**Комунальний заклад освіти**

**«Дніпровський науковий ліцей інформаційних технологій»**

**Курсова робота**

**на тему:**

**Цікавий дизайн сайту ЛІТу**

**Виконавець:**

**ліцеїст 10-В-1 класу**

**Кравець Кирило**

**Керівник роботи:**

**Лавренюк Ірина Валеріївна**

**Дніпро**

**2025**

Зміст

[1. ВСТУП 3](#_Toc193189866)

[1.1. Актуальність роботи 3](#_Toc193189867)

[1.2. Мета роботи 4](#_Toc193189868)

[1.3. Додаткові аспекти та значення роботи 5](#_Toc193189869)

[2. ОСНОВНА ЧАСТИНА 6](#_Toc193189870)

[2.1. Теоретична частина з предмету 6](#_Toc193189871)

[2.2. Теоретична частина з інформатики 7](#_Toc193189872)

[2.2.1. Засоби та технології 7](#_Toc193189877)

[2.2.2. Засоби та технології 11](#_Toc193189882)

[2.2.3. Елементи управління та їх властивості 13](#_Toc193189888)

[2.2.4. Цікаві фрагменти програмного коду 16](#_Toc193189889)

[2.3. Опис роботи 20](#_Toc193189890)

[2.3.1. Головна сторінка (index.html) 20](#_Toc193189891)

[2.3.2. Головна сторінка (index.html) 21](#_Toc193189893)

[2.3.3. Контакти (contacts.html) 22](#_Toc193189901)

[2.3.4. Кнопка перемикання теми (light-theme-btn) 22](#_Toc193189902)

[2.3.5. Анімаційний фон за допомогою Vanta.js 24](#_Toc193189913)

[2.4. Посібник користувача 25](#_Toc193189919)

[2.4.1. Зміна теми сайту (світла/темна) 25](#_Toc193189920)

[2.4.2. Навігація по сайту 25](#_Toc193189928)

[2.4.3. Контакти та зворотний зв'язок 25](#_Toc193189937)

[2.5. Структурна схема програми 26](#_Toc193189938)

[2.6. Напрямки, де можна використати роботу 27](#_Toc193189951)

[2.7. Програмно-апаратні вимоги 28](#_Toc193189952)

[2.8. Програмно-апаратні вимоги 29](#_Toc193189953)

[2.9. Використані програмні засоби 30](#_Toc193189954)

[3. ВИСНОВКИ 31](#_Toc193189955)

[4. СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 34](#_Toc193189956)

# ВСТУП

Сучасний розвиток цифрових технологій значно вплинув на організацію освітнього процесу та способи комунікації між навчальними закладами, викладачами та студентами. Особливо це стосується цифровізації освітнього контенту, де якісний веб-сайт може стати ключовим інструментом для ефективної взаємодії з учасниками навчального процесу.

Тема даної роботи – «Цікавий дизайн сайту ліцею» – спрямована на створення сучасного, функціонального та зручного веб-ресурсу для навчального закладу. Основна мета проєкту – розробка інтуїтивного та естетично привабливого сайту, що полегшить доступ до важливої інформації, сприятиме інтерактивному навчанню та підвищить зацікавленість користувачів.

Реалізація цього проєкту передбачає використання сучасних веб-технологій для створення платформи, яка дозволяє учням та їхнім батькам ознайомлюватися з розкладом занять, новинами ліцею, навчальними матеріалами та результатами успішності.

Такий підхід дозволить створити не лише інформаційну платформу, а й динамічний цифровий простір, який відповідатиме сучасним вимогам освітньої сфери та підвищить імідж навчального закладу в цифровому середовищі.

# 1.1. Актуальність роботи

Сучасний світ активно переходить у цифровий формат, і навчальні заклади не є винятком. Веб-сайти стали невід’ємною частиною освітньої інфраструктури, виконуючи роль основного джерела інформації для учнів, батьків та викладачів. Проте багато навчальних закладів все ще використовують застарілі веб-ресурси з незручним інтерфейсом та обмеженим функціоналом, що значно знижує ефективність комунікації між адміністрацією закладу та його аудиторією.

Забезпечення сучасного дизайну та інтуїтивно зрозумілого інтерфейсу сайту є ключовим аспектом для підвищення його функціональності та привабливості. Структурований доступ до розкладу занять, навчальних матеріалів, оголошень, новин та інших важливих ресурсів значно спрощує взаємодію всіх учасників освітнього процесу.

Важливим компонентом даного проекту є також створення адаптивного дизайну, що дозволяє сайту коректно відображатися на різних пристроях, включаючи смартфони та планшети. Оскільки значна частина користувачів заходить на сайти саме з мобільних пристроїв, адаптивність є критично важливим фактором для зручності навігації.

Розробка сучасного веб-сайту для навчального закладу не лише сприятиме покращенню комунікації, але й допоможе сформувати позитивний імідж ліцею, відображаючи його прагнення до впровадження інноваційних освітніх рішень.

* 1. Мета роботи

Головною метою даної роботи є розробка сучасного модульного веб-сайту для ліцею, який забезпечує зручний доступ до інформації про навчальні програми, розклади, умови вступу та контактні дані. Сайт має бути адаптивним, з інтуїтивно зрозумілим інтерфейсом та інтегрованими функціональними можливостями, що підвищують зручність користування. Досягнення цієї мети передбачає вирішення низки завдань, які включають:

* **Створення інтуїтивно зрозумілого користувацького інтерфейсу.** Розробка адаптивного інтерфейсу для зручного перегляду інформації про ліцей, умови вступу, навчальні програми та розклад занять і т.і.. Особлива увага приділяється зручності користування як на десктопних, так і на мобільних пристроях.
* **Розробка структурованої інформаційної системи.** Інтеграція різних типів інформації, таких як новини ліцею, розклад занять, розклад класів та курси. Створення чіткої структури контенту для полегшення навігації та доступу до важливої інформації.
* **Інтеграція функціоналу для зручної взаємодії користувачів.** Розробка можливості для користувачів переглядати умови вступу, курсів та програм, з можливістю перегляду конкретних розкладів класів, подій або курсів, а також зворотного зв'язку.
* **Розробка функціоналу для адаптивного перегляду на мобільних пристроях.** Забезпечення коректного відображення всього контенту на різних пристроях з різними розмірами екрану, зберігаючи всі функціональні можливості та зручність у використанні.

# 1.3. **Додаткові аспекти та значення роботи**

Розробка даного веб-сайту має значний практичний потенціал для навчальних закладів, які прагнуть впровадити сучасні цифрові рішення для покращення комунікації та організації освітнього процесу. Впровадження інтерактивного дизайну та зручної навігації дозволить зробити сайт не лише інформаційним ресурсом, а й ефективним інструментом для взаємодії між адміністрацією, викладачами, учнями та їхніми батьками.

З точки зору користувачів, сайт стане центральною платформою для доступу до розкладу занять, навчальних матеріалів, оголошень та інших важливих ресурсів.

Додатково, інтеграція бази даних дозволить автоматизувати процеси управління навчальним контентом та забезпечить актуальність інформації у режимі реального часу. Це важливо як для викладачів, які зможуть оперативно оновлювати навчальні матеріали та виставляти оцінки, так і для учнів, які отримають доступ до необхідної інформації у зручному форматі(Це буде реалізовано у майбутньому).

Окрім цього, розробка адаптивного дизайну забезпечить коректне відображення сайту на різних пристроях, включаючи мобільні телефони та планшети. Це особливо важливо в сучасному середовищі, де більшість користувачів отