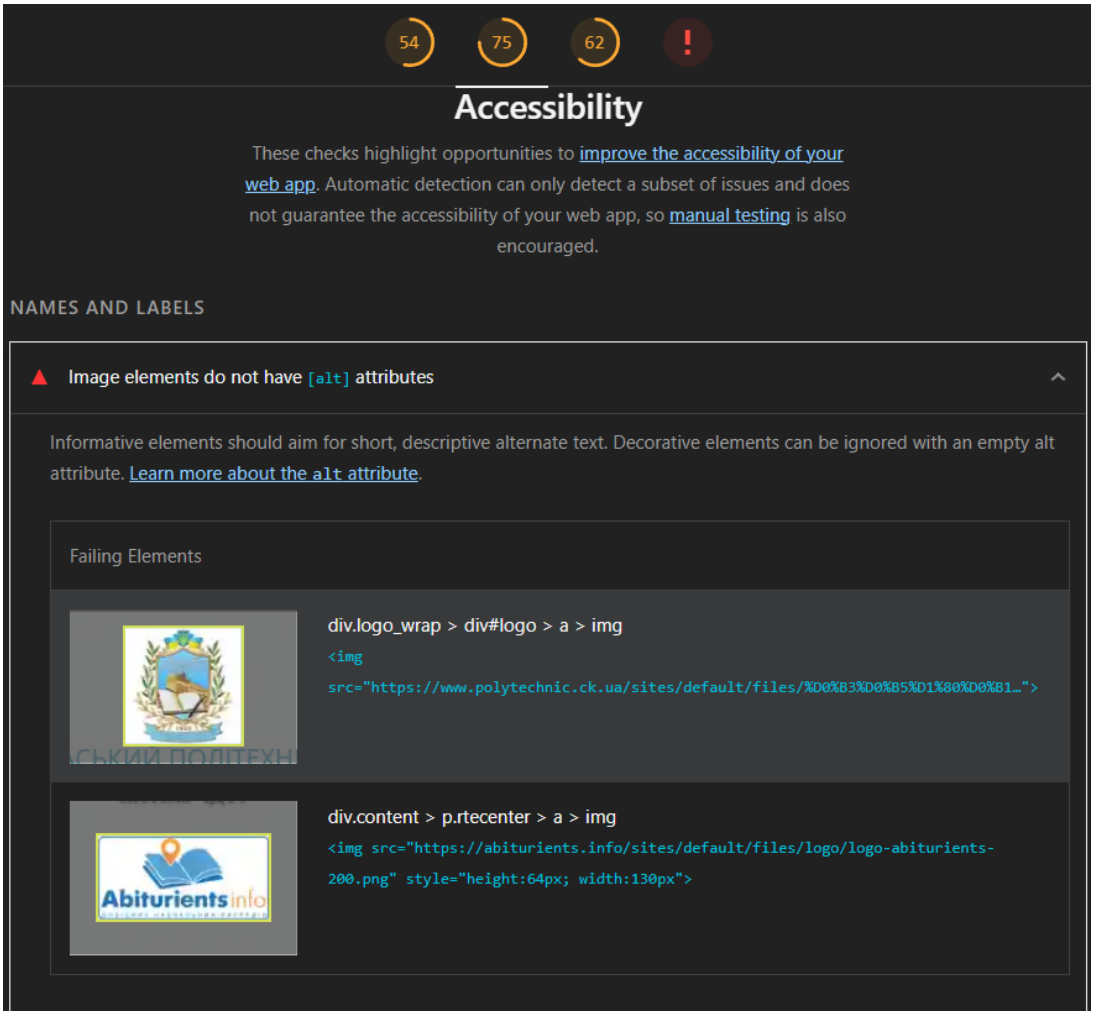


Рисунок 2.5 – Помилки доступності: відсутність атрибута alt.

Показник доступності у 75 балів вказує на наявність конкретних бар'єрів для користувачів з особливими потребами. Lighthouse виявив такі порушення:

зображення не мають атрибута alt – логотип коледжу та зображення партнерів (зокрема Abiturients.info) завантажуються без описового тексту, що унеможливлює їх сприйняття екранними читалками для користувачів з порушеннями зору;

посилання не мають описового тексту – частина посилань на сайті не містить зрозумілого текстового опису, що ускладнює навігацію за допомогою допоміжних технологій;

недостатній контрастний коефіцієнт між кольором тексту та фону в окремих елементах навігації та блоках контенту.

Відповідно до стандарту WCAG 2.1, усі зображення, що несуть інформаційне навантаження, зобов'язані мати атрибут alt з коротким

описом. Відсутність таких атрибутів є прямим порушенням принципу сприйнятності (Perceivable).

2.3.3 Проблеми безпеки та найкращих практик (Best Practices: 62)

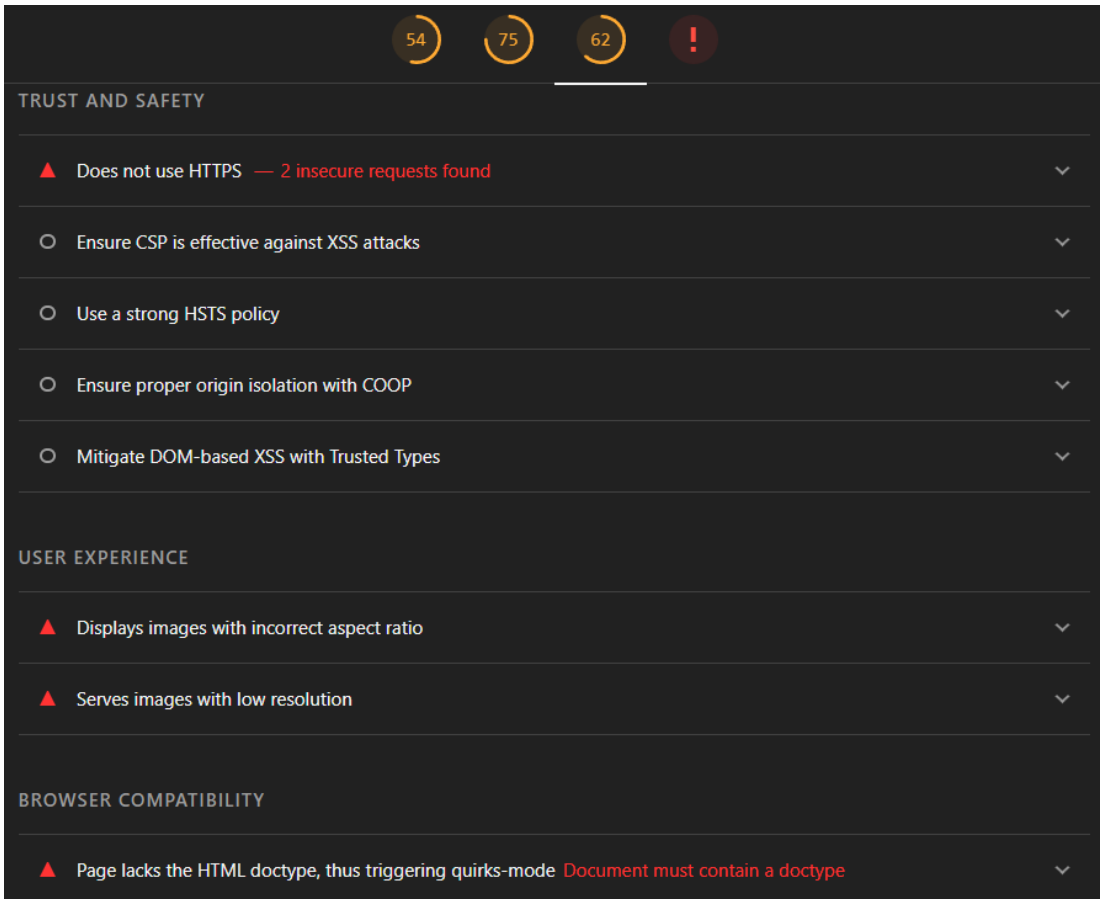


Рисунок 2.6 – Виявлені незахищені HTTP-запити.

Оцінка 62 у категорії Best Practices свідчить про наявність серйозних технічних порушень. Основна проблема, виявлена Lighthouse, – використання незахищених HTTP-запитів на сайті, що працює за протоколом HTTPS. Зокрема виявлено два незахищені запити:

ресурс зі стороннього домену creativecommons.org – шрифт /presskit/cc-icons.ttf завантажується за протоколом HTTP і був заблокований браузером;

ресурс Google Fonts через CDN – завантажується без коректних заголовків безпеки.

Змішаний контент (HTTP-ресурси на HTTPS-сторінці) є суттєвою вразливістю безпеки: зловмисники можуть перехопити або підмінити незахищені ресурси. Сучасні браузери блокують такий контент, що додатково знижує функціональність сайту.

Окрім проблем із HTTPS, аудит виявив додаткові порушення у категорії Best Practices. Сторінка не містить обов'язкового оголошення типу документа DOCTYPE, що змушує браузер переходити у режим сумісності (quirks-mode). У цьому режимі браузер застосовує застарілі правила відображення, що може призводити до некоректного рендерингу елементів у різних браузерах. Також Lighthouse зафіксував дві проблеми із зображеннями: вони відображаються з порушеними пропорціями (incorrect aspect ratio) та мають недостатню роздільну здатність для екранів з високою щільністю пікселів (low resolution). Це безпосередньо погіршує візуальне сприйняття сторінки.

2.3.4 Критичні проблеми SEO.

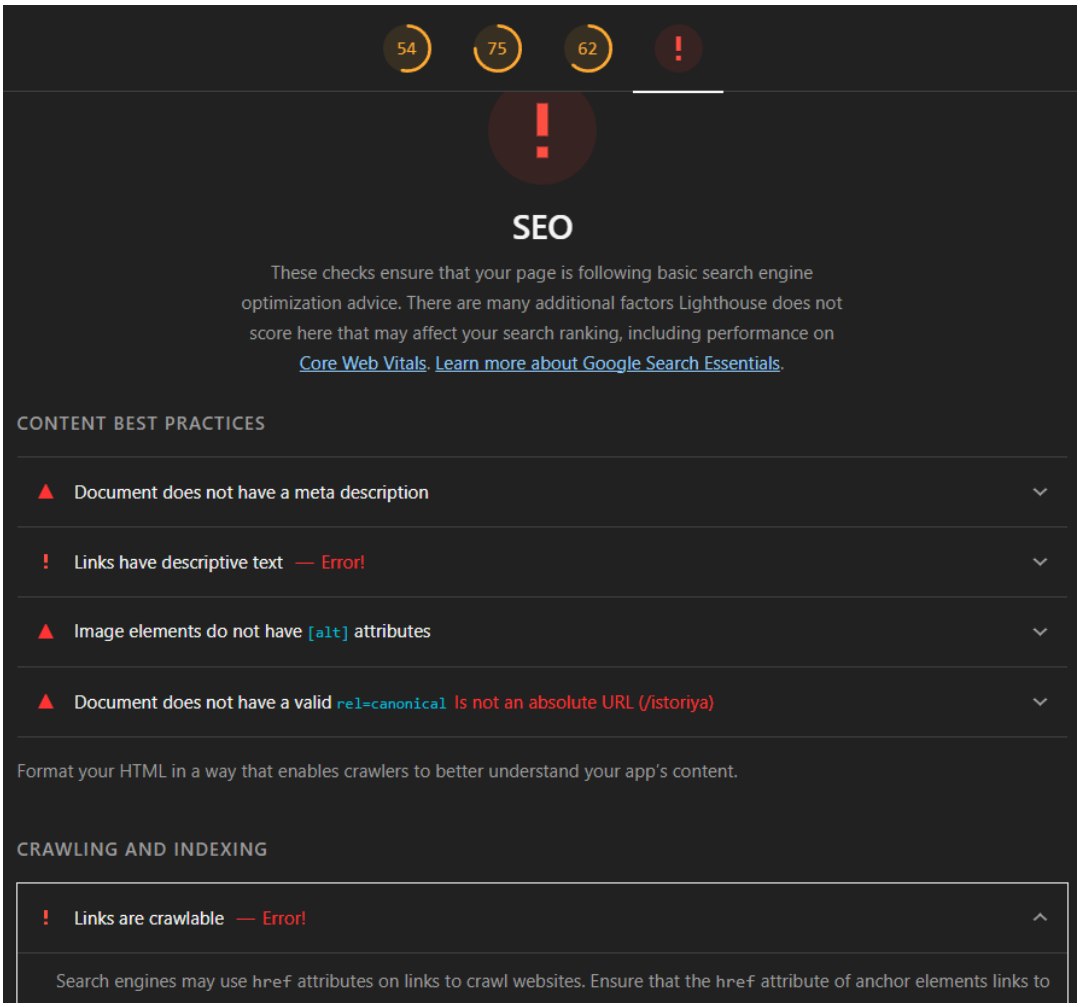


Рисунок 2.7 – Критичні SEO-помилки.

Категорія SEO отримала статус помилки (Error), що є найгіршим можливим результатом. Lighthouse виявив дві критичні проблеми:

відсутній мета-опис сторінки (meta description) – сторінка не має тегу <meta name="description">, що безпосередньо впливає на відображення сайту у результатах пошуку Google та інших пошукових систем;

посилання не є придатними для сканування (Links are crawlable – Error) – частина посилань на сайті має некоректні атрибути href, що унеможливлює їх виявлення та індексацію пошуковими роботами;

некоректний атрибут rel=canonical – сторінка використовує відносне посилання (/istoriya) замість абсолютного URL, що порушує правила визначення канонічної сторінки для пошукових систем.

Сукупність цих помилок призводить до того, що сайт коледжу фактично є частково невидимим для пошукових систем, що суттєво знижує його досяжність для потенційних абітурієнтів.

2.4 UX-аналіз існуючого вебінтерфейсу

UX-аналіз проведено на основі візуального огляду сайту з позиції трьох основних груп користувачів та перевірки відповідності принципам проєктування, розглянутим у розділі 1.

2.4.1 Аналіз користувацьких сценаріїв

Для глибшого розуміння проблем інтерфейсу розглянуто типові сценарії взаємодії трьох основних груп користувачів із сайтом.

Сценарій 1. Абітурієнт шукає інформацію про вступ.

Користувач відкриває сайт з метою дізнатися про спеціальності та умови вступу. Перша проблема виникає одразу: замість інформативної головної сторінки відкривається розділ "Історія коледжу". Абітурієнт змушений самостійно шукати потрібний розділ у перевантаженому меню з 8 пунктами верхнього рівня та паралельною бічною навігацією. Пошук контактів для уточнення деталей потребує додаткових переходів. На мобільному пристрої задача ускладнюється через відсутність адаптивного дизайну.

Рисунок 2.8 – Схема користувацького сценарію. Абітурієнт.

Сценарій 2. Студент переглядає розклад занять.

Студент відкриває сайт для перегляду розкладу. Пункт "Розклад занять" винесено окремо в нижній рядок меню, що не відповідає його пріоритетності для цієї групи користувачів. На смартфоні текст розкладу виходить за межі екрану, що унеможливлює нормальне читання без горизонтального прокручування.

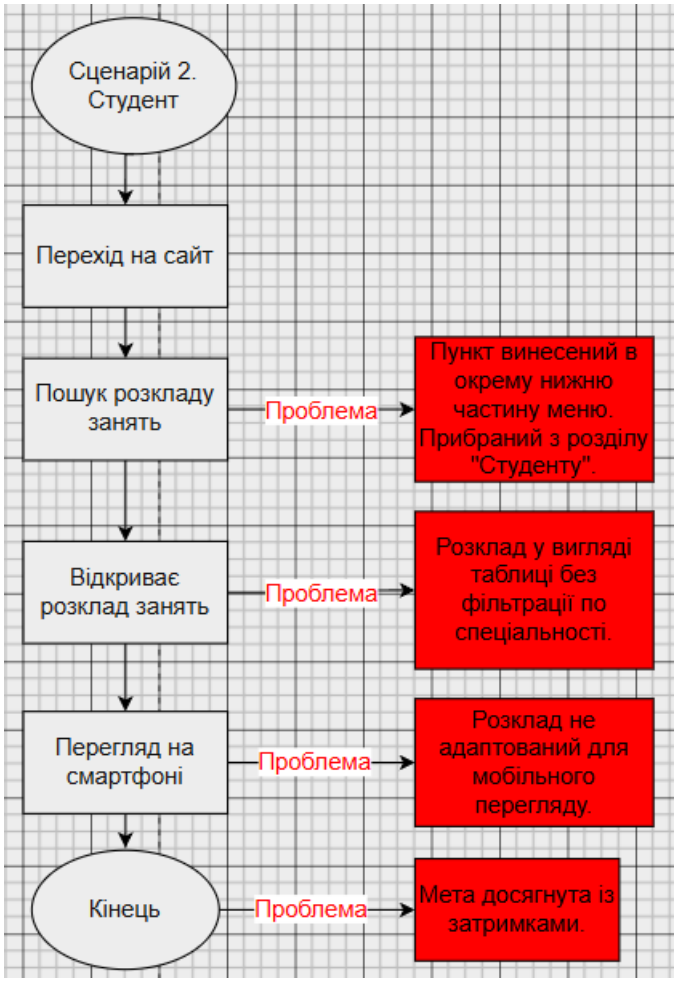


Рисунок 2.9 – Схема користувацького сценарію. Студент.

Сценарій 3. Викладач шукає документи.

Викладач намагається знайти нормативні документи або опублікувати інформацію. Через відсутність чіткої структури та неконсистентне оформлення різних підсторінок пошук потрібного розділу займає більше часу ніж необхідно.

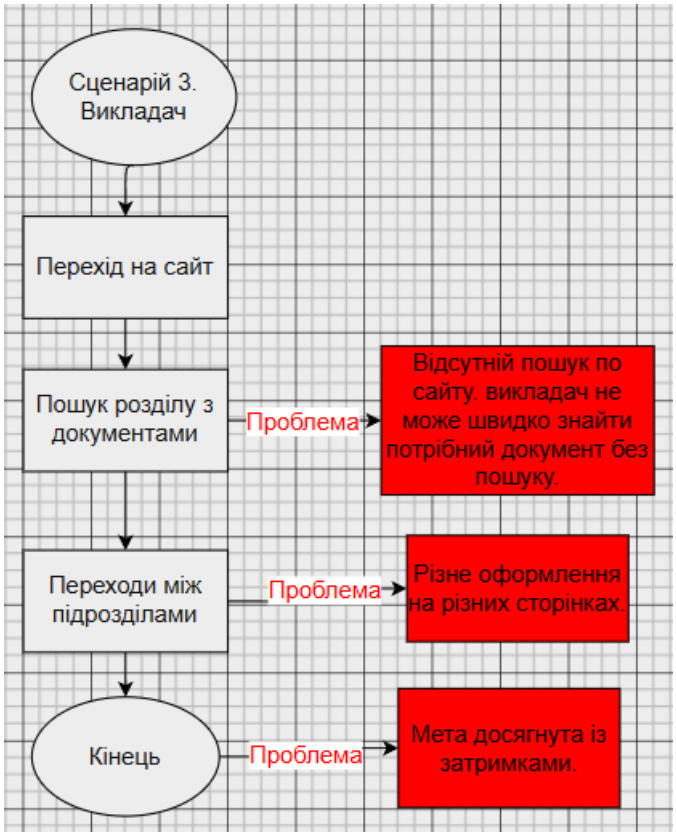


Рисунок 2.10 – Схема користувацького сценарію. Викладач.

Аналіз сценаріїв підтверджує що проблеми навігації та адаптивності однаково негативно впливають на всі групи користувачів.

2.4.2 Аналіз адаптивності вебінтерфейсу

Перевірку адаптивності проведено за допомогою інструментів Chrome DevTools у режимі емуляції пристроїв для трьох контрольних точок. Результати наведено у таблиці 2.3.

Таблиця 2.3

Результати перевірки адаптивності сайту polytechnic.ck.ua.

Пристрій	**Роздільна здатність**	**Виявлені проблеми**
Desktop	1920px	Працює задовільно, контент відображається коректно
Планшет	768px	Зміщення елементів навігації, часткове перекриття блоків
Смартфон	375px	Гамбургер-меню дублюється бічною панеллю "НАВІГАЦІЯ", зображення не мають

		фіксованої позиції і хаотично зміщуються залежно від ширини екрану

Аналіз мобільної версії сайту виявив дві характерні проблеми. По-перше, незважаючи на наявність гамбургер-меню (іконка з трьох горизонтальних ліній у верхньому лівому куті), на сторінці одночасно відображається бічна панель "НАВІГАЦІЯ" з дублюючим набором посилань. Це займає значну частину екрану смартфону і вводить користувача в оману щодо того, яким меню слід користуватись.

По-друге, зображення на сторінці не мають коректно визначених розмірів та правил позиціонування для різних розмірів екрану. Через відсутність медіа-запитів вони хаотично зміщуються – то ліворуч, то праворуч – залежно від ширини вікна браузера, що порушує візуальну структуру сторінки.

Сайт має базовий тег viewport що дозволяє масштабувати контент, однак не використовує медіа-запити (media queries) та точки перелому (breakpoints) для адаптації макету під різні розміри екрану. Це означає що мобільна версія є лише зменшеною копією десктопної, а не повноцінним адаптивним інтерфейсом.

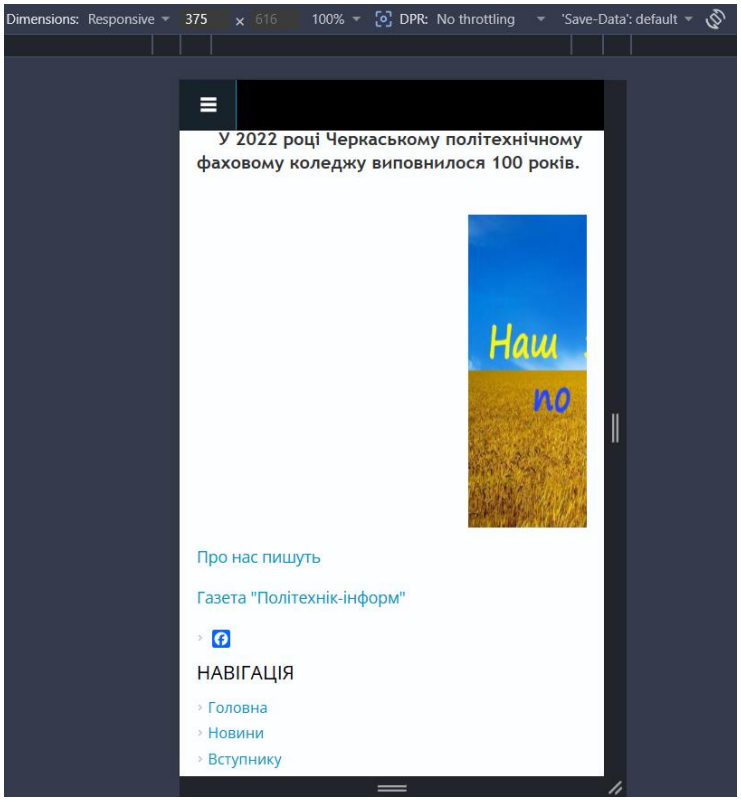


Рисунок 2.11 – Мобільна версія сайту polytechnic.ck.ua (ЧПФК) (375px): дублювання навігації та некоректне відображення зображень.

Таблиця 2.4

Виявлені UX-проблеми вебінтерфейсу polytechnic.ck.ua.

Категорія	Виявлена проблема	Вплив на користувача
Навігація	Дублювання навігаційного меню – верхнє горизонтальне меню та ліва бічна панель "НАВІГАЦІЯ" містять ті самі або схожі посилання	Когнітивне перевантаження, незрозуміло яким меню користуватись
Навігація	Надмірна кількість пунктів у меню (8 пунктів верхнього рівня + підменю)	Ускладнений пошук потрібного розділу, особливо для нових відвідувачів
Візуальна ієрархія	Відсутність чіткої головної сторінки – завантажується сторінка "Історія коледжу" замість інформативної головної	Абітурієнт не отримує ключову інформацію з першого екрану
Адаптивність	Відсутність адаптивного дизайну – при перегляді на мобільному пристрої елементи перекриваються, текст виходить за межі екрану	Неможливість зручного використання сайту на смартфоні
Консистентність	Різний стиль оформлення на різних підсторінках – різні шрифти, кольори та розміщення елементів	Відчуття ненадійності, складність орієнтування
Доступність	Низький контраст тексту в деяких блоках,	Недоступність для людей з порушеннями зору

	відсутність alt-текстів у зображень	
Контент	Наявність стороннього спам-контенту (посилань на сторонні сайти годинників у HTML-коді), high-endrolex.com, що є ознакою злому або компрометації сайту.	Загроза безпеці, зниження довіри до ресурсу

Таблиця 2.5

Пріоритетність виявлених проблем вебінтерфейсу.

Проблема	**Критичність**	**Вплив**
Критичні SEO-помилки	Висока	Сайт не знаходиться у пошукових системах
Відсутність адаптивного дизайну	Висока	Неможливість використання на мобільних пристроях
Час завантаження 12.9 секунди	Висока	Користувач залишає сайт до його відкриття
Подвійна навігація	Середня	Когнітивне перевантаження, втрата орієнтації
Відсутність alt-атрибутів	Середня	Недоступність для користувачів з порушеннями зору
Відсутність DOCTYPE	Середня	Некоректний рендеринг у різних браузерах
Змішаний HTTP/HTTPS контент	Низька	Часткове блокування ресурсів браузером

Особливої уваги заслуговує проблема дублювання навігації: сайт одночасно має горизонтальне головне меню та ліву бічну панель "НАВІГАЦІЯ", що містять перетинаючийся набір посилань. Відповідно до принципу **"Don't Make Me Think"** Стіва Круга, користувач не повинен витрачати зусилля на розуміння структури сайту – всі дії мають бути очевидними. Подвійна навігація прямо суперечить цьому принципу.

2.5 Вибір інструментів та технологій для розробки оптимізованого інтерфейсу

На основі проведеного аналізу визначено технологічний стек для розробки покращеної версії вебінтерфейсу. При виборі інструментів враховувались такі критерії: відповідність обсягу проєкту, простота підтримки, продуктивність, підтримка адаптивного дизайну та доступність.

Для реалізації оптимізованого інтерфейсу обрано такі технології:

- HTML5 та CSS3 – семантична верстка із використанням структурних елементів (<header>, <nav>, <main>, <section>, <footer>) забезпечить коректну роботу екранних читалок та покращить SEO;
- CSS Grid та Flexbox – сучасні інструменти для побудови адаптивних макетів без залежності від зовнішніх бібліотек;
- JavaScript (Vanilla JS) – для реалізації інтерактивних елементів (мобільне меню, анімації) без надлишкових залежностей;
- Figma – для проєктування прототипів та дизайн-макетів перед реалізацією.

Вибір на користь Vanilla JS замість фреймворків (React, Vue) обґрунтований тим, що сайт коледжу є переважно інформаційним ресурсом без складної динамічної логіки. Використання фреймворків у такому випадку призвело б до збільшення розміру бандлу та зниження початкової продуктивності.

Висновки до розділу 2

У другому розділі проведено комплексний аналіз існуючого вебінтерфейсу Черкаського політехнічного фахового коледжу. UX-аналіз виявив сім основних проблем, що негативно впливають на досвід різних груп користувачів, зокрема дублювання навігації, відсутність адаптивного дизайну та компрометацію коду стороннім контентом. За результатами технічного аудиту Google Lighthouse встановлено незадовільні оцінки за всіма категоріями: продуктивність 54/100, доступність 75/100, найкращі практики 62/100, SEO – критичні помилки. Порівняльний аналіз чотирьох українських освітніх вебсайтів підтвердив що проблеми з продуктивністю є системними, а polytechnic.ck.ua (ЧПФК) має найбільш комплексний набір недоліків, що робить його оптимальним об'єктом для демонстрації методів оптимізації.

На основі виявлених недоліків у третьому розділі буде розроблено оновлений вебінтерфейс із застосуванням адаптивного дизайну, оптимізації продуктивності, семантичної верстки та сучасних підходів UX/UI відповідно до стандартів WCAG 2.1.

РОЗДІЛ 3. ПРОЄКТУВАННЯ ТА РОЗРОБКА ОПТИМІЗОВАНОГО ВЕБІНТЕРФЕЙСУ ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1 Технічне завдання на розробку оптимізованого вебінтерфейсу

На підставі результатів технічного аудиту та UX-аналізу сформульовано технічне завдання на розробку покращеної версії вебінтерфейсу. Новий інтерфейс повинен відповідати таким вимогам:

Вимоги до продуктивності:

- показник Performance у Lighthouse не нижче 85 балів;
- оптимізація всіх зображень (стиснення, формат WebP, атрибути width/height);
- впровадження lazy loading для зображень;
- мінімізація CSS та JavaScript файлів.

Вимоги до доступності:

- показник Accessibility у Lighthouse не нижче 90 балів;
- усі зображення повинні мати атрибут alt;
- семантична верстка з коректним використанням HTML5-елементів;
- коефіцієнт контрастності тексту відповідно до WCAG 2.1 рівня AA (мінімум 4.5:1);
- підтримка навігації за допомогою клавіатури.

Вимоги до SEO:

- наявність мета-тегів (title, description) на кожній сторінці;
- коректний атрибут rel=canonical з абсолютними URL;
- всі посилання повинні мати коректні атрибути href;
- показник SEO у Lighthouse не нижче 90 балів.

Вимоги до UX та дизайну:

- єдина система навігації без дублювання;
- адаптивний дизайн для екранів шириною від 320px до 1920px;
- інформативна головна сторінка з чітким описом коледжу, ключовими посиланнями та контактами;
- консистентна система кольорів, шрифтів та відступів на всіх сторінках;
- використання виключно безпечних HTTPS-посилань для зовнішніх ресурсів.

3.2 Проєктування інтерфейсу у Figma

Перед початком верстки було виконано проєктування інтерфейсу у середовищі Figma. Такий підхід є обов'язковим кроком у сучасній веброзробці – він дозволяє виявити структурні та навігаційні проблеми до написання коду, перевірити логіку взаємодії користувача з інтерфейсом та суттєво скоротити кількість переробок на пізніх етапах.

Проєктування виконувалось з урахуванням усіх проблем, виявлених у другому розділі: дублювання навігації, неінформативна головна сторінка, відсутність адаптивності та чітких закликів до дії. Кожне рішення в макеті безпосередньо вирішує конкретну задачу, виявлену під час аналізу. Було розроблено макети для двох основних сторінок – головної та сторінки «Вступнику», оскільки саме вони охоплюють найважливіші сценарії використання ресурсу.

3.2.1 Wireframe головної сторінки

На першому етапі розроблено чорно-білий wireframe – схематичне зображення інтерфейсу без кольорів, шрифтів та зображень. Мета wireframe полягає у визначенні структури та порядку розміщення блоків без відволікання на візуальне оформлення. Це дозволяє швидко перевірити архітектуру інтерфейсу і переконатись що всі необхідні елементи присутні та розміщені логічно.

Структура головної сторінки побудована за принципом послідовного розкриття інформації – користувач рухається від загального знайомства з коледжем до конкретних дій:

Топбар з контактним номером телефону розміщений у самому верху. Це рішення обумовлене сценарієм абітурієнта який часто шукає контакти для уточнення деталей – тепер номер видно одразу без прокручування.

Шапка з логотипом, навігацією та кнопкою «Подати заявку» є фіксованою при прокручуванні. На відміну від оригінального сайту де навігація дублювалась у верхньому меню та бічній панелі «НАВІГАЦІЯ», нова версія містить єдину навігаційну систему. Це усуває когнітивне перевантаження описане у підрозділі 2.4.

Hero-секція – головний екран який користувач бачить першим. На оригінальному сайті відвідувач одразу потрапляв до підрозділу «Історія коледжу» що не відповідало потребам жодної з цільових груп. Новий hero-блок одразу представляє коледж, містить фотографію та дві кнопки переходу до ключових дій.

Блок швидкого доступу розміщений одразу після hero-секції і містить чотири посилання: Вступнику, Розклад занять, Спеціальності, Контакти. Це рішення безпосередньо відповідає сценаріям усіх трьох груп користувачів виявлених у підрозділі 2.4.1 – кожна група отримує швидкий перехід до потрібного розділу без пошуку в меню.

Далі розташовані секції у порядку зменшення пріоритетності: інформація про коледж зі статистикою, спеціальності у форматі карток, новини, розклад занять та контакти. Футер дублює основні посилання для зручності користувачів які дійшли до кінця сторінки.

3.2.2 Wireframe сторінки «Вступнику»

Сторінка «Вступнику» спроєктована для обслуговування найпріоритетнішого сценарію – абітурієнт шукає інформацію про вступ. На оригінальному сайті ця інформація була розпорошена між кількома підрозділами без чіткої логічної послідовності, що значно ускладнювало її пошук.

Нова структура сторінки побудована за принципом воронки – від загального до конкретного і до цільової дії:

Page hero з заголовком, підзаголовком та хлібними крихтами орієнтує користувача де він знаходиться і дозволяє повернутись на головну. Поруч з текстом розміщено фотографію студентів коледжу для візуального залучення.

Блок «Як вступити» з чотирма кроками подає процес вступу у вигляді послідовної інструкції. Це усуває невизначеність абітурієнта і формує чітке розуміння що потрібно зробити.

Таблиця спеціальностей з умовами вступу зібрана в одному місці у структурованому вигляді: назва, термін навчання, база вступу та форма навчання. На оригінальному сайті ця інформація була відсутня у такому форматі.

Блок документів та важливих дат розміщені поруч – абітурієнт одночасно бачить що зібрати і коли подати.

Форма онлайн-заявки завершує сторінку. Розміщення форми після всієї інформації є свідомим рішенням – до моменту її заповнення користувач вже ознайомився з умовами і готовий до дії. Наявність онлайн-форми усуває необхідність особистого візиту для первинного звернення, що є суттєвою перевагою для абітурієнтів з інших міст.

3.2.3 Кольорова схема та типографіка

Після затвердження wireframes розроблено повноцінний дизайн-макет із кольоровою схемою та типографікою. Для збереження впізнаваності бренду за основу взято синій колір присутній на офіційному сайті коледжу. Таблиця 3.1 містить повний перелік кольорів інтерфейсу.

Таблиця 3.1

Кольорова схема вебінтерфейсу.

Змінна	Значення	Призначення
--color-primary	#1565c0	Основний синій – фон hero, активна навігація
--color-primary-dark	#0d47a1	Темніший синій – кнопки, топбар
--color-primary-light	#e3f2fd	Світлий синій – фони при hover
--color-accent	#ffa726	Жовтогарячий – hover телефону, акценти
--color-footer	#0d1b2a	Майже чорний – фон футера

Типографіка побудована на двох шрифтах: Montserrat використовується для заголовків і навігації – геометричний гротеск що надає сучасного академічного вигляду. Open Sans застосовується для основного тексту – він відзначається широкими міжлітерними відстанями що забезпечує комфортне читання на будь-якому екрані.

Особлива увага приділена кнопкам заклику до дії. В оригінальному інтерфейсі кнопки для переходу до вступу були відсутні взагалі. У новій версії у hero-секції реалізовано дві кнопки з чіткою ієрархією: основна «Стати студентом» виконана з темним заповненням що добре читається на синьому фоні, допоміжна «Дізнатися більше» має прозору рамку і є візуально підпорядкованою. Така ієрархія відповідає принципу візуальної ієрархії описаному у підрозділі 1.2.5 і спрямовує увагу до цільової дії. Кнопка «Подати заявку» у фіксованій шапці залишається доступною на всіх сторінках незалежно від позиції прокручування.

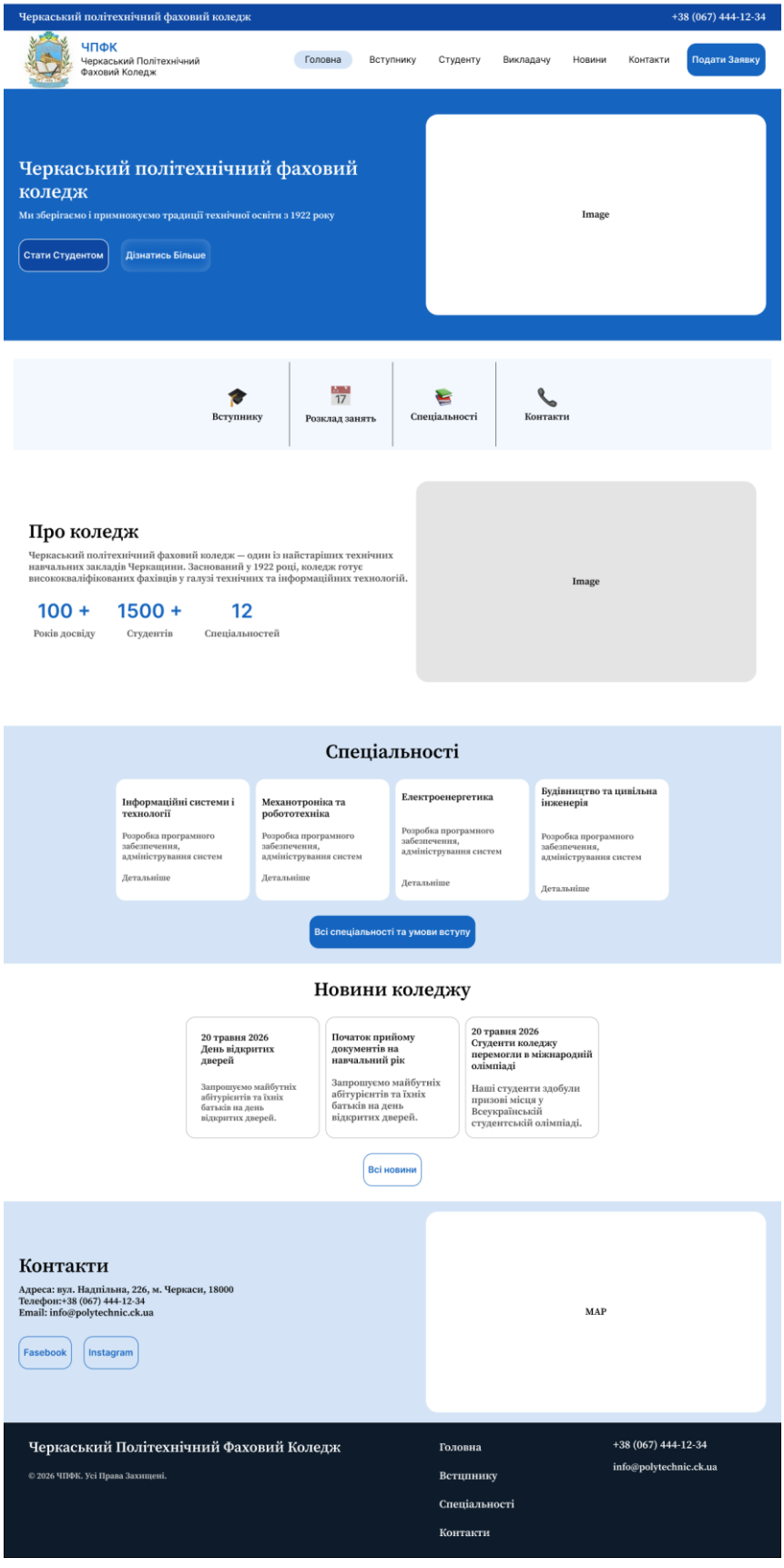


Рисунок 3.1 – Макет головної сторінки, розроблений в середовищі Figma.

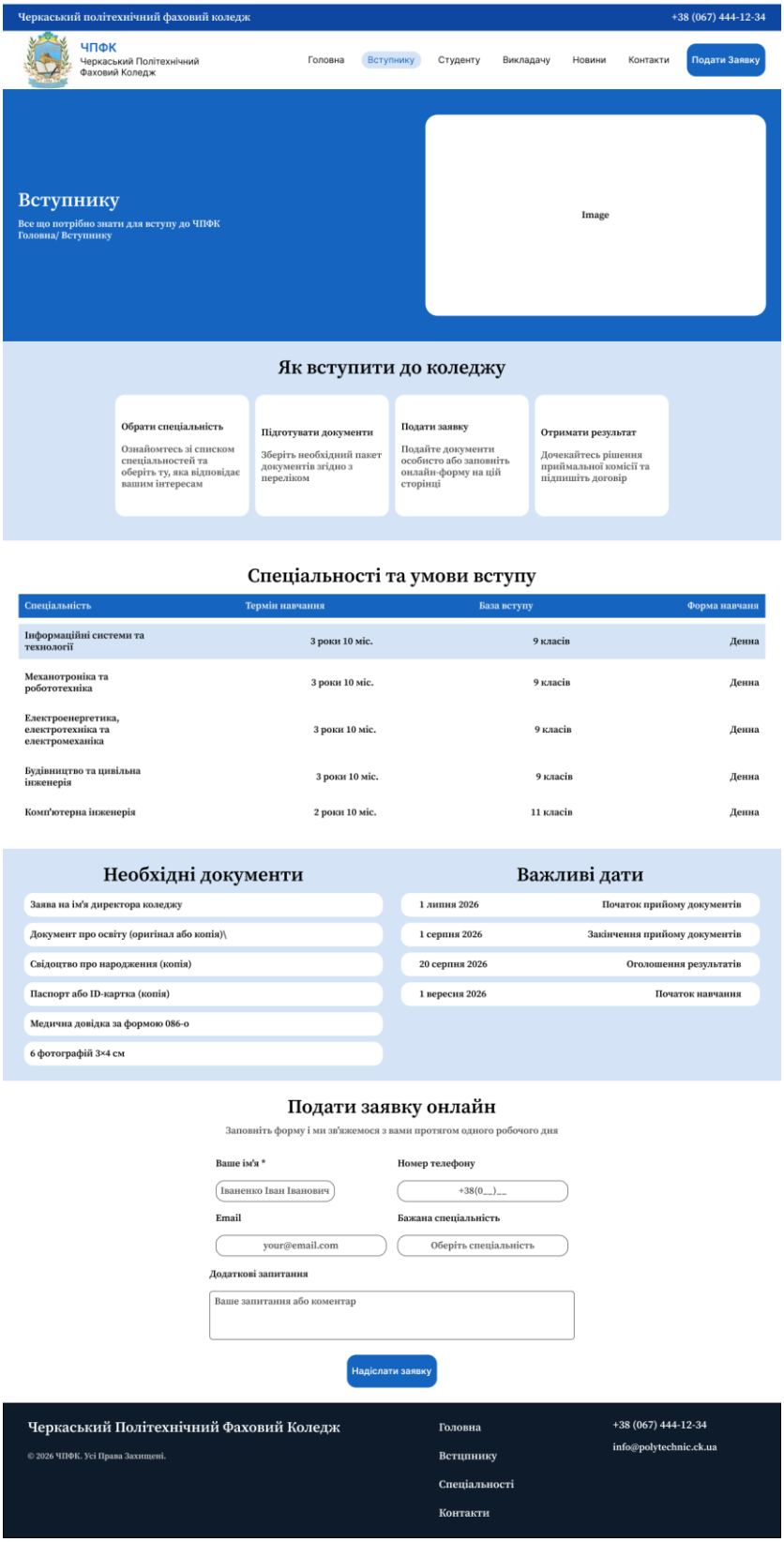


Рисунок 3.2 – Дизайн-макет сторінки «Вступнику», розроблений в середовищі Figma.

3.2.4 Адаптивні версії інтерфейсу

Wireframes розроблялись одразу для трьох контрольних точок відповідно до технічного завдання. Такий підхід називається responsive design і детально описаний у підрозділі 1.2.2.

Desktop від 1101px – повна версія з горизонтальною навігацією, hero у двох колонках (текст і фотографія), картки спеціальностей у чотири колонки.

Планшет від 876px до 1100px – навігація скорочується до основних пунктів, зображення у hero ховається і залишається тільки текстовий блок з кнопками, картки переходять у двоколонковий формат.

Мобільний до 876px – горизонтальна навігація замінюється гамбургер-меню, всі блоки відображаються в одну колонку, кнопки у hero займають повну ширину екрану для зручності натискання пальцем. Номер телефону у топбарі залишається видимим – він є прямим посиланням для дзвінка з мобільного пристрою.

У результаті розроблені макети стали повноцінною основою для верстки що розглядається у наступному підрозділі.

3.3 Розробка вебінтерфейсу інформаційної системи

Розробка оптимізованого вебінтерфейсу виконувалась на основі затверджених макетів інформаційної архітектури з послідовним застосуванням методів програмної оптимізації, визначених у першому розділі та сформульованих у технічному завданні. Реалізація проєкту здійснювалась у середовищі розробки (IDE) IntelliJ IDEA з використанням вбудованого локального сервера для рендерингу та тестування проміжних результатів.

Функціональну структуру вебінтерфейсу, що відображає архітектуру сторінок та навігаційні зв'язки між ними (включаючи внутрішню систему якорів), наведено на рисунку 3.3.

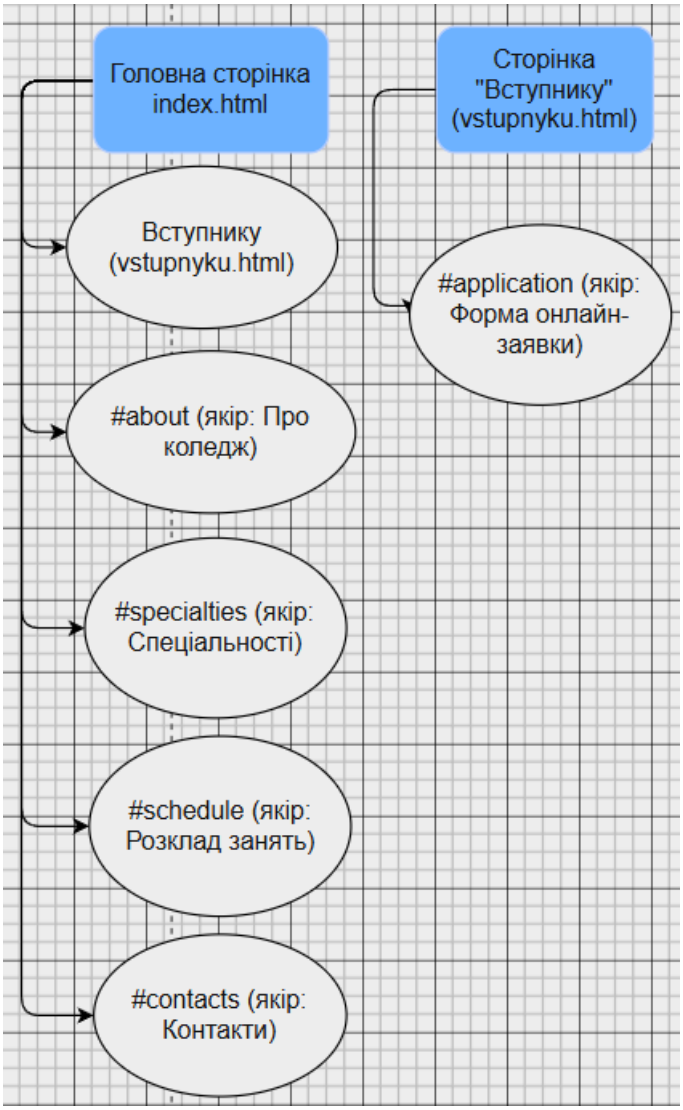


Рисунок 3.3 – Функціональна структура вебінтерфейсу коледжу.

3.3.1 Структура проєкту

Проєкт побудований за принципом мінімальної залежності від зовнішніх бібліотек та фреймворків. Як обґрунтовано у підрозділі 2.5, сайт коледжу є переважно інформаційним ресурсом без складної динамічної логіки, тому використання React або Vue призвело б до надлишкового збільшення розміру бандлу без практичної користі.

Розроблений вебінтерфейс розміщено у відкритому репозиторії GitHub за адресою: https://sm-spider.github.io/University/

Проєкт складається з двох HTML-сторінок, централізованої таблиці стилів, JavaScript-модуля та каталогу графічних ресурсів. Структуру файлів проєкту наведено на рисунку 3.4.

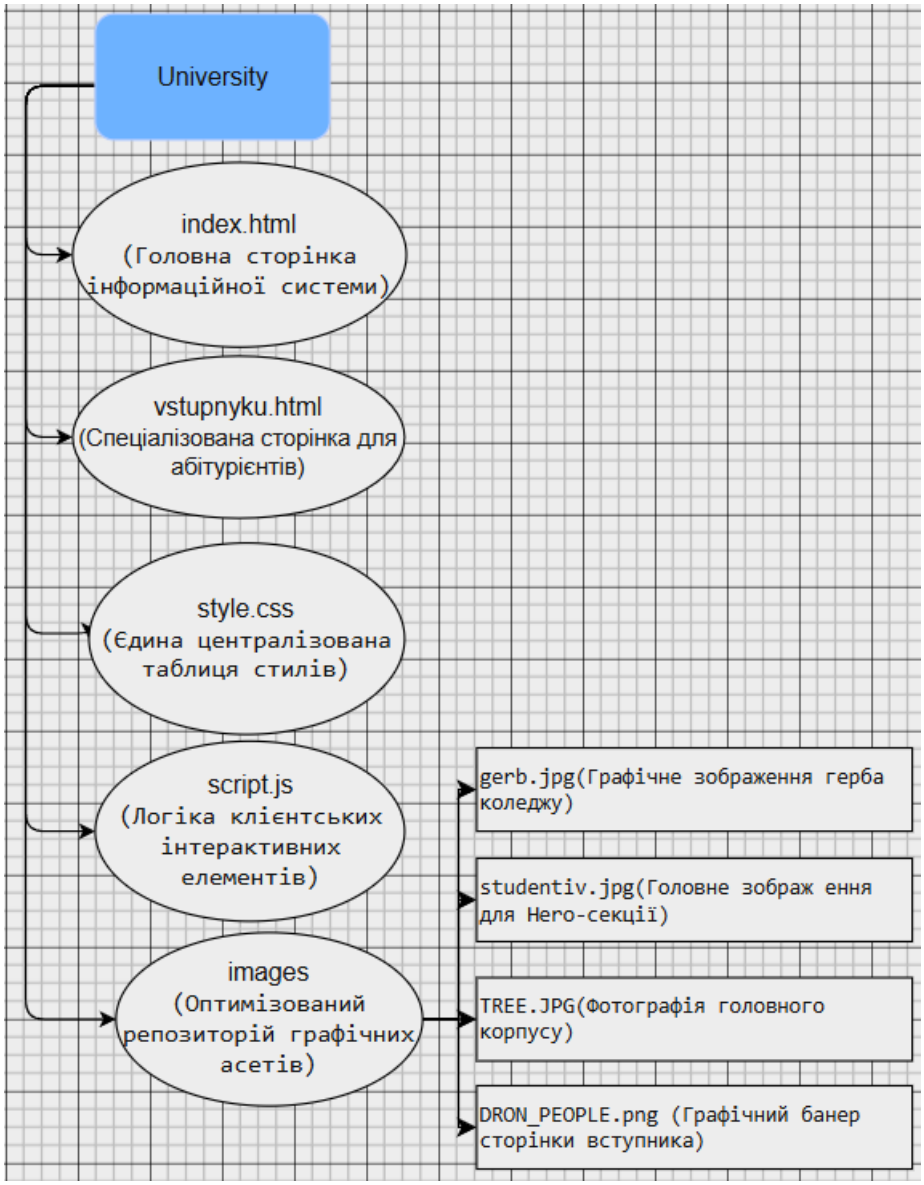


Рисунок 3.4 – Структура файлів проекту веб інтерфейсу.

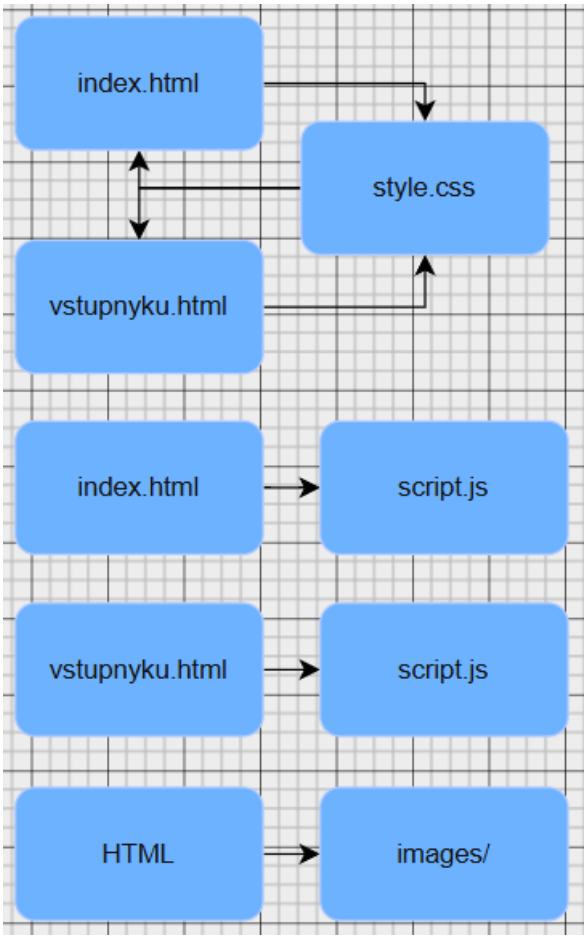


Рисунок 3.5 – Структурна схема взаємодії файлів проєкту.

Використання єдиного файлу style.css для обох сторінок є свідомим рішенням: браузер завантажує і кешує його один раз при першому відкритті, а при переході на другу сторінку файл береться з кешу без повторного завантаження. Це безпосередньо покращує показник Performance у Lighthouse для сторінки «Вступнику». Технічні характеристики розробленого вебінтерфейсу зведено у таблицю 3.2.

Таблиця 3.2

Технічні характеристики розробленого вебінтерфейсу.

Характеристика	**Значення**
Мова розмітки	HTML5
Мова стилів	CSS3 (BEM, Custom Properties)
Мова програмування	JavaScript (Vanilla JS, ES6+)

Шрифти	Montserrat, Open Sans (Google Fonts)
Кількість сторінок	2 (index.html, vstupnyku.html)
Breakpoints	1100px, 876px, 768px, 480px
Підтримка браузерів	Chrome, Firefox, Edge, Safari
Підтримка мобільних	від 320px
Валідація форми	JS (regex для email і телефону)

Навігаційна структура вебінтерфейсу спроєктована відповідно до основних сценаріїв взаємодії користувачів із системою. Схему навігаційних переходів між сторінками наведено на рисунку 3.6.

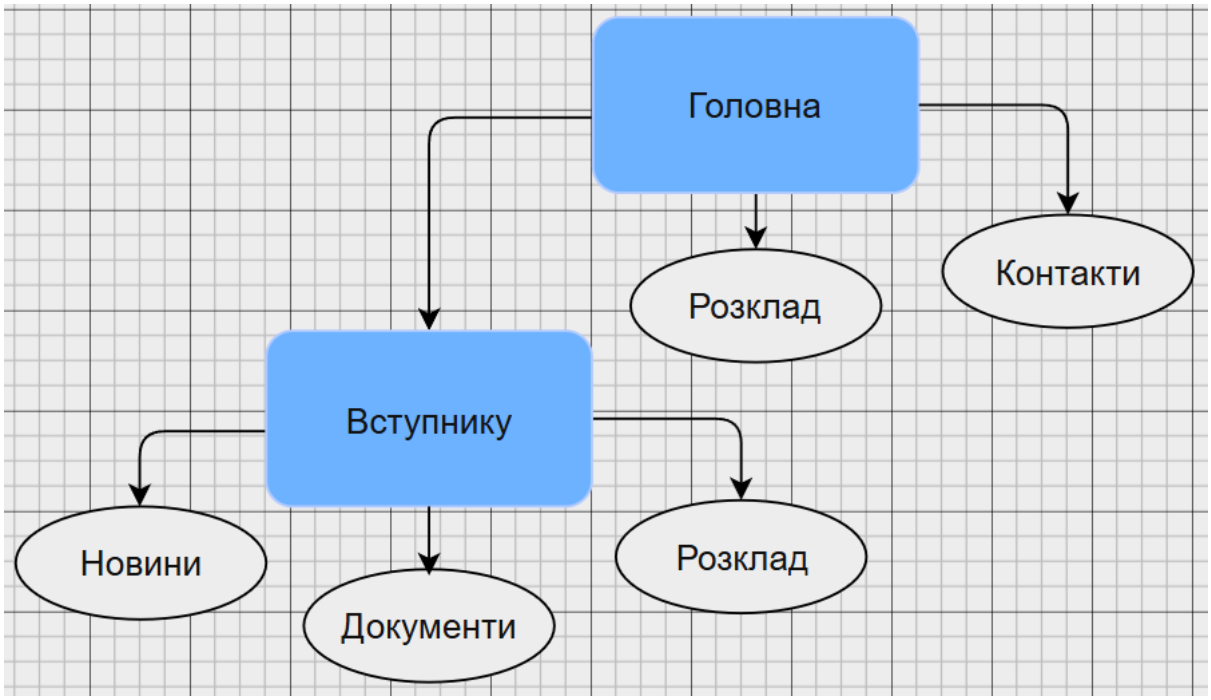


Рисунок 3.6 – Схема навігації між сторінками.

3.3.2 Семантична верстка HTML5

Семантична верстка є одним із ключових методів оптимізації доступності та SEO, описаних у підрозділі 1.4.3. На відміну від оригінального сайту коледжу, що був побудований на Drupal 7 з переважним використанням нейтральних тегів та , новий інтерфейс повністю побудований на семантичних елементах HTML5.

Детальну ієрархію елементів HTML-документа наведено на рисунку 3.7.

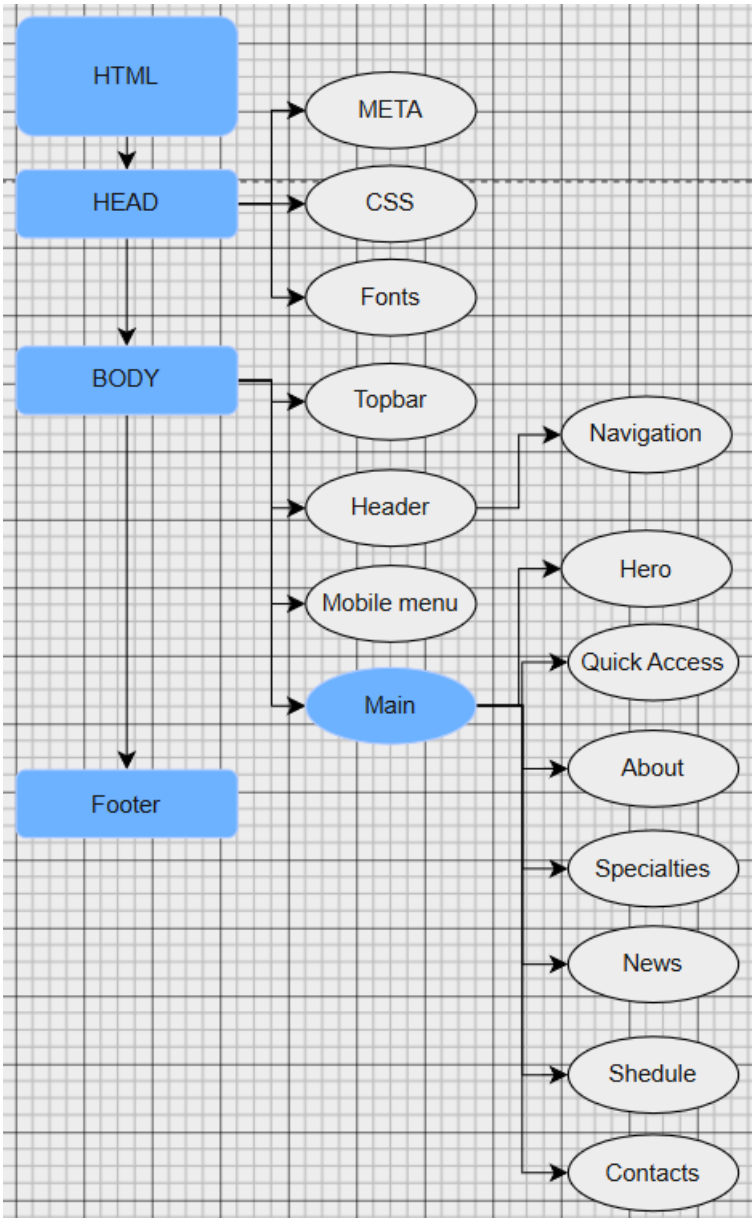


Рисунок 3.7 – Схема структури HTML-документа (ієрархічна діаграма елементів сторінки).

Такий підхід дає три практичні переваги. По-перше, екранні читалки для користувачів з порушеннями зору автоматично розпізнають структуру сторінки і дозволяють переходити між розділами без перегляду всього контенту. По-друге, пошукові роботи Google краще розуміють ієрархію контенту, що безпосередньо впливає на SEO. По-третє, семантичні теги містять вбудований функціонал доступності – наприклад, тег <button> автоматично підтримує керування клавіатурою і озвучення екранними читалками, чого не забезпечує нейтральний .

Для реалізації семантичної структури використано елементи <header>, <nav>, <main>, <section> та <footer>. Це система базових та специфічних інклюзивних елементів розмітки, характеристики яких наведено у таблиці 3.3.

Таблиця 3.3

Архітектура семантичних елементів та атрибутів вебінтерфейсу.

Елемент / Атрибут	**Призначення у структурі інтерфейсу**	**Вплив на доступність та SEO**
<header>, <footer>	Оголошення глобальної шапки та підвалу сайту	Чітке структурування зон наскрізного контенту
<nav>	Контейнер головної та мобільної навігації	Дозволяє скрінрідерам миттєво знаходити меню
<main>, <section>	Визначення зони головного контенту та логічних блоків	Пріоритизація індексації роботами Google
aria-label	Додатковий текстовий опис для блоків навігації	Пояснює призначення меню для незрячих користувачів
alt	Текстовий опис для всіх графічних елементів	Пряма вимога WCAG 2.1; усуває помилки розділу 2
width та height	Явне вказання фізичних пропорцій зображень	Запобігає зміщенню контенту, покращує метрику CLS
aria-labelledby	Логічний зв'язок контенту секції з її заголовком	Формує дерево розділів для засобів інклюзії
<address>, <time>	Розмітка контактних даних та часових міток	Переводить дані у машинозрозумілий формат

Аналогічний системний підхід застосовано до всіх секцій контенту обох розроблених сторінок. Повний код розробленого вебінтерфейсу наведено у Додатку А.

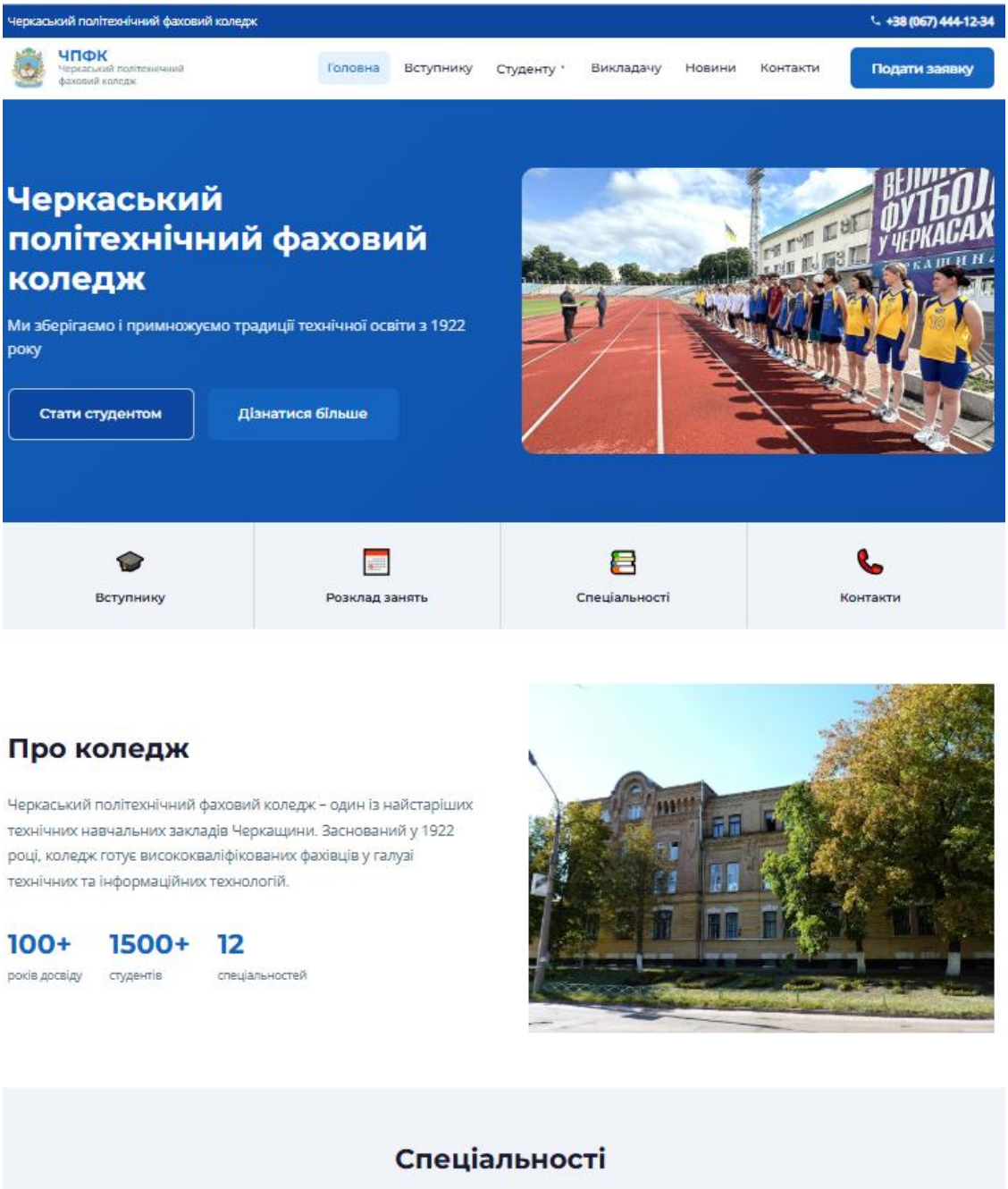


Рисунок 3.8 – Головна сторінка вебінтерфейсу, desktop версія.

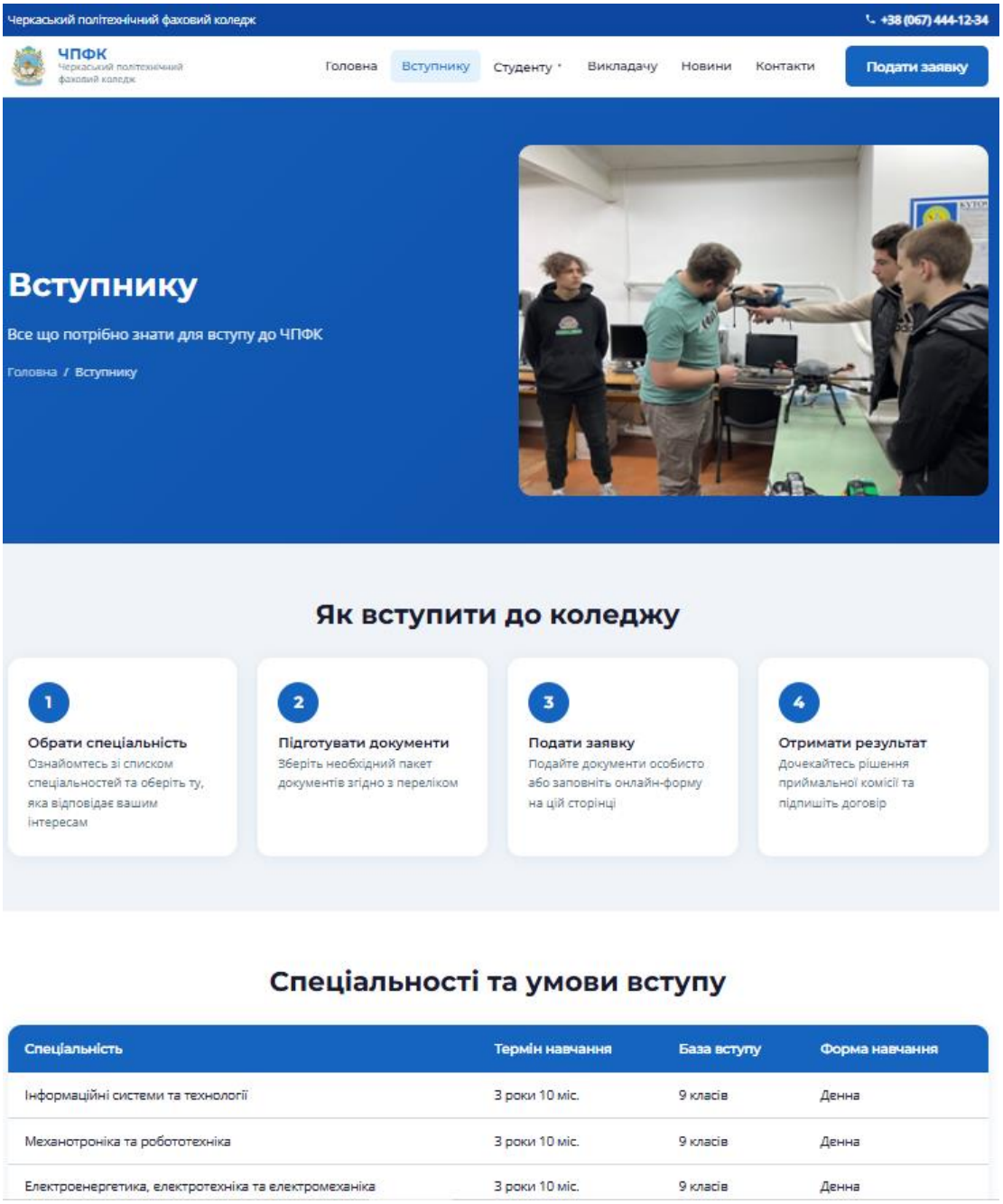


Рисунок 3.9 – Сторінка «Вступнику», desktop версія.

3.3.3 Адаптивний CSS та система стилів

Таблиця стилів організована за методологією BEM (Block Element Modifier) – підходом, де назви класів відображають ієрархію компонентів: блок, елемент всередині блоку та модифікатор стану. Наприклад, клас nav__link--active означає елемент link блоку nav у стані active. Це робить код передбачуваним, усуває конфлікти між стилями різних компонентів і спрощує підтримку проєкту.

Централізоване керування кольорами, шрифтами та розмірами реалізовано за допомогою CSS Custom Properties (змінних). Такий підхід дозволяє змінювати параметри інтерфейсу в одному місці без редагування окремих компонентів. Повний код розробленого вебінтерфейсу наведено у Додатку А.

Адаптивність вебінтерфейсу базується на поєднанні двох сучасних специфікацій CSS: одновимірного моделювання компонентів Flexbox (використано для вирівнювання елементів у топбарі, шапці, кнопках та меню) та двовимірних гнучких сіток CSS Grid (застосовано для секцій зі складною геометрією – картки спеціальностей, блоки новин, футер).

Трансформація інтерфейсу під різні пристрої регулюється за допомогою медіа-запитів (@media). Схему роботи адаптивної навігації залежно від ширини екрана пристрою наведено на рисунку 3.10.

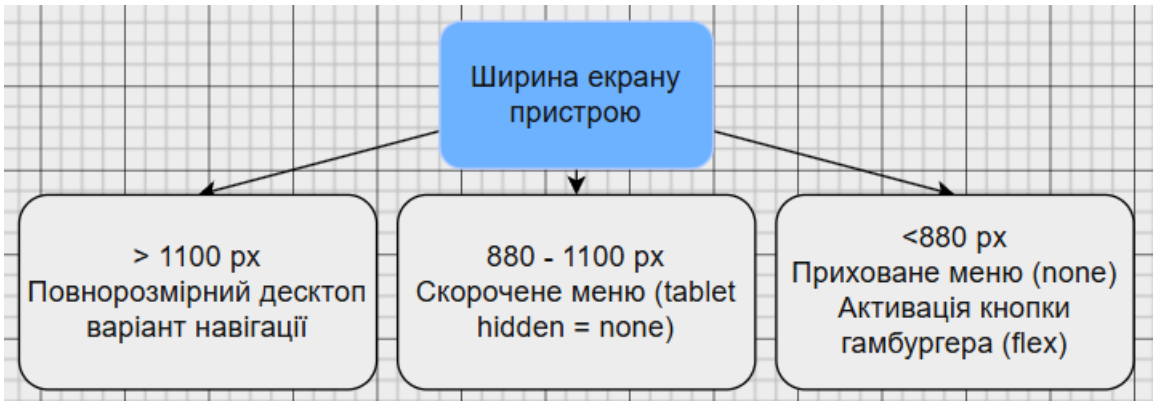


Рисунок 3.10 - Схема адаптивної навігації залежно від ширини екрана.

Адаптація інтерфейсу до різних розмірів екранів реалізована за допомогою медіа-запитів CSS. При переході до планшетних та мобільних роздільних здатностей відбувається перебудова багатоколонкових сіток, приховування другорядних елементів та заміна горизонтальної навігації на гамбургер-меню.

Головна відмінність цієї реалізації від оригінального сайту полягає в динамічній структурній перебудові. Наприклад, властивість grid-template-columns: 1fr (скидання колонок до одного значення) перетворює сітку на строго вертикальний стек, що виключає горизонтальну прокрутку (viewport overflow) на екранах мобільних телефонів. При ширині менше 480px другорядна кнопка заклику до дії (header__cta) повністю вилучається з потоку рендерингу за допомогою display: none, звільняючи корисну площу екрана під мобільну навігацію. Повний CSS-код системи стилізації додано у відповідний розділ додатків.

Принциповою відмінністю від оригіналу є те, що кожен брейкпоінт не лише масштабує контент, а й змінює структуру макета відповідно до особливостей пристрою користувача.

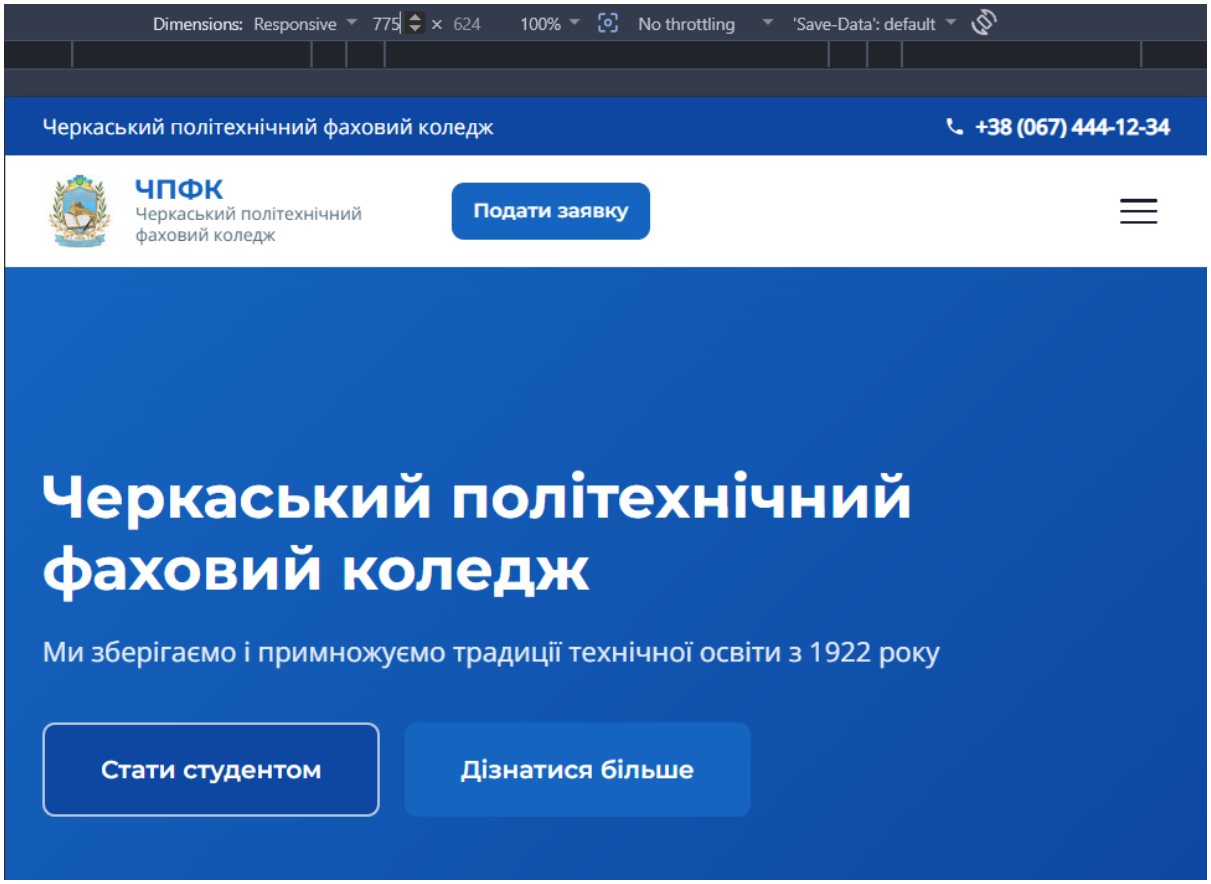


Рисунок 3.11– Адаптивна версія головної сторінки на планшеті.

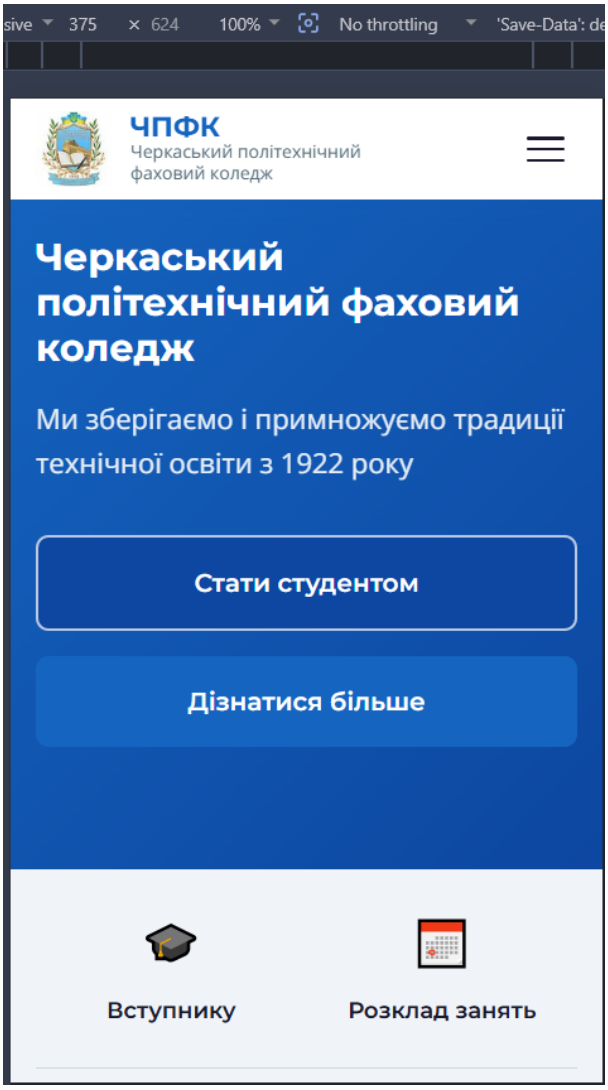


Рисунок 3.12 – Адаптивна версія головної сторінки на смартфоні 375px.

3.3.4 JavaScript: інтерактивні елементи

Логіка додається виключно туди, де стандартних можливостей HTML5 та CSS3 недостатньо для забезпечення повноцінного UX-сценарію. Такий підхід гарантує високу швидкість парсингу скриптового потоку. У проєкті реалізовано чотири клієнтських модулі:

1. **Гамбургер-меню.** Відстежує натискання кнопки і перемикає клас is-open на мобільному меню та is-active на кнопці, що також анімує три риски у хрестик. Одночасно оновлюються aria-атрибути для підтримки доступності – aria-expanded повідомляє допоміжним технологіям, чи відкрите меню в даний момент. Меню автоматично закривається при кліку на будь-яке посилання всередині або при кліку поза його межами.

2. **Dropdown-навігація** реалізована для пункту «Студенту» – при натисканні розкривається підменю з трьома посиланнями. Атрибут aria-expanded динамічно змінюється при відкритті та закритті, що дозволяє екранним читалкам коректно повідомляти стан меню.
3. **Клієнтська валідація форми онлайн-заявки:** Найскладніший логічний блок клієнтської частини. Валідація форми перевіряє послідовно три умови: заповненість обов'язкових полів (ім'я, телефон, спеціальність), коректність формату телефону через регулярний вираз та коректність формату email, якщо поле заповнене. Email є необов'язковим полем – якщо воно порожнє, перевірка пропускається. При успішному проходженні всіх перевірок відображається повідомлення про прийняту заявку.
4. **Плавний скрол**, який реалізований для всіх посилань з якорями. При кліку на посилання типу «Розклад занять» у блоці швидкого доступу сторінка плавно прокручується до відповідного розділу замість миттєвого стрибка – це покращує відчуття від взаємодії з інтерфейсом.

Схему алгоритму найскладнішого з них, валідації форми, наведено на рисунку 3.13.

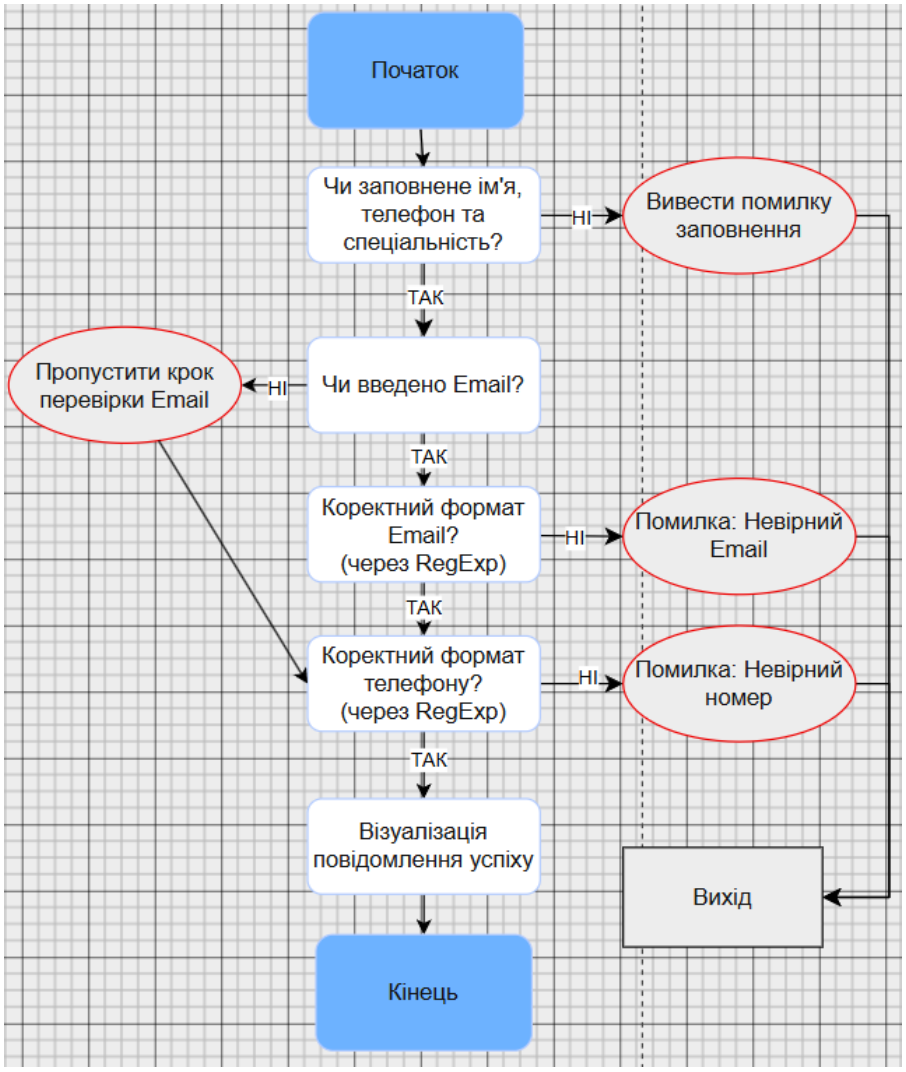


Рисунок 3.13 – Блок-схема алгоритму клієнтської валідації форми заявки вступника.

Візуальне відображення розроблених інтерактивних модулів на реальних пристроях у процесі взаємодії з користувачем представлено на рисунках 3.14 та 3.15.

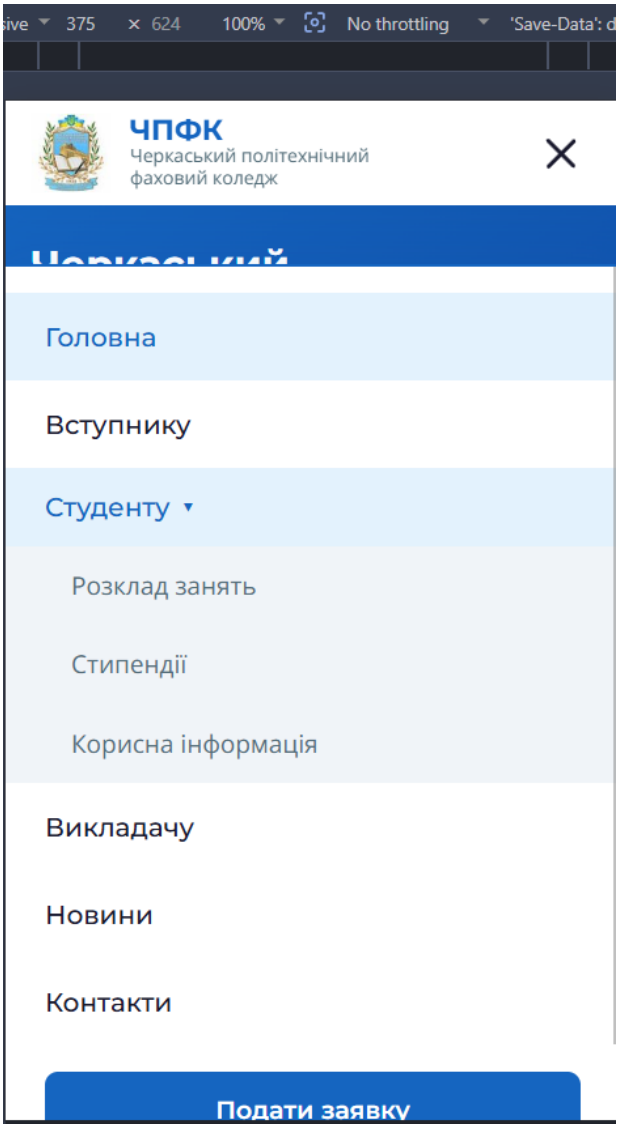


Рисунок 3.14 – Відкрите мобільне меню з dropdown «Студенту».

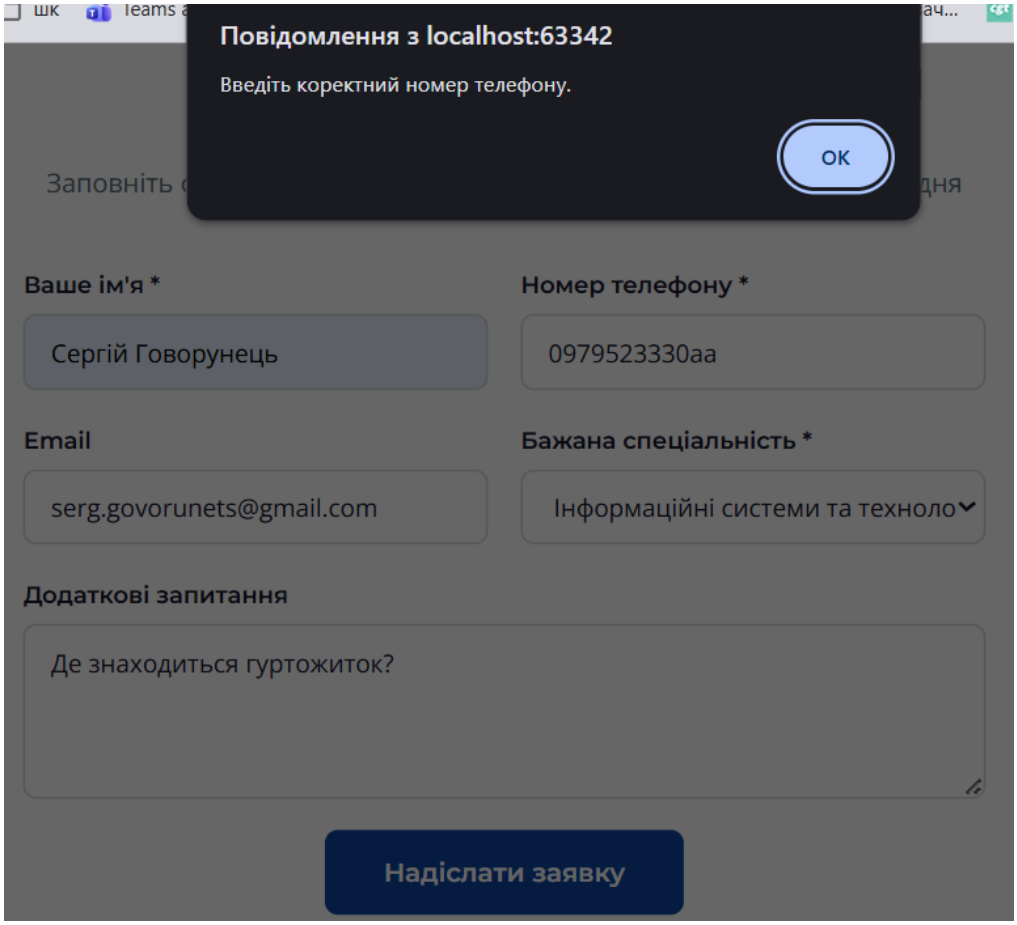


Рисунок 3.15 – Форма заявки з валідацією.

3.3.5 Оптимізація продуктивності при розробці

Паралельно з верстkою впроваджувались методи оптимізації продуктивності, описані у підрозділі 1.4.1.

Директива <link rel="preload"> в <head> кожної сторінки вказує браузеру завантажити головне зображення hero-секції до початку рендерингу контенту. Це безпосередньо покращує показник Largest Contentful Paint – один з ключових показників Core Web Vitals, що впливає на позицію сайту у пошукових системах.

Атрибут loading="lazy" застосовано до всіх зображень, крім головного у першому екрані. Це означає, що зображення секцій «Про коледж» та «Вступнику» завантажуються лише тоді, коли користувач прокручує до них сторінку, а не всі одразу при ініціалізації сесії. Для сайту з кількома великими фотографіями це суттєво скорочує початковий час завантаження.

Підключення шрифтів Google Fonts оптимізовано через директиви <link rel="preconnect">, що встановлюють TCP-з'єднання з серверами Google до

початку безпосереднього завантаження самих шрифтів і скорочують затримку на 100–300 мс залежно від швидкості мережі.

Зовнішні ресурси підключаються виключно за захищеним протоколом HTTPS, що усуває проблему змішаного контенту (Mixed Content), виявлену на оригінальному сайті. На оригінальному сайті два HTTP-ресурси блокувались сучасними браузерами і знижували бал Best Practices до 62 – в новій версії ця вразливість усунена повністю.

Семантична верстка з коректними атрибутами alt, наявність мета-тегу <meta name="description">, валідний rel="canonical" з абсолютними URL-адресами та правильні атрибути href у посиланнях забезпечують усунення всіх чотирьох категорій SEO-помилок, виявлених у розділі 2.

