**МІНІСТЕРСТВО ОСВІТИ І НАУКИ  УКРАЇНИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"**

**ІКНІ**

Кафедра **ПЗ**

****

**ЗВІТ**

До лабораторної роботи №1

**З дисципліни:** «*Програмування в Інтернет*»

**На тему:** *“Структура DOM та методи доступу до вузлів дерева“*

**Лектор:**

асистент каф.ПЗ

Степанов Д.С.

**Виконав:**

студент групи ПЗ-22

Курман В.В.

**Прийняла:**

старша викладачка кафедри ПЗ

Грицай О. Д.

« \_\_\_\_ » \_\_\_\_\_\_\_\_  2025 р.

∑= \_\_\_\_                              .

Львів – 2025

**Тема роботи:** Структура DOM та методи доступу до вузлів дерева.

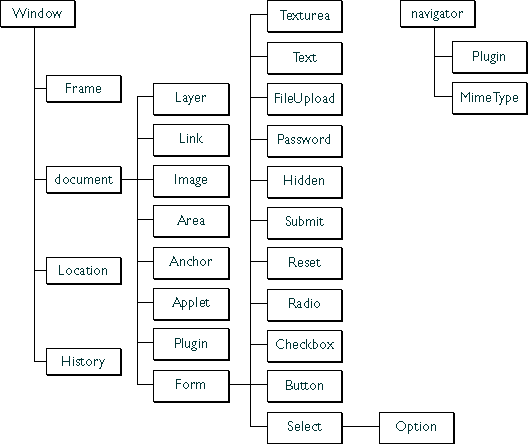
**Мета роботи:** Ознайомитись з ієрархічною структурою об’єктів JavaScript та об’єктами документа і браузера.

**Теоретичні відомості**

## *Модель DOM*

web-сторінка може мати вигляд дерева, вузли якого є об’єктами, до властивостей яких доступаються операторами мови програмування Javascript.

Модель DOM (Document Object Model, об’єктна модель документа) містить низку стандартних глобальних об’єктів. Зокрема, це window, navigator, document, screen, history, location. Ієрархію основних об’єктів у моделі DOM подано   
на рис. 1.2.



*Рис. 1.2. Дерево моделі DOM об’єкта window браузера*

Перерахуємо глобальні об’єкти та їх призначення, а також розглянемо методи та властивості глобальних об’єктів DOM:

* [window](http://www.w3schools.com/htmldom/dom_obj_window.asp) – вікно браузера, найвищий об’єкт ієрархії DOM. Усі глобальні змінні стають частинами об’єкта window. Його методи такі:

[alert](http://www.w3schools.com/htmldom/met_win_alert.asp), [blur](http://www.w3schools.com/htmldom/met_win_blur.asp), [clearInterval](http://www.w3schools.com/htmldom/met_win_clearInterval.asp), [clearTimeout](http://www.w3schools.com/htmldom/met_win_clearTimeout.asp), [close](http://www.w3schools.com/htmldom/met_win_close.asp), [confirm](http://www.w3schools.com/htmldom/met_win_confirm.asp), [focus](http://www.w3schools.com/htmldom/met_win_focus.asp), [moveBy](http://www.w3schools.com/htmldom/met_win_moveBy.asp), [moveTo](http://www.w3schools.com/htmldom/met_win_moveTo.asp), [open](http://www.w3schools.com/htmldom/met_win_open.asp), [print](http://www.w3schools.com/htmldom/met_win_print.asp), [prompt](http://www.w3schools.com/htmldom/met_win_prompt.asp), [resizeBy](http://www.w3schools.com/htmldom/met_win_resizeBy.asp), [resizeTo](http://www.w3schools.com/htmldom/met_win_resizeTo.asp), [scrollBy](http://www.w3schools.com/htmldom/met_win_scrollBy.asp), [scrollTo](http://www.w3schools.com/htmldom/met_win_scrollTo.asp), [setInterval](http://www.w3schools.com/htmldom/met_win_setInterval.asp), [setTimeout](http://www.w3schools.com/htmldom/met_win_setTimeout.asp)

Властивості: [document](http://www.w3schools.com/htmldom/dom_obj_document.asp), [history](http://www.w3schools.com/htmldom/dom_obj_history.asp), [location](http://www.w3schools.com/htmldom/dom_obj_location.asp), [name](http://www.w3schools.com/htmldom/prop_win_name.asp)

* [navigator](http://www.w3schools.com/htmldom/dom_obj_navigator.asp) – інформація про браузер.

Властивості: [appName](http://www.w3schools.com/htmldom/prop_nav_appname.asp), [appVersion](http://www.w3schools.com/htmldom/prop_nav_appversion.asp), [browserLanguage](http://www.w3schools.com/htmldom/prop_nav_browserlanguage.asp), [cookieEnabled](http://www.w3schools.com/htmldom/prop_nav_cookieenabled.asp), [platform](http://www.w3schools.com/htmldom/prop_nav_platform.asp), [userAgent](http://www.w3schools.com/htmldom/prop_nav_useragent.asp)

* [screen](http://www.w3schools.com/htmldom/dom_obj_screen.asp) – інформація про екран, який використовує браузер.

Властивості: [availHeight](http://www.w3schools.com/htmldom/prop_screen_availheight.asp), [availWidth](http://www.w3schools.com/htmldom/prop_screen_availwidth.asp), [colorDepth](http://www.w3schools.com/htmldom/prop_screen_colordepth.asp), [height](http://www.w3schools.com/htmldom/prop_screen_height.asp), [pixelDepth](http://www.w3schools.com/htmldom/prop_screen_pixeldepth.asp), [width](http://www.w3schools.com/htmldom/prop_screen_width.asp)

* [history](http://www.w3schools.com/htmldom/dom_obj_history.asp) – список викликаних сторінок.

Властивості: [length](http://www.w3schools.com/htmldom/prop_his_length.asp)

Методи: [back](http://www.w3schools.com/htmldom/met_his_back.asp), [forward](http://www.w3schools.com/htmldom/met_his_forward.asp), [go](http://www.w3schools.com/htmldom/met_his_go.asp)

* [location](http://www.w3schools.com/htmldom/dom_obj_location.asp) – URL поточної web-сторінки.

Властивості: [host](http://www.w3schools.com/htmldom/prop_loc_host.asp), [hostname](http://www.w3schools.com/htmldom/prop_loc_hostname.asp), [href](http://www.w3schools.com/htmldom/prop_loc_href.asp), [pathname](http://www.w3schools.com/htmldom/prop_loc_pathname.asp), [port](http://www.w3schools.com/htmldom/prop_loc_port.asp), [protocol](http://www.w3schools.com/htmldom/prop_loc_protocol.asp), [search](http://www.w3schools.com/htmldom/prop_loc_search.asp)

Методи: [assign](http://www.w3schools.com/htmldom/met_loc_assign.asp), [reload](http://www.w3schools.com/htmldom/met_loc_reload.asp), [replace](http://www.w3schools.com/htmldom/met_loc_replace.asp)

* [document](http://www.w3schools.com/htmldom/dom_obj_document.asp) – поточна Web-сторінка.

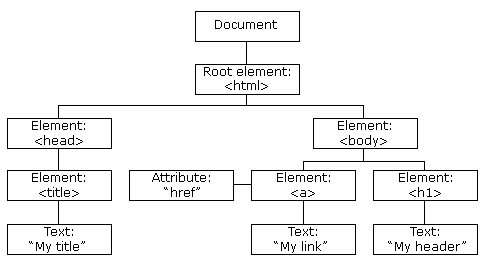
Властивості: [anchors](http://www.w3schools.com/htmldom/coll_doc_anchors.asp), body, [cookie](http://www.w3schools.com/htmldom/prop_doc_cookie.asp), [domain](http://www.w3schools.com/htmldom/prop_doc_domain.asp), [forms](http://www.w3schools.com/htmldom/coll_doc_forms.asp), [images](http://www.w3schools.com/htmldom/coll_doc_images.asp), [links](http://www.w3schools.com/htmldom/coll_doc_links.asp), [referrer](http://www.w3schools.com/htmldom/prop_doc_referrer.asp), [title](http://www.w3schools.com/htmldom/prop_doc_title.asp), [URL](http://www.w3schools.com/htmldom/prop_doc_url.asp)

Методи: [close](http://www.w3schools.com/htmldom/met_doc_close.asp), [getElementById](http://www.w3schools.com/htmldom/met_doc_getelementbyid.asp), [getElementsByName](http://www.w3schools.com/htmldom/met_doc_getelementsbyname.asp), [open](http://www.w3schools.com/htmldom/met_doc_open.asp), [write](http://www.w3schools.com/htmldom/met_doc_write.asp), [writeln](http://www.w3schools.com/htmldom/met_doc_writeln.asp), [getElementsByTagName](http://www.w3schools.com/htmldom/met_doc_getelementsbytagname.asp)

Приклад ієрархічного дерева DOM для об’єкта document наведено на рис. 1.3.

За змістом вузли дерева поділяють на три типи:

* **елементів** (HTML теги) – породжені вузли або атрибути;
* **тексту**  (текст або блоки) – не мають породжених вузлів чи атрибутів;
* атрибутів (пари атрибут/значення) – не мають породжених вузлів чи атрибутів.

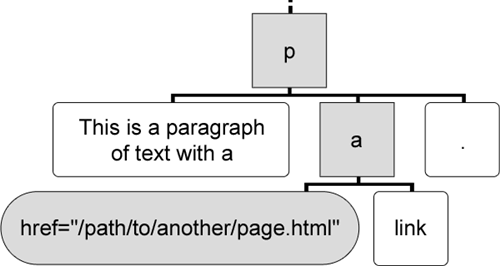


*Рис. 1.3. Дерево об’єкта document у моделі DOM*

Наведемо фрагмент сторінки і відповідне йому дерево   
(рис. 1.4)

<p>This is a paragraph of text with a

<a href="/path/to/another/page.html">link inside </a>.



*Рис. 1.4. Фрагмент дерева з вузлом параграфа*

**Типи вузлів та методи доступу до них**

За місцем розташування на дереві виділено такі типи вузлів: parent – батьківський, child – породжений, sibling – сусідній, firstchild – перший породжений, lastchild – останній породжений.

Назви вузлів є складовими назвами властивостей вузлів та параметрів методів, які ними оперують (ліквідують, модифікують, додають тощо). Вузли DOM-дерева мають такі властивості:

* firstChild, lastChild – початковий та кінцевий породжені вузли;
* childNodes – масив породжених вузлів;
* nextSibling, previousSibling – сусідні вузли (по одному), що мають одного батька;
* parentNode – елемент, що породив цей вузол.

Мова програмування дає змогу змінювати склад вузлів, вигляд та зміст web-сторінки. На моделі DOM програміст має можливість виконувати операції двох типів: на рівні вузлів для зміни структури дерева додаванням або відніманням вузла та заміни вузла; на рівні вузла для зміни змісту елемента. Зокрема, для першого типу операцій застосовуються методи:

* [appendChild](http://www.w3schools.com/dom/met_node_appendchild.asp)(node): розміщає вузол у кінці списку породжених вузлів до заданого;
* [insertBefore](http://www.w3schools.com/dom/met_node_insertbefore.asp)(newChild, oldChild): розміщає новий вузол перед вузлом oldChild у списку породжених вузлів до заданого;
* [removeChild](http://www.w3schools.com/dom/met_node_removechild.asp)(node): знищує наведений вузол зі списку породжених вузлів до заданого;
* [replaceChild](http://www.w3schools.com/dom/met_node_replacechild.asp)(newChild, oldChild): заміняє пород­жений на новий вузол.

Властивості набувають значення

* + nodeType: 1 – елемент, 2 – атрибут, 3 – текст, 4 – коментарі, 9 – документ.
  + nodeName: повертає версію тега, наприклад, "div" чи "article". Текстові вузли мають назву "#text" . Назва вузла document є "#document" ,
  + nodeValue: текст вузла або значення атрибута.

Доступ до вузлів DOM за тегами чи ідентифікаторами здійснюється методами:

* document.getElementById("id") – елемент;
* element.getElementsByTagName("tag") – усі по­роджені вузли;

document.createElement("tag") - команда створює новий порожній вузол для подання елемента певного типу. Рис.1. Загальна схема ієрархічної структури об’єктів.

**Опрацювання подій**

Події та обробники подій є дуже важливою частиною у програмування на JavaScript. ***Події*** (*Events*), головним чином, ініціюються тими або іншими діями користувача. Події – це дії, які відбуваються, внаслідок того, що робить користувач. Наприклад, якщо користувач клацає по деякій кнопці, відбувається подія Click. Якщо миша перетинає яке-небудь посилання - відбувається подія MouseOver. Існує певний набір подій, які розпізнає той чи інший броузер.

Ми можемо примусити нашу JavaScript-програму реагувати на деякі з них. І це може бути виконано за допомогою спеціальних програм ***обробки подій***. Так, в результаті клацання по кнопці може створюватися випадаюче вікно. Це означає, що створення вікна повинно бути реакцією на подію Click. Програма - обробник подій, яку ми повинні використати в даному випадку, називається onClick. І вона повідомляє комп'ютер, що потрібно робити, якщо відбудеться дана подія.

Обробник подій записується в документ як атрибут тега HTML, до якого ви приписуєте код JavaScript. Наприклад, ви створили ***функцію*** JavaScript, і назвали її compute. Ви можете примусити броузер виконувати цю функцію, коли користувач натискає на кнопку, в якій до onClick приписаний обробник результату кнопки:

INPUT TYPE="button" VALUE="Calculate" onClick="compute(this.form)"

Ви можете поміщати будь-які твердження JavaScriptусередині кавичок onClick. Ці твердження будуть виконані, коли користувач натискатиме на кнопку. Якщо Ви хочете включити більш ніж одне твердження, то окремі твердження записуються через крапку з комою (;).

Взагалі, це – непогана ідея визначати ***функцію*** для обробників подій тому що:

* це робить ваш код мобільним, оскільки ви можете використовувати ту ж саму функцію в багатьох різних місцях.
* це робить ваші твердження більш легкими для читання.

В даному прикладі використовується this.form, щоб звернутися до поточної форми. Ключове слово звертається до об'єкту (об'єкту кнопки у вищезазначеному прикладі). Потім конструкція this.form звертається до форми, що містить кнопку. Далі – обробник події onClick робить запит до функції compute(), з поточною формою this.form, як параметр функції.

Події звертаються до тегів HTML таким чином:

* події **Focus, Blur, Change**: text fields, textareas, і selections;
* подія **Click**: buttons, radio buttons, checkboxes, submit buttons, reset buttons, links;
* подія **Select**: text fields, textareas;
* подія **MouseOver**: links.

Ви можете використовувати в скрипті безліч різних типів функцій обробки подій. Щоб дізнатись про усі існуючі обробники подій, звертайтеся до відповідного довідника. Наведемо лише деякі з них:

* **onLoad** - виконання скрипта або функції при завантаженні;
* **onChange** - породжується при зміні значення елемента форми;
* **onClick** - породжується при виборі об'єкту (button, checkbox і т.п.);
* **onSelect** - породжується при виборі текстового об'єкту (text, textarea);
* **onSubmit** - при натисненні на кнопку Submit;
* **onUnload** - при переході до іншої сторінки.

**Завдання**

1. Розробити web-сторінку згідно [макета](https://cacoo.com/diagrams/ZvVhYS3UpG5PdbBy/EDE3A) (wireframe).

2. Сторінка повинна відповідати наступним [вимогам](https://docs.google.com/document/d/1Lqwpmc2RlPVP-4p0pOTFw4pHGf_1tCLdpdMZKlAMBZQ/edit?tab=t.0).

**Виконання**

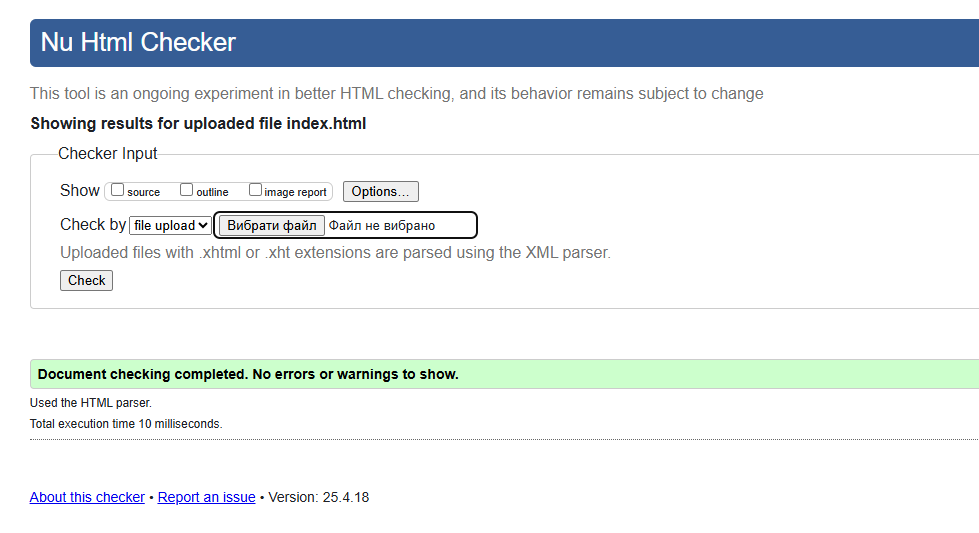
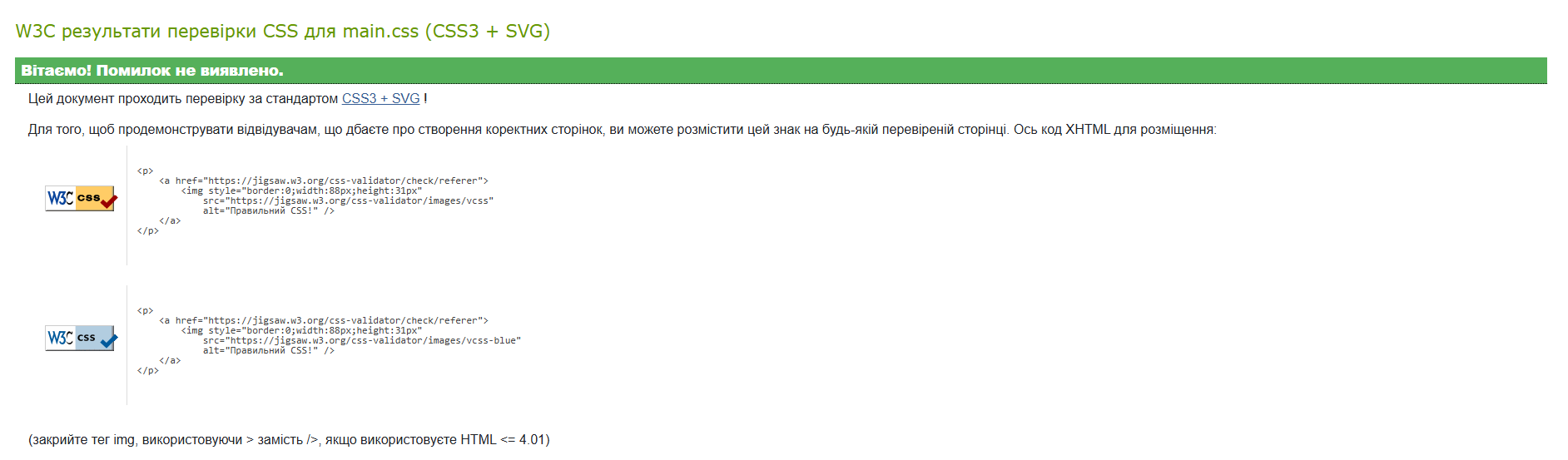


Рис. 1. Результат валідації сайту на <https://validator.w3.org/>



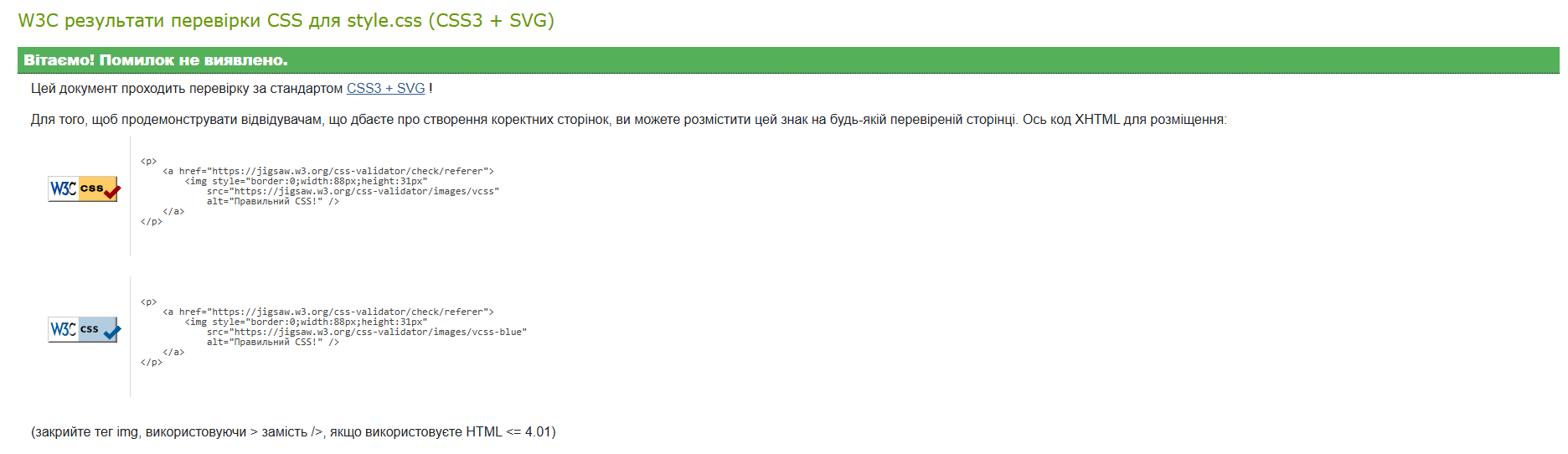


Рис. 2-3. Результат валідації сайту на <https://jigsaw.w3.org/css-validator/>



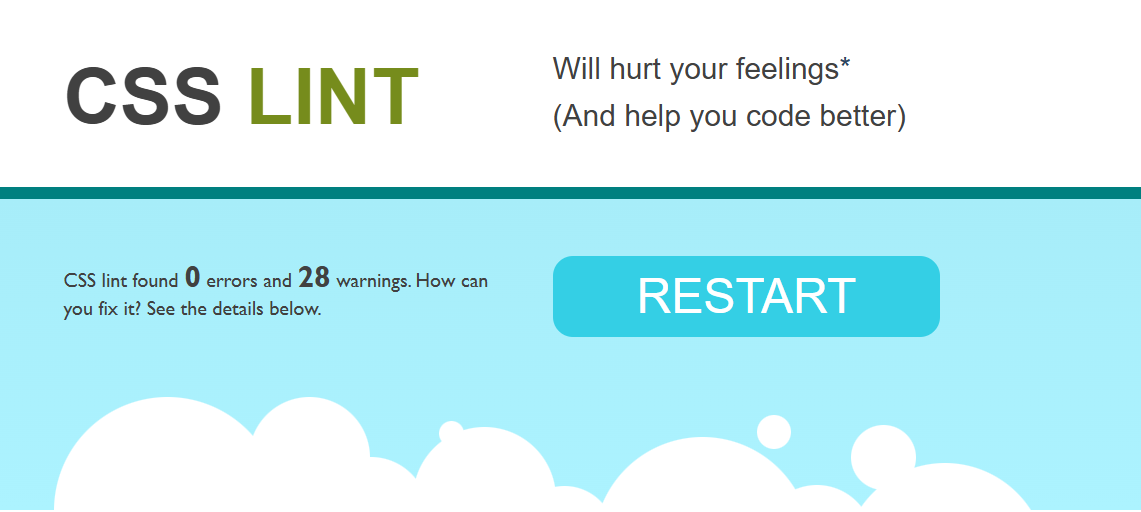
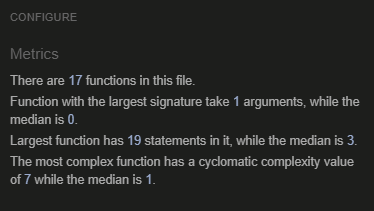


Рис. 4-5. Результат валідації сайту на <https://csslint.net/>



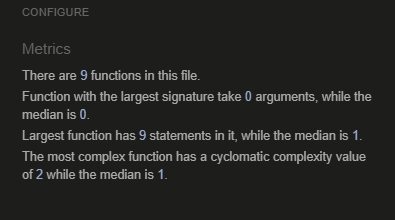


Рис. 6-7. Результат валідації сайту на <https://jshint.com/>

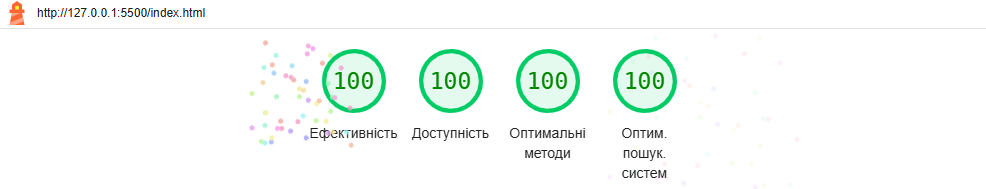


Рис. 8. Результат валідації сайту на Lighthouse

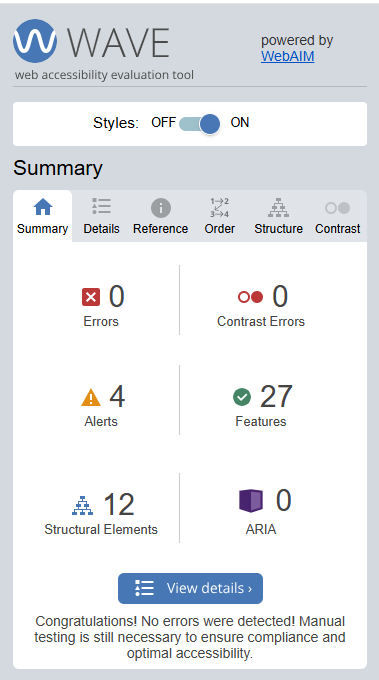
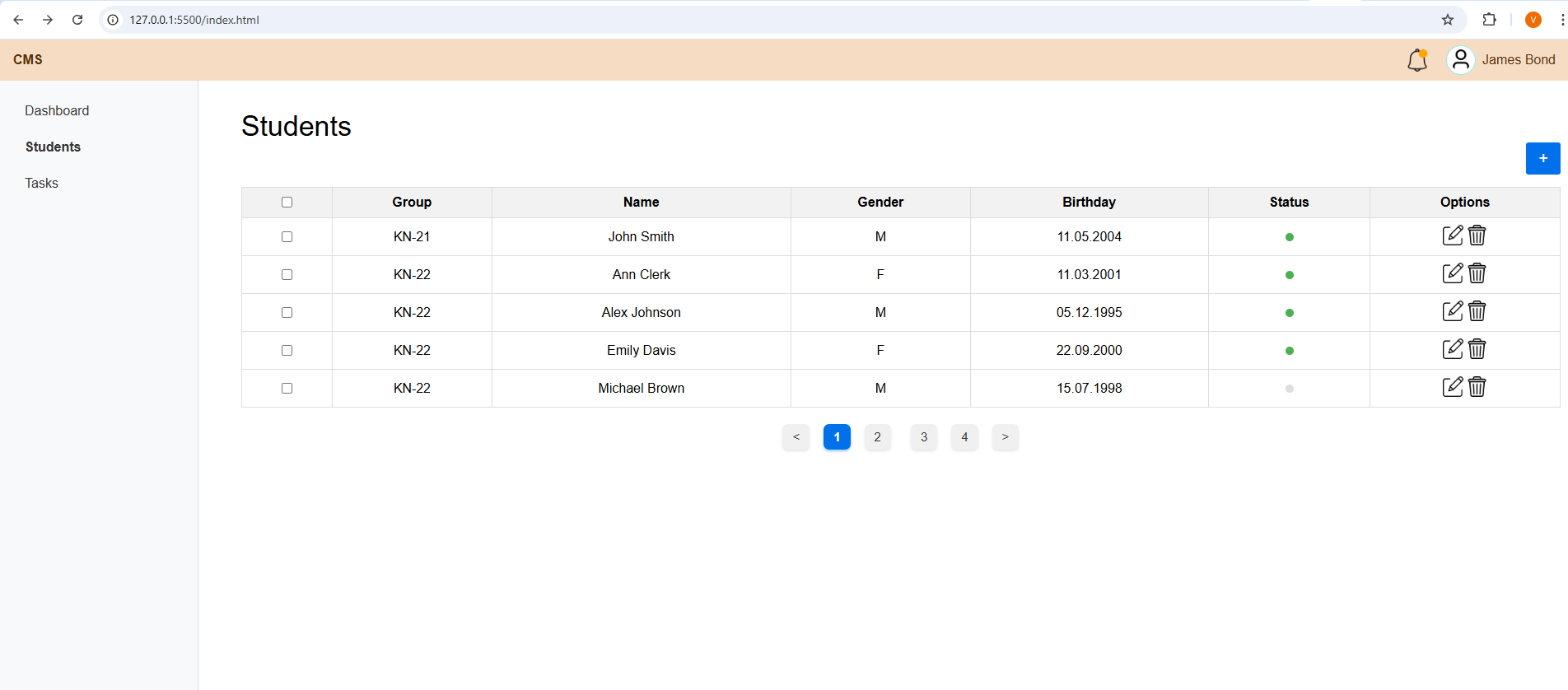
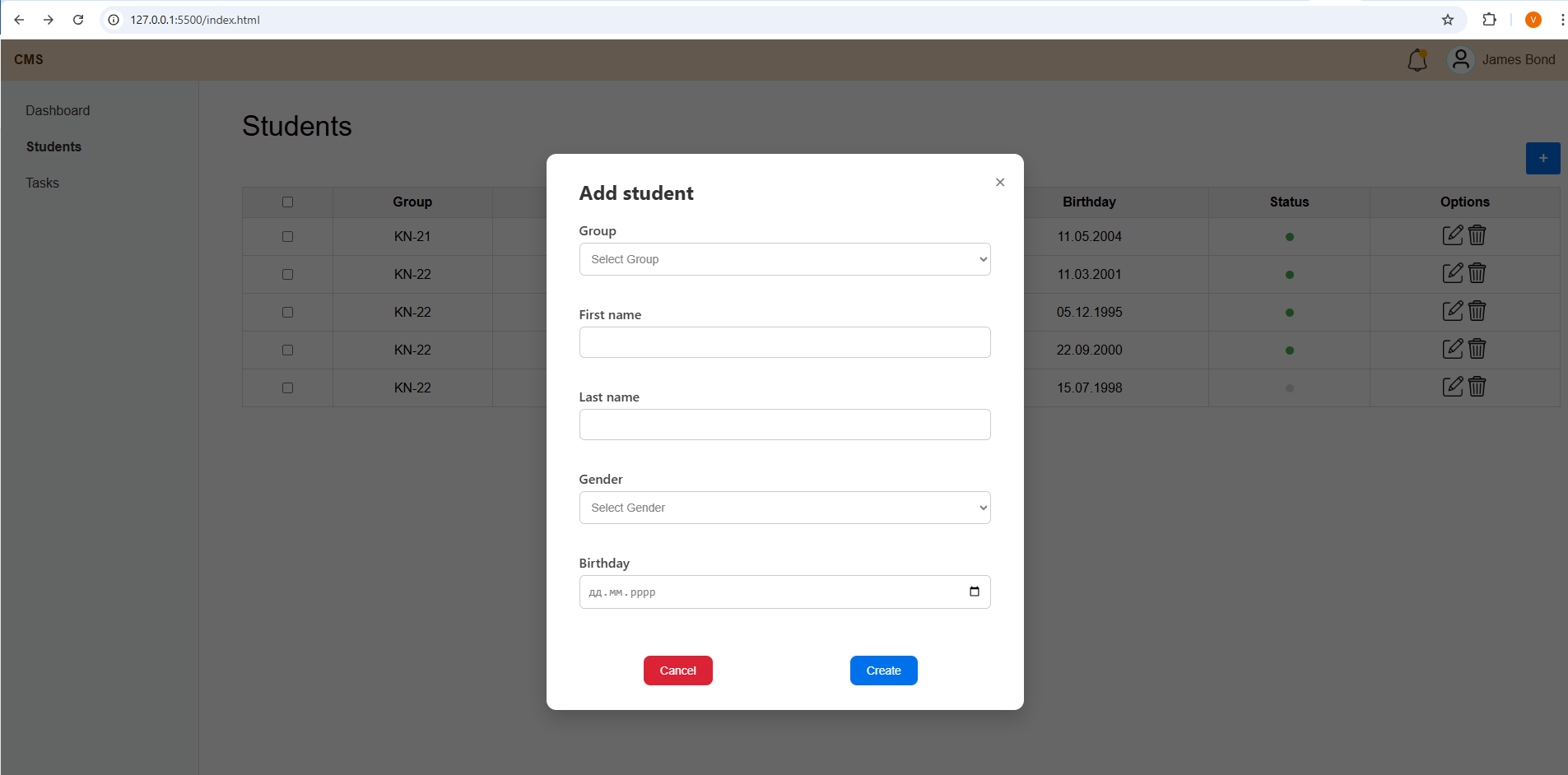
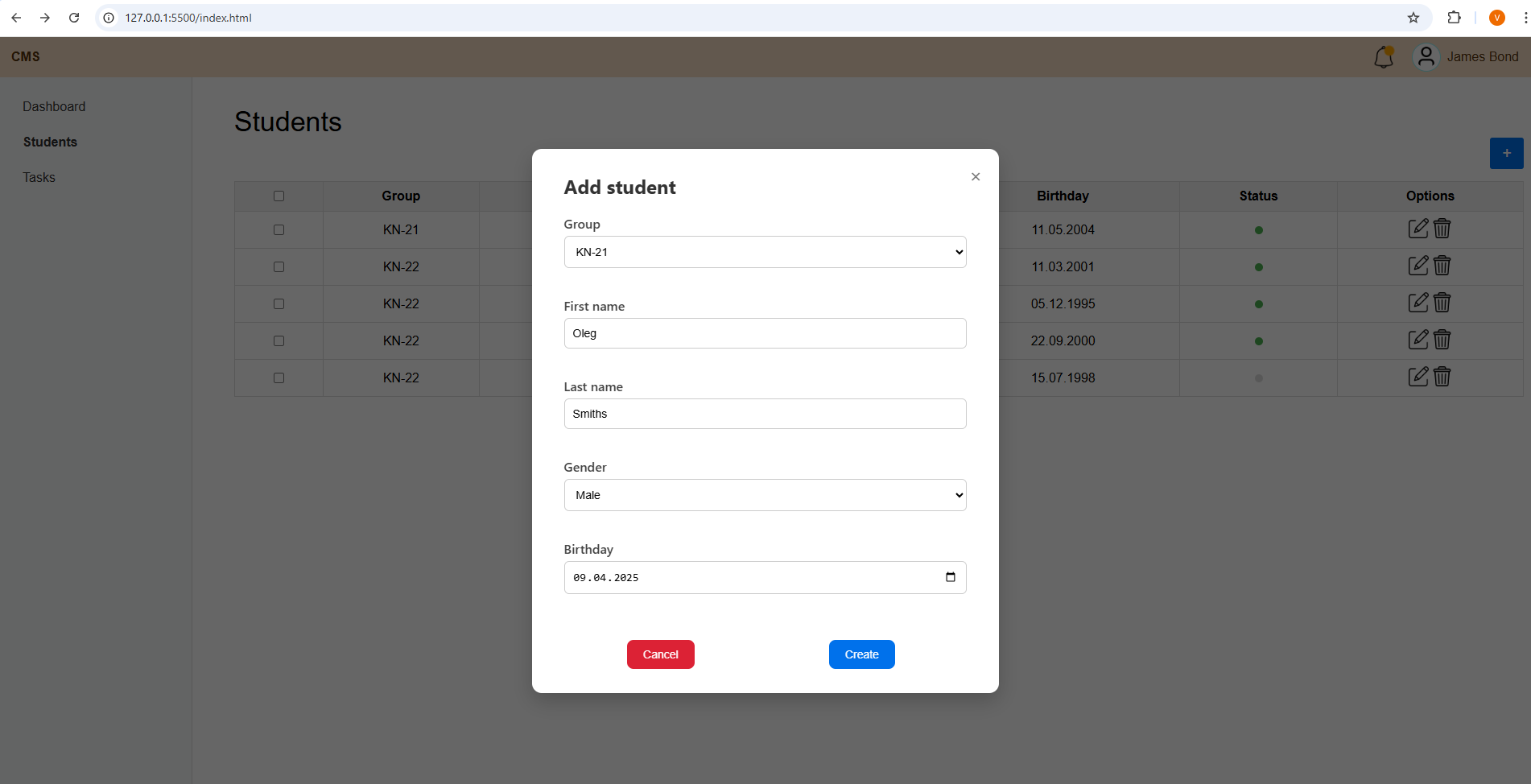
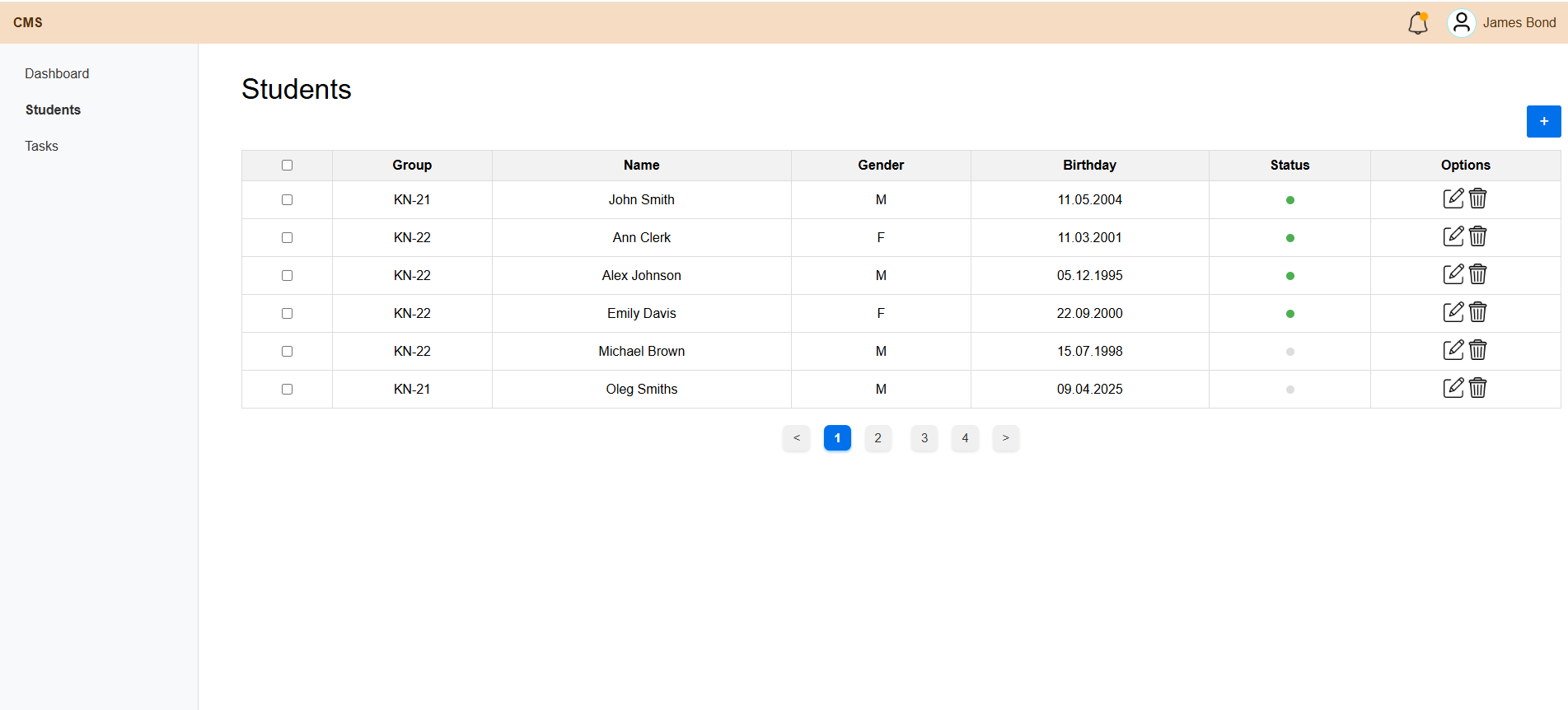


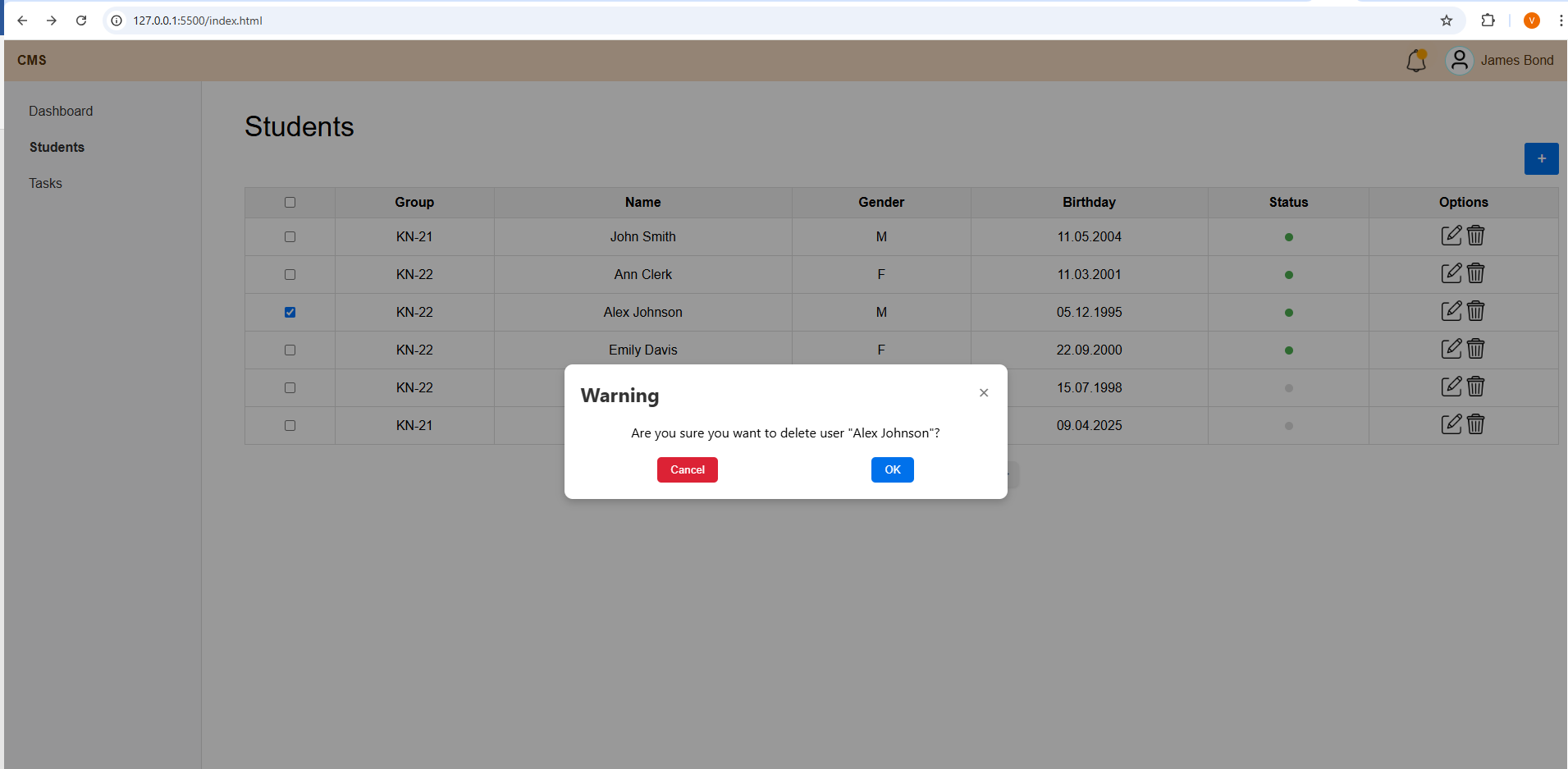
Рис. 9. Результат валідації сайту на <https://wave.webaim.org/>

