УКРАЇНСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ЗАЛІЗНИЧНОГО ТРАНСПОРТУ

ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КЕРУЮЧИХ СИСТЕМ
ТА ТЕХНОЛОГІЙ

Кафедра інформаційних технологій

МЕТОДИЧНІ ВКАЗІВКИ

для виконання лабораторних робіт
з освітнього компонента

«*WEB-ТЕХНОЛОГІЇ НАВЧАННЯ*»

Частина 1

Харків 2026

Методичні вказівки розглянуто і рекомендовано до друку на засіданні кафедри інформаційних технологій 21 травня 2026 р., протокол № 11.

Рекомендовано для здобувачів вищої освіти першого (бакалаврського) рівня спеціальності В11 «Філологія» усіх форм здобуття освіти.

Укладач

доц. О. І. Іванюк

Рецензент

доц. Н. А. Корольова

ЗМІСТ

ВСТУП

Методичні вказівки для виконання лабораторних робіт розроблено з метою надання здобувачам теоретичних знань і практичних навичок, необхідних для опанування основ веброзроблення та застосування сучасних засобів створення і оформлення вебсторінок. У межах лабораторних робіт передбачено вивчення мови гіпертекстової розмітки HTML, каскадних таблиць стилів CSS, фреймворку Bootstrap як засобу швидкого створення структурованих, зручних і адаптивних вебінтерфейсів.

Лабораторні роботи спрямовані на формування у здобувачів умінь самостійно створювати HTML-документи, організовувати структуру вебсторінок, додавати текстові, графічні, табличні, мультимедійні елементи і форми введення даних, реалізовувати навігацію між сторінками, стилізувати вебвміст засобами CSS, використовувати селектори, класи, ідентифікатори, псевдокласи і псевдоелементи, застосовувати готові компоненти і службові класи Bootstrap для оформлення інтерфейсу.

Виконання лабораторних робіт передбачає послідовне ускладнення навчального матеріалу: від створення базової HTML-структури вебсторінки до її стилізації засобами CSS і подальшого використання Bootstrap як альтернативного підходу для оформлення. Поетапне засвоєння здобувачами основних принципів побудови сучасних вебінтерфейсів сприяє розвитку навичок самостійної практичної роботи і формує цілісне уявлення про взаємозв’язок структури, оформлення і подання вебвмісту.

Методичні вказівки містять теоретичні відомості, покрокові інструкції з виконання лабораторних робіт, приклади програмного коду, контрольні запитання та індивідуальні завдання.

Під час розроблення методичних вказівок було використано інструменти штучного інтелекту OpenAI, зокрема ChatGPT [1] і Codex [2], а також Claude [3] компанії Anthropic. ChatGPT застосовано для опрацювання

формулювань, уточнення структури навчального матеріалу, редагування пояснювальних фрагментів, уніфікації стилю викладу та перевірки логічної послідовності інструкцій. Codex і Claude використано для опрацювання технічних прикладів, перевірки узгодженості фрагментів HTML-, CSS-, Bootstrap-коду. Використання зазначених інструментів мало допоміжний характер і було спрямоване на підвищення якості, структурованості та зрозумілості навчально-методичного матеріалу.

Лабораторна робота 1
ОСНОВИ HTML

Мета роботи – формування навичок створення та редагування вебсторінок за допомогою мови розмітки HTML, ознайомлення з основними HTML-елементами та їх використанням для побудови структурованого і змістовного контенту.

1.1 Теоретичні відомості

HTML (HyperText Markup Language) є стандартною мовою розмітки, яку використовують для створення вебсторінок. За допомогою HTML формують структуру інтернет-ресурсу, визначають склад сторінки, логічні зв'язки між її частинами та спосіб подання вмісту в браузері. HTML слугує для опису структури документа і не належить до мов програмування в традиційному розумінні, оскільки не містить алгоритмічних конструкцій і не виконує обчислень.

Основою HTML є теги, які записують у кутових дужках. Більшість тегів мають відкривальну і закривальну форму та окреслюють межі певного елемента, наприклад заголовка, абзацу, посилання чи таблиці. Структура HTML-документа зазвичай починається з оголошення типу документа `<!DOCTYPE html>`, після чого розміщують кореневий елемент `<html>`, службовий блок `<head>` і основний блок вмісту `<body>`. У блоці `<head>` містяться службові дані про сторінку, у блоці `<body>` розташовують увесь вміст, який буде відображений у вікні браузера.

HTML дає змогу описувати окремі елементи і логічну будову сторінки. Для цього використовують семантичні теги, зокрема `<header>`, `<nav>`, `<main>`, `<section>` і `<footer>`, що допомагають впорядкувати документ, зробити його зрозумілішим для людини, браузера та допоміжних технологій.

Атрибути HTML задають додаткові властивості елементів і записані у відкривальному тегу. Наприклад, атрибут `lang` визначає мову документа, `href` задає адресу посилання, `src` указує шлях до файлу, `alt` містить альтернативний текст для зображення, `id` і `class` допомагають ідентифікувати елементи. Атрибут `href` використовують як для переходів на зовнішні ресурси, так і посилань на інші сторінки або розділи в межах одного проєкту, атрибут `src` вказує браузеру, який саме файл зображення, відео або аудіо потрібно відкрити. Атрибут `id` застосовують для унікального позначення конкретного елемента сторінки, `class` використовують для позначення групи однотипних елементів.

Важливими складовими HTML є різні групи елементів, кожна з яких виконує власну роль. Заголовки `<h1>`-`<h6>` задають ієрархію змісту, абзаци `<p>` використовують для основного тексту, списки `<ul>` та `<ol>` допомагають впорядковувати відомості, посилання `<a>` забезпечують навігацію, таблиці дають змогу подати структуровані дані, форми призначені для введення інформації користувачем. Окремо HTML підтримує мультимедійні елементи: зображення, аудіо та відео, тому одна сторінка може поєднувати текстовий, табличний і мультимедійний вміст.

У структурі HTML важливу роль, окрім семантичних тегів, відіграють універсальні контейнери `<div>`. Цей контейнер є універсальним блочним елементом, який використовують для логічного групування вмісту. Він не належить до семантичних тегів, однак є корисним для подальшого структурування сторінки та її оформлення засобами CSS.

Для запису спеціальних символів у HTML застосовують сутності. Наприклад, `&copy;` відображає символ авторського права, `&lt;`, `&gt;` дають змогу подати кутові дужки як текст, `&amp;` використовують для символу амперсанда. Такі позначення потрібні в тих випадках, коли символ має службове значення або має бути відображений у спеціальній формі.

Додаткові відомості щодо HTML, структури документа та основних елементів наведено в джерелах [4, 5].

1.2 Засоби виконання

Для виконання лабораторної роботи необхідні:

– персональний комп’ютер або ноутбук з операційною системою Windows, macOS або Linux;
– браузер актуальної версії (Google Chrome, Mozilla Firefox або Microsoft Edge);
– редактор Visual Studio Code; за потреби – розширення `Auto Close Tag`;
– онлайн-сервіс W3C Markup Validation Service [7].

1.3 Основи використання HTML

1 Завантаження і встановлення Visual Studio Code

Для створення HTML-документів зручно використовувати редактор Visual Studio Code. Він підтримує підсвічування синтаксису, розширення для веброзроблення та зручне редагування коду.

1 Відкрийте браузер і перейдіть на офіційний сайт Visual Studio Code [6].
2 Завантажте інсталяційний файл для відповідної операційної системи. На сторінці завантаження виберіть версію для Windows, macOS або Linux.
3 Запустіть інсталяційний файл і встановіть програму, дотримуючись стандартних кроків інсталятора. Підтвердьте параметри встановлення та дочекайтеся завершення процесу.
4 Після завершення встановлення відкрийте Visual Studio Code. Переконайтеся, що стартове вікно редактора запускається коректно.

2 Налаштування Visual Studio Code для роботи з HTML

Перед початком роботи необхідно налаштувати редактор так, щоб він полегшував введення HTML-коду та прискорював створення базової структури документа.

1 У вікні Visual Studio Code відкрийте вкладку `Extensions`. Скористайтеся піктограмою на лівій бічній панелі або натисніть комбінацію клавіш `Ctrl + Shift + X`.

2 У полі пошуку вкладки `Extensions` знайдіть і встановіть розширення `Auto Close Tag` (рисунок 1.1). Після встановлення редактор автоматично підставлятиме закривальні теги, що допомагає уникати помилок під час введення парних тегів.

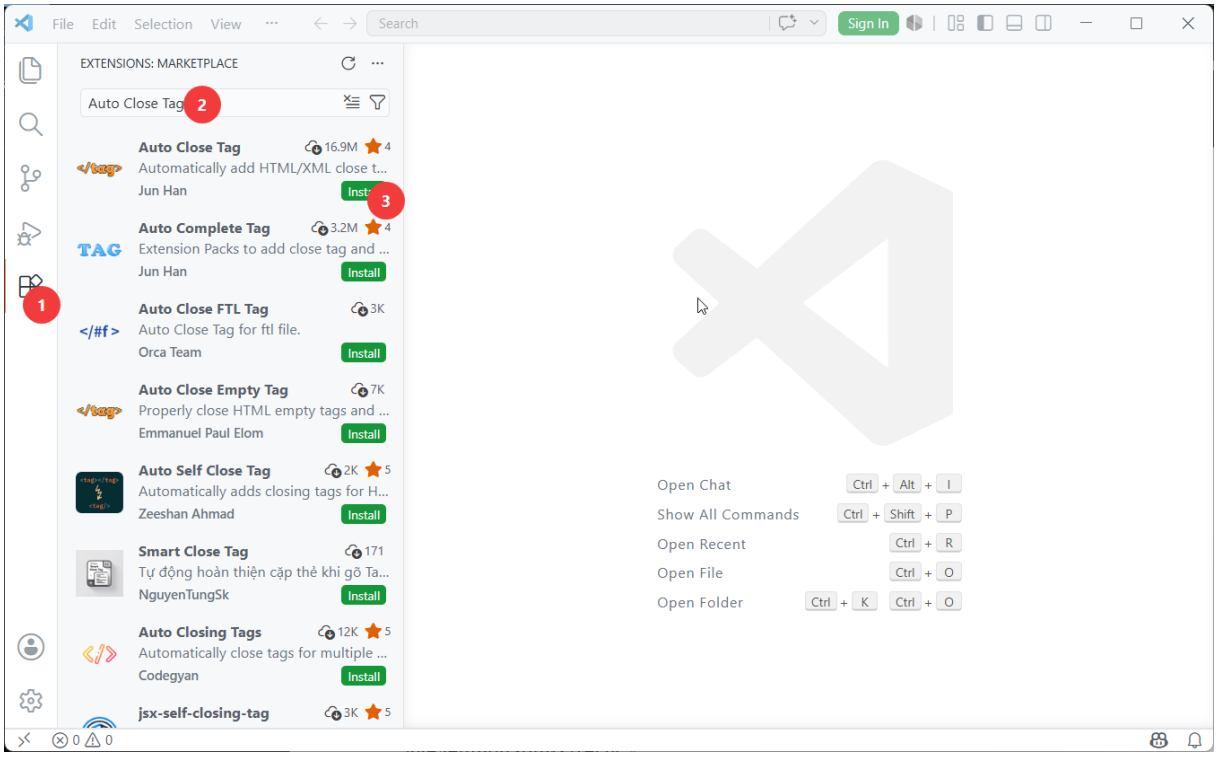

Рисунок 1.1 – Встановлення розширення `Auto Close Tag`

3 Створіть на комп’ютері окрему папку для виконання лабораторної роботи. Наприклад, можна створити папку з назвою `lab_1`, щоб усі матеріали першої лабораторної роботи зберігалися разом.

4 Відкрийте створену папку в редакторі через команду `File > Open Folder`. Після відкриття назва папки має з'явитися в лівій частині вікна Visual Studio Code.

3 Створення та збереження HTML-файлу

У цьому кроці створимо основний файл вебсторінки, у якому далі буде послідовно сформований HTML-код.

1 Створіть новий файл у відкритій робочій папці. Натисніть `File > New File` або використовуйте комбінацію клавіш `Ctrl + N` (`Cmd + N` на macOS).

2 Збережіть файл із назвою `index.html`. Виконайте команду `File > Save` або натисніть `Ctrl + S` (`Cmd + S` на macOS), потім введіть назву `index.html`.

Назву `index.html` традиційно використовують для головної сторінки вебпроєкту. Цей файл браузер і вебсервер сприймають як початкову сторінку, тому необхідно дотримуватися загальноприйнятого іменування.

4 Формування базової структури HTML-документа

Кожна HTML-сторінка повинна мати базову службову структуру: оголошення типу документа, кореневий елемент `<html>`, блок службової інформації `<head>` і блок вмісту сторінки `<body>`.

1 Відкрийте файл `index.html` у редакторі. Натисніть на назві файлу, щоб відкрити вкладку редагування.

2 Скористайтеся вбудованим скороченням `Emmet`: у порожньому файлі `index.html` введіть символ `!` і натисніть `Tab` або `Enter`, щоб редактор автоматично вставив базовий шаблон HTML-сторінки.

3 Якщо автоматичний шаблон не з'являється, вставте в порожній файл `index.html` такий код вручну.

```
<!DOCTYPE html>
<html lang="uk">
```

```
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Навчальна вебсторінка студента-
філолога</title>
</head>
<body>

</body>
</html>
```

4 Збережіть файл `index.html`. Після збереження документ уже матиме коректну базову структуру, хоча вміст сторінки в області `<body>` поки що залишатиметься порожнім.

У наведеному фрагменті `<!DOCTYPE html>` повідомляє браузеру про використання HTML5, атрибут `lang="uk"` задає мову документа, тег `<meta charset="UTF-8">` забезпечує коректне відображення українських символів, тег `<title>` визначає назву сторінки на вкладці браузера.

5 Створення семантичної структури вебсторінки

Щоб сторінка мала логічну будову, потрібно одразу створити її основні змістові блоки. Надалі це дасть змогу поступово наповнювати документ вмістом, не перебудовуючи його повністю.

1 У файлі `index.html` додайте між тегами `<body>` та `</body>` основні структурні блоки сторінки.

```
<header></header>
<main></main>
<footer></footer>
```

Елемент `<header>` використовують для верхньої частини сторінки, `<main>` містить основний вміст, `<footer>` призначений для нижньої частини документа.

2 Збережіть файл.

6 Додавання заголовка сторінки і вступного тексту

Після створення каркаса сторінки необхідно додати перші змістові елементи: заголовок, вступний блок і короткий текст у нижній частині сторінки.

1 Додайте між тегами <header> і </header> головний заголовок сторінки.

```
<h1>Навчальна вебсторінка студента-філолога</h1>
```

2 Додайте між тегами <main> і </main> вступну секцію, яка міститиме заголовок розділу і два абзаци пояснювального тексту.

```
<section id="intro">
    <h2>Вступ</h2>
    <p>Ця вебсторінка містить навчальні матеріали для
студентів спеціальності "Філологія".</p>
    <p>На сторінці зібрано приклади з мовознавства,
літературознавства та перекладознавства.</p>
</section>
```

3 Додайте між тегами <footer> і </footer> короткий підпис.

```
<p>Навчальний приклад для студентів-філологів</p>
```

4 Після додавання елементів збережіть зміни та перевірте в браузері, що з’явилися заголовок, вступний блок і підпис у нижній частині сторінки (рисунок 1.2).

У цьому кроці використано тег <h1> для головного заголовка сторінки, тег <h2> для заголовка розділу, тег <p> для абзаців тексту і тег <section> для логічного виділення змістового блоку.

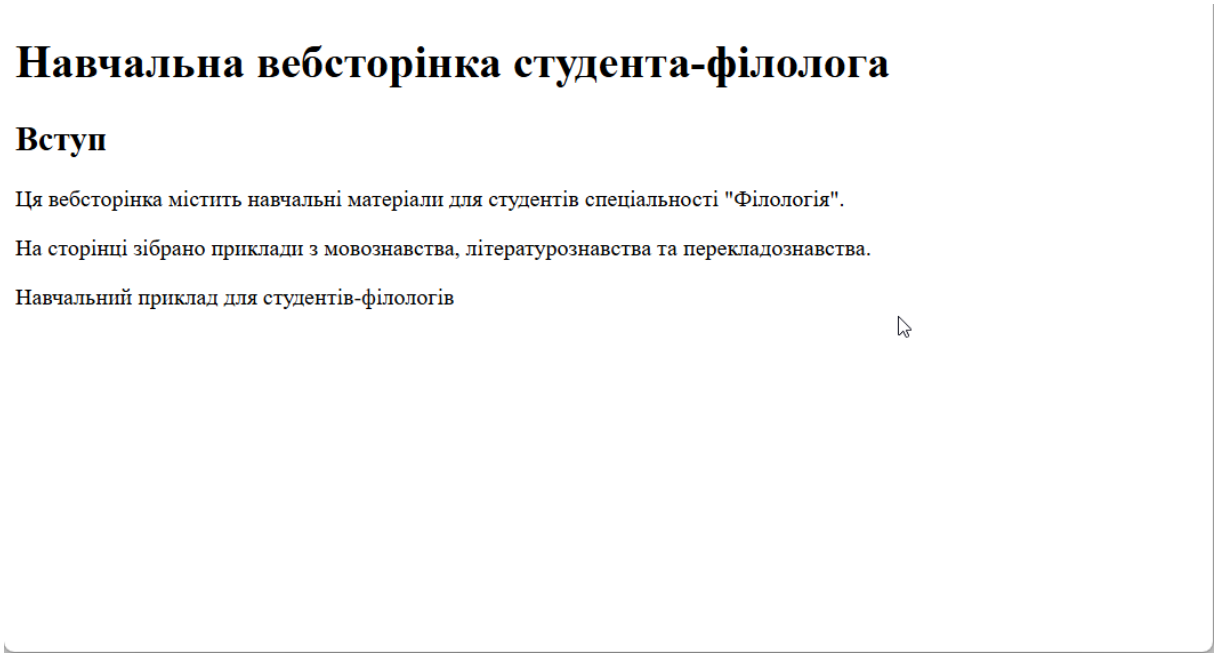

Рисунок 1.2 – Результат додавання заголовка сторінки і вступного тексту

7 Формування основних структурних розділів вебсторінки

Щоб надалі зручно додавати різні елементи HTML, потрібно заздалегідь підготувати окремі секції для списків, зображень, таблиці, форми і мультимедійних матеріалів.

1 У блоці `<main>` після закривального тегу `</section>` вступної секції `intro`, але до закривального тегу `</main>`, додайте такі порожні тематичні секції:

```
<section id="lists">
    <h2>Списки</h2>
</section>

<section id="gallery">
    <h2>Зображення</h2>
</section>

<section id="data-table">
    <h2>Таблиця</h2>
</section>

<section id="feedback">
    <h2>Форма зворотного зв’язку</h2>
</section>
```

```
<section id="media">
    <h2>Мультимедіа</h2>
</section>
```

Кожна секція має атрибут `id`, який пізніше буде використано для створення внутрішньої навігації на сторінці.

2 Збережіть файл і перегляньте сторінку в браузері (рисунок 1.3).

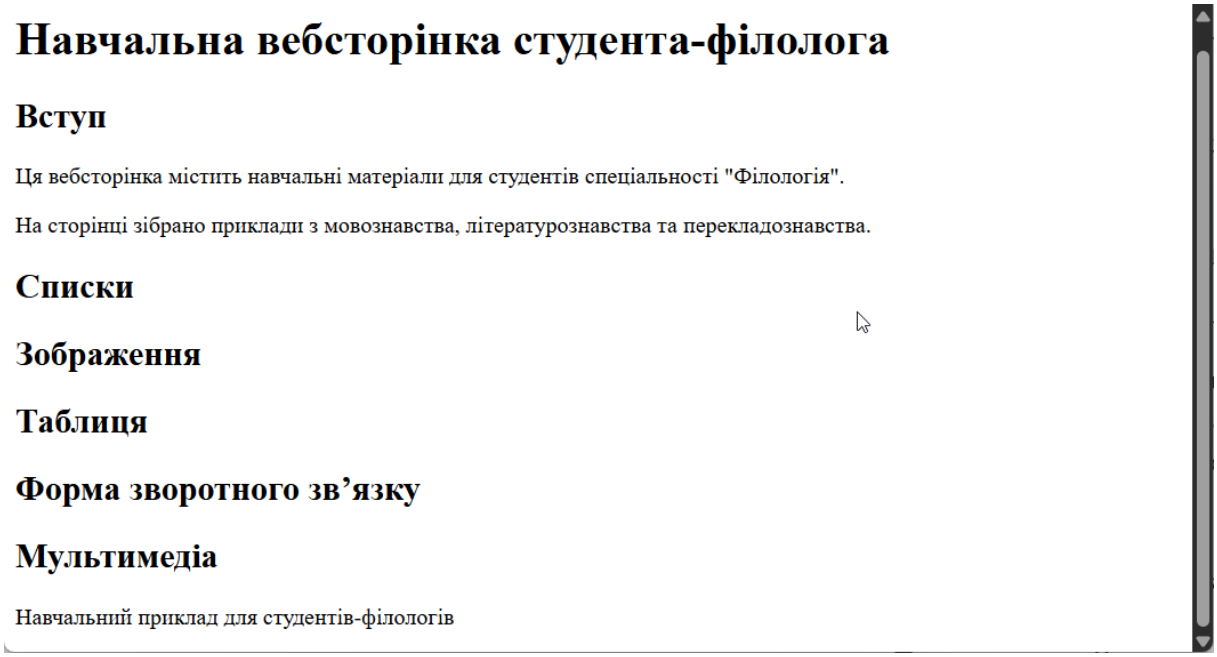

Рисунок 1.3 – Результат створення структурних розділів

Зараз сторінка ще не містить наповнення в кожному розділі, проте вже формується її логічний каркас, що спрощує подальшу роботу, оскільки кожен наступний елемент буде вставлений у вже підготовлене місце.

8 Створення маркованих і нумерованих списків

Списки є одним із базових способів подання інформації на вебсторінці. HTML підтримує як марковані, так і нумеровані списки.

1 Додайте в секцію `lists`, одразу після заголовка `<h2>Списки</h2>`, приклад маркованого списку.

```
<h3>Неупорядкований список</h3>
```

```
<ul>
    <li>Мовознавство</li>
    <li>Літературознавство</li>
    <li>Перекладознавство</li>
</ul>
```

2 Додайте одразу після маркованого списку приклад нумерованого списку.

```
<h3>Упорядкований список</h3>
<ol>
    <li>Прочитати уривок твору</li>
    <li>Визначити мовні особливості</li>
    <li>Підготувати короткий коментар</li>
</ol>
```

Тег <ul> створює маркований список, тег <ol> створює нумерований список, тег <li> задає окремий пункт списку.

3 Збережіть файл і перегляньте сторінку в браузері, щоб перевірити відображення обох типів списків (рисунок 1.4).

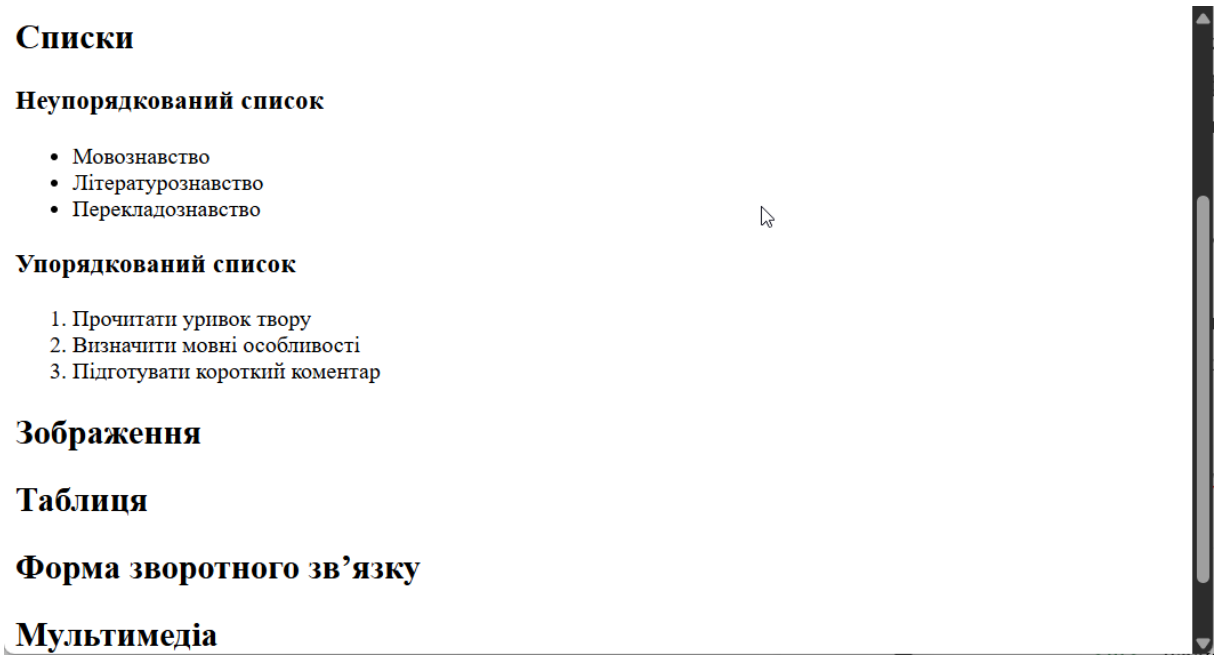

Рисунок 1.4 – Результат додавання списків

9 Створення гіперпосилань у HTML-документі

Гіперпосилання є однією з важливих можливостей HTML, оскільки вони дають змогу переходити до інших частин тієї самої сторінки або відкривати зовнішні ресурси.

1 Додайте в секцію `intro` після двох наявних абзаців внутрішнє посилання на секцію з формою.

```
<p><a href="#feedback">Перейти до форми зворотного зв’язку</a></p>
```

2 Додайте одразу після внутрішнього посилання зовнішнє посилання на інший вебресурс.

```
<p><a href="https://example.com" target="_blank">Відкрити електронний ресурс</a></p>
```

Атрибут `href` у тегу `<a>` задає адресу переходу. Якщо в `href` записано `#feedback`, браузер переходить до елемента з ідентифікатором `feedback` у межах цієї самої сторінки. Якщо в `href` записано повну вебадресу, наприклад `https://example.com`, посилання відкриває зовнішній ресурс. Атрибут `target="_blank"` відкриває зовнішню сторінку в новій вкладці браузера.

3 Збережіть файл і перевірте результат у браузері. Внутрішнє посилання має переводити до блоку форми в межах тієї самої сторінки, зовнішнє має відкривати інший ресурс (рисунок 1.5).

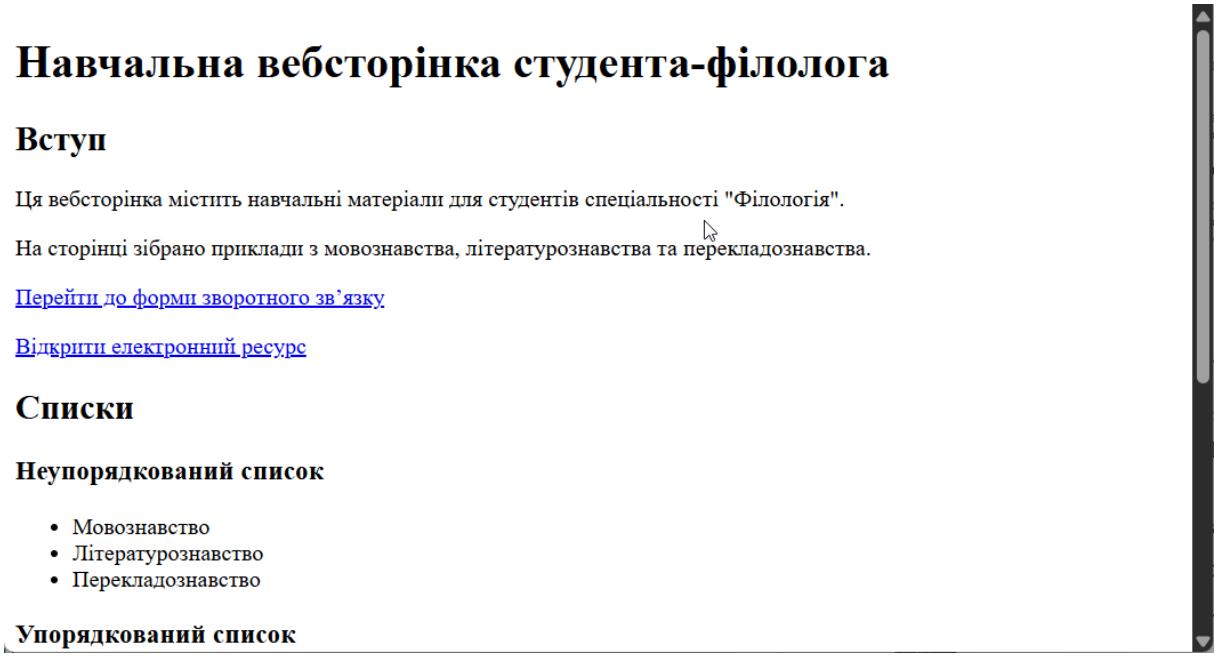

Рисунок 1.5 – Результат додавання гіперпосилань

10 Додавання графічних елементів на вебсторінку

HTML дає змогу вставляти на сторінку графічні зображення. Для цього потрібно підготувати відповідний графічний файл і правильно вказати шлях до нього в коді.

1 Підготуйте файл зображення і помістіть його в ту саму папку, що і файл `index.html`. Назвіть файл `image.jpg`, щоб назва в коді збігалася з назвою файлу в папці.

2 Додайте в секцію `gallery`, одразу після заголовка `<h2>Зображення</h2>` фрагмент, наведений нижче.

```
<img src="image.jpg" alt="Навчальна ілюстрація з
філології" width="300">
<p>Приклад ілюстрації до мовного або літературного
матеріалу.</p>
```

Атрибут `src` містить ім'я графічного файлу, атрибут `alt` задає альтернативний текст, атрибут `width` визначає ширину зображення.

3 Збережіть файл і перевірте в браузері, що зображення відображено коректно (рисунок 1.6).

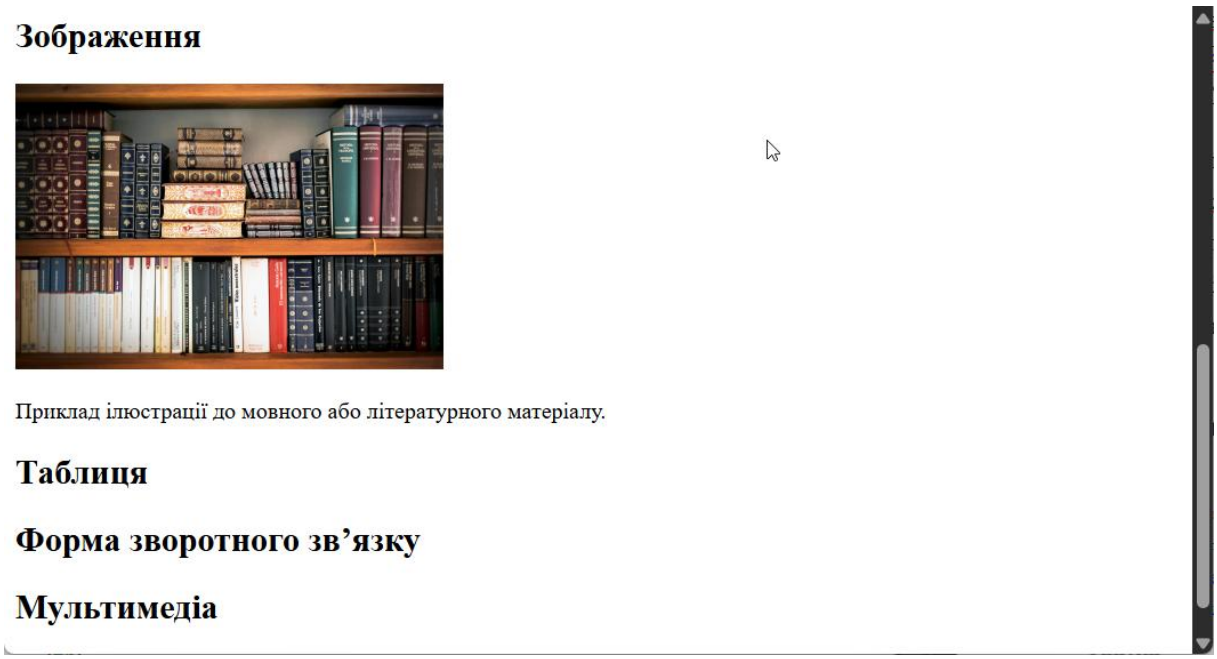

Рисунок 1.6 – Результат додавання зображення

11 Створення таблиці для подання структурованих даних

Таблиці використовують для впорядкованого подання даних у рядках і стовпцях. Створимо просту таблицю із заголовковим рядком і кількома рядками даних.

1 Додайте в секцію `data-table`, нижче заголовка `<h2>Таблиця</h2>`, код, наведений нижче.

```
<table border="1" cellpadding="8">
    <tr>
        <th>Розділ</th>
        <th>Приклад навчальної теми</th>
    </tr>
    <tr>
        <td>Мовознавство</td>
        <td>Фразеологізми в сучасній українській
мові</td>
    </tr>
    <tr>
        <td>Літературознавство</td>
        <td>Образ автора в новелі</td>
    </tr>
    <tr>
        <td>Перекладознавство</td>
```

```
        <td>Передавання культурно маркованої лексики</td>
    </tr>
</table>
```

Тег <tr> створює рядок таблиці, тег <th> визначає заголовкову клітинку, тег <td> задає звичайну клітинку з даними.

У цьому прикладі таблиця показує базову структуру табличних даних: заголовковий рядок, кілька рядків вмісту і поділ на стовпці. Атрибути *border* і cellpadding не є сучасним способом оформлення таблиць. У сучасних вебпроєктах для стилізації таблиць зазвичай використовують CSS. У прикладі вони застосовані, щоб одразу побачити межі клітинок і перевірити структуру таблиці без засобів CSS.

2 Збережіть файл і перегляньте сторінку в браузері, щоб переконатися у правильному відображенні рядків і стовпців (рисунок 1.7).

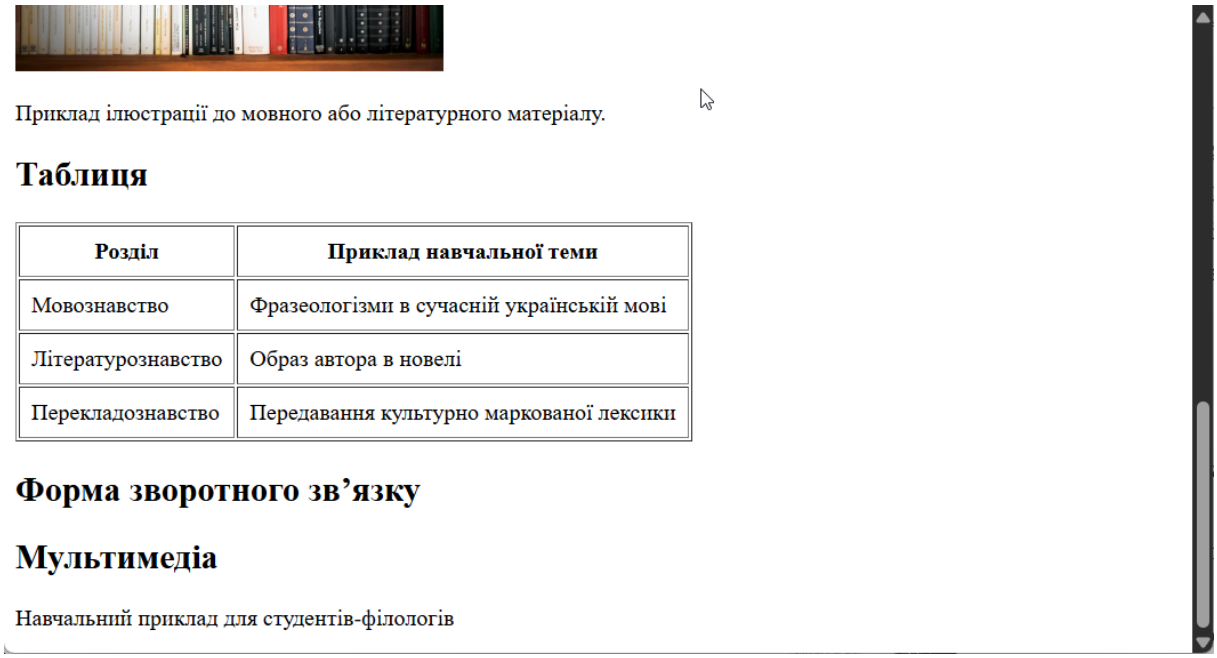

Рисунок 1.7 – Результат додавання таблиці

12 Організація навігації в межах вебсторінки

Коли сторінка вже містить кілька розділів, необхідно додати меню навігації, що дасть змогу швидко переходити до потрібної частини документа.

1 Додайте в елемент <header>, безпосередньо після наявного заголовка <h1>, меню.

```
<nav>
    <ul>
        <li><a href="#intro">Вступ</a></li>
        <li><a href="#lists">Списки</a></li>
        <li><a href="#gallery">Зображення</a></li>
        <li><a href="#data-table">Таблиця</a></li>
        <li><a href="#feedback">Форма</a></li>
        <li><a href="#media">Мультимедіа</a></li>
    </ul>
</nav>
```

Елемент <nav> призначений для навігаційних посилань. Усі пункти меню ведуть до відповідних секцій сторінки за їхніми ідентифікаторами.

2 Збережіть файл і перевірте в браузері, що переходи до відповідних розділів сторінки працюють коректно (рисунок 1.8).

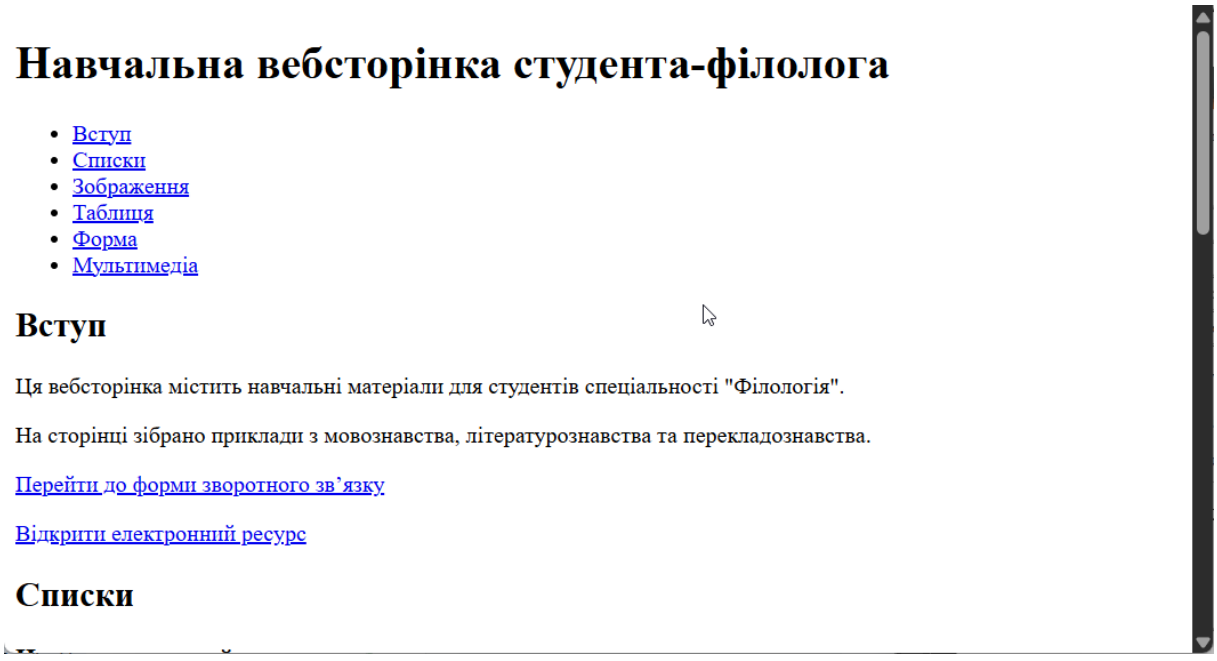

Рисунок 1.8 – Результат додавання навігаційного меню

13 Створення форми введення даних

Форми призначені для введення та надсилання даних користувача. Спочатку необхідно створити базову форму з текстовими полями, полем електронної пошти, текстовою областю та кнопкою надсилання.

1 Додайте в секцію `feedback`, одразу після заголовка `<h2>Форма зворотного зв’язку</h2>`, код, наведений нижче.

```
<form action="#" method="post">
    <label for="name">Ім’я:</label><br>
    <input type="text" id="name" name="name"
required><br><br>

    <label for="email">Електронна пошта:</label><br>
    <input type="email" id="email" name="email"
required><br><br>

    <label for="message">Повідомлення:</label><br>
    <textarea id="message" name="message" rows="4"
cols="40"></textarea><br><br>

    <button type="submit">Надіслати</button>
</form>
```

У цьому прикладі тег `<label>` створює підпис для поля, `<input type="text">` додає текстове поле, `<input type="email">` призначений для введення електронної пошти, `<textarea>` створює багаторядкове поле, `<button>` додає кнопку надсилання, `<br>` переносить наступний елемент на новий рядок. Атрибут `action` указує, куди мають бути надіслані дані форми; значення `#` у цьому прикладі означає, що форма не пов’язана з реальним сервером. Атрибут `method="post"` задає спосіб передавання даних, атрибут `for` у тегу `<label>` пов’язує підпис із відповідним полем, атрибут `required` робить поле обов’язковим для заповнення. Атрибути `rows` і `cols` у `<textarea>` задають початкові розміри текстової області.

2 Збережіть файл і перегляньте результат у браузері, щоб переконатися у правильному відображенні полів введення (рисунок 1.9).

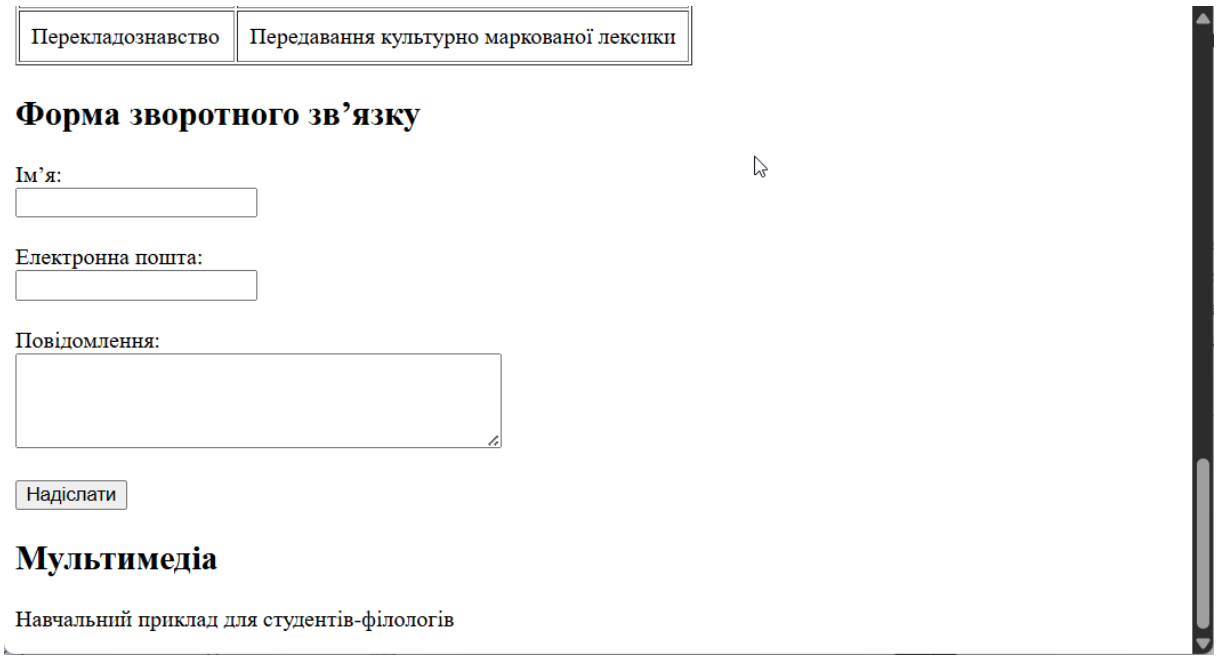

Рисунок 1.9 – Результат додавання форми

14 Розширення форми додатковими елементами керування

Після створення базової форми додамо ще кілька поширених елементів керування: список вибору, перемикачі та прапорець.