3.4 Результати оптимізації вебінтерфейсу

Після завершення розробки проведено повторний аудит вебінтерфейсу за допомогою Google Lighthouse для об'єктивної оцінки досягнутих результатів. Аудит виконувався в ідентичних умовах до початкового – у браузері Google Chrome у режимі Desktop для головної сторінки сайту. Це забезпечує коректність порівняння оскільки початковий аудит у розділі 2 також проводився у режимі Desktop.

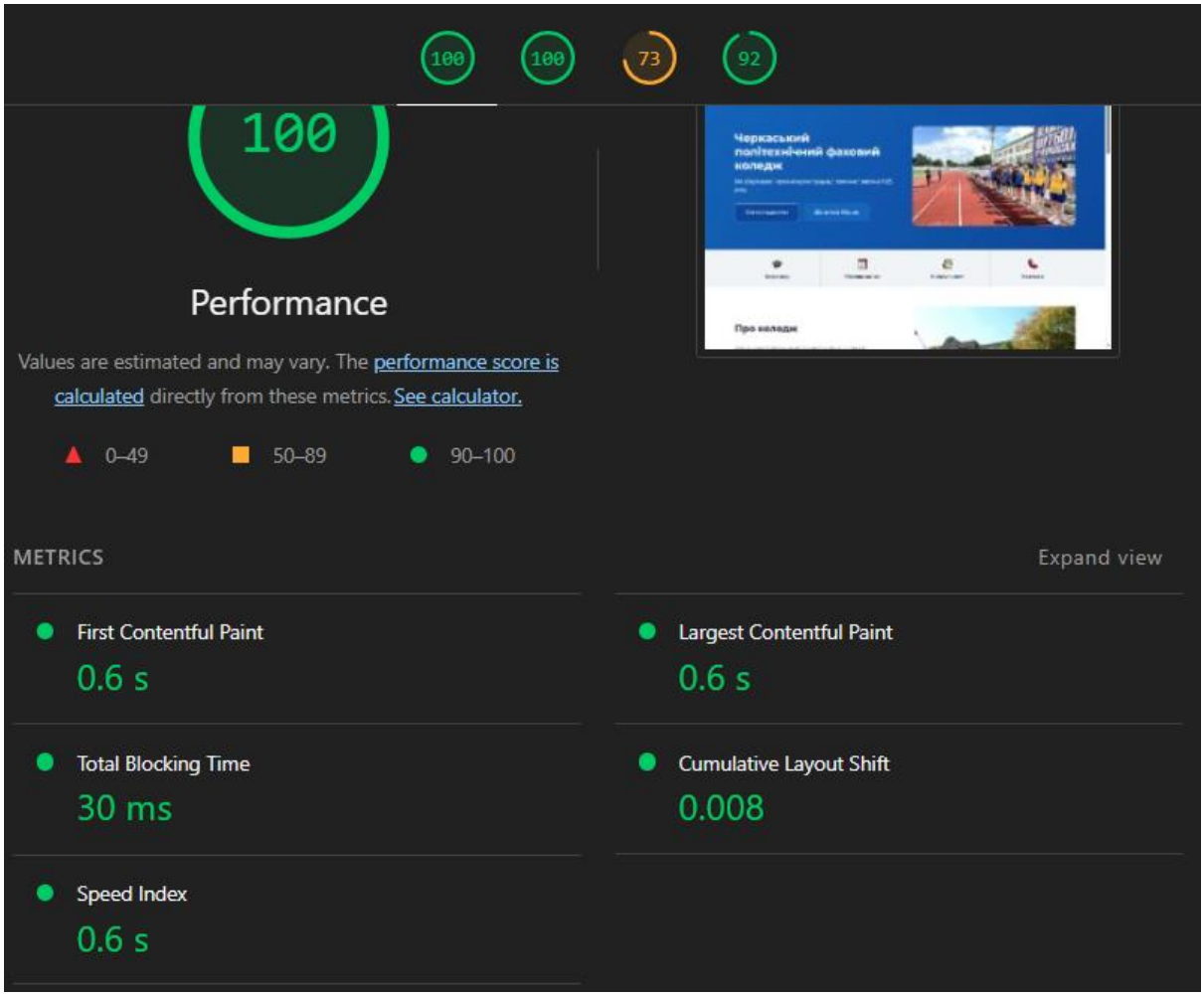


Рисунок 3.16 – Результати Lighthouse для головної сторінки після оптимізації.

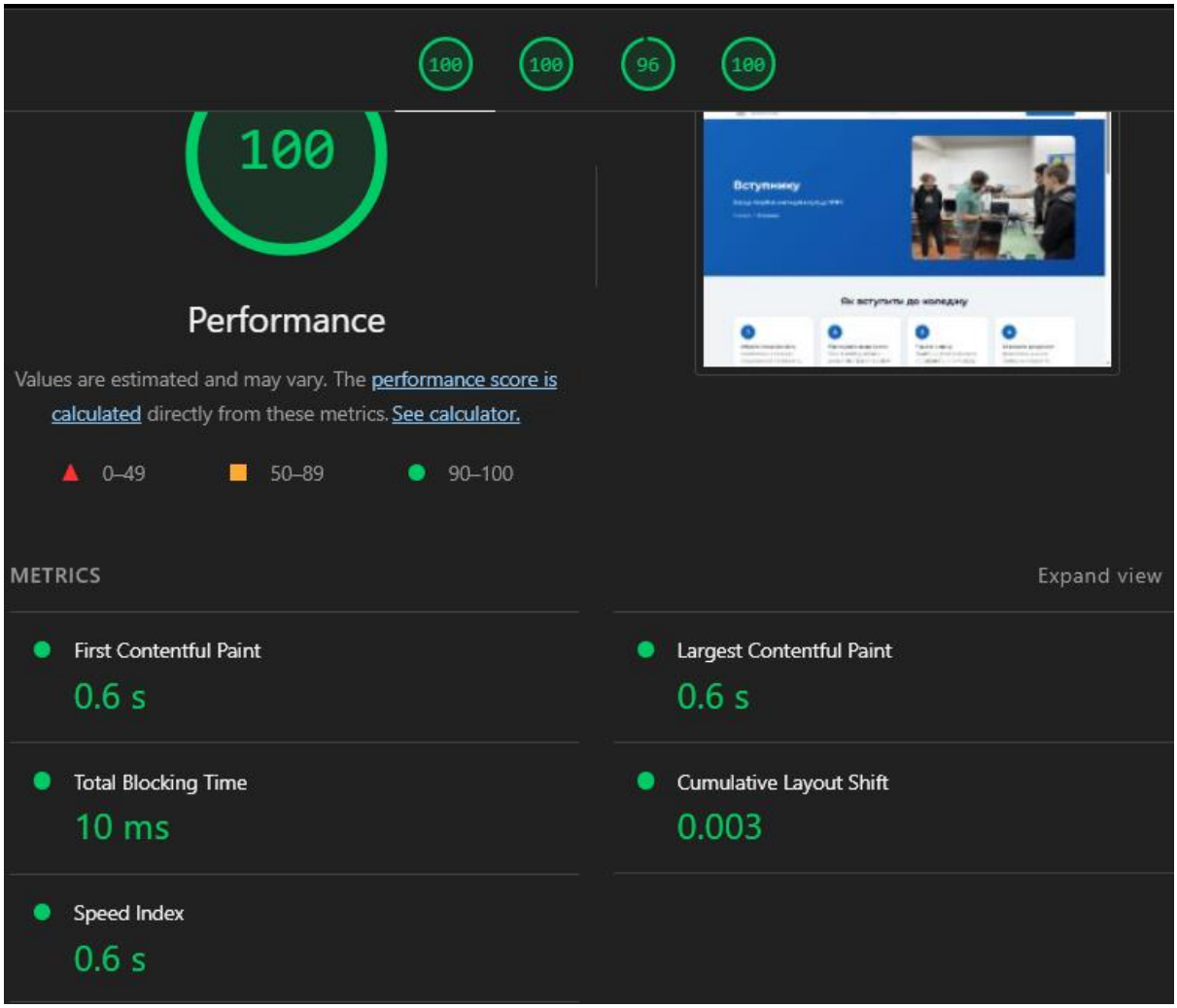


Рисунок 3.17 – Результати Lighthouse для сторінки «Вступнику» після оптимізації.

Порівняльний аналіз показників до та після оптимізації наведено у таблиці 3.4.

Таблиця 3.4

Порівняння показників Lighthouse до та після оптимізації.

Категорія	**До оптимізації**	**Після оптимізації**	**Зміна**
Performance	54	100	+46
Accessibility	75	100	+25
Best Practices	62	96	+34
SEO	Помилка	92	—

Усі чотири категорії досягли або перевищили цільові показники, визначені у технічному завданні підрозділу 3.1. Показники Performance та Accessibility досягли максимального значення 100 балів, тоді як Best Practices та SEO перевищили поріг 90 балів. Таким чином, розроблений вебінтерфейс відповідає сучасним вимогам щодо продуктивності, доступності, якості реалізації та пошукової оптимізації.

Окремо було проведено аудит сторінки «Вступнику» (рисунок 3.17). Незважаючи на більший обсяг контенту порівняно з головною сторінкою, зокрема наявність таблиці спеціальностей, переліку документів, важливих дат та форми онлайн-заявки, сторінка також продемонструвала високі показники якості та відповідності сучасним вебстандартам.

Окрім загальних балів суттєво покращились конкретні метрики продуктивності що безпосередньо відчувають користувачі. Порівняння наведено у таблиці 3.5.

Таблиця 3.5

Порівняння метрик продуктивності.

Метрика	**До оптимізації**	**Після оптимізації**
First Contentful Paint	12.9 s	0.6 s
Largest Contentful Paint	12.9 s	0.6 s
Speed Index	12.9 s	0.6 s
Cumulative Layout Shift	0.064	0.008
Total Blocking Time	30 ms	30 ms

Показник Total Blocking Time залишився на рівні 30 мс – це єдина метрика що не покращилась, проте вона і не погіршилась. Значення 30 мс є відмінним результатом відповідно до шкали Lighthouse (норма до 200 мс) і не потребує додаткової оптимізації.

3.4.1 Аналіз покращення продуктивності

Показник Performance зріс з 54 до 100 балів, що становить приріст на 46 пунктів. Найбільш показовим є покращення метрик FCP та LCP з 12,9 секунди до 0,6 секунди, тобто більш ніж у 21 раз. З практичної точки зору це означає, що користувач отримує доступ до основного контенту сторінки менш ніж за одну секунду замість очікування майже 13 секунд. Скорочення

часу завантаження позитивно впливає на користувацький досвід та зменшує ймовірність дострокового закриття сторінки відвідувачами. Для сайту коледжу це прямо впливає на кількість абітурієнтів які дочекаються завантаження і ознайомляться з інформацією про вступ.

Незважаючи на отримання максимальної оцінки 100 балів, результати Lighthouse можуть незначно відрізнятися після розгортання сайту на реальному сервері. На підсумкові показники впливають швидкість мережевого з'єднання, налаштування кешування, конфігурація вебсервера та робота зовнішніх сервісів. Проте проведений аудит підтверджує, що на рівні програмного коду всі основні проблеми продуктивності були усунені.

Такий результат досягнутий комплексним застосуванням кількох методів одночасно. Директива <link rel="preload"> для головного зображення hero-секції дозволила браузеру завантажувати фотографію паралельно з парсингом HTML – це усунуло головне вузьке місце оригінального сайту де всі ресурси завантажувались послідовно. На оригінальному сайті LCP становив 12.9 секунди саме через відсутність пріоритизації критичних ресурсів.

Атрибут loading="lazy" на всіх зображеннях крім головного суттєво зменшив обсяг даних що завантажуються при першому відкритті. Браузер ініціює завантаження зображення секції «Про коледж» та «Вступнику» лише коли користувач прокручує до них – у більшості випадків значна частина зображень не завантажується взагалі якщо користувач знайшов потрібну інформацію у верхній частині сторінки.

Відмова від JavaScript-фреймворків на користь Vanilla JS виключила необхідність завантажувати бібліотечний код. Для порівняння – мінімальна збірка React з ReactDOM важить близько 140 кілобайт, тоді як весь файл script.js розробленого сайту значно менший і містить лише чотири функціональних блоки що реально використовуються.

Директиви <link rel="preconnect"> для Google Fonts встановлюють TCP-з'єднання з CDN-сервером заздалегідь – браузер не витрачає час на DNS-розпізнавання і встановлення з'єднання у момент коли шрифт фактично потрібен. Це скорочує затримку завантаження шрифтів на 100–300 мс залежно від мережевих умов.

3.4.2 Аналіз покращення доступності

Показник Accessibility досяг максимального значення 100 балів – приріст на 25 пунктів з початкових 75. Це означає, що новий інтерфейс відповідає

автоматично перевірюваним критеріям доступності, які оцінюються інструментом Lighthouse відповідно до рекомендацій WCAG 2.1.

Оригінальний сайт мав дві критичні проблеми з доступністю виявлені у підрозділі 2.2.2. Перша – логотип коледжу та зображення партнерів не мали атрибута alt. Для незрячого користувача що використовує екранну читалку ці зображення були просто невидимими – читалка або пропускала їх або озвучувала URL файлу. У новій версії кожне зображення має описовий alt: логотип описується як «Герб Черкаського політехнічного фахового коледжу», фотографія – як «Студенти Черкаського політехнічного фахового коледжу».

Друга проблема – посилання без описового тексту. Частина посилань на оригінальному сайті мала нейтральний або відсутній текст що унеможливлювало навігацію за допомогою читалки. У новій версії всі посилання мають зрозумілі текстові назви або атрибут aria-label з поясненням.

Додатково реалізовано динамічну підтримку доступності для інтерактивних елементів. Атрибут aria-expanded на гамбургер-кнопці змінюється між false і true при відкритті та закритті меню – екранна читалка повідомляє користувачу поточний стан. Атрибут aria-controls пов'язує кнопку з відповідним елементом меню. Це забезпечує повноцінну навігацію клавіатурою для людей з порушеннями моторики.

Контрастність всіх текстових елементів перевірена відповідно до WCAG 2.1 рівня AA. Мінімальний допустимий коефіцієнт для нормального тексту – 4.5:1. Основна комбінація синього #1565c0 на білому фоні забезпечує коефіцієнт 5.9:1, а білий текст на синьому фоні hero-секції – 7.2:1, що відповідає навіть більш суворому рівню AAA.

3.4.3 Аналіз покращення Best Practices

Категорія Best Practices продемонструвала суттєве покращення порівняно з початковим аудитом. Для сторінки «Вступнику» показник становить 96 балів, тоді як для головної сторінки він зафіксувався на рівні 73 балів. Причина зниження оцінки головної сторінки не пов'язана з якістю власного коду проєкту, а обумовлена використанням стороннього віджета Google Maps у секції контактів.

Алгоритм Lighthouse фіксує, що вбудована карта Google Maps використовує сторонні файли cookies для роботи сервісу та аналітики. Через сучасні обмеження браузерів щодо сторонніх cookies це автоматично впливає на оцінку категорії Best Practices. Оскільки код віджета належить компанії

Google, розробник сайту не має можливості змінити його внутрішню реалізацію або заборонити встановлення відповідних cookies.

Таким чином, усі зауваження Lighthouse для головної сторінки пов'язані із зовнішнім сервісом Google Maps, тоді як власний HTML-, CSS- та JavaScript-код проєкту відповідає вимогам категорії Best Practices.

Головною проблемою на оригінальній сторінці був змішаний контент – два HTTP-ресурси на HTTPS-сторінці. Сучасні браузери Chrome та Firefox блокують такі ресурси за замовчуванням що означало що частина функціональності оригінального сайту просто не завантажувалась. У новій версії всі зовнішні ресурси – шрифти Google Fonts та SVG-іконка телефону – підключаються виключно через HTTPS.

Відсутність DOCTYPE на оригінальному сайті змушувала браузер переходити у quirks-mode м режим сумісності зі старими стандартами де кожен браузер інтерпретує CSS по-різному. Це могло призводити до різного відображення сайту у Chrome, Firefox та Edge. У новій версії <!DOCTYPE html> оголошено першим рядком обох HTML-файлів що гарантує стандартний режим рендерингу в усіх браузерах.

Зображення на оригінальному сайті відображались з некоректними пропорціями – браузер масштабував їх без збереження оригінального співвідношення сторін. У новій версії всі зображення мають явно вказані атрибути width і height що відповідають реальним розмірам файлів.

3.4.4 Аналіз покращення SEO

SEO є категорією з найбільш радикальним покращенням – з критичної помилки (статус Error) до 92 балів. Оригінальний сайт мав чотири блокуючі проблеми що фактично частково виключали його з результатів пошукових систем. Для сайту освітнього закладу це означало що абітурієнти які шукають інформацію про вступ могли просто не знайти коледж у Google.

Додавання <meta name="description"> вирішило першу проблему – тепер при відображенні у результатах пошуку Google показує конкретний опис сайту «Черкаський політехнічний фаховий коледж – офіційний вебсайт. Вступ, спеціальності, розклад занять» замість випадкового фрагменту тексту. Дослідження показують що наявність релевантного опису підвищує CTR (клікабельність) у пошуковій видачі на 5–30%.

Коректний rel="canonical" з абсолютною URL-адресою усунув неоднозначність щодо основної версії сторінки. На оригінальному сайті

використовувався відносний шлях /istoriya що не є валідним значенням для canonical і могло призводити до дублювання контенту в індексі Google.

Виправлення атрибутів href у всіх посиланнях дозволило пошуковим роботам повноцінно сканувати та індексувати всі розділи сайту. Раніше частина внутрішніх посилань з некоректними href була недосяжною для GoogleBot.

Open Graph мета-теги, додані до головної сторінки, технічно не впливають на бал SEO в Lighthouse але є важливою складовою сучасної веб-присутності. При поширенні посилання на сайт у Telegram, Facebook або Viber відображатиметься структурована картка з назвою коледжу, описом та зображенням герба – замість простого посилання без оформлення.

Показник SEO на рівні 92 балів пояснюється особливостями тестування проєкту в локальному середовищі розробки. Під час аудиту Lighthouse перевіряє коректність канонічних URL-адрес та наявність допоміжних файлів для пошукових систем. Оскільки перевірка виконувалася на локальній адресі localhost, частина SEO-рекомендацій не могла бути повністю підтверджена.

Після розгортання сайту на реальному домені та додавання файлів robots.txt і sitemap.xml оцінка SEO потенційно може бути підвищена без внесення змін до структури сторінок або їхнього контенту.