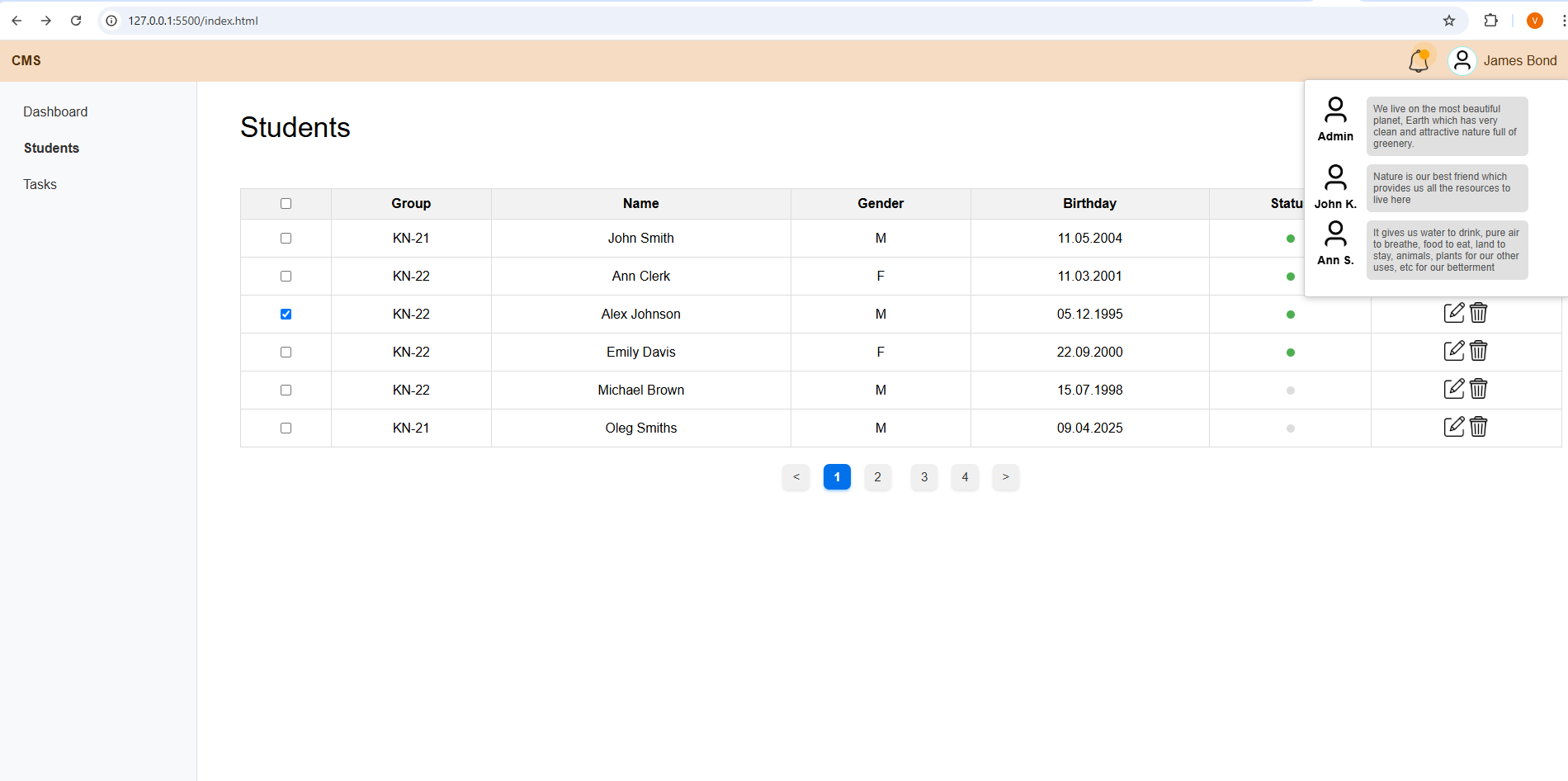


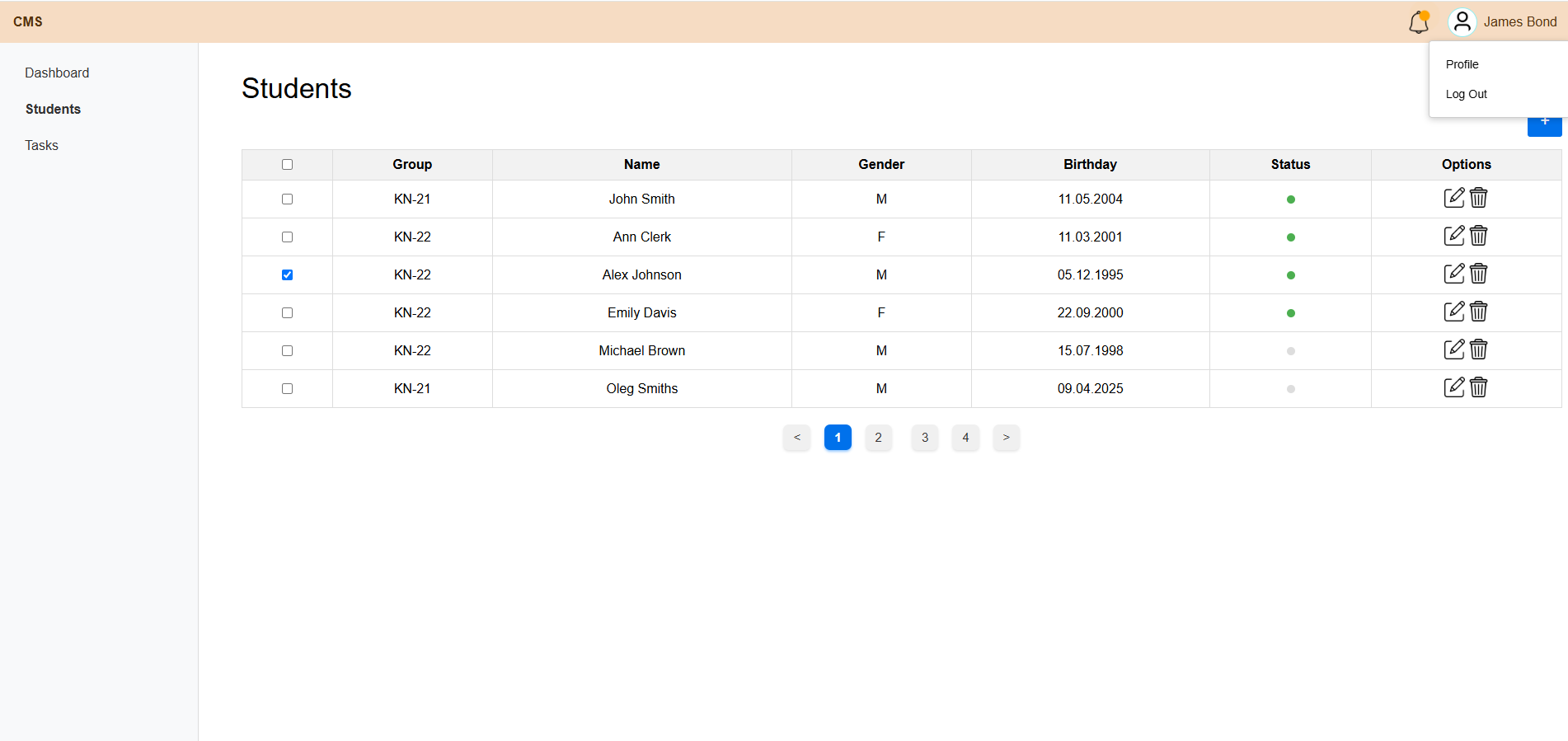


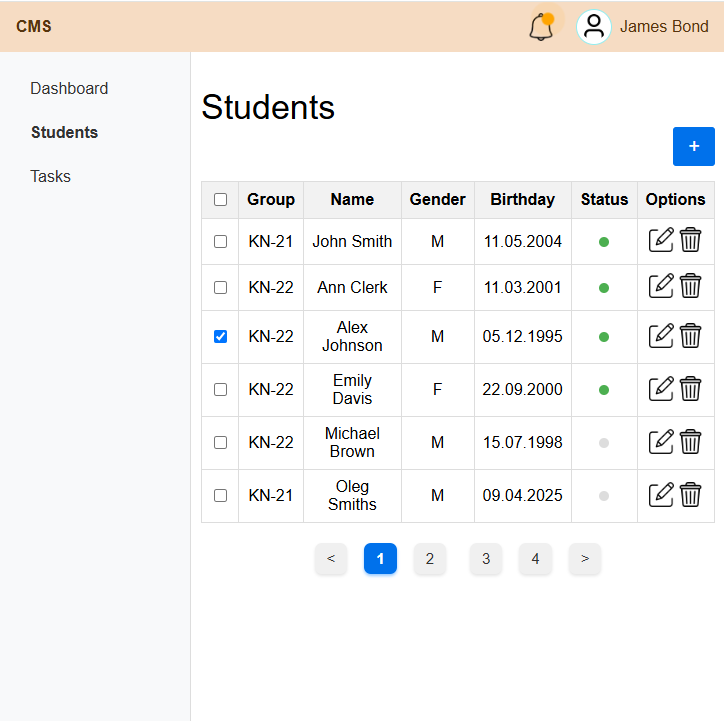












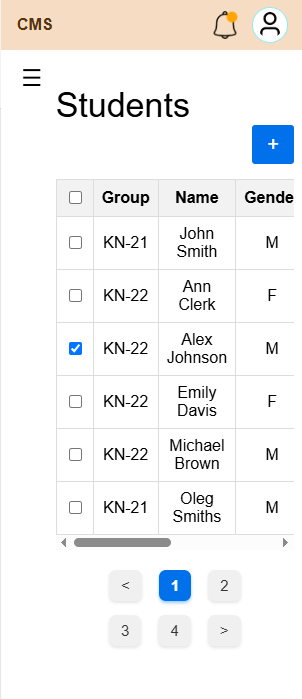


Рис. 10-18. Сайт

**Висновки**

На даній лабораторній роботі, я ознайомився з ієрархічною структурою об’єктів JavaScript та об’єктами документа і браузера та методами доступу до вузлів дерева.В ході виконання лабораторної роботи Було розроблено інтерактивний інтерфейс, який забезпечує динамічну зміну контенту за допомогою скриптів, а також реалізовано адаптивне відображення елементів для різних розмірів екранів. Завдяки цьому проєкту вдалося краще зрозуміти принципи структурованої розмітки, стилізації через каскадні таблиці стилів та програмної взаємодії за допомогою JavaScript, що є важливою основою для подальшого розвитку у сфері веб-розробки.