1 Додайте всередину вже створеної форми, перед кнопкою `<button type="submit">Надіслати</button>`, список вибору теми звернення.

```
<label for="topic">Напрям запиту:</label><br>
<select id="topic" name="topic">
    <option value="linguistics">Мовознавчий
аналіз</option>
    <option value="literature">Літературознавчий
коментар</option>
    <option value="translation">Переклад тексту</option>
</select><br><br>
```

2 Додайте нижче списку вибору перемикачі для вибору формату відповіді.

```
<p>Оберіть формат відповіді:</p>
<input type="radio" id="reply-email" name="reply_format"
value="email" checked>
<label for="reply-email">Електронна пошта</label><br>
```

```
<input type="radio" id="reply-phone" name="reply_format"
value="phone">
<label for="reply-phone">Телефон</label><br><br>
```

3 Додайте після групи перемикачів прапорець підтвердження згоди.

```
<input type="checkbox" id="agree" name="agree" required>
<label for="agree">Погоджуюся на обробку
даних</label><br><br>
```

Тег `<select>` створює список вибору, тег `<option>` задає його варіанти, елементи з атрибутом `type="radio"` дають змогу вибрати один варіант із кількох, елемент з атрибутом `type="checkbox"` створює прапорець, який можна встановити або зняти незалежно від інших полів; у цьому прикладі такий прапорець використано для підтвердження згоди на обробку даних. У радіокнопок однакове значення атрибута `name` об'єднує їх в одну групу, тому користувач може вибрати лише один варіант, атрибут `checked` задає початково вибраний пункт.

4 Збережіть зміни і перевірте в браузері, що список вибору, перемикачі та прапорець відображені коректно (рисунок 1.10).

Рисунок 1.10 – Результат розширення форми

15 Використання контейнера у структурі вебсторінки

Окрім семантичних тегів, у HTML часто використовують універсальний контейнер , який дає змогу логічно групувати пов'язані елементи сторінки.

1 Додайте в секцію intro, після вже вставлених посилань, контейнер, наведений нижче.

```
<div>
    <h3>Філологічна довідка</h3>
    <p>У цьому блоці можна розмістити коротку довідку про
мовне явище або літературний твір.</p>
</div>
```

 не задає змістового призначення блоку, але є корисним під час подальшого оформлення сторінки засобами CSS. Подібні контейнери часто використовують для об'єднання елементів, які потрібно спільно оформити або розташувати на сторінці.

16 Коментування структури HTML-документа

Коментарі в HTML допомагають краще орієнтуватися в коді, особливо тоді, коли сторінка стає більшою за обсягом. Вони не відображені в браузері, але полегшують читання структури документа.

1 Додайте перед відкривальним тегом <header> коментар, наведений нижче.

```
<!-- Верхня частина сторінки -->
```

2 Додайте перед відкривальним тегом <main> коментар, наведений нижче.

```
<!-- Основний вміст сторінки -->
```

3 Додайте перед відкривальним тегом <footer> коментар, наведений нижче.

```
<!-- Нижня частина сторінки -->
```

4 Замініть наявний вміст елемента <footer>, тобто попередній абзац із текстом, кодом, наведеним нижче.

```
<!-- &copy; — HTML-сутність для символу авторського права
-->
<p>&copy; 2026 Навчальний приклад для студентів-
філологів</p>
```

У цьому фрагменті також використано HTML-сутність ©, яка відображає символ авторського права.

5 Збережіть файл і відкрийте сторінку в браузері. Коментарі не будуть видимі у вікні браузера, проте символ авторського права в нижньому колонтитулі має бути коректно відображений.

17 Додавання мультимедійних елементів

HTML підтримує вставлення відео- та аудіофайлів без використання сторонніх засобів. Для цього достатньо підготувати відповідні файли в робочій папці та правильно вказати їх у коді.

1 Підготуйте відеофайл і помістіть його в ту саму папку, що й index.html. Назвіть відеофайл video.mp4, щоб назва в коді збігалася з назвою файлу в папці.

2 Додайте в секцію media, одразу після заголовка <h2>Мультимедіа</h2>, відео.

```
<h3>Відео</h3>
<video width="320" controls>
    <source src="video.mp4" type="video/mp4">
    Браузер не підтримує відтворення відео.
</video>
```

3 Підготуйте аудіофайл і помістіть його в ту саму папку, що й index.html. Назвіть аудіофайл audio.mp3, щоб назва в коді збігалася з назвою файлу в папці.

4 Додайте безпосередньо під блоком відео блок аудіо.

```
<h3>Аудіо</h3>
<audio controls>
    <source src="audio.mp3" type="audio/mpeg">
    Браузер не підтримує відтворення аудіо.
</audio>
```

Атрибут `controls` додає стандартні елементи керування відтворенням, тому користувач може запускати, зупиняти і переглядати мультимедійний матеріал без додаткових інструментів. У тегу `<source>` атрибут `src` указує шлях до мультимедійного файлу, атрибут `type` допомагає браузеру правильно визначити формат цього файлу.

Відео і аудіо доцільно розміщувати в окремій секції мультимедіа, щоб сторінка зберігала зрозумілу структуру. Якщо файл знаходиться в тій самій папці, що й `index.html`, достатньо вказати лише його ім’я в атрибуті `src`.

5 Збережіть файл і перевірте в браузері, що відео і аудіо можна відтворити (рисунок 1.11).

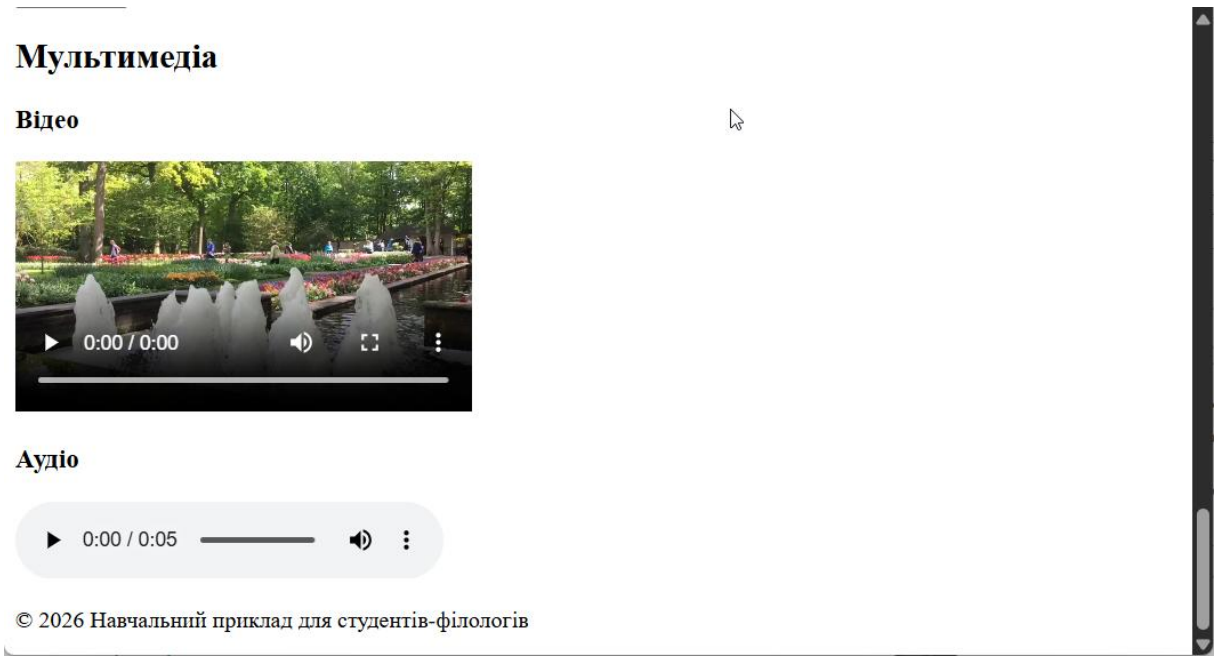

Рисунок 1.11 – Результат додавання мультимедійних елементів

18 Створення додаткової HTML-сторінки

Після опрацювання основних елементів головної сторінки можна створити ще один HTML-документ у межах того самого проєкту.

1 Створіть у тій самій папці новий файл `about.html`. Можна скористатися командою `File > New File` або кнопкою створення нового файлу в бічній панелі провідника Visual Studio Code.

2 Додайте у файл `about.html` код, наведений нижче.

```
<!DOCTYPE html>
<html lang="uk">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Додатковий матеріал</title>
</head>

<body>
    <header>
        <h1>Додатковий матеріал</h1>
        <nav>
            <ul>
                <li><a href="index.html">Головна</a></li>
            </ul>
        </nav>
    </header>

    <main>
        <section>
            <h2>Про навчальний матеріал</h2>
            <p>Ця сторінка може містити короткий опис
мовного явища, твору або перекладацького завдання.</p>
        </section>
    </main>

    <footer>
        <p>&copy; 2026 Навчальний приклад для студентів-
філологів</p>
    </footer>
</body>
```

```
</html>
```

Ця сторінка є окремим HTML-документом, але належить до того самого мініпроєкту, тому з нею можна організувати навігаційний зв’язок. Вебпроєкт може складатися з кількох взаємопов’язаних HTML-файлів.

3 Збережіть файл і відкрийте його в браузері, щоб перевірити коректність структури документа (рисунок 1.12).

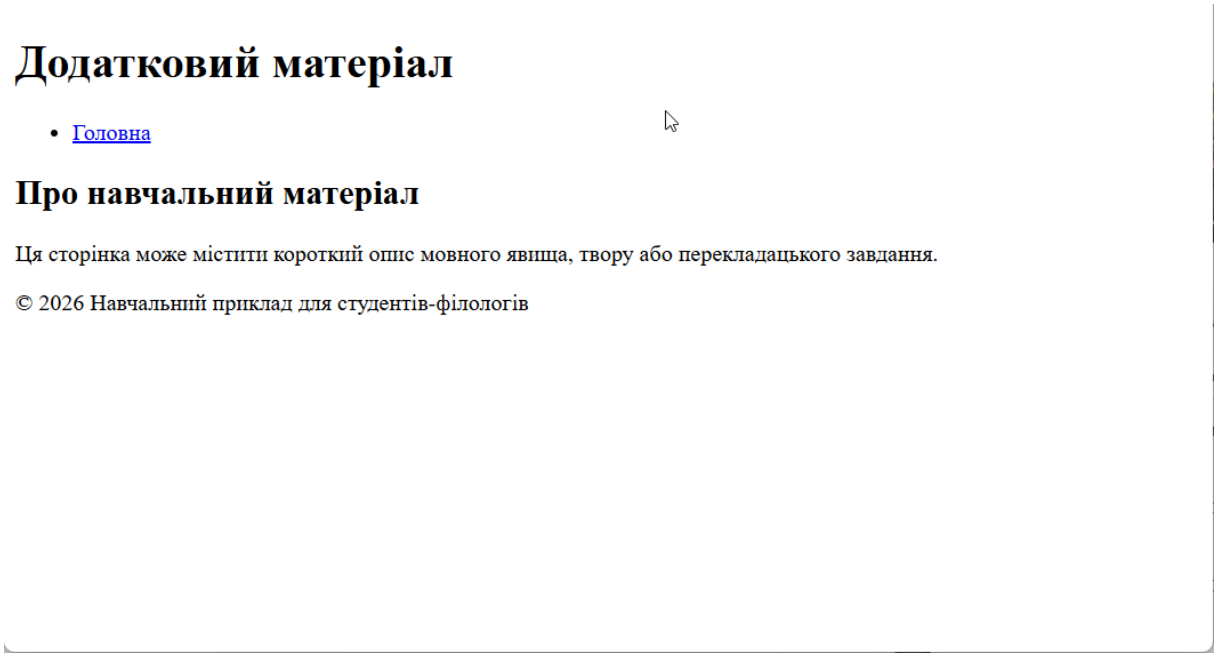

Рисунок 1.12 – Результат створення додаткової сторінки

19 Організація навігації між HTML-сторінками

Після створення другої сторінки потрібно додати посилання між файлами, щоб користувач міг переходити від головної сторінки до додаткової і назад.

1 Поверніться до файлу `index.html`. Якщо в редакторі відкрита вкладка `about.html`, натисніть на вкладці `index.html`, щоб продовжити редагування головної сторінки.

2 Додайте в навігаційне меню, у кінець уже наявного списку посилань, пункт, наведений нижче.

```
<li><a href="about.html">Додатковий матеріал</a></li>
```

3 Додайте в секцію `intro`, після зовнішнього посилання і перед контейнером `<div>`, ще одне текстове посилання.

```
<p><a href="about.html">Перейти до додаткового
матеріалу</a></p>
```

У цьому випадку адреса `about.html` указує на інший HTML-файл, що знаходиться в тій самій папці.

4 Збережіть зміни і перевірте в браузері переходи між сторінками (рисунок 1.13).

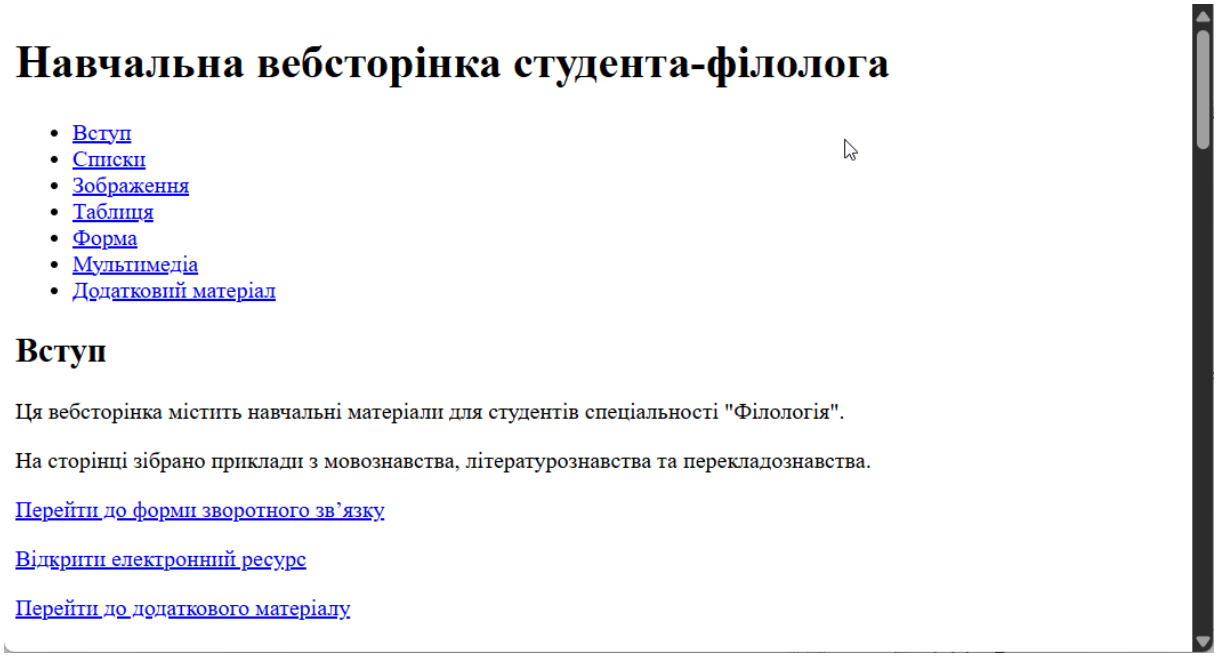

Рисунок 1.13 – Результат додавання міжсторінкової навігації

20 Валідація HTML-коду та підсумкове впорядкування структури документа

Завершальним етапом є перевірка правильності HTML-коду. Це допомагає виявити помилки у вкладеності тегів, неправильні атрибути або пропущені елементи.

1 Відкрийте онлайн-валідатор W3C Markup Validation Service [7].

2 Завантажте до валідатора файл `index.html`. Скористайтеся режимом перевірки файлу.

3 Уважно перегляньте повідомлення валідатора і виправте знайдені помилки. Звертайте увагу на неправильну вкладеність тегів, пропущені закривальні теги та некоректні атрибути.

4 Аналогічно перевірте файл `about.html`. Після завантаження другого файлу переконайтеся, що додаткова сторінка також не містить критичних помилок.

5 Після цього ще раз перегляньте код у Visual Studio Code і переконайтеся, що HTML-документи мають охайне форматування, однакові відступи і правильну логічну структуру.

Валідація є важливою для пошуку синтаксичних помилок і формування звички писати коректний і акуратно структурований HTML-код. Якщо валідатор не показує критичних зауважень, це означає, що документ побудовано правильно і його буде легше розширювати в подальшому (рисунок 1.14).

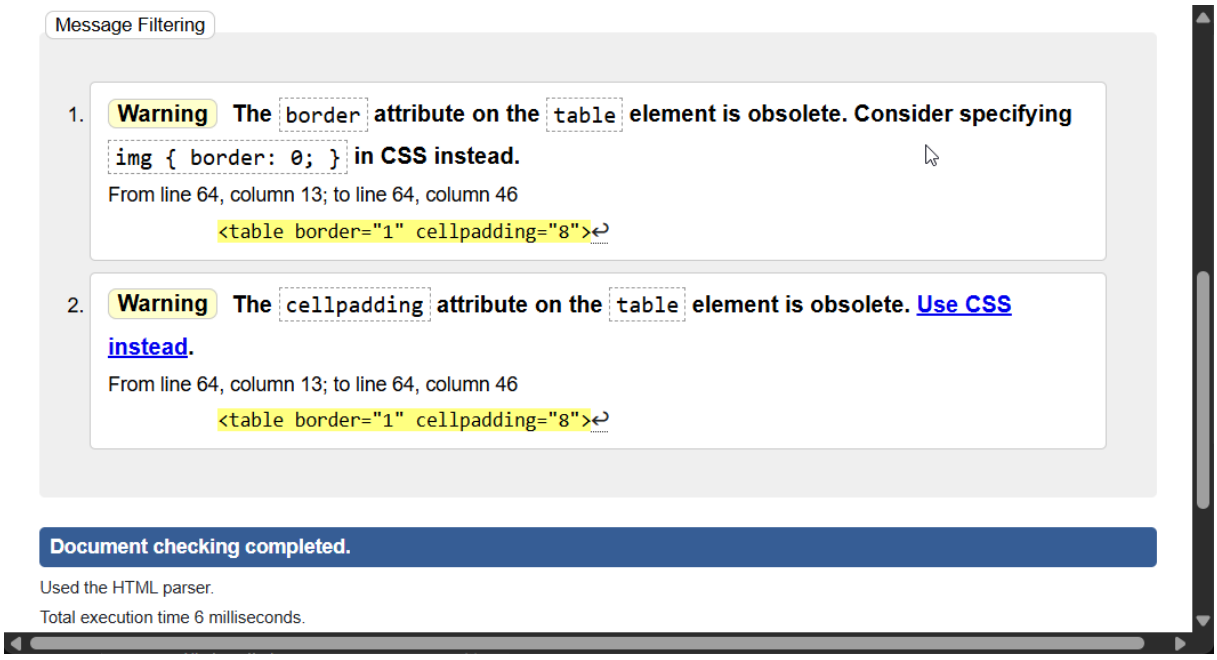

Рисунок 1.14 – Результат перевірки структури документа

1.4 Порядок виконання роботи

Індивідуальне завдання виконують на основі вебсторінки структурного підрозділу організації, яку необхідно вибрати відповідно до варіанта. Ця сторінка є основним джерелом змісту – факти, назви, контакти і напрями діяльності потрібно відтворювати точно, без довільних змін.

1 Відповідно до варіанта необхідно вибрати вебсторінку структурного підрозділу організації та переконатися, що сторінка відкривається коректно.

2 Проаналізувати зміст наданого джерела та виписати відомості, які можна використати в роботі. Необхідно підготувати назву структурного підрозділу організації, короткий опис, напрями діяльності, контактні дані, відомості про співробітників і наявні ілюстративні матеріали.

3 Створити головну сторінку у файлі `index.html` і наповнити її так, щоб були використані всі основні HTML-елементи, розглянуті в роботі. Відомості з наданої сторінки можна впорядковувати у вигляді заголовків, абзаців, списків, таблиці, форми, посилань, зображень і мультимедійних елементів, але без спотворення змісту джерела.

4 Структуру сторінки побудувати за допомогою семантичних тегів `<header>`, `<nav>`, `<main>`, `<section>` і `<footer>`. Внутрішні назви розділів, текст навігації та підписи елементів мають відповідати тематиці вибраного структурного підрозділу організації.

5 Створити додаткову сторінку у файлі `about.html` і використати її для подання окремого змістового блоку, наприклад короткої історії структурного підрозділу організації, опису напрямів діяльності або контактної інформації. Зв’язок між файлами `index.html` і `about.html` потрібно організувати через коректні гіперпосилання.

6 Готову роботу перевірити в браузері та виконати валідацію HTML-коду. Необхідно переконатися, що сторінки відкриваються без помилок, посилання працюють, зображення і мультимедійні елементи відображені коректно, код має охайну структуру.

1.5 Зміст звіту

За результатами виконання індивідуального завдання підготувати звіт у форматі PDF, який має містити:

1 Титульний аркуш із зазначенням дисципліни, номера лабораторної роботи, прізвища та імені здобувача, номера групи.
2 Номер варіанта і посилання на вебсторінку структурного підрозділу організації, яка була джерелом вмісту.
3 Короткий опис виконаних дій.
4 Скриншоти сторінок `index.html` та `about.html` у браузері.
5 Результат перевірки HTML-коду валідатором W3C.
6 Висновки про виконану роботу.

Разом зі звітом необхідно надати файли `index.html`, `about.html` і використані медіафайли (зображення, відео, аудіо).

1.6 Варіанти вихідних даних

Варіанти індивідуальних завдань наведено в електронному ресурсі [8].

Контрольні запитання

1 Що таке HTML і яке його призначення для веброзроблення?
2 Чим HTML відрізняється від мов програмування?
3 Що таке тег? Яка різниця між відкривальним і закривальним тегом?

4 Із яких основних блоків складається базова структура HTML-документа?

5 Яке призначення оголошення `<!DOCTYPE html>`?

6 Що таке семантичні теги та навіщо вони потрібні? Наведіть приклади.

7 Яке призначення атрибута `lang` у тегу `<html>`?

8 Що забезпечує тег `<meta charset="UTF-8">`?

9 Яка різниця між маркованим (`<ul>`) і нумерованим (`<ol>`) списком?

10 Яке призначення атрибута `href` у тегу `<a>`? Чим внутрішнє посилання відрізняється від зовнішнього?

11 Яке призначення атрибута `alt` у тегу `<img>` і чому його важливо заповнювати?

12 Які теги використовують для створення таблиці в HTML і яке призначення кожного з них?

13 Яке призначення тегу `<form>` і атрибутів `action` і `method`?

14 Чим відрізняються елементи `<input type="radio">` і `<input type="checkbox">`?

15 Що таке HTML-валідація і яке її значення для якості вебдокумента?

Лабораторна робота 2
ОСНОВИ CSS

Мета роботи – формування навичок використання CSS для оформлення вебсторінок, роботи з селекторами та послідовного покращення візуального подання вже наявної структури документа.

2.1 Теоретичні відомості

CSS (Cascading Style Sheets) є мовою стилів, яку застосовують для оформлення вебсторінок. HTML визначає структуру документа і його змістові елементи, а CSS відповідає за зовнішній вигляд: кольори, шрифти, відступи, рамки, розміри, положення елементів і візуальні ефекти.

CSS-правило складається із селектора і блоку оголошень. Селектор визначає, до яких елементів буде застосовано оформлення; усередині фігурних дужок записують властивості та їхні значення, наприклад `color`, `font-size`, `background-color`, `margin` або `padding`.

Стилі можна підключати до HTML різними способами. Вбудовані стилі задають безпосередньо в атрибуті `style` окремого елемента, внутрішні стилі розміщують у тегу `<style>` усередині документа; зовнішні стилі виносять до окремого файлу з розширенням `.css` і підключають через тег `<link>`.

У CSS важливими є каскадність, успадкування та специфічність. Каскадність означає, що для одного елемента можуть одночасно бути прийнятними кілька правил, і браузер має визначити, яке з них буде пріоритетним. Успадкування пояснює, чому деякі властивості, наприклад колір тексту чи шрифт, можуть бути передані від батьківського елемента до вкладених елементів. Специфічність показує, що селектор ідентифікатора є точнішим за селектор класу, селектор класу є точнішим за селектор елемента.

Під час стилізації важливо розуміти блокову модель CSS (рисунок 2.1). Кожен елемент можна розглядати як прямокутний блок, який має область вмісту, внутрішні відступи `padding`, рамку і зовнішні відступи `margin`. Завдяки цим властивостям налаштовують відстані між елементами, межі блоків і загальну візуальну впорядкованість сторінки. Також потрібно звернути увагу на одиниці вимірювання, які часто використовують у CSS, наприклад `px`, `%` і `rem`, оскільки вони визначають розміри шрифтів, ширину елементів, відступи та інші параметри оформлення.

Рисунок 2.1 – Блокова модель CSS

За допомогою CSS можна задавати базове оформлення та керувати поведінкою елементів у різних станах. Псевдокласи дають змогу змінювати вигляд елементів під час наведення курсора, встановлення фокуса або інших дій користувача. Псевдоелементи використовують для декоративного додавання вмісту або оформлення окремої частини елемента без зміни HTML-коду. Властивість `transition` допомагає зробити зміну станів плавною, для того щоби сторінка виглядала охайніше, і її сприймали як завершену.

Додаткові відомості про CSS-селектори, властивості і способи підключення стилів наведено в джерелах [9, 10].

2.2 Засоби виконання

Для виконання лабораторної роботи необхідні:

– персональний комп'ютер або ноутбук з операційною системою Windows, macOS або Linux;
– браузер актуальної версії (Google Chrome, Mozilla Firefox або Microsoft Edge);
– редактор Visual Studio Code із файлами `index.html` та `about.html`, підготовленими в лабораторній роботі 1;
– онлайн-сервіс W3C CSS Validation Service [11].

2.3 Основи використання CSS

Ця робота є прямим продовженням попередньо підготовленої HTML-сторінки. CSS додають до файлів `index.html` і `about.html`, які вже були створені раніше.

1 Підготовка HTML-структури для стилізації

1 Скопіюйте до папки `lab_2` файли `index.html`, `about.html`, `image.jpg`, `video.mp4` і `audio.mp3` із папки `lab_1`.

Отже, у папці лабораторної роботи вже буде підготовлена HTML-основа, до якої можна послідовно додавати стилі.

2 Створення файлу стилів `styles.css`

Потрібно створити окремий файл стилів. У ньому поступово накопичуватимуть усі CSS-правила для обох HTML-сторінок.

1 Створіть у папці `lab_2` новий файл `styles.css`. Скористайтеся командою `File > New File` або кнопкою створення нового файлу в лівій частині вікна Visual Studio Code.

2 Збережіть новий файл із назвою `styles.css`. Натисніть `Ctrl + S` (`Cmd + S` на macOS), введіть назву файлу та підтвердьте збереження.

3 Додайте до файлу `styles.css` початковий службовий коментар.

```
/* Базові стилі сторінки */
```

4 Збережіть файл `styles.css`, щоб надалі всі наступні правила додавати до вже створеного документа.

Окремий файл стилів дає змогу централізовано керувати оформленням одразу кількох сторінок і не змішувати HTML-структуру з CSS-оформленням.

3 Підключення зовнішнього CSS-файлу до HTML-документів

Щоб браузер почав застосовувати стилі, файл `styles.css` потрібно підключити до кожної HTML-сторінки. Для цього використовують тег `<link>` у блоці `<head>`.

1 Відкрийте файл `index.html` і додайте перед закривальним тегом `</head>` рядок, наведений нижче.

```
<link rel="stylesheet" href="styles.css">
```

2 Відкрийте файл `about.html` і додайте перед закривальним тегом `</head>` такий самий рядок.

```
<link rel="stylesheet" href="styles.css">
```

3 Збережіть файли `index.html` і файл `about.html`.

У тегу `<link>` атрибут `rel="stylesheet"` повідомляє браузеру, що підключена таблиця стилів, атрибут `href="styles.css"` указує шлях до файлу стилів.

Після підключення зовнішнього файлу стилів обидві сторінки почнуть використовувати спільне CSS-оформлення.

4 Застосування базових стилів для всієї вебсторінки

У цьому кроці додамо базові правила, які впливатимуть на всю сторінку та формуватимуть початковий вигляд документа і робитимуть подальшу стилізацію передбачуваною.

1 Додайте у файл `styles.css` базові правила, наведені нижче.

```
* {
    box-sizing: border-box;
}

body {
    margin: 0;
    font-family: Arial, sans-serif;
    line-height: 1.6;
    color: #1f2a44;
    background-color: #eef3f8;
}
```

2 Збережіть файл `styles.css`, оновіть сторінку `index.html` у браузері, переконайтеся, що текст став рівномірнішим, зовнішній відступ браузера зник (рисунок 2.2).

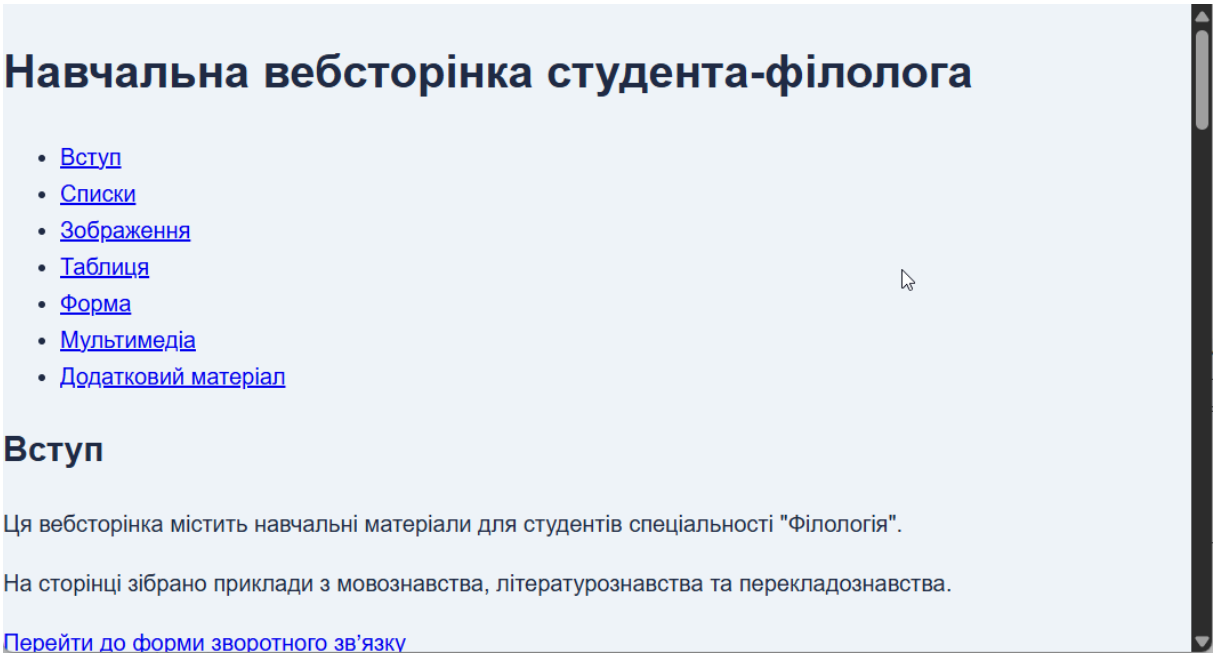

Рисунок 2.2 – Результат застосування базових стилів до `index.html`

Селектор * називають універсальним, оскільки його застосовують для всіх елементів сторінки. У цьому прикладі він задає `box-sizing: border-box`, тому `padding` і `border` ураховані всередині вказаної ширини елемента. Селектор `body` задає спільні параметри для всього документа: `margin: 0` прибирає стандартний зовнішній відступ, `font-family: Arial, sans-serif` задає основний шрифт і запасний беззарубковий варіант, `line-height: 1.6` збільшує міжрядковий інтервал для зручнішого читання, `color: #1f2a44` визначає базовий колір тексту, `background-color: #eef3f8` задає світлий фон сторінки.

5 Стилізація основних текстових елементів вебсторінки

Після базового оформлення доцільно впорядкувати текстові елементи. Заголовки і абзаци повинні мати зрозумілі розміри та відступи, щоб сторінка була охайною і послідовною.

1 Додайте у файл `styles.css` правила для заголовків і абзаців.

```
h1, h2, h3 {
    margin-top: 0;
}

h1 {
    font-size: 2.2rem;
    margin-bottom: 16px;
}

h2 {
    font-size: 1.6rem;
    margin-bottom: 16px;
}

h3 {
    font-size: 1.15rem;
    margin-bottom: 12px;
}

p {
    margin-bottom: 16px;
}
```

2 Збережіть файл `styles.css`, оновіть сторінку в браузері та перевірте, що заголовки різних рівнів відрізняються за розміром, абзаци мають однакові відступи.

Запис `h1, h2, h3` через кому означає, що одне правило одночасно застосовують для кількох селекторів. Властивість `margin-top: 0` прибирає стандартний верхній відступ заголовків, `margin-bottom` залишає контрольований проміжок після них. Значення в одиницях `rem` залежать від базового розміру шрифту документа, тому таке оформлення легше масштабувати і підтримувати. У результаті сформована зрозуміла візуальна ієрархія: головний заголовок сприймають як назву сторінки, заголовки другого рівня як назви розділів, заголовки третього рівня як підзаголовки всередині секцій.

6 Оформлення кольорової гами та фону сторінки

На цьому кроці потрібно зробити сторінку візуально виразнішою, узгодити кольори фону, заголовків і допоміжного тексту.

1 Замініть правило `body` у файлі `styles.css`.

```
body {
    margin: 0;
    font-family: Arial, sans-serif;
    line-height: 1.6;
    color: #1f2a44;
    background: linear-gradient(180deg, #eef3f8 0%,
#f8fafc 100%);
}
```

2 Замініть поточне правило `h2` у файлі `styles.css` на нове та додайте нижче окреме правило для абзацу в нижньому колонтитулі.

```
h2 {
    color: #23408e;
    border-bottom: 2px solid #d7dff0;
    padding-bottom: 8px;
}
```

```
footer p {
    color: #5d6785;
}
```

3 Збережіть файл `styles.css` і перевірте сторінку в браузері. Зверніть увагу, що фон став м'якшим, назви секцій краще відокремлені від основного тексту.

Функція `linear-gradient(180deg, ...)` створює плавний вертикальний перехід між кольорами зверху вниз. Властивість `color` у правилі `h2` змінює колір назв розділів, `border-bottom` додає нижню межу під заголовком, `padding-bottom` створює невеликий внутрішній відступ між текстом і цією межею. Селектор `footer p` звертається лише до абзацу всередині елемента `footer`, тому текст у нижньому колонтитулі можна оформити окремо від основного вмісту сторінки.

7 Використання селекторів елементів у CSS

У попередніх кроках уже було використано селектори елементів для `body`, `h1`, `h2`, `h3` і `p`. Тепер розширимо цей підхід на інші типи елементів, які вже є в HTML-документі.

1 Додайте у файл `styles.css` правила для секцій, списків і підписів форми.

```
section {
    margin-bottom: 36px;
}

ul, ol {
    padding-left: 24px;
}

label {
    font-weight: 600;
}
```

2 Збережіть файл `styles.css`, оновіть сторінку в браузері та перевірте, що секції краще відділені одна від одної, списки мають зручні відступи.

Селектори елементів дають змогу швидко стилізувати всі теги певного типу без додавання додаткових атрибутів до HTML-коду. Властивість `margin-bottom` у правилі `section` додає проміжок між секціями, `padding-left: 24px` у правилах для `ul` і `ol` залишає місце для маркерів або нумерації, `font-weight: 600` робить підписи `label` помітнішими поруч із полями форми.

8 Додавання класів та ідентифікаторів для подальшої стилізації

Щоб точніше керувати оформленням окремих частин сторінки, потрібно додати кілька класів до тих елементів, які мають отримати індивідуальний вигляд. Наявні ідентифікатори секцій, створені в попередній роботі, залишаються без змін і далі використовувані для навігації та стилізації.

1 Додайте клас `site-nav` до тегу `<nav>` у файлі `index.html` і у файлі `about.html`.

```
<nav class="site-nav">
```

2 Додайте клас `external-link` до зовнішнього посилання і клас `note-box` до відкривального тегу контейнера довідки у файлі `index.html`.

```
<p><a href="https://example.com" target="_blank"
class="external-link">Відкрити електронний ресурс</a></p>

<div class="note-box">
```

3 Додайте потрібні класи до тегів зображення, таблиці, форми, відео і аудіо у файлі `index.html`, щоб вони мали вигляд, показаний нижче.

```
<img class="main-image" src="image.jpg" alt="Навчальна
ілюстрація з філології">
```

```
<table class="info-table">

<form class="feedback-form" action="#" method="post">

<video class="media-player" controls>

<audio class="media-player" controls>
```

4 Додайте клас `note-box` до відкривального тегу секції у файлі `about.html`.

```
<section class="note-box">
```

5 Збережіть файли `index.html` і `about.html`.

Атрибут `class` дає змогу позначити елемент для подальшого CSS-оформлення. Один і той самий клас можна використати для кількох елементів, тоді як `id` залишається унікальним позначенням конкретного блоку сторінки.

9 Використання селекторів класів та ідентифікаторів у CSS

Після додавання класів і наявних `id` можна перейти до точнішої стилізації окремих фрагментів сторінки, що дає змогу виділити важливі блоки та не змінювати вигляд усіх елементів одного типу одночасно.

1 Додайте у файл `styles.css` правила для ідентифікаторів і класів.

```
#intro {
    scroll-margin-top: 24px;
}

#feedback {
    background-color: #f7faff;
    padding: 24px;
    border-radius: 16px;
}

.site-nav ul {
    list-style: none;
```

```
    padding-left: 0;
}

.note-box {
    margin-top: 20px;
    padding: 16px 18px;
    background-color: #fff7ec;
    border-left: 4px solid #c2643f;
    border-radius: 12px;
}
```

2 Збережіть файл `styles.css`, оновіть сторінку `index.html` і перевірте, що секція форми відрізняється від інших блоків, блок довідки має окреме акцентне оформлення.

Селектори ідентифікаторів зручні для унікальних блоків сторінки, селектори класів краще використовувати там, де однаковий стиль повторюється на кількох сторінках або в кількох місцях документа. Складений селектор `.site-nav ul` означає список `ul`, що знаходиться всередині елемента з класом `site-nav`. Властивість `scroll-margin-top` додає верхній відступ під час переходу за внутрішнім посиланням, тому секція не прилягає впритул до верхнього краю вікна. У правилі `#feedback` значення `background-color`, `padding` і `border-radius` формують окремий м'яко виділений блок форми, `list-style: none` і `padding-left: 0` прибирають стандартні маркери та відступ списку меню, `border-left` у `.note-box` створює акцентну вертикальну смугу ліворуч.

10 Стилізація верхньої частини сторінки, основного вмісту і нижнього колонтитула

На цьому етапі оформимо великі структурні блоки сторінки. Так сторінка отримає більш завершений вигляд, основний вміст буде візуально відокремлений від фону.

1 Додайте у файл `styles.css` правила для `header`, `main` і `footer`.

```css
header, main, footer {
    width: min(1000px, calc(100% - 40px));
    margin: 0 auto;
}

header {
    padding: 28px 32px;
    background-color: #23408e;
    color: #ffffff;
    border-radius: 0 0 18px 18px;
}

main {
    margin-top: 24px;
    padding: 32px;
    background-color: #ffffff;
    border-radius: 18px;
    box-shadow: 0 12px 30px rgba(35, 64, 142, 0.08);
}

footer {
    padding: 24px 0 40px;
    text-align: center;
}
```

2 Збережіть файл `styles.css`, оновіть сторінку в браузері та перевірте, що верхня частина сторінки виглядає як окремий блок, основний вміст отримав білий фон і м'яку тінь.

Функція `min(1000px, calc(100% - 40px))` дає змогу задати ширину, яка не перевищує `1000px`, але водночас залишає бічні поля на вузьких екранах. Вираз `calc(100% - 40px)` віднімає від повної ширини `40px`, запис `margin: 0 auto` центрує блок по горизонталі. Властивість `padding` у `header` і `main` створює внутрішні поля, `background-color` задає фон блоків, `color: #ffffff` робить текст у верхній частині сторінки контрастним, `border-radius` заокруглює кути. Властивість `margin-top` відсуває основний вміст від верхньої частини сторінки, `text-align:`

`center` центрує текст у нижньому колонтитулі, функція `rgba(35, 64, 142, 0.08)` у `box-shadow` задає колір тіні з частковою прозорістю. Таке оформлення допомагає відразу візуально відокремити верхню частину сторінки, основний зміст і нижню частину сторінки.

11 Стилізація навігаційного меню вебсторінки

Меню у верхній частині сторінки має бути читабельним і зручним для натискання. Для цього потрібно оформити список навігації як компактний горизонтальний набір посилань.

1 Додайте у файл `styles.css` правила для меню навігації.

```
.site-nav ul {
    display: flex;
    flex-wrap: wrap;
    gap: 12px;
    margin: 20px 0 0;
}

.site-nav a {
    display: block;
    padding: 10px 14px;
    border-radius: 999px;
    background-color: rgba(255, 255, 255, 0.18);
    color: #ffffff;
    text-decoration: none;
}
```

2 Збережіть файл `styles.css` і перевірте сторінку `index.html` і сторінку `about.html` у браузері (рисунок 2.3). Переконайтеся, що меню в обох документах оформлено однаково.

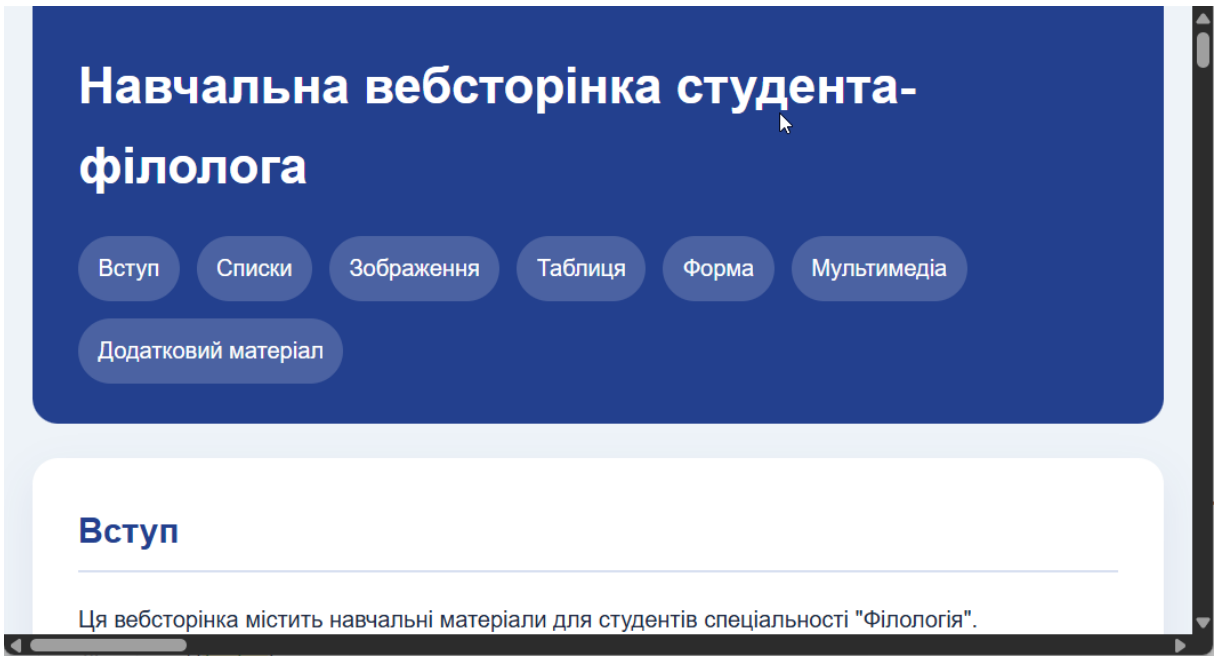

Рисунок 2.3 – Результат стилізації навігаційного меню

Властивість `display: flex` розміщує елементи списку в один ряд, `flex-wrap: wrap` дає змогу переносити їх на новий рядок за нестачі місця, `gap` задає відстань між пунктами меню, `margin: 20px 0 0` відсуває список від заголовка сторінки. У правилі для `.site-nav a` значення `display: block` робить усе посилання клікабельним як єдиний блок, `padding: 10px 14px` збільшує зручну область натискання, `background-color: rgba(...)` створює напівпрозорий фон, `color: #ffffff` задає світлий текст, `border-radius: 999px` надає посиланню округлої форми. На цьому кроці список із посиланнями перетворюється на зручне меню, яке можна використовувати і на головній, і на додатковій сторінці.

12 Оформлення списків і гіперпосилань

Списки та посилання належать до базових елементів будь-якої вебсторінки. Вони мають бути достатньо помітними, але не перевантажувати загальний вигляд сторінки.

1 Додайте у файл `styles.css` правила для пунктів списку та звичайних посилань.

```
li {
    margin-bottom: 8px;
}

a {
    color: #23408e;
    text-decoration: none;
    font-weight: 600;
}

.external-link {
    display: inline-block;
    margin-top: 4px;
    color: #c2643f;
}
```

2 Збережіть файл `styles.css`, оновіть сторінку в браузері та перевірте, що пункти списків читабельні рівномірно, зовнішнє посилання помітно відрізняється від інших гіперпосилань у тексті (рисунок 2.4).

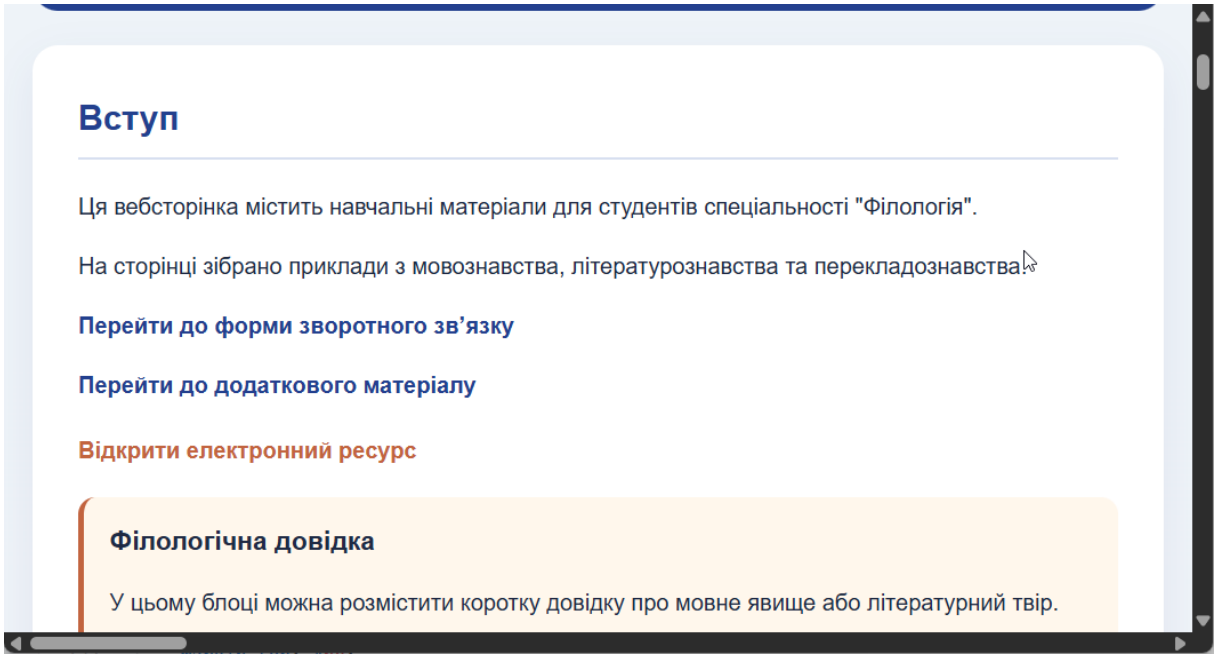

Рисунок 2.4 – Результат стилізації гіперпосилань

Загальне правило `a` задає спільний вигляд усіх посилань: `text-decoration: none` прибирає стандартне підкреслення, `font-weight: 600` робить текст виразнішим, `color: #23408e` задає основний колір гіперпосилань. Для `li` властивість `margin-bottom` додає невеликий проміжок між пунктами списку. У правилі `.external-link` значення `display: inline-block` дає змогу коректно застосувати верхній відступ для посилання всередині абзацу, `margin-top: 4px` трохи відокремлює його від попереднього тексту, `color: #c2643f` візуально виділяє зовнішнє посилання.

13 Стилізація зображень і мультимедійних елементів

Графічні та мультимедійні елементи мають гармонійно вписуватися в сторінку. Для цього потрібно обмежити їхню ширину, додати відступи і надати охайного вигляду.

1 Додайте у файл `styles.css` правила для зображення, відео та аудіо.

```
.main-image, .media-player {
    display: block;
    max-width: 100%;
    margin-top: 16px;
}

.main-image {
    border: 4px solid #e1e8f5;
    border-radius: 14px;
    box-shadow: 0 10px 24px rgba(31, 42, 68, 0.12);
}

.media-player {
    width: 100%;
    border-radius: 12px;
}
```

2 Збережіть файл `styles.css`, оновіть сторінку в браузері та перевірте, що зображення не виходить за межі основного блоку, відео і аудіо акуратно розміщені в секції мультимедіа.

Властивість `display: block` прибирає зайвий проміжок, який часто виникає біля рядкових елементів, `max-width: 100%` допомагає уникнути виходу елемента за межі контейнера, `margin-top: 16px` створює відступ над медіаелементом. Для `.main-image` властивості `border`, `border-radius` і `box-shadow` додають рамку, заокруглення та м'яку тінь. Для `.media-player` властивість `width: 100%` розтягує відео або аудіо на всю доступну ширину батьківського блоку, `border-radius` пом'якшує форму контейнера.

14 Стилізація таблиці для подання структурованих даних

Таблиця має бути зручною для читання, тому її потрібно оформити через CSS замість застарілих HTML-атрибутів оформлення, що, крім того, допоможе відокремити структуру від зовнішнього вигляду.

1 Додайте у файл `styles.css` правила для таблиці.

```
.info-table {
    width: 100%;
    border-collapse: collapse;
    margin-top: 16px;
    background-color: #fbfcff;
}

.info-table th,
.info-table td {
    padding: 12px 14px;
    border: 1px solid #cfd8ea;
    text-align: left;
}

.info-table th {
    background-color: #23408e;
    color: #ffffff;
}
```

2 Збережіть файл `styles.css`, оновіть сторінку в браузері, переконайтеся, що заголовки таблиці виділені кольором, клітинки мають однакові межі та відступи (рисунок 2.5).

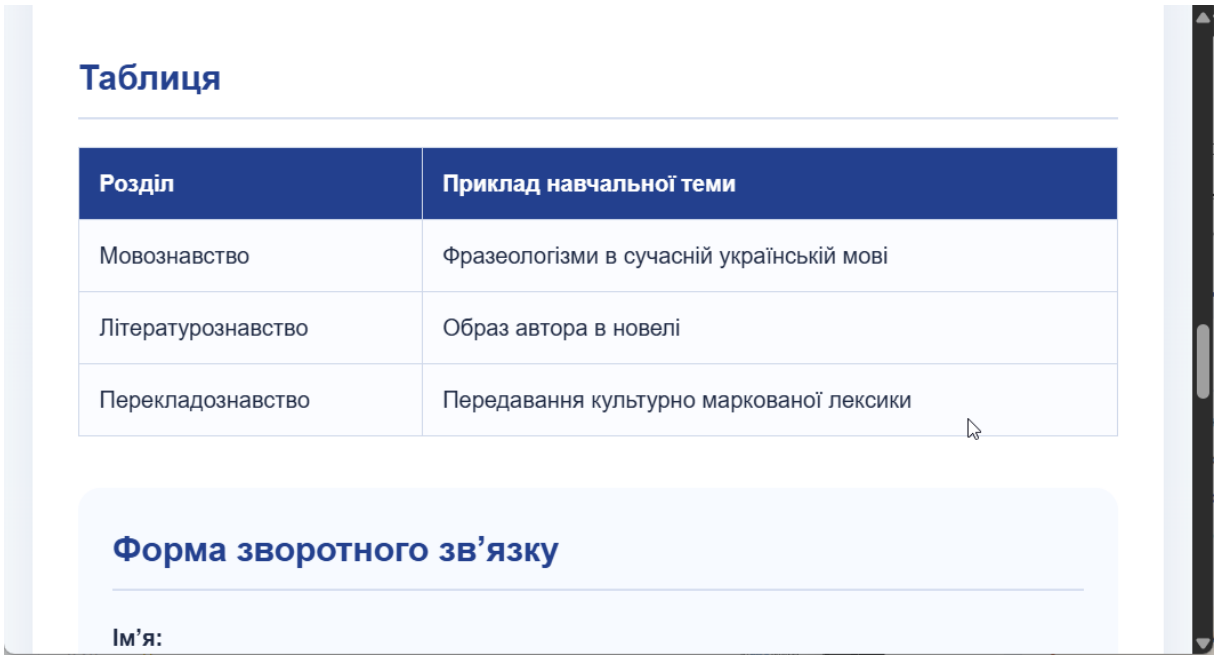

Рисунок 2.5 – Результат стилізації таблиці

Властивість `width: 100%` розтягує таблицю на всю ширину доступного блоку, `margin-top: 16px` відокремлює її від попереднього тексту, `background-color` задає світлий фон таблиці. Властивість `border-collapse: collapse` об'єднує межі сусідніх клітинок, завдяки чому таблиця виглядає акуратно і не має подвійних ліній. У правилах для `th` і `td` значення `padding` додає внутрішні відступи в клітинках, `border` створює межі, `text-align: left` вирівнює текст по лівому краю.

15 Стилізація форми введення даних

Форма зворотного зв'язку має бути структурно правильною та зручною для заповнення. Для цього потрібно оформити поля введення, список вибору та кнопку надсилання.

1 Додайте у файл `styles.css` правила для форми та її елементів.

```css
.feedback-form {
    max-width: 640px;
}

.feedback-form input[type="text"],
.feedback-form input[type="email"],
.feedback-form textarea,
.feedback-form select {
    width: 100%;
    padding: 10px 12px;
    margin: 6px 0 16px;
    border: 1px solid #bcc7dd;
    border-radius: 8px;
    background-color: #ffffff;
    font: inherit;
}

.feedback-form textarea {
    resize: vertical;
    min-height: 120px;
}

.feedback-form input[type="radio"],
.feedback-form input[type="checkbox"] {
    margin-right: 8px;
}

.feedback-form button {
    padding: 10px 18px;
    border: none;
    border-radius: 8px;
    background-color: #23408e;
    color: #ffffff;
    cursor: pointer;
    font: inherit;
}
```

2 Збережіть файл `styles.css`, оновіть сторінку в браузері, перевірте, що текстові поля, список вибору і кнопка мають узгоджений вигляд, форму можна сприймати як цілісний блок (рисунок 2.6).

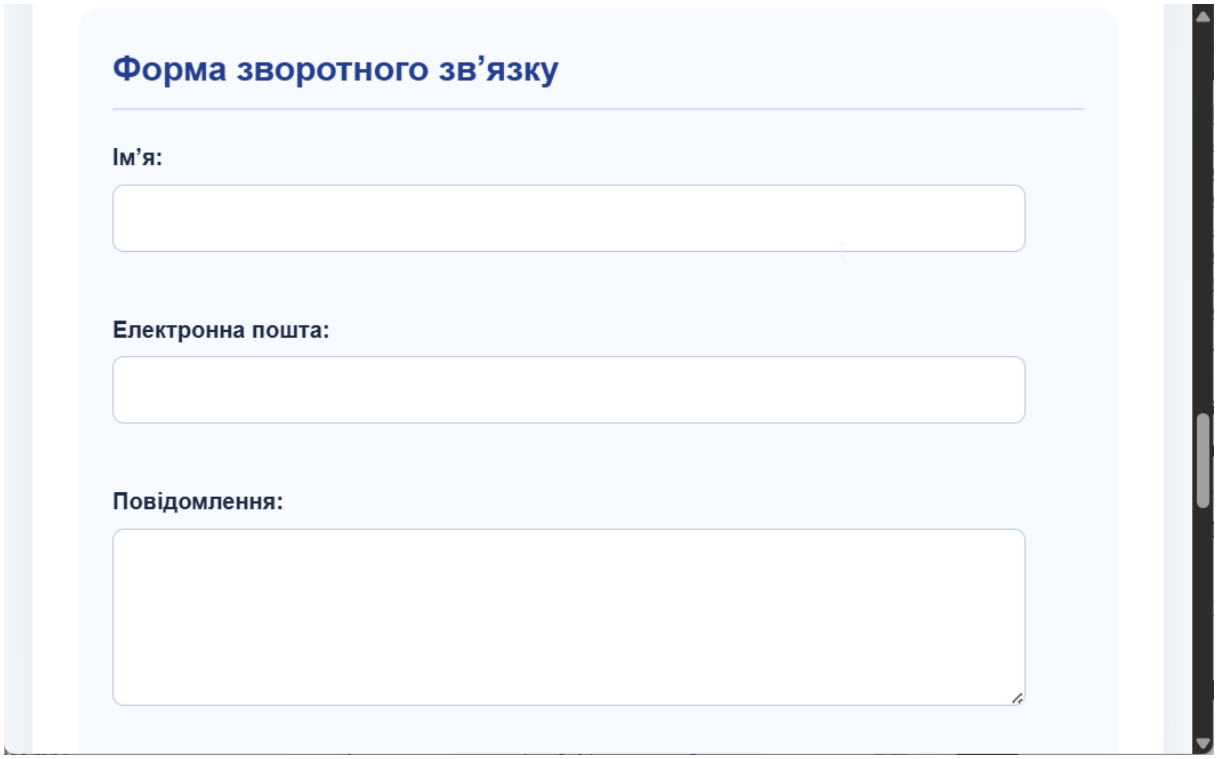

Рисунок 2.6 – Результат стилізації форми

Селектори на кшталт `input[type="text"]` називають селекторами атрибутів: вони дають змогу звернутися лише до тих полів `input`, у яких атрибут `type` має вказане значення. Властивість `max-width` обмежує загальну ширину форми, `width: 100%` розтягує поля на ширину доступного контейнера, `padding` створює внутрішній простір для тексту, `border` і `border-radius` формують охайні межі полів. Властивість `font: inherit` успадковує шрифт від батьківського елемента, `resize: vertical` дає змогу змінювати розмір текстової області лише по вертикалі, `margin-right` відокремлює радіокнопки і прапорець від підписів, `cursor: pointer` показує, що кнопку можна натиснути. На цьому етапі форма стає зручнішою для введення даних, однакова гарнітура і відступи роблять усі елементи оформлення послідовними.

16 Використання псевдокласів для інтерактивного оформлення елементів

Псевдокласи дають змогу змінювати вигляд елементів у відповідь на дії користувача, що робить сторінку зрозумілішою, оскільки під час наведення курсора або встановлення фокуса елемент візуально реагує на взаємодію.

1 Додайте у файл `styles.css` правила з псевдокласами.

```
a:hover {
    color: #c2643f;
}

.site-nav a:hover,
.site-nav a:focus {
    background-color: rgba(255, 255, 255, 0.32);
    color: #ffffff;
}

.feedback-form input:focus,
.feedback-form textarea:focus,
.feedback-form select:focus {
    outline: none;
    border-color: #23408e;
    box-shadow: 0 0 0 3px rgba(35, 64, 142, 0.15);
}

.feedback-form button:hover {
    background-color: #1a316f;
}
```

2 Збережіть файл `styles.css` і перевірте сторінку в браузері. Наведіть курсор на посилання і кнопку, потім натисніть на текстовому полі або полі електронної пошти, щоб перевірити роботу стану `:focus`.

Псевдокласи `:hover` і `:focus` покращують зручність роботи зі сторінкою та підказують, який саме елемент зараз активний або готовий до введення. Властивість `outline: none` прибирає стандартну рамку браузера навколо активного поля, `box-shadow` у цьому разі створює власне м’яке підсвічування елемента під час фокуса.

17 Використання псевдоелементів для декоративного оформлення вмісту

Псевдоелементи дають змогу додавати декоративні деталі без зміни тексту в HTML-документі. Такий підхід зручний, коли потрібно зробити оформлення виразнішим, але не перевантажувати саму розмітку зайвими символами.

1 Додайте у файл `styles.css` правило з псевдоелементом.

```
h2::before {
    content: ">> ";
    color: #c2643f;
}
```

2 Збережіть файл `styles.css`, оновіть сторінку в браузері, перевірте, що перед кожним заголовком другого рівня з’явився декоративний префікс (рисунок 2.7).

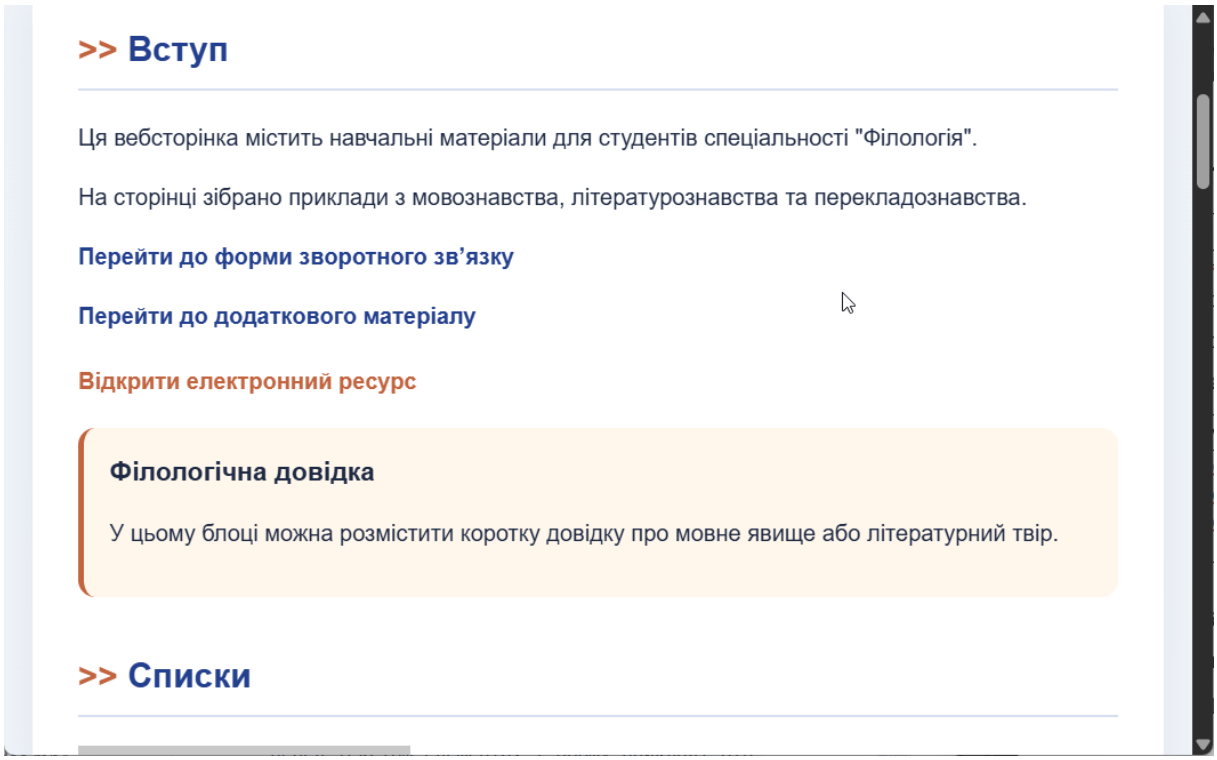

Рисунок 2.7 – Результат використання псевдоелемента для заголовків `h2`

Псевдоелемент `::before` вставляє додатковий вміст перед текстом елемента. У цьому прикладі він використаний лише як декоративний акцент.

18 Додавання плавних переходів і простих візуальних ефектів

Після введення станів `:hover` і `:focus` зробимо зміну оформлення плавнішою, використовуючи властивість `transition`.

1 Додайте у файл `styles.css` правила для плавних переходів.

```
a,
.main-image,
.feedback-form button,
.feedback-form input,
.feedback-form textarea,
.feedback-form select {
    transition: color 0.2s ease, background-color 0.2s
ease, border-color 0.2s ease, box-shadow 0.2s ease,
transform 0.2s ease;
}

.main-image:hover {
    transform: scale(1.01);
}

.feedback-form button:hover {
    transform: translateY(-1px);
}
```

2 Збережіть файл `styles.css`, оновіть сторінку в браузері, наведіть курсор на зображення, кнопку і посилання, щоб перевірити плавність зміни станів.

У властивості `transition` послідовно перелічено властивості, для яких потрібно зробити плавну зміну, тривалість переходу і тип часової функції `ease`. Функція `scale(1.01)` трохи збільшує зображення під час наведення, `translateY(-1px)` злегка зміщує кнопку вгору. Плавні переходи роблять взаємодію зі сторінкою приємнішою та прибирають різкі візуальні зміни під час наведення курсора або роботи з формою.

19 Валідація CSS-коду та підсумкове впорядкування файлу стилів

Завершальним етапом є перевірка коректності CSS-коду. Валідація допомагає знайти синтаксичні помилки, пропущені символи або неточні назви властивостей.

1 Відкрийте браузер і перейдіть на сторінку W3C CSS Validation Service [11].

2 Завантажте файл `styles.css` або вставте його вміст у поле перевірки.

3 Проаналізуйте повідомлення валідатора та виправте помилки, якщо вони будуть виявлені (рисунок 2.8).

4 Перегляньте файл `styles.css` після виправлень і переконайтеся, що правила розташовані в логічній послідовності.

5 Збережіть файл `styles.css` і повторно перевірте сторінки в браузері.

Рисунок 2.8 – Результат перевірки коректності CSS-коду

2.4 Порядок виконання роботи

Індивідуальне завдання виконують на основі HTML-сторінки про структурний підрозділ організації, створеної в першій роботі, і сайту-зразка, який необхідно вибрати відповідно до варіанта. Із сайту-зразка потрібно використовувати лише візуальні орієнтири, а не зміст сторінок.

1 За основу взяти файли `index.html` і `about.html`, підготовлені в межах першої роботи. Зміст про структурний підрозділ організації,

назви розділів, контактні дані та інші фактичні відомості потрібно зберегти без заміни.

2 Відповідно до варіанта вибрати сайт-зразок, який буде використано як стилістичний орієнтир. Необхідно переглянути головну сторінку та інші розділи сайту, визначити основні візуальні риси оформлення; проаналізувати кольорову гаму, типографіку, вигляд навігації, кнопок, посилань, таблиць, форм, блоків із фоном і відступами – ці елементи будуть орієнтиром для створення власного `styles.css`.

3 Створити файл `styles.css` так, щоб сторінка структурного підрозділу організації набула стилістики, подібної до сайту-зразка. Стилі застосувати для файлів `index.html` і `about.html`. Якщо для цього потрібно додати окремі `class` або `id`, до HTML-розмітки вносять лише мінімальні зміни без перебудови змісту сторінок.