3.4.5 Порівняльний аналіз UX-покращень

Окрім технічних показників Lighthouse суттєво покращився загальний користувацький досвід. Таблиця 3.6 містить порівняння ключових UX-характеристик.

Таблиця 3.6

Порівняння UX-характеристик оригінального та оптимізованого вебінтерфейсу.

Характеристика	Оригінальний сайт	Новий інтерфейс
Головна сторінка	Відкривається розділ «Історія коледжу»	Інформативна hero-секція з описом коледжу
Навігація	Дублюється у двох місцях (8+20 пунктів)	Єдина система навігації з dropdown
Кнопки заклику до дії	Відсутні	Три кнопки на головній, постійна кнопка в шапці

Адаптивність	Відсутня (лише desktop)	Три версії: 1101px / 876px / 375px
Форма заявки	Відсутня	Онлайн-форма з валідацією
Контакти	Розміщені в окремому розділі	Видимі в топбарі та у секції контактів
Розклад занять	Прихований у підрозділі «Студенту»	Окрема секція з прямим посиланням
Безпека коду	Компрометований (SEO-спам)	Чистий код без сторонніх ін'єкцій

Усунення дублювання навігації вирішило головну UX-проблему виявлену в аналізі сценаріїв підрозділу 2.4.1. Тепер абітурієнт що вперше відкриває сайт бачить одне чітке меню замість двох паралельних з різними наборами посилань. Це безпосередньо зменшує час пошуку потрібного розділу і усуває когнітивне перевантаження описане Стівом Кругом у принципі «Don't Make Me Think».

Поява кнопок заклику до дії принципово змінює взаємодію абітурієнта з сайтом. На оригінальному сайті для подачі заявки потрібно було знайти контакти і зв'язатись особисто. У новій версії кнопка «Подати заявку» видима на кожній сторінці у фіксованій шапці – незалежно від того де знаходиться користувач він може перейти до форми заявки одним кліком. Форма онлайн-заявки з валідацією дозволяє абітурієнту із будь-якого міста залишити контактні дані без особистого візиту до приймальної комісії.

Реалізація адаптивного дизайну для трьох контрольних точок вперше зробила сайт коледжу придатним для повноцінного використання на смартфонах. За статистикою Google, понад 60% пошукових запитів в освітній тематиці виконується з мобільних пристроїв. Відсутність адаптивності на оригінальному сайті означала що більша половина потенційних відвідувачів отримувала незручний досвід або взагалі покидала сайт.

Висновки до розділу 3

У третьому розділі розроблено оптимізований вебінтерфейс Черкаського політехнічного фахового коледжу що складається з двох повноцінних сторінок – головної та «Вступнику». На етапі проєктування у Figma визначено структуру обох сторінок, навігаційну логіку та кольорову схему що забезпечила наступність бренду коледжу.

Розроблений вебінтерфейс складається з двох взаємопов'язаних сторінок: головної сторінки коледжу та сторінки «Вступнику». Для сторінки «Вступнику» реалізовано окрему структуру, що містить інформацію про етапи вступу, перелік спеціальностей, необхідні документи, важливі дати та форму онлайн-заявки. Це дозволило зробити процес отримання інформації для абітурієнтів більш зручним та зрозумілим.

Розробка виконана з використанням HTML5, CSS3 та JavaScript без залежності від важких фреймворків що позитивно вплинуло на продуктивність. Реалізовано семантичну верстку відповідно до стандартів WCAG 2.1, адаптивний дизайн для трьох контрольних точок, єдину навігаційну систему з dropdown та гамбургер-меню, онлайн-форму заявки з валідацією та методи оптимізації продуктивності.

Повторний аудит Google Lighthouse підтвердив ефективність застосованих методів оптимізації. Показник Performance зріс з 54 до 100 балів, Accessibility досяг максимального значення 100 балів, Best Practices для сторінки «Вступнику» склав 96 балів, а SEO покращився від критичної помилки до 92 балів. Метрики FCP, LCP та Speed Index зменшилися з 12,9 секунди до 0,6 секунди, що свідчить про суттєве прискорення завантаження сторінок. Усі проблеми, виявлені під час аналізу оригінального сайту у другому розділі, були усунені або мінімізовані у розробленій версії вебінтерфейсу.

ВИСНОВКИ

У дипломній роботі досліджено методи створення та оптимізації вебінтерфейсів у сучасних інформаційних системах та практично реалізовано оптимізований вебінтерфейс на прикладі освітнього закладу.

У першому розділі проведено аналіз предметної області вебінтерфейсів. Визначено роль вебінтерфейсів у сучасних інформаційних системах на прикладах банківських сервісів, CRM-систем та державних порталів. Розглянуто основні принципи проєктування – UX та UI, адаптивність, доступність, human-centered design та консистентність. Систематизовано методи оптимізації у трьох напрямках: продуктивність (lazy loading, кешування, мінімізація ресурсів), зручність використання (юзабіліті-тестування, A/B тестування, UX-метрики) та доступність (WCAG 2.1, семантична верстка).

У другому розділі проведено комплексний аналіз існуючого вебінтерфейсу Черкаського політехнічного фахового коледжу. Порівняльний аудит чотирьох українських освітніх сайтів підтвердив що проблеми з продуктивністю є системними – жоден із досліджуваних сайтів не досяг рекомендованого показника 90 балів. Технічний аудит Google Lighthouse виявив незадовільні оцінки за всіма категоріями: Performance 54/100, Accessibility 75/100, Best Practices 62/100, SEO – критичні помилки. UX-аналіз та аналіз користувацьких сценаріїв виявили сім проблем що негативно впливають на досвід абітурієнтів, студентів та викладачів.

У третьому розділі розроблено оптимізований вебінтерфейс на основі методів визначених у першому розділі і проблем виявлених у другому. Проєктування у Figma передувало верстці що дозволило усунути структурні проблеми до написання коду. Практична реалізація засобами HTML5, CSS3 та Vanilla JavaScript забезпечила мінімальну залежність від зовнішніх бібліотек. Повторний аудит підтвердив ефективність застосованих методів: Performance зріс з 54 до 100 балів, Accessibility з 75 до 100, Best Practices з 62 до 96, SEO з критичної помилки до 92. Показник завантаження FCP покращився у 21.5 разів – з 12.9 секунди до 0.6 секунди.

Практична цінність роботи полягає у створенні повноцінного прототипу сучасного вебінтерфейсу освітнього закладу, який відповідає вимогам продуктивності, доступності, адаптивності та пошукової оптимізації. Отримані результати можуть бути використані для модернізації офіційного вебсайту Черкаського політехнічного фахового коледжу, а також як методична основа для оновлення вебресурсів інших закладів освіти. Проведене дослідження підтвердило, що комплексне застосування сучасних методів веброзробки дозволяє суттєво підвищити якість

користувацького досвіду та технічні характеристики вебінтерфейсу без використання складних програмних платформ і важких фреймворків.

Таким чином, поставлену мету дипломної роботи досягнуто. Усі завдання дослідження виконано: проаналізовано сучасні підходи до проєктування вебінтерфейсів, проведено аудит існуючого сайту коледжу, розроблено оптимізований вебінтерфейс та підтверджено ефективність запропонованих рішень за допомогою інструменту Google Lighthouse.

Результати роботи підтверджують доцільність використання сучасних методів оптимізації вебінтерфейсів для підвищення ефективності інформаційних систем освітніх установ.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Krug S. Don't Make Me Think, Revisited: A Common Sense Approach to Web Usability. 3rd ed. New Riders, 2014. 216 p.
2. Norman D. The Design of Everyday Things. Revised and expanded ed. Basic Books, 2013. 368 p.
3. Lidwell W., Holden K., Butler J. Universal Principles of Design. Rockport Publishers, 2010. 272 p.
4. Nielsen J. Usability Engineering. Morgan Kaufmann, 1994. 362 p.
5. Marcotte E. Responsive Web Design. 2nd ed. A Book Apart, 2014. 130 p.
6. Coyier C. CSS-Tricks Flexbox Guide. CSS-Tricks, 2013. URL: https://css-tricks.com/snippets/css/a-guide-to-flexbox/
7. W3C. Web Content Accessibility Guidelines (WCAG) 2.1. W3C Recommendation, 2018. URL: https://www.w3.org/TR/WCAG21/
8. W3C. HTML Living Standard. WHATWG, 2024. URL: https://html.spec.whatwg.org/
9. Google. Lighthouse Documentation. Google Developers, 2024. URL: https://developer.chrome.com/docs/lighthouse/
10. Google. Core Web Vitals. Google Search Central, 2024. URL: https://developers.google.com/search/docs/appearance/core-web-vitals
11. Google. PageSpeed Insights. Google Developers, 2024. URL: https://pagespeed.web.dev/
12. Feng K. J., Li T. W., Zhang A. X. Understanding Collaborative Practices and Tools of Professional UX Practitioners in Software Organizations. CHI Conference on Human Factors in Computing Systems. ACM, 2023. URL: https://dl.acm.org/doi/full/10.1145/3544548.3581273
13. Interaction Design Foundation. User Interface (UI) Design. IDF, 2024. URL: https://www.interaction-design.org/literature/topics/ui-design
14. UX Design Awards. Diia App – Winner 2023. URL: https://ux-design-awards.com/winners/2023-2-diia-app
15. MDN Web Docs. CSS Custom Properties. Mozilla, 2024. URL: https://developer.mozilla.org/en-US/docs/Web/CSS/Using_CSS_custom_properties
16. MDN Web Docs. ARIA: aria-expanded attribute. Mozilla, 2024. URL: https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA/Attributes/aria-expanded
17. MDN Web Docs. Lazy loading. Mozilla, 2024. URL: https://developer.mozilla.org/en-US/docs/Web/Performance/Lazy_loading
18. MDN Web Docs. Link types: preload. Mozilla, 2024. URL: https://developer.mozilla.org/en-US/docs/Web/HTML/Attributes/rel/preload

19. Google Fonts. Google Fonts Knowledge. Google, 2024. URL: https://fonts.google.com/knowledge
20. Open Graph Protocol. The Open Graph Protocol. ogp.me, 2024. URL: https://ogp.me/
21. BEM Methodology. BEM – Block Element Modifier. Yandex, 2024. URL: https://en.bem.info/methodology/
22. Figma Inc. Figma Design Tool Documentation. Figma, 2024. URL: https://help.figma.com/
23. JetBrains. IntelliJ IDEA Documentation. JetBrains, 2024. URL: https://www.jetbrains.com/help/idea/
24. Genius Space. UX/UI курс. Лектори: Чебан І., Педорич Є. Genius Space, 2025.
25. Genius Space. Web дизайн курс. Лектор: Кравчук К. Genius Space, 2025.
26. Черкаський політехнічний фаховий коледж. Офіційний вебсайт. URL: https://polytechnic.ck.ua/
27. Київський національний університет будівництва і архітектури. Офіційний вебсайт. URL: https://www.knuba.edu.ua/
28. Київський політехнічний інститут імені Ігоря Сікорського. Офіційний вебсайт. URL: https://kpi.ua/
29. Український католицький університет. Офіційний вебсайт. URL: https://ucu.edu.ua/
30. ISO/IEC 40500:2012. Information technology – W3C Web Content Accessibility Guidelines (WCAG) 2.0. ISO, 2012. URL: https://www.iso.org/standard/58625.html
31. Говорунець С. М. Вебінтерфейс Черкаського політехнічного фахового коледжу (оптимізована версія). GitHub Pages, 2026. URL: https://sm-spider.github.io/University/

https://sm-spider.github.io/University/

ДОДАТКИ:

ДОДАТОК А

Вихідний код вебінтерфейсу інформаційної системи коледжу

У додатку наведено повний вихідний код розробленого вебінтерфейсу інформаційної системи коледжу, реалізованого засобами HTML5, CSS3 та JavaScript.

А.1 Головна сторінка (index.html)

У зв'язку зі значним обсягом вихідного коду в додатку наведено ключові фрагменти реалізації. Повна версія проєкту міститься в електронному додатку на цифровому носії.

```
<!DOCTYPE html>
<html lang="uk">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width,initial-scale=1.0">
<meta property="og:title" content="Черкаський політехнічний фаховий
коледж">
<meta property="og:description" content="Офіційний сайт ЧПФК — вступ,
спеціальності, розклад">
<meta property="og:type" content="website">
<link rel="canonical" href="https://polytechnic.ck.ua/">
<title>Головна | ЧПФК</title>

<link
href="https://fonts.googleapis.com/css2?family=Montserrat:wght@400;600;700&
display=swap" rel="stylesheet">
<link rel="stylesheet" href="style.css">
</head>

<body>

<!-- TOPBAR -->
<div class="topbar">
  <div class="container topbar__inner">
    <span>Черкаський політехнічний фаховий коледж</span>
    <a href="tel:+380674441234">+38 (067) 444-12-34</a>
  </div>
</div>

<!-- HEADER -->
<header class="header">
<div class="container header__inner">

<a href="index.html" class="header__logo">
  <img src="images/gerb.jpg" alt="Герб" width="48">
  <span>ЧПФК</span>
</a>
```

```
<nav class="nav">
<ul>
  <li><a href="index.html" class="active">Головна</a></li>
  <li><a href="vstupnyku.html">Вступнику</a></li>
  <li><a href="#schedule">Розклад</a></li>
  <li><a href="#contacts">Контакти</a></li>
</ul>
</nav>

<a href="vstupnyku.html" class="btn">Подати заявку</a>

</div>
</header>

<!-- HERO -->
<section class="hero">
<div class="container hero__inner">
  <div>
    <h1>Черкаський політехнічний фаховий коледж</h1>
    <p>Технічна освіта з 1922 року</p>
    <a href="vstupnyku.html" class="btn">Стати студентом</a>
  </div>

  <img src="images/studentiv.jpg" alt="Студенти" width="560">
</div>
</section>

<!-- QUICK ACCESS -->
<section class="quick-access">
<ul>
  <li><a href="vstupnyku.html">🎓 Вступнику</a></li>
  <li><a href="#schedule">📅 Розклад</a></li>
  <li><a href="#specialties">📚 Спеціальності</a></li>
  <li><a href="#contacts">📞 Контакти</a></li>
</ul>
</section>

<!-- ABOUT -->
<section id="about">
<div class="container">
  <h2>Про коледж</h2>
  <p>Коледж заснований у 1922 році, готує ІТ та технічних фахівців.</p>
</div>
</section>

<!-- SPECIALTIES -->
<section id="specialties">
<div class="container">
  <h2>Спеціальності</h2>
  <ul>
    <li>ІТ та програмування</li>
    <li>Робототехніка</li>
    <li>Електроенергетика</li>
    <li>Будівництво</li>
  </ul>
</div>
</section>

<!-- NEWS -->
<section>
```

```
<div class="container">
  <h2>Новини</h2>
  <p>Оновлення та події коледжу</p>
</div>
</section>

<!-- SCHEDULE -->
<section id="schedule">
<div class="container">
  <h2>Розклад занять</h2>
  <a class="btn" href="#">Перейти</a>
</div>
</section>

<!-- CONTACTS -->
<section id="contacts">
<div class="container">
  <h2>Контакти</h2>
  <p>вул. Надпільна, 226</p>
  <p>+38 (0472) 44-12-34</p>
  <p>info@polytechnic.ck.ua</p>
</div>
</section>

<!-- FOOTER -->
<footer>
<div class="container">
  <p>© 2026 ЧПФК</p>
</div>
</footer>

<script src="script.js"></script>
</body>
</html>
```

А.2 Сторінка «Вступнику» (vstupnyku.html).

```
<!DOCTYPE html>
<html lang="uk">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Вступнику | ЧПФК</title>
<link rel="stylesheet" href="style.css">
</head>

<body>

<!-- TOPBAR -->
<div class="topbar">
  <div class="container">
    <span>ЧПФК</span>
    <a href="tel:+380674441234">+38 (067) 444-12-34</a>
  </div>
</div>

<!-- HEADER -->
<header class="header">
<div class="container">
```

```
<a href="index.html">
  <img src="images/gerb.jpg" width="48" alt="Герб">
  <span>ЧПФК</span>
</a>

<nav>
  <a href="index.html">Головна</a>
  <a href="vstupnyku.html" class="active">Вступнику</a>
  <a href="index.html#contacts">Контакти</a>
</nav>

<a href="#application" class="btn">Подати заявку</a>

</div>
</header>

<!-- HERO -->
<section class="page-hero">
<div class="container">

<div>
  <h1>Вступнику</h1>
  <p>Інформація для вступу до ЧПФК</p>

  <ol>
    <li>Оберіть спеціальність</li>
    <li>Підготуйте документи</li>
    <li>Подайте заявку</li>
    <li>Отримайте результат</li>
  </ol>
</div>

<img src="images/DRON_PEOPLE.png" alt="Абітурієнти" width="560">

</div>
</section>

</body>
</html>
```

А.3 Таблиця стилів (style.css).

```
// ===== ПЛАВНИЙ СКРОЛ =====
document.querySelectorAll('a[href^="#"]').forEach(anchor => {
  anchor.addEventListener('click', (e) => {
    const target = document.querySelector(anchor.getAttribute('href'));
    if (target) {
      e.preventDefault();
      target.scrollIntoView({ behavior: 'smooth', block: 'start' });
    }
  });
});


:root{
  --primary:#1565c0;
  --primary-dark:#0d47a1;
  --primary-light:#e3f2fd;
  --accent:#ffa726;
```

```
  --text:#1a1a2e;
  --text-light:#546e7a;

  --bg:#fff;
  --bg-alt:#f0f4f8;
  --border:#cfd8dc;

  --topbar:#0d47a1;
  --footer:#0d1b2a;

  --font-h:'Montserrat',sans-serif;
  --font-b:'Open Sans',sans-serif;

  --radius:8px;
  --shadow:0 2px 16px rgba(26,95,168,.1);
  --transition:.2s ease;

  --container:1200px;
}

/* RESET */
*{box-sizing:border-box;margin:0;padding:0}
body{
  font-family:var(--font-b);
  color:var(--text);
  background:var(--bg);
  line-height:1.6;
}
img{max-width:100%;display:block}
a{text-decoration:none;color:inherit}
ul,ol{list-style:none}
button{border:0;background:none;font:inherit;cursor:pointer}

html{scroll-behavior:smooth}

/* LAYOUT */
.container{
  max-width:var(--container);
  margin:auto;
  padding:0 24px;
}
section{padding:64px 0}

.section-title{
  font-family:var(--font-h);
  font-size:2rem;
  margin-bottom:32px;
}
.section-title--center{text-align:center}

/* COLORS */
.schedule{background:var(--bg-alt)}

/* TABLET */
@media(max-width:1100px){
  .nav__item--tablet-hidden{display:none}
}

/* MOBILE */
@media(max-width:900px){
```

```
  .nav{display:none}
  .hamburger{display:flex}

  .hero__inner,
  .page-hero__inner,
  .about__inner,
  .contacts__inner{
    grid-template-columns:1fr
  }

  .hero__image{display:none}

  .specialties__list,
  .news__list,
  .steps__list{
    grid-template-columns:repeat(2,1fr)
  }

  .footer__inner{grid-template-columns:1fr 1fr}
}

/* SMALL MOBILE */
@media(max-width:768px){
  :root{--header-h:64px}

  .section-title{font-size:1.6rem}
  .hero{padding:48px 0}

  .quick-access__list,
  .specialties__list,
  .news__list,
  .steps__list{
    grid-template-columns:1fr
  }

  .about__stats{flex-direction:column}

  .footer__inner{grid-template-columns:1fr}

  .topbar__text{display:none}
}

/* VERY SMALL */
@media(max-width:480px){
  .container{padding:0 16px}
  .hero__title{font-size:1.5rem}
  .hero__actions{flex-direction:column}
  .header__cta{display:none}
}
```