4 Забезпечити цілісність оформлення. Однакові кольори, шрифти, відступи, рамки та акцентні елементи мають повторюватися на головній і додатковій сторінках і підтримувати спільний візуальний стиль.

5 Виконати валідацію CSS-коду та переглянути файл `styles.css` щодо повторів або неузгоджених правил.

2.5 Зміст звіту

За результатами виконання індивідуального завдання підготувати звіт у форматі PDF, який має містити:

1 Титульний аркуш із зазначенням дисципліни, номера лабораторної роботи, прізвища та імені здобувача, номера групи.

2 Номер варіанта, посилання на вебсторінку структурного підрозділу організації з першої роботи і посилання на сайт-зразок, який було взято за візуальний орієнтир.

3 Короткий опис виконаних дій.

4 Скриншоти стилізованих сторінок `index.html` і `about.html` у браузері.

5 Результат перевірки CSS-коду валідатором W3C.

6 Висновки про виконану роботу.

Разом зі звітом необхідно надати файли `index.html`, `about.html`, `styles.css` і використані медіафайли.

2.6 Варіанти вихідних даних

Варіанти індивідуальних завдань наведено в електронному ресурсі [12].

Контрольні запитання

1 Що таке CSS і яке його призначення для веброзроблення?

2 Яку роль відіграє CSS порівняно з HTML?

3 Із яких частин складається CSS-правило?

4 Назвіть три способи підключення CSS до HTML-документа і вкажіть, який із них є рекомендованим і чому.

5 Яке призначення тегу `<link>` і його атрибутів `rel` і `href`?

6 Що таке каскадність у CSS і як вона впливає на застосування правил?

7 Що таке специфічність селектора? Який селектор є пріоритетнішим: елемента, класу чи ідентифікатора?

8 Яка різниця між селектором елемента, селектором класу та селектором ідентифікатора?

9 Що таке блокова модель CSS? Які складові вона включає?

10 Яка різниця між властивостями `margin` і `padding`?

11 Що таке псевдоклас у CSS? Наведіть два приклади та поясніть їхнє призначення.

12 Що таке псевдоелемент у CSS? Наведіть приклад його використання.

13 Яке призначення властивості `transition` і як вона покращує взаємодію зі сторінкою?

14 Що означає запис `color: rgba(0, 0, 0, 0.5)` і яке призначення має четвертий параметр?

15 Що таке CSS-валідація і яке її значення для якості таблиці стилів?

Лабораторна робота 3
ОСНОВИ BOOTSTRAP 5

Мета роботи – ознайомлення з фреймворком Bootstrap і формування навичок використання готових класів і компонентів для швидкого оформлення вебсторінок, створених засобами HTML.

3.1 Теоретичні відомості

Bootstrap є популярним фронтенд-фреймворком, який містить готову систему стилів, набір службових класів і типові компоненти для побудови вебінтерфейсів. Його основна перевага полягає в тому, що значну частину оформлення сторінки можна виконати без самостійного написання великої кількості CSS-правил. Завдяки цьому Bootstrap часто використовують для швидкого створення навчальних, демонстраційних і прикладних інтерфейсів.

Оформлення в Bootstrap базовано на використанні класів, які додають безпосередньо до HTML-елементів. Ці класи керують відступами, кольорами, розмірами тексту, межами, фоном, вирівнюванням і поведінкою елементів на сторінці. Умовно ці класи можна поділити на службові та компонентні: службові класи швидко задають окремі параметри на кшталт `p-4`, `mb-3`, `text-white` або `bg-primary`, компонентні класи формують готові елементи інтерфейсу, наприклад кнопки, таблиці, картки чи навігаційні блоки.

Фреймворк можна підключати локально або через CDN. Такий підхід зручний, оскільки в цьому разі достатньо додати до HTML-документа посилання на готові файли Bootstrap. Після цього сторінка отримує доступ до системи оформлення, сітки і компонентів без додаткового налаштування проєкту.

Важливою частиною Bootstrap є система компонування сторінки на основі контейнерів, рядків і стовпців. Контейнер задає межі основної області вмісту, рядок організує горизонтальне розміщення блоків, стовпці дають змогу розподілити ці блоки по ширині сторінки. Це допомагає швидко будувати охайні макети і адаптувати їх до різних розмірів екрана. Для цього Bootstrap використовує адаптивні позначення на кшталт `col-md-6` або `col-lg-4`, де частина після дефіса вказує, із якого розміру екрана починає діяти відповідне правило.

Окрім сітки, Bootstrap містить багато готових компонентів, зокрема кнопки, навігацію, таблиці, форми, картки, попереджувальні повідомлення та інші елементи інтерфейсу.

Додаткові відомості про підключення Bootstrap, систему сіток і готові компоненти наведено в джерелах [13–15].

3.2 Засоби виконання

Для виконання лабораторної роботи необхідні:

- персональний комп’ютер або ноутбук з операційною системою Windows, macOS або Linux;
- браузер актуальної версії (Google Chrome, Mozilla Firefox або Microsoft Edge);
- редактор Visual Studio Code із файлами `index.html` та `about.html`, підготовленими в лабораторній роботі 1;
- доступ до мережі Інтернет для підключення Bootstrap через CDN.

3.3 Основи використання Bootstrap 5

Ця робота продовжує опрацювання вже підготовленої HTML-структури. Тут використовують ту саму навчальну сторінку, але замість

ручної стилізації через окремий CSS-файл оформлення виконують засобами Bootstrap.

1 Підготовка HTML-структури для використання Bootstrap

1 Скопіюйте до папки `lab_3` файли `index.html`, `about.html`, `image.jpg`, `video.mp4` і `audio.mp3` із папки `lab_1`.

Оскільки Bootstrap є самодостатнім засобом стилізації і замінює окремий файл `styles.css`, за основу беруть чисту HTML-структуру без підключеного CSS-файлу, тобто файли з папки `lab_1`. Після копіювання в папці лабораторної роботи вже буде підготовлена HTML-основа, до якої можна послідовно додавати Bootstrap-класи.

2 Підключення Bootstrap до HTML-документів

Щоб отримати доступ до класів і компонентів Bootstrap, потрібно підключити готові файли фреймворку до кожної HTML-сторінки. Для цього використовують CDN-посилання.

1 Додайте у файл `index.html` перед закривальним тегом `</head>` рядок, наведений нижче.

```
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.8/dist/c
ss/bootstrap.min.css" rel="stylesheet" integrity="sha384-
sRIl4kxILFvY47J16cr9ZwB07vP4J8+LH7qKQnuqkuIAvNWLzeN8tE5YB
ujZqJLB" crossorigin="anonymous">
```

2 Повторіть таку саму дію для файлу `about.html`.

3 Збережіть файл `index.html` і файл `about.html`.

У тегу `<link>` атрибут `href` указує адресу CSS-файлу Bootstrap, `rel="stylesheet"` повідомляє браузеру, що потрібно підключити таблицю стилів. Атрибути `integrity` і `crossorigin` використані для перевірки цілісності підключеного файлу.

Підключення через CDN дає змогу одразу використовувати Bootstrap без локального встановлення додаткових файлів у проєкт.

3 Перевірка коректності підключення Bootstrap на вебсторінці

Після підключення Bootstrap доцільно відразу виконати невелику перевірку. Для цього достатньо додати кілька базових класів, зміни від яких буде легко помітити в браузері.

1 Додайте до відкривального тегу `<body>` у файлі `index.html` клас.

```
<body class="bg-body-tertiary">
```

2 Додайте до головного заголовка сторінки клас `display-5`.

```
<h1 class="display-5">Навчальна вебсторінка студента-філолога</h1>
```

3 Збережіть файл `index.html` і відкрийте сторінку в браузері. Переконайтеся, що фон сторінки змінився, заголовок став більшим (рисунок 3.1).

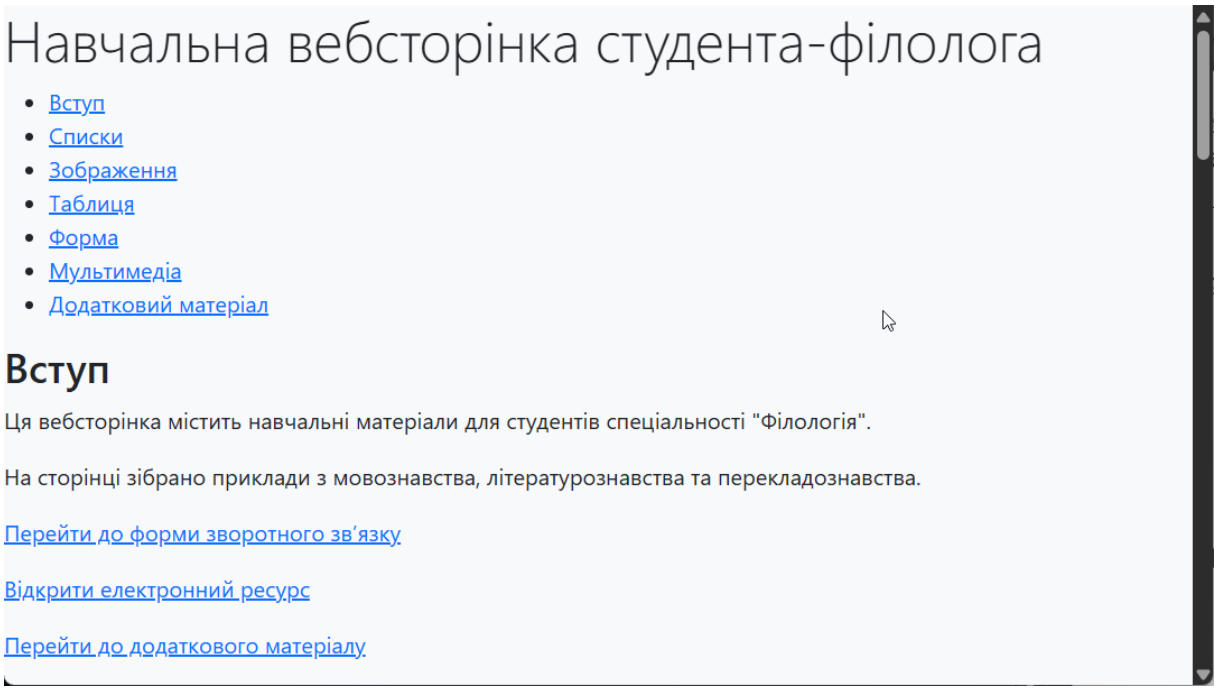

Рисунок 3.1 – Результат перевірки коректності підключення Bootstrap

У Bootstrap більшість оформлення задано через атрибут `class`. Клас `bg-body-tertiary` застосовує готове фонове оформлення сторінки, клас `display-5` надає заголовку великого виразного стилю.

4 Використання контейнерів Bootstrap для впорядкування вмісту сторінки

Контейнери Bootstrap допомагають впорядкувати вміст і не дають змогу елементам розтягуватися на всю ширину вікна без потреби. Тому доцільно додати контейнерні блоки для основних частин сторінки.

1 У файлі `index.html` одразу після відкривального тегу `<header>` додайте відкривальний тег контейнера з класом `container`.

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

2 Перед закривальним тегом `</header>` додайте закривальний тег контейнера.

```
</div>
```

3 Додайте клас `container` до відкривального тегу `<main>`.

```
<main class="container">
```

4 Додайте клас `container` до відкривального тегу `<footer>`.

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

5 Збережіть файл `index.html` і перевірте сторінку в браузері (рисунок 3.2).

Клас `container` задає максимальну ширину блоку і центрує його в межах сторінки, тому вміст не розтягується безконтрольно на всю ширину вікна. Якщо розмістити такий контейнер усередині елемента `<header>`, фон верхньої частини сторінки може залишатися на всю ширину вікна, сам вміст вирівнюється в охайних межах.

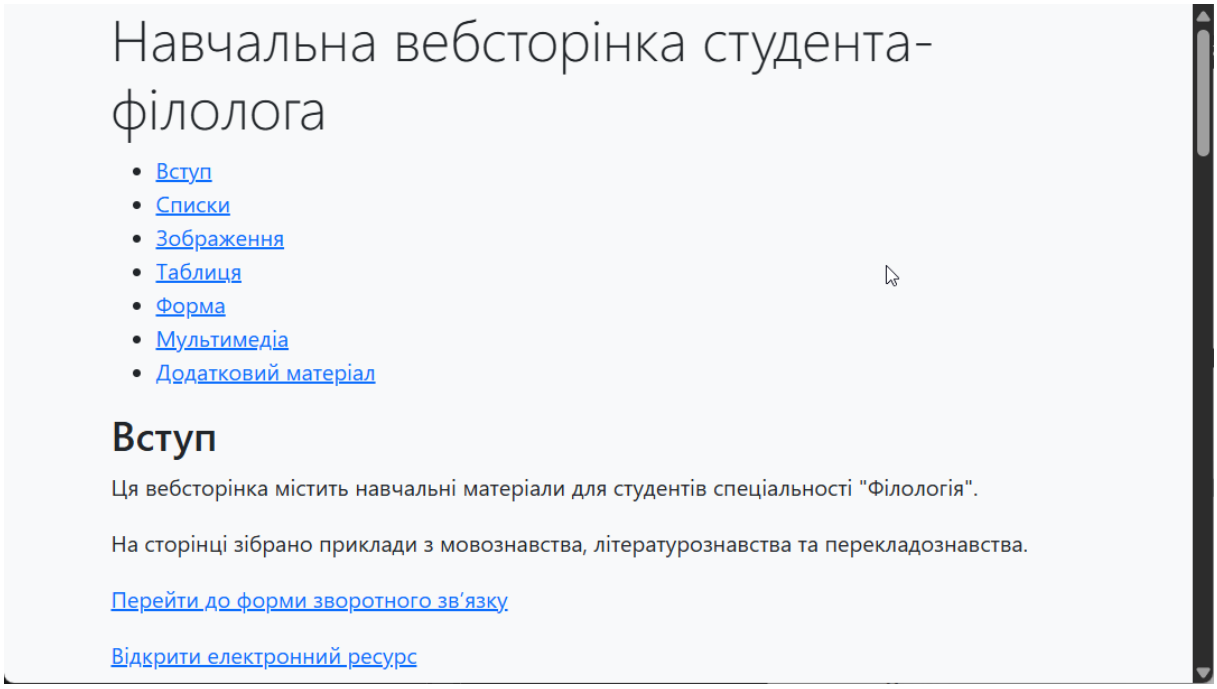

Рисунок 3.2 – Результат використання контейнерів

5 Оформлення текстових елементів засобами Bootstrap

Bootstrap містить готові класи для роботи з розмірами тексту, товщиною накреслення та службовими відступами. Це допомагає покращити вигляд заголовків і абзаців без написання власного CSS.

1 У верхній частині сторінки додайте до відкривального тегу `<header>` класи `bg-primary`, `text-white`, `py-4` і `mb-4`.

```
<header class="bg-primary text-white py-4 mb-4">
```

2 У тому самому блоці додайте до головного заголовка класи `display-5`, `fw-semibold` і `mb-3`, щоб він мав вигляд, як наведено нижче.

```
<h1 class="display-5 fw-semibold mb-3">Навчальна
вебсторінка студента-філолога</h1>
```

3 Додайте класи `bg-white border rounded-4 shadow-sm p-4 mb-4` до відкривального тегу секції `intro`, класи `h3 mb-3` до її заголовка

і клас `lead` до першого абзацу. Після змін ці теги мають виглядати так, як показано нижче.

```
<section id="intro" class="bg-white border rounded-4
shadow-sm p-4 mb-4">
    <h2 class="h3 mb-3">Вступ</h2>
    <p class="lead">Ця вебсторінка містить навчальні
матеріали для студентів спеціальності "Філологія".</p>
```

4 Збережіть файл `index.html` і перегляньте результат у браузері (рисунок 3.3).

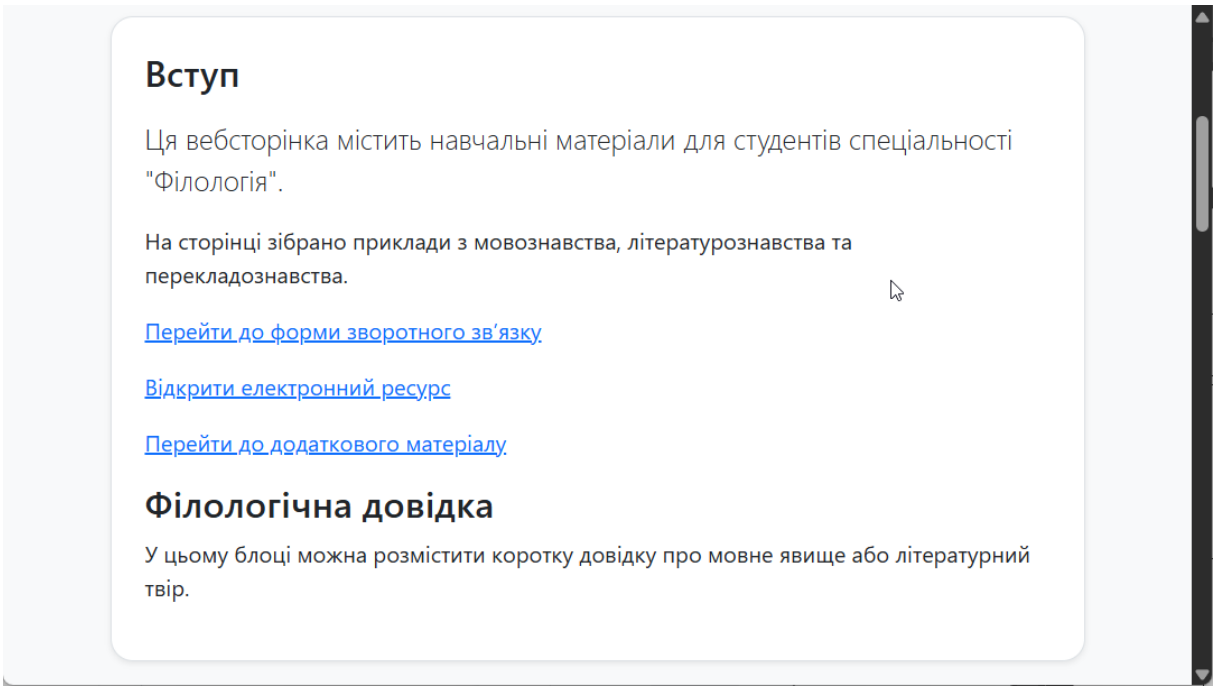

Рисунок 3.3 – Результат оформлення текстових елементів

Класи `bg-primary` і `text-white` задають фон і колір тексту, `py-4` додає вертикальні внутрішні відступи, `mb-4` і `mb-3` задають нижні зовнішні відступи, `fw-semibold` робить накреслення напівжирним. Клас `lead` виділяє вступний абзац збільшеним шрифтом, `bg-white` задає білий фон секції, `border` додає межу, `rounded-4` заокруглює кути, `p-4` створює внутрішні відступи, `shadow-sm` додає легку тінь, `h3` дає змогу взяти готовий розмір і стиль заголовка третього рівня без зміни семантичного тегу `<h2>`.

6 Стилізація кнопок і гіперпосилань за допомогою класів Bootstrap

Bootstrap дає змогу легко перетворювати звичайні гіперпосилання на кнопки з готовим оформленням, що є зручним для важливих дій і переходів між сторінками.

1 Замініть у секції `intro` три абзаци з посиланнями на блок, наведений нижче.

```
<div class="d-flex flex-wrap gap-2 mb-4">
    <a class="btn btn-primary" href="#feedback">Перейти
до форми зворотного зв’язку</a>
    <a class="btn btn-outline-secondary"
href="https://example.com" target="_blank">Відкрити
електронний
        ресурс</a>
    <a class="btn btn-success" href="about.html">Перейти
до додаткового матеріалу</a>
</div>
```

2 Збережіть файл `index.html`, оновіть сторінку в браузері та перевірте, що всі три посилання відображені як кнопки різних типів (рисунок 3.4).

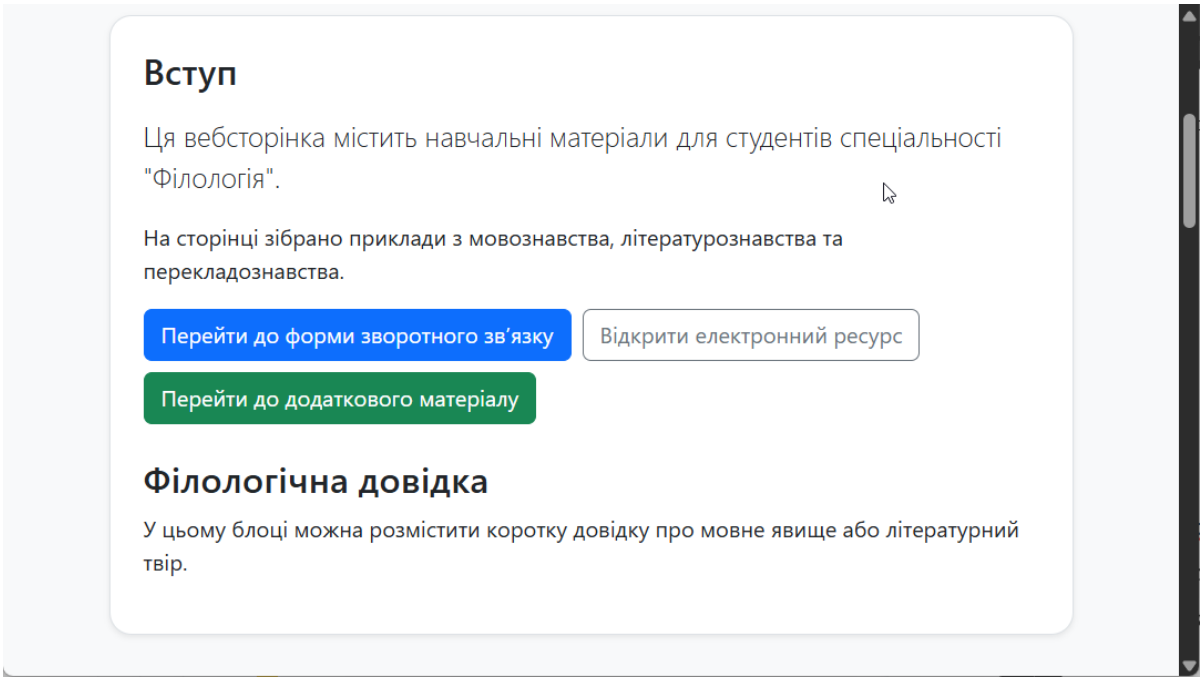

Рисунок 3.4 – Результат стилізації гіперпосилань

Клас `btn` перетворює посилання на кнопку, `btn-primary`, `btn-outline-secondary` і `btn-success` задають три готові варіанти її оформлення: основний, контурний і акцентний. Клас `d-flex` робить контейнер гнучким, `flex-wrap` дає змогу кнопкам бути перенесеними на новий рядок, `gap-2` задає проміжки між ними, `mb-4` залишає відступ після всього блоку кнопок.

7 Оформлення навігаційного меню із застосуванням Bootstrap

У верхній частині сторінки має бути зручне навігаційне меню. Для цього можна використати класи Bootstrap для навігації та службові класи для гнучкого розташування посилань.

1 Додайте до відкривального тегу `<nav>` класи `navbar navbar-dark p-0`, одразу після нього додайте відкривальний тег внутрішнього контейнера з класами `container-fluid px-0`. Початок блоку навігації після змін має виглядати так, як показано нижче.

```
<nav class="navbar navbar-dark p-0">
    <div class="container-fluid px-0">
```

2 Перед закривальним тегом `</nav>` додайте закривальний тег внутрішнього контейнера. Кінець блоку навігації має виглядати так, як показано нижче.

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

3 Додайте клас `navbar-nav flex-row flex-wrap gap-2` до відкривального тегу `<ul>`, до кожного тегу `<li>` у меню додайте клас `nav-item`.

4 Додайте до кожного посилання `<a>` у навігаційному меню класи `nav-link px-3 rounded-pill bg-white bg-opacity-10`. Початок

списку із навігаційними посиланнями після редагування має виглядати так, як показано нижче.

```
<ul class="navbar-nav flex-row flex-wrap gap-2">
    <li class="nav-item">
        <a class="nav-link px-3 rounded-pill bg-white bg-
opacity-10" href="#intro">Вступ</a>
    </li>
    <li class="nav-item">
        <a class="nav-link px-3 rounded-pill bg-white bg-
opacity-10" href="#lists">Списки</a>
    </li>
```

5 Збережіть файл `index.html` і перевірте в браузері, що меню стало компактним, помітним і зручним для переходу між розділами (рисунок 3.5).

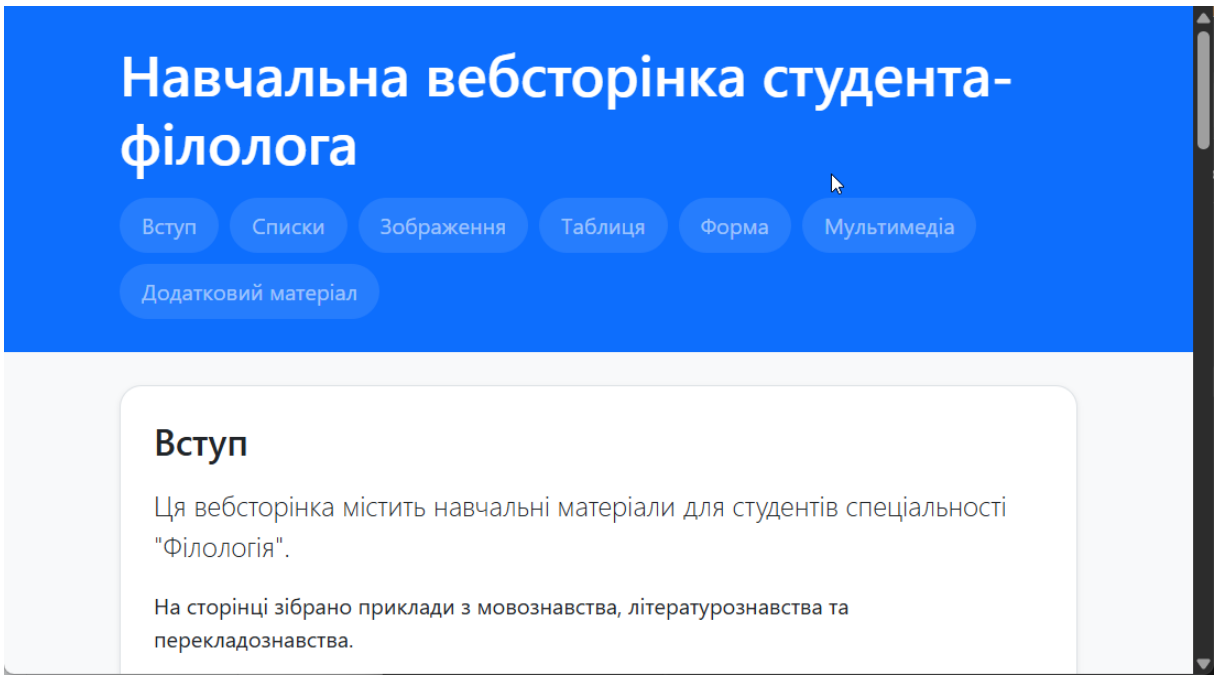

Рисунок 3.5 – Результат оформлення навігаційного меню

У цьому кроці використано компонент `navbar` і пов’язані з ним класи `navbar-nav`, `nav-item`, і `nav-link`, які формують структуру навігаційного меню. Клас `navbar-dark` підлаштовує вигляд меню під темний фон

верхньої частини сторінки, `container-fluid` займає всю доступну ширину всередині навігації, `p-0` і `px-0` прибирають стандартні внутрішні відступи, `flex-row` розташовує пункти меню в один ряд, `flex-wrap` дозволяє їм переноситися, `px-3` додає горизонтальні відступи до посилань, `rounded-pill` робить їхню форму округлою, поєднання `bg-white` і `bg-opacity-10` створює напівпрозорий світлий фон для кожного посилання.

8 Використання системи сітки Bootstrap для розміщення блоків на сторінці

Щоб окремі секції сторінки не були розташовані лише вертикально одна під одною, використаємо сітку Bootstrap. Для цього потрібно поєднати рядки `row` і стовпці `col`.

1 Перед відкривальним тегом `<section id="lists">` додайте фрагмент, поданий нижче.

```
<div class="row g-4 mb-4">
    <div class="col-lg-6">
```

2 Замініть фрагмент від закривального тегу `</section>` секції `lists` перед відкривальним тегом `<section id="gallery">` на такий, що подано нижче.

```
    </section>
</div>

<div class="col-lg-6">
    <section id="gallery">
```

3 Після закривального тегу `</section>` секції `gallery` додайте фрагмент, що подано нижче.

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

4 Перед відкривальним тегом `<section id="data-table">` додайте фрагмент, поданий нижче.

```
<div class="row g-4 mb-4">
    <div class="col-lg-5">
```

5 Замініть фрагмент від закривального тегу `</section>` секції `data-table` перед відкривальним тегом `<section id="feedback">` на такий, що подано нижче.

```
    </section>
</div>

<div class="col-lg-7">
    <section id="feedback">
```

6 Після закривального тегу `</section>` секції `feedback` додайте фрагмент, поданий нижче.

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

7 Збережіть файл `index.html` і перевірте в браузері, що окремі секції розташовані в кілька колонок на широкому екрані (рисунок 3.6).

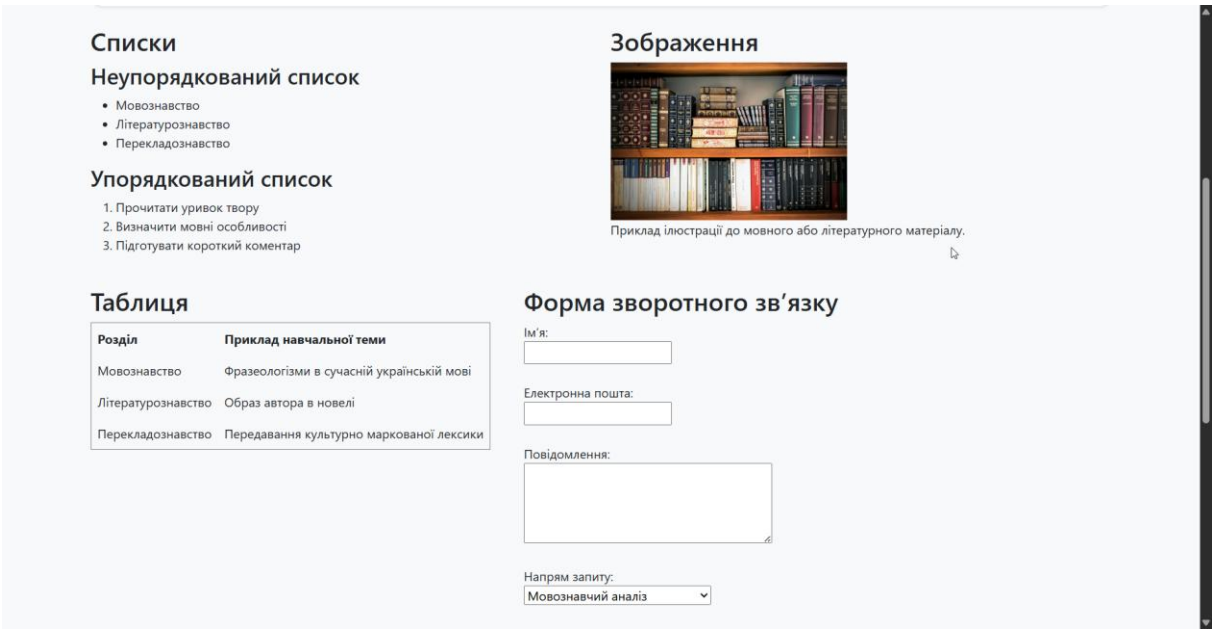

Рисунок 3.6 – Результат використання системи сітки для розміщення блоків на сторінці

Клас `row` створює рядок сітки, класи `col-lg-6`, `col-lg-5` і `col-lg-7` задають ширину колонок, яка починає діяти на великих екранах. Клас `g-4` додає проміжки між колонками та рядками, `mb-4` залишає відступ після всього рядка. Сітка Bootstrap допомагає швидко керувати макетом сторінки і робить розміщення блоків більш гнучким.

9 Оформлення секцій сторінки за допомогою відступів, кольорів і службових класів Bootstrap

1 Додайте до відкривальних тегів секцій `lists`, `gallery`, `data-table` і `feedback` однакові класи `h-100 bg-white border rounded-4 shadow-sm p-4`. Після змін ці теги мають виглядати так, як показано нижче.

```
<section id="lists" class="h-100 bg-white border rounded-
4 shadow-sm p-4">
<section id="gallery" class="h-100 bg-white border
rounded-4 shadow-sm p-4">
<section id="data-table" class="h-100 bg-white border
rounded-4 shadow-sm p-4">
<section id="feedback" class="h-100 bg-white border
rounded-4 shadow-sm p-4">
```

2 Додайте до заголовків другого рівня в цих секціях класи `h3 mb-3`. Після змін відповідні теги мають виглядати так, як показано нижче.

```
<h2 class="h3 mb-3">Списки</h2>
<h2 class="h3 mb-3">Зображення</h2>
<h2 class="h3 mb-3">Таблиця</h2>
<h2 class="h3 mb-3">Форма зворотного зв’язку</h2>
```

3 У секції `lists` додайте до обох заголовків третього рівня клас `h5`. Після змін вони мають виглядати так, як показано нижче.

```
<h3 class="h5">Неупорядкований список</h3>
<h3 class="h5">Упорядкований список</h3>
```

4 Збережіть файл `index.html` і перевірте в браузері, що назви секцій і підзаголовки списків стали візуально узгодженими (рисунок 3.7).

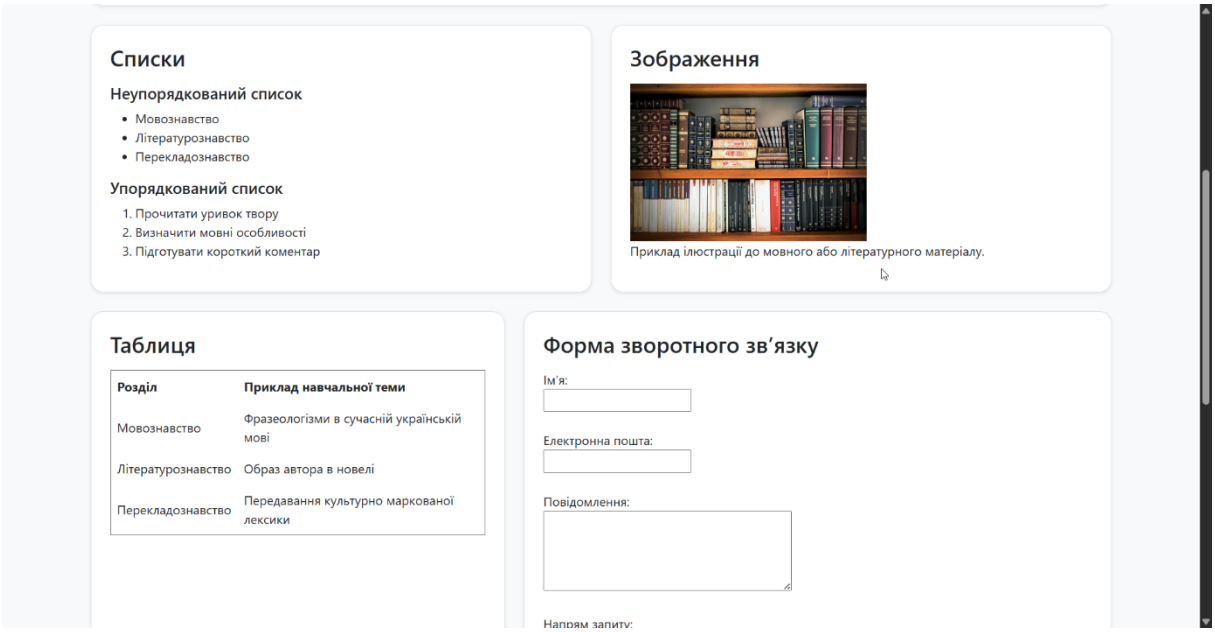

Рисунок 3.7 – Результат оформлення секцій сторінки

Класи `h3` і `h5` використані тут як типографічні класи Bootstrap: вони задають готовий розмір і насиченість тексту без зміни самого рівня заголовка в структурі документа. Клас `mb-3` додає нижній відступ після назви секції, `h-100` розтягує секцію на всю висоту батьківської колонки. Після цього всі тематичні секції матимуть узгоджене оформлення, заголовки різних рівнів будуть подані в одному стилі. Це допомагає сприймати сторінку як єдине ціле.

10 Стилізація зображень і мультимедійних елементів засобами Bootstrap

1 Додайте до тегу зображення у файлі `index.html` класи `img-fluid`, `rounded`, `shadow-sm` і `mb-3`, щоб він мав такий вигляд, як показано нижче.

```
<img src="image.jpg" alt="Навчальна ілюстрація з
філології" class="img-fluid rounded shadow-sm mb-3">
```

2 У секції `media` додайте Bootstrap-класи до відкривального тегу і перебудуйте внутрішній вміст за зразком.

```
<section id="media" class="bg-white border rounded-4
shadow-sm p-4">
    <h2 class="h3 mb-3">Мультимедіа</h2>
    <div class="row g-4 align-items-start">
        <div class="col-lg-7">
            <h3 class="h5">Відео</h3>
            <video class="w-100 rounded shadow-sm mb-3"
controls>
                <source src="video.mp4" type="video/mp4">
                Браузер не підтримує відтворення відео.
            </video>
        </div>
        <div class="col-lg-5">
            <div class="bg-light border rounded-4 p-3 h-
100">
                <h3 class="h5">Аудіо</h3>
                <audio class="w-100" controls>
                    <source src="audio.mp3"
type="audio/mpeg">
                    Браузер не підтримує відтворення
аудіо.
                </audio>
            </div>
        </div>
    </div>
</section>
```

3 Збережіть файл `index.html`, оновіть сторінку в браузері та перевірте, що зображення не виходить за межі секції, відео й аудіо розміщені акуратно (рисунок 3.8).

Клас `img-fluid` забезпечує адаптивне відображення зображення, `rounded` заокруглює його кути, `shadow-sm` додає легку тінь, `mb-3` залишає нижній відступ після зображення або відео. Для мультимедійної секції `w-100` розтягує елемент на ширину колонки, `align-items-start` вирівнює вміст колонок по верхньому краю, `bg-light` задає світліший фон внутрішньому блоку з аудіо, `p-3` додає внутрішні відступи, `h-100` дозволяє цьому блоку займати всю висоту колонки.

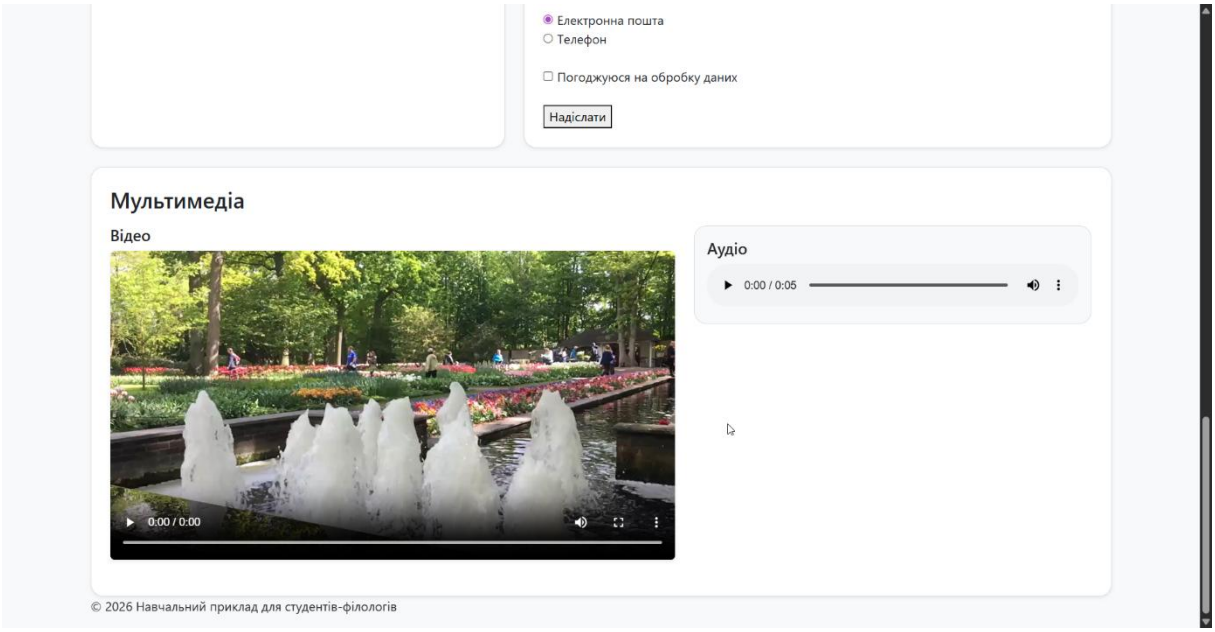

Рисунок 3.8 – Результат стилізації мультимедійних елементів

11 Оформлення таблиці за допомогою класів Bootstrap

Bootstrap містить готові класи для таблиць, завдяки яким можна швидко отримати зрозуміле і читабельне подання даних.

1 Додайте до тегу таблиці у файлі `index.html` класи `table`, `table-striped`, `table-bordered`, `align-middle` і `mb-0`, щоб він мав вигляд, показаний нижче.

```
<table class="table table-striped table-bordered align-
middle mb-0">
```

2 Збережіть файл `index.html` і перевірте сторінку в браузері. Переконайтеся, що таблиця отримала межі, чергування рядків і охайне вирівнювання вмісту (рисунок 3.9).

Клас `table` задає базове оформлення таблиці, `table-striped` додає чергування кольору рядків, `table-bordered` створює межі навколо клітинок, `align-middle` вирівнює вміст клітинок по вертикалі, `mb-0` прибирає нижній зовнішній відступ таблиці всередині секції.

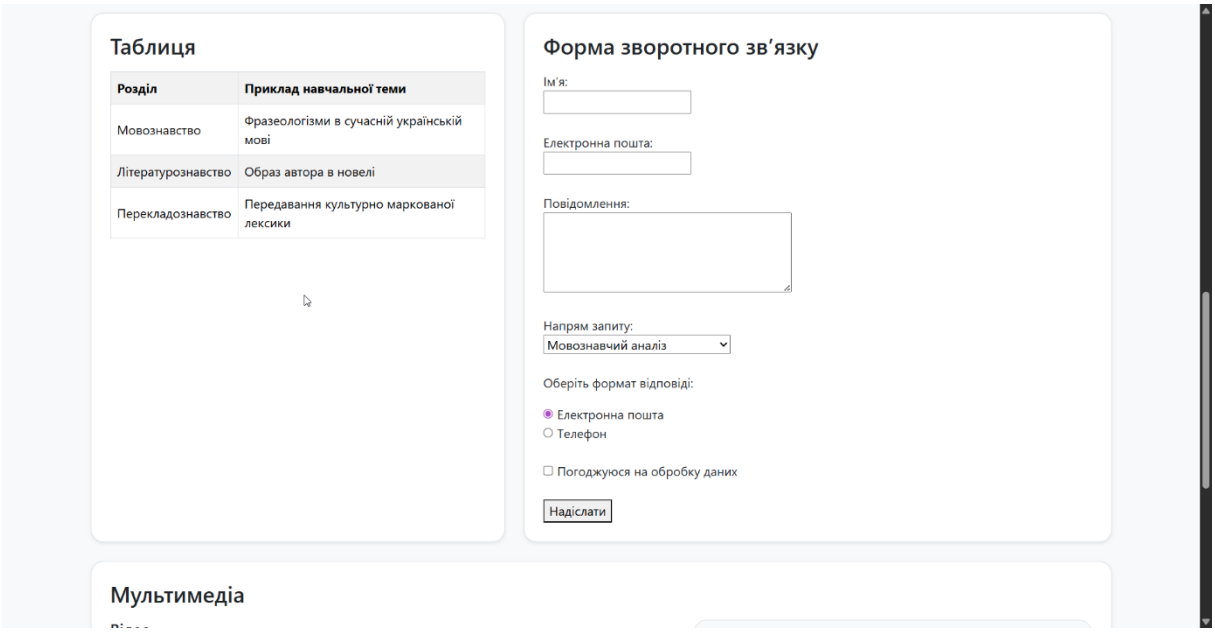

Рисунок 3.9 – Результат оформлення таблиці

12 Оформлення форми введення даних за допомогою Bootstrap

Форма є одним із найпоширеніших елементів вебінтерфейсу, тому її доцільно оформити засобами Bootstrap. Для цього використовують класи `form-label`, `form-control`, `form-select`, `form-check` і класи кнопок.