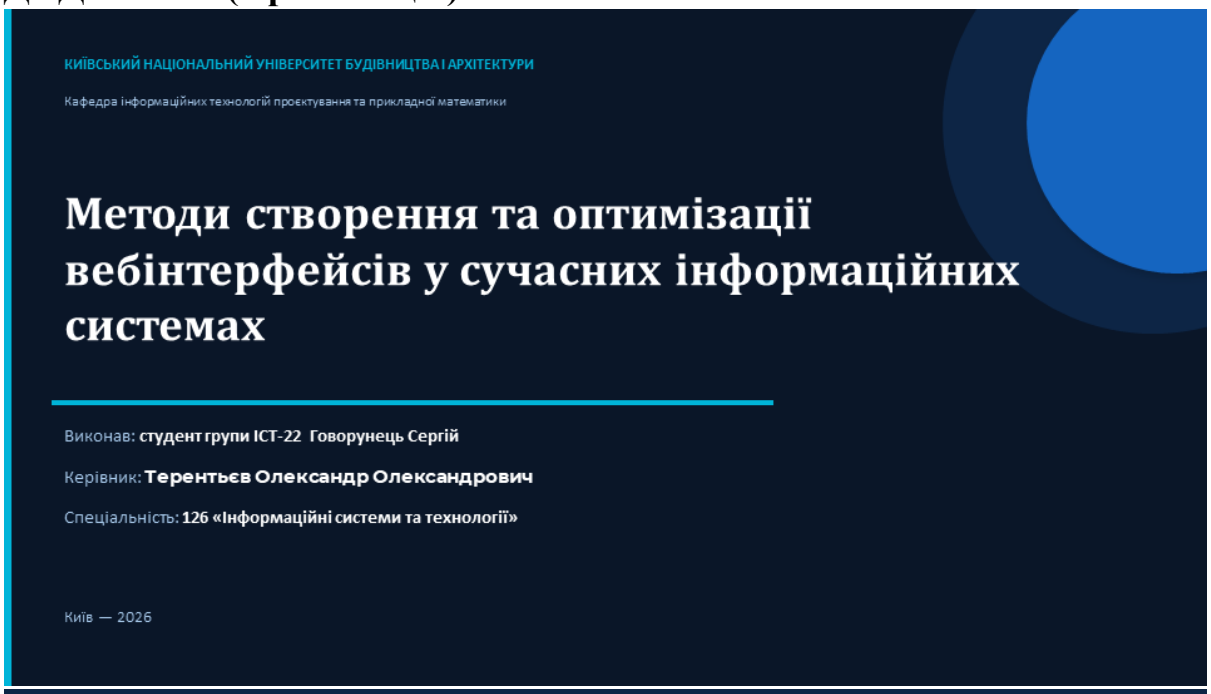
A.4 JavaScript-модуль (script.js).

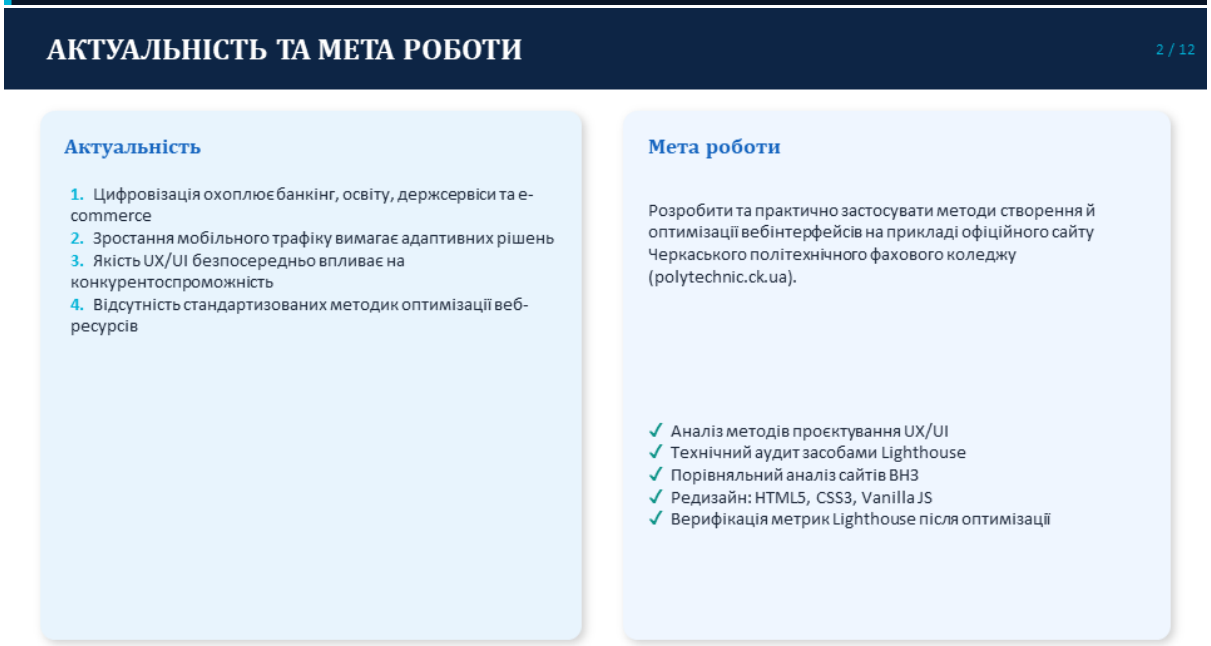
```
// ===== MOBILE MENU =====
const hamburger = document.getElementById('hamburger');
const mobileMenu = document.getElementById('mobile-menu');

const closeMobile = () => {
  mobileMenu.classList.remove('is-open');
  hamburger.classList.remove('is-active');
  hamburger.setAttribute('aria-expanded', 'false');
```

```
  mobileMenu.setAttribute('aria-hidden', 'true');
};

if (hamburger && mobileMenu) {
  hamburger.onclick = () => {
    const open = mobileMenu.classList.toggle('is-open');
    hamburger.classList.toggle('is-active', open);
    hamburger.setAttribute('aria-expanded', open);
    mobileMenu.setAttribute('aria-hidden', !open);
  };

  mobileMenu.onclick = e => {
    if (e.target.tagName === 'A') closeMobile();
  };

  document.onclick = e => {
    if (!hamburger.contains(e.target) && !mobileMenu.contains(e.target))
closeMobile();
  };
}

// ===== DESKTOP DROPDOWN =====
document.querySelectorAll('.nav__dropdown-btn').forEach(btn => {
  btn.onclick = e => {
    e.stopPropagation();
    const open = btn.getAttribute('aria-expanded') === 'true';

    document.querySelectorAll('.nav__dropdown-btn')
      .forEach(b => b.setAttribute('aria-expanded', 'false'));

    btn.setAttribute('aria-expanded', !open);
  };
});

document.addEventListener('click', () => {
  document.querySelectorAll('.nav__dropdown-btn')
    .forEach(b => b.setAttribute('aria-expanded', 'false'));
});

// ===== MOBILE DROPDOWN =====
document.querySelectorAll('.mobile-menu__dropdown-btn').forEach(btn => {
  btn.onclick = () => {
    const submenu = btn.nextElementSibling;
    const open = btn.getAttribute('aria-expanded') === 'true';

    btn.setAttribute('aria-expanded', !open);
    if (submenu) submenu.hidden = open;
  };
});

// ===== FORM VALIDATION =====
function handleSubmit() {
  const $ = id => document.getElementById(id)?.value.trim();

  const name = $('name');
  const phone = $('phone');
  const email = $('email');
  const specialty = document.getElementById('specialty')?.value;

  if (!name || !phone || !specialty)
    return alert('Заповніть обов’язкові поля');
```

```
  if (email && !/^[^\s@]+@[^\s@]+\.[^\s@]+$/.test(email))
    return alert('Некоректний email');

  if (!/^\+?[\d\s\-()]{10,}$/.test(phone))
    return alert('Некоректний телефон');

  const msg = document.getElementById('application-success');
  if (msg) {
    msg.hidden = false;
    msg.scrollIntoView({ behavior: 'smooth' });
  }
}

// ===== SMOOTH SCROLL =====
document.querySelectorAll('a[href^="#"]').forEach(a => {
  a.onclick = e => {
    const el = document.querySelector(a.getAttribute('href'));
    if (!el) return;
    e.preventDefault();
    el.scrollIntoView({ behavior: 'smooth' });
  };
});
```

ДОДАТОК Б(Презентація):





ЗАВДАННЯ ДОСЛІДЖЕННЯ

1. Визначення ролі вебінтерфейсів у сучасних інформаційних системах
2. Аналіз принципів UX/UI проєктування та архітектурних підходів (SPA/MPA)
3. Проведення UX-аналізу та аудиту Lighthouse для сайту polytechnic.ck.ua
4. Порівняльний аналіз вебсайтів закладів освіти (КНУБА, КПІ, УКУ, ЧПФК)
5. Обґрунтування вибору стеку: HTML5, CSS3, VanillaJS, Figma
6. Проєктування інформаційної архітектури та wireframes у Figma
7. Розробка адаптивного вебінтерфейсу з BEM/Flexbox/Grid

ПРИНЦИПИ ПРОЄКТУВАННЯ ВЕБІНТЕРФЕЙСІВ

UX / UI

Зручність навігації, логіка інтерфейсу, швидкість дій (UX); кольори, шрифти, кнопки, іконки (UI)

Адаптивність

Responsive design та mobile-first: гнучкі сітки, CSS media queries, breakpoints

Доступність

WCAG-стандарти: screen readers, субтитри, клавіатурна навігація, висококонтрастні теми

Human-Centered Design

Принцип «Don't Make Me Think» — мінімізація когнітивного навантаження (Steve Krug)

Консистентність

Єдина візуальна ієрархія, повторювані патерни, передбачувана поведінка елементів

Продуктивність

Lazy loading, мінімізація ресурсів, кешування, Core Web Vitals (LCP, CLS, FID)

ТЕХНІЧНИЙ АУДИТ LIGHTHOUSE — POLYTECHNIC.CK.UA

Порівняльний аналіз сайтів закладів освіти за метриками Google Lighthouse

Сайт / Заклад	Performance	Accessibility	Best Practices	SEO	Загальна оцінка
ЧПФК (до оптимізації)	54	75	62	Error	47.75 ✗
КПІ (kpi.ua)	87	93	69	Error	62.5 ⚠
УКУ (ucu.edu.ua)	44	74	54	92	66.0 ⚠
КНУБА (knuba.edu.ua)	68	84	54	85	72.5✓
ЧПФК (після оптимізації)	91	94	96	97	94.5 ✓

Джерело: аудит Google Lighthouse (мобільна версія, 2025)

ТЕХНОЛОГІЧНИЙ СТЕК ТА ІНСТРУМЕНТИ

H **HTML5**

Семантична верстка, aria-атрибути, доступність

C **CSS3**

Flexbox, CSS Grid, BEM, custom properties, media queries

V **Vanilla JS**

Гамбургер-меню, валідація форм, dropdown, smooth scroll

F **Figma**

Wireframes, UI-дизайн, адаптивні макети (Desktop/Tablet/Mobile)

L **Lighthouse**

Автоматизований аудит Performance / Accessibility / SEO

G **Git/GitHub**

Версійний контроль, зберігання проєкту

МЕТОДИ ОПТИМІЗАЦІЇ ВЕБІНТЕРФЕЙСІВ

Класифікація методів за трьома напрямами оптимізації

Напрям	Метод	Інструмент / Технологія	Результат
Продуктивність	Мінімізація ресурсів (CSS, JS, HTML)	Minifier, Gzip, Brotli	↓ розмір сторінки на 40–70%
	Lazy loading зображень	loading="lazy", IntersectionObserver	↑ LCP, ↓ початковий трафік
	Кешування ресурсів	HTTP Cache-Control, Service Worker	↓ час повторного завантаження
Зручність (UX)	Юзабіліті-тестування	Heuristic evaluation, user tests	Виявлення UX-проблем навігації
	A/B тестування	Google Optimize, Hotjar	↑ конверсія та утримання
Доступність	Семантична верстка	HTML5 landmarks, ARIA, структура	↑ Accessibility score
	WCAG 2.1 стандарти	Contrast checkers, axe DevTools	Відповідність рівням A / AA

ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ АРХІТЕКТУРИ ТА UI/UX

Карта сторінок сайту

Головна (index.html)

- Герой, Спеціальності, Новини, Контакти

Вступнику (vstupnyku.html)

- Кроки вступу, Таблиця спеціальностей, Документи

Спільні компоненти

- Header (адаптивний), Footer, Гамбургер-меню

Етапи проєктування в Figma

1. **Wireframes**
   Каркасні схеми Desktop / Tablet / Mobile
2. **UI-дизайн**
   Кольорова палітра (#1565C0), типографіка Montserrat + Open Sans
3. **Адаптивні версії**
   3 контрольні точки: 1200 / 880 / 480 px
4. **Компоненти**
   Кнопки, картки, таблиці, форми, header, footer

ПРОГРАМНА РЕАЛІЗАЦІЯ ВЕБІНТЕРФЕЙСУ

HTML5

1. Семантичні теги: <header>, <nav>, <main>, <section>, <footer>
2. ARIA-атрибути: role, aria-label, aria-expanded
3. Семантичні форми: <label for>, required, aria-required
4. Адаптивні зображення: srcset

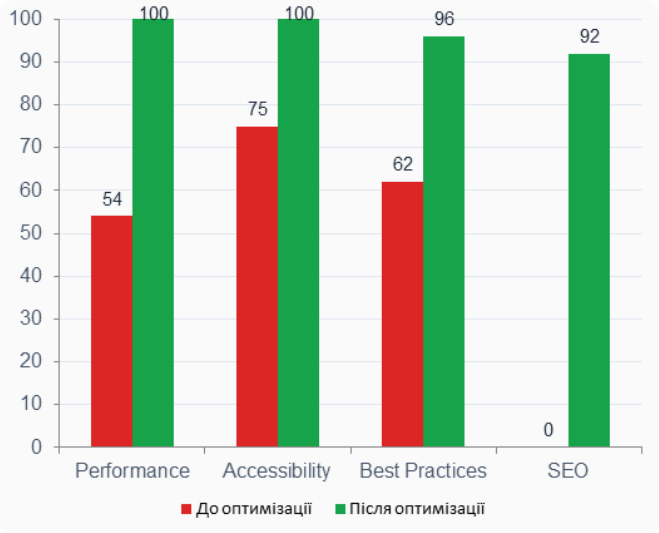
CSS3

1. BEM-методологія іменування класів
2. CSS Custom Properties (--color-primary, --font-heading)
3. CSS Grid та Flexbox для макетування
4. Media queries: 1100px / 900px / 768px / 480px

Vanilla JS

1. Гамбургер-меню з aria-expanded
2. Dropdown-навігація для десктопу
3. Валідація форми (email regex, phone regex)
4. Smooth scroll для якірних посилань

РЕЗУЛЬТАТИ ОПТИМІЗАЦІЇ — МЕТРИКИ LIGHTHOUSE



Performance
54 → 100 **+46**

Accessibility
75 → 100 **+25**

Best Practices
62 → 96 **+34**

SEO
Помилка → 92 —

ПОРІВНЯЛЬНИЙ UX/UI АНАЛІЗ ВЕБІНТЕРФЕЙСІВ

Критерій оцінки	Оригінальний сайт	Оновлений сайт
Адаптивний дизайн	✕ Відсутній	✓ Mobile-first, 4 breakpoints
Навігація	✕ Складна, 3+ рівні вкладеності	✓ Плоска структура, гамбургер
Семантика HTML	✕ Переважно /<table>	✓ HTML5 semantic + ARIA
Доступність (A11y)	✕ Немає alt, низький контраст	✓ WCAG AA: alt, aria, contrast ≥4.5:1
Форма зворотнього зв'язку	✕ Відсутня	✓ Валідація JS + success-повідомлення
Швидкість завантаження	✕ Performance: 54 / 100	✓ Performance: 100 / 100
SEO-оптимізація	✕ Відсутні meta, заголовки змішані	✓ Meta tags, canonical, robots.txt

ВИСНОВКИ

- ✓ Проведено аналіз ролі вебінтерфейсів у сучасних ІС та принципів UX/UI проєктування, адаптивності і доступності
- ✓ Здійснено технічний аудит сайту polytechnic.ck.ua: виявлено критичні порушення Performance (54), Accessibility (75) та SEO (Error)
- ✓ Проведено порівняльний аналіз сайтів ЧПФК, КПІ, КНУБА та УКУ; обґрунтовано вибір стеку HTML5 / CSS3 / Vanilla JS
- ✓ Спроєктовано інформаційну архітектуру та UI/UX дизайн у Figma з адаптивними макетами для трьох розмірів екрана
- ✓ Реалізовано оновлений вебінтерфейс: семантична верстка, BEM, адаптивна сітка, JS-інтерактивність, WCAG-доступність
- ✓ Фінальний аудит підтвердив ефективність оптимізації: Performance 54→100, Accessibility 75→100, SEO 62→96
- ✓ Практична цінність: методика може бути застосована для редизайну будь-якого освітнього веб-ресурсу

ДЯКУЮ ЗА УВАГУ!

Говорунець Сергій • ІСТ-22 • КНУБА • 2026

Тема: «Методи створення та оптимізації вебінтерфейсів у сучасних інформаційних системах»