1 Додайте до відкривального тегу `<form>` у секції `feedback` класи `row g-3`, щоб він мав такий вигляд, як показано нижче.

```
<form class="row g-3" action="#" method="post">
```

2 Замініть фрагмент для поля імені від `<label for="name">Ім'я:</label><br>` до `<input type="text" id="name" name="name" required><br><br>` на такий, що подано нижче.

```
<div class="col-12">
    <label for="name" class="form-label">Ім'я</label>
    <input type="text" class="form-control" id="name"
name="name" required>
</div>
```

3 Замініть фрагмент для поля електронної пошти від `<label for="email">Електронна пошта:</label><br>` до `<input type="email" id="email" name="email" required><br><br>` на такий, що подано нижче.

```
<div class="col-12">
    <label for="email" class="form-label">Електронна
пошта</label>
    <input type="email" class="form-control" id="email"
name="email" required>
</div>
```

4 Замініть фрагмент для повідомлення від `<label for="message">Повідомлення:</label><br>` до `<textarea id="message" name="message" rows="4" cols="40"></textarea><br><br>` на такий, що подано нижче.

```
<div class="col-12">
    <label for="message" class="form-
label">Повідомлення</label>
    <textarea class="form-control" id="message"
name="message" rows="4"></textarea>
</div>
```

5 Замініть фрагмент зі списком вибору теми від `<label for="topic">Напрям запиту:</label><br>` до `</select><br><br>` на такий, що подано нижче.

```
<div class="col-md-6">
    <label for="topic" class="form-label">Напрям
запиту</label>
    <select class="form-select" id="topic" name="topic">
        <option value="linguistics">Мовознавчий
аналіз</option>
        <option value="literature">Літературознавчий
коментар</option>
        <option value="translation">Переклад
тексту</option>
    </select>
</div>
```

6 Замініть фрагмент із перемикачами від абзацу `Оберіть формат відповіді` до другого рядка з міткою `Телефон` на такий, що подано нижче.

```
<div class="col-md-6">
    <p class="form-label mb-2">Оберіть формат
відповіді</p>
    <div class="form-check">
        <input class="form-check-input" type="radio"
id="reply-email" name="reply_format" value="email"
checked>
        <label class="form-check-label" for="reply-
email">Електронна пошта</label>
    </div>
    <div class="form-check">
        <input class="form-check-input" type="radio"
id="reply-phone" name="reply_format" value="phone">
        <label class="form-check-label" for="reply-
phone">Телефон</label>
    </div>
</div>
```

7 Замініть фрагмент із прапорцем згоди та кнопкою надсилання на такий, що подано нижче.

```
<div class="col-12">
    <div class="form-check">
        <input class="form-check-input" type="checkbox"
id="agree" name="agree" required>
        <label class="form-check-label"
for="agree">Погоджуюся на обробку даних</label>
    </div>
</div>

<div class="col-12">
    <button type="submit" class="btn btn-success">
Надіслати</button>
</div>
```

8 Збережіть файл `index.html`, оновіть сторінку в браузері та перевірте, що форма має сучасний вигляд, а поля введення оформлені однаково (рисунок 3.10).

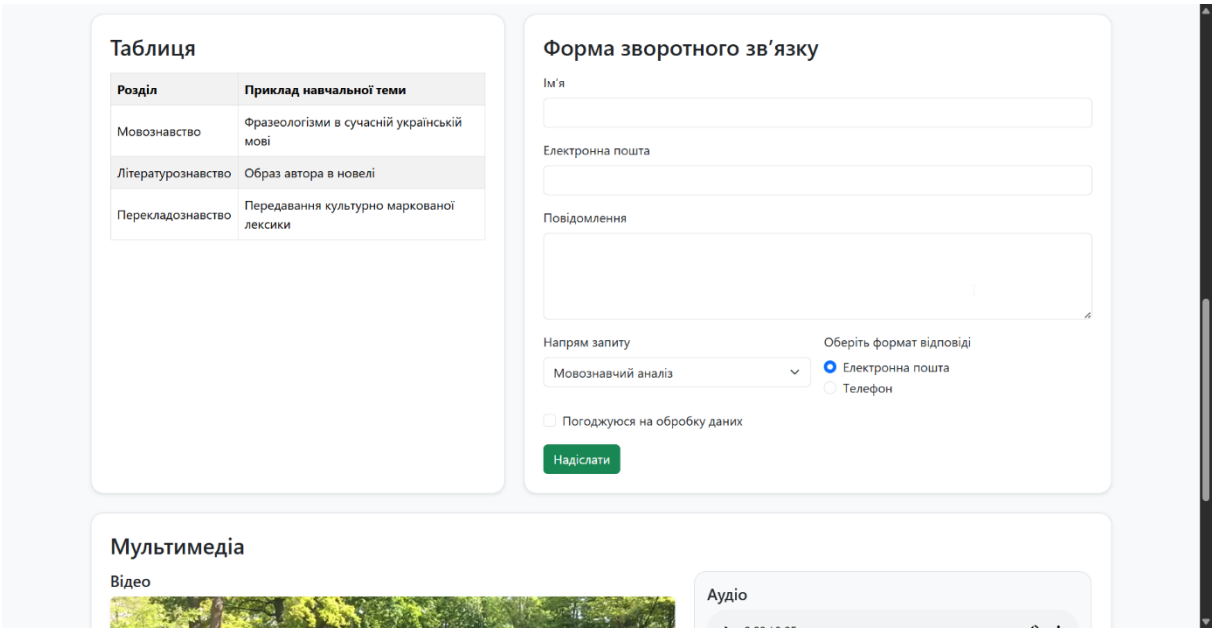

Рисунок 3.10 – Результат оформлення форми введення даних

Клас `row g-3` організує елементи форми в сітку з проміжками, `col-12` займає всю ширину рядка, `col-md-6` ділить простір навпіл, починаючи із середніх екранів. Клас `form-label` оформлює підписи полів, `form-control` і `form-select` надають єдиний вигляд полям введення та списку вибору, `form-check`, `form-check-input` і `form-check-label` призначені для груп радіокнопок і прапорців. Клас `btn-success` додає готове зелене оформлення кнопки надсилання. Bootstrap значно скорочує обсяг ручного оформлення форми і дає змогу зосередитися на структурі та логіці елементів.

13 Використання готових компонентів Bootstrap у структурі сторінки

Однією з переваг Bootstrap є наявність готових компонентів, які можна вставити в уже існуючу HTML-структуру з мінімальними змінами. На цьому етапі доцільно додати хоча б один такий приклад.

1 Додайте до контейнера довідки в секції `intro` класи `alert alert-warning mb-0` і атрибут `role="alert"`, до його внутрішніх елементів додайте відповідні Bootstrap-класи. Після змін цей фрагмент має виглядати так, як показано нижче.

```
<div class="alert alert-warning mb-0" role="alert">
    <h3 class="h5 alert-heading">Філологічна довідка</h3>
    <p class="mb-0">У цьому блоці можна розмістити
коротку довідку про мовне явище або літературний
твір.</p>
</div>
```

2 У секції `lists` додайте клас `h5` до обох заголовків третього рівня, клас `list-group mb-4` до маркованого списку, клас `list-group list-group-numbered` до нумерованого списку, клас `list-group-item` до кожного пункту списку. Після змін фрагменти мають виглядати так, як показано нижче.

```
<h3 class="h5">Неупорядкований список</h3>
<ul class="list-group mb-4">
    <li class="list-group-item">Мовознавство</li>
    <li class="list-group-item">Літературознавство</li>
    <li class="list-group-item">Перекладознавство</li>
</ul>

<h3 class="h5">Упорядкований список</h3>
<ol class="list-group list-group-numbered">
    <li class="list-group-item">Прочитати уривок
твору</li>
    <li class="list-group-item">Визначити мовні
особливості</li>
    <li class="list-group-item">Підготувати короткий
коментар</li>
</ol>
```

3 Збережіть файл `index.html` і перевірте сторінку в браузері. Зверніть увагу, що списки і попереджувальний блок тепер оформлені готовими компонентами Bootstrap (рисунок 3.11).

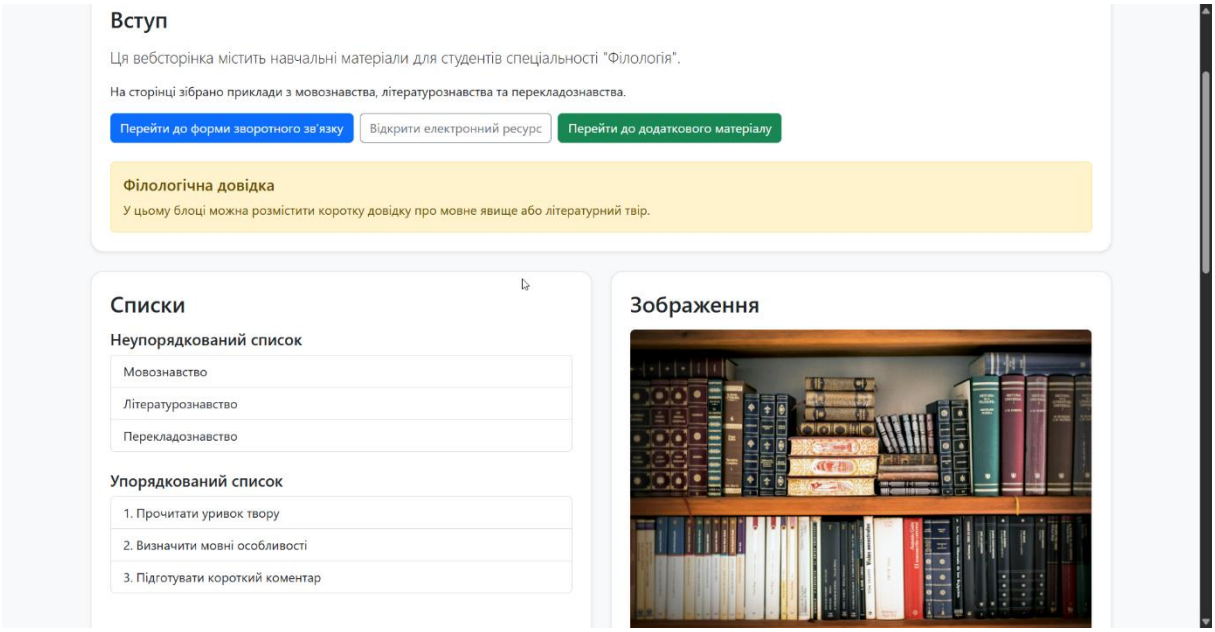

Рисунок 3.11 – Приклад використання готових компонентів

У цьому кроці використано компоненти `alert` і `list-group`. Атрибут `role="alert"` позначає повідомлення як важливий інформаційний блок для допоміжних технологій, `alert-warning` задає попереджувальне оформлення, `alert-heading` стилізує заголовок усередині цього блоку, `mb-0` прибирає зайвий нижній відступ у завершальному абзаці. Для списків `list-group` формує єдиний компонент, `list-group-item` оформлює кожен окремий пункт, `mb-4` залишає відступ після першого списку, `list-group-numbered` автоматично додає нумерацію до впорядкованого списку без ручного прописування маркерів у стилях.

14 Адаптація додаткової HTML-сторінки до спільного використання Bootstrap

Окрім головної сторінки, оформлення потрібно поширити на додатковий документ `about.html`. На цьому етапі потрібно послідовно додати відповідні Bootstrap-класи і до другої сторінки проєкту.

1 Додайте до відкривального тегу `<body>` у файлі `about.html` клас `bg-body-tertiary`.

```
<body class="bg-body-tertiary">
```

2 Додайте до відкривального тегу `<header>` класи `bg-primary text-white py-4 mb-4`. Одразу після нього додайте відкривальний тег контейнера `<div class="container">`, перед закривальним тегом `</header>` додайте закривальний тег `</div>`.

3 Додайте до заголовка `<h1>` класи `display-6 fw-semibold mb-3`.

4 Додайте до тегу `<nav>` класи `navbar navbar-dark p-0`. Одразу після нього додайте відкривальний тег внутрішнього контейнера `<div class="container-fluid px-0">`, перед `</nav>` додайте закривальний тег `</div>`. До тегу `<ul>` додайте класи `navbar-nav flex-row flex-wrap gap-2`, до тегу `<li>` клас `nav-item`, до посилання на головну сторінку класи `nav-link px-3 rounded-pill bg-white bg-opacity-10`.

5 Додайте до відкривального тегу `<main>` класи `container pb-4`, до відкривального тегу `<section>` класи `card border-0 shadow-sm`.

6 Одразу після відкривального тегу `<section class="card border-0 shadow-sm">` додайте відкривальний тег внутрішнього блоку `<div class="card-body p-4">`, перед `</section>` додайте його закривальний тег `</div>`.

7 Додайте до заголовка `<h2>` класи `card-title h4 mb-3`, до абзацу під ним клас `card-text`, після цього абзацу додайте посилання, наведене нижче.

```
<a class="btn btn-outline-primary mt-2"
href="index.html">Повернутися на головну</a>
```

8 Додайте до відкривального тегу `<footer>` класи `container text-center text-secondary py-4`, до абзацу в нижньому колонтитулі клас `mb-0`.

9 Збережіть файл `about.html` і відкрийте сторінку `about.html` у браузері (рисунок 3.12). Перевірте, що вона стилістично узгоджена з головною сторінкою.

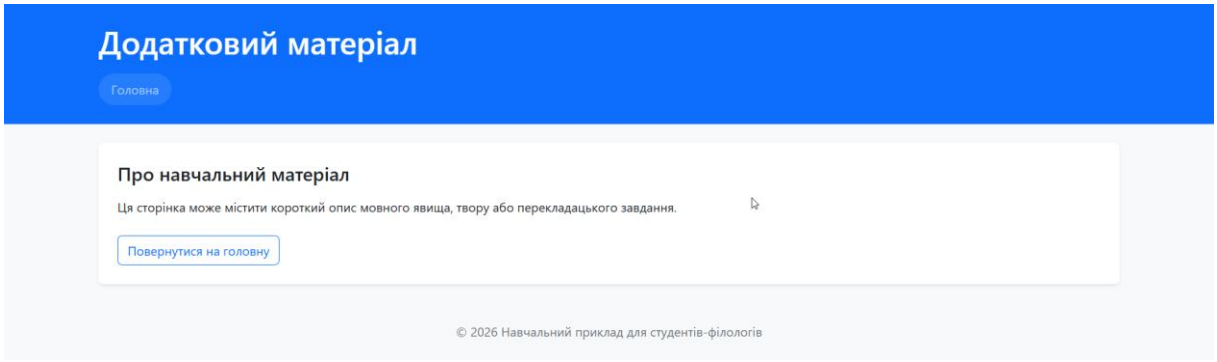

Рисунок 3.12 – Результат адаптації додаткової сторінки

Компонент `card` формує окремий візуально відокремлений блок, `card-body` додає внутрішній контейнер для його вмісту. Класи `card-title` і `card-text` стилізують заголовок і основний текст картки, `border-0` прибирає стандартну рамку, `pb-4` додає нижній внутрішній відступ у головному блоці, `display-6` задає великий стиль заголовка сторінки, `btn-outline-primary` оформлює посилання як контурну кнопку, `mt-2` додає відступ над нею, `text-center` і `text-secondary` у нижньому колонтитулі центрують допоміжний текст і задають йому приглушений колір. У такому

вигляді картка зручна і прийнятна для компактного подання окремого змістового блоку.

15 Підсумкове впорядкування коду HTML-документів

1 Перегляньте файл `index.html` і файл `about.html` і переконайтеся, що всі класи Bootstrap записані без помилок.
2 Вирівняйте відступи в коді, якщо після редагування вони стали нерівномірними.
3 Збережіть обидва HTML-файли і повторно відкрийте сторінки в браузері.
4 Виконайте валідацію обох сторінок онлайн-сервісом W3C Markup Validation Service [7]. Зверніть увагу, що Bootstrap-класи самі по собі не порушують HTML-валідації, тому будь-які виявлені помилки стосуються структури документа.

3.4 Порядок виконання роботи

Індивідуальне завдання виконують на основі HTML-сторінки про структурний підрозділ організації, створеної в першій роботі. На цьому етапі зміст сторінок збережений, самі сторінки адаптовані до використання Bootstrap як готового засобу оформлення інтерфейсу.

1 За основу взяти файли `index.html` і `about.html`, підготовлені в першій роботі.
2 Bootstrap підключити до обох HTML-документів через CDN і перевірити, що сторінки коректно отримують стилі фреймворку. Після цього наявну HTML-структуру послідовно доповнювати Bootstrap-класами та потрібними локальними обгортками.
3 До сторінки додати Bootstrap-класи для контейнерів, заголовків, кнопок, навігації, таблиці, форми, зображень і службових блоків. Зміни в HTML мають бути виправданими, мінімальними та

спрямованими на використання компонентів і службових класів Bootstrap. Потрібно використати хоча б один приклад сітки Bootstrap для компонування вмісту сторінки.

4 Додаткову сторінку `about.html` оформити в тій самій логіці, що і головну. Зберегти єдиний вигляд верхньої частини сторінки, навігації, основного змісту та нижнього колонтитула, щоб обидва документи були сприймані як частини одного проєкту.

5 Сторінки перевірити в браузері, переконатися, що всі елементи коректно відображені після підключення Bootstrap, і виконати валідацію HTML-коду. Окремо звернути увагу на навігацію, форми, таблицю, кнопки, мультимедійні елементи і відображення сторінок на різній ширині вікна.

3.5 Зміст звіту

За результатами виконання індивідуального завдання підготувати звіт у форматі PDF, який має містити:

1 Титульний аркуш із зазначенням дисципліни, номера лабораторної роботи, прізвища та імені здобувача, номера групи.

2 Номер варіанта і посилання на вебсторінку структурного підрозділу організації, яка була джерелом змісту в першій роботі.

3 Короткий опис виконаних дій.

4 Скриншоти сторінок `index.html` та `about.html` у браузері.

5 Стислий перелік Bootstrap-компонентів і службових класів, використаних у роботі.

6 Результат перевірки HTML-коду валідатором W3C.

7 Висновки про виконану роботу.

Разом зі звітом надати файли `index.html`, `about.html` і використані медіафайли.

3.6 Варіанти вихідних даних

Варіант завдання відповідає варіанту, вибраному в лабораторній роботі 1.

Контрольні запитання

1 Що таке Bootstrap і яке його призначення для веброзроблення?
2 Яка принципова різниця між оформленням сторінки через власний CSS-файл і засобами Bootstrap?
3 Що таке службові класи Bootstrap? Наведіть три приклади та поясніть їхню дію.
4 Що таке компонентні класи Bootstrap? Наведіть два приклади готових компонентів.
5 Яке призначення класу `container` у Bootstrap і як він впливає на ширину вмісту?
6 Як працює система сітки Bootstrap? Які три основні елементи вона використовує?
7 Що означають позначення `col-md-6` та `col-lg-4` у системі сітки?
8 Яке призначення класу `row` і що станеться, якщо розмістити колонки без нього?
9 Яке призначення класів `bg-primary` та `text-white`? Які ще кольорові варіанти пропонує Bootstrap?
10 Що таке компонент `navbar` у Bootstrap і які класи необхідні для його базової структури?
11 Яке призначення класу `img-fluid` і чому його рекомендують додавати до зображень?
12 Що таке компонент `alert` у Bootstrap і яке призначення атрибута `role="alert"`?

13 Яке призначення класів `form-control` і `form-label` для оформлення форми?

14 Що означає клас `d-flex` і для чого його використовують разом із `flex-wrap` і `gap`?

15 Чим підключення Bootstrap через CDN відрізняється від локального підключення і які переваги та обмеження має кожен підхід?

СПИСОК ЛІТЕРАТУРИ

1 OpenAI. ChatGPT. URL: https://openai.com/chatgpt (дата звернення: 23.03.2026).

2 OpenAI. Codex. URL: https://openai.com/codex (дата звернення: 23.03.2026).

3 Anthropic. Claude. URL: https://claude.ai (дата звернення: 30.04.2026).

4 MDN Web Docs. HTML: HyperText Markup Language. URL: https://developer.mozilla.org/en-US/docs/Web/HTML (дата звернення: 23.03.2026).

5 W3Schools. HTML Tutorial. URL: https://www.w3schools.com/html (дата звернення: 23.03.2026).

6 Visual Studio Code. URL: https://code.visualstudio.com (дата звернення: 23.03.2026).

7 W3C Markup Validation Service. URL: https://validator.w3.org (дата звернення: 23.03.2026).

8 Іванюк О. І. Lab 1 HTML data.xlsx. URL: https://github.com/oleksa-iv/web-tech-ed-course/blob/master/lab_1_HTML/lab_1_data.xlsx (дата звернення: 23.03.2026).

9 MDN Web Docs. CSS: Cascading Style Sheets. URL: https://developer.mozilla.org/en-US/docs/Web/CSS (дата звернення: 23.03.2026).

10 W3Schools. CSS Tutorial. URL: https://www.w3schools.com/css (дата звернення: 23.03.2026).

11 W3C CSS Validation Service. URL: https://jigsaw.w3.org/css-validator (дата звернення: 23.03.2026).

12 Іванюк О. І. Lab 2 CSS data.xlsx. URL: https://github.com/oleksa-iv/web-tech-ed-course/blob/master/lab_2_CSS/lab_2_data.xlsx (дата звернення: 23.03.2026).

13 Bootstrap. Get started with Bootstrap. URL: https://getbootstrap.com/docs/5.3/getting-started/introduction (дата звернення: 23.03.2026).
14 Bootstrap. Examples. URL: https://getbootstrap.com/docs/5.3/examples (дата звернення: 23.03.2026).
15 W3Schools. Bootstrap 5 Tutorial. URL: https://www.w3schools.com/bootstrap5 (дата звернення: 23.03.2026).

МЕТОДИЧНІ ВКАЗІВКИ

для виконання лабораторних робіт
з освітнього компонента
«WEB-ТЕХНОЛОГІЇ НАВЧАННЯ»

Частина 1

Відповідальний за випуск Іванюк О. І.

Редактор Ібрагімова Н. В.

Підписано до друку 19.06.2026 р.
Умовн. друк. арк. 5,25. Тираж . Замовлення № .
Видавець та виготовлювач Український державний університет
залізничного транспорту,
61050, Харків-50, майдан Фейєрбаха,7.
Свідоцтво суб'єкта видавничої справи ДК № 6100 від 21.03.2018 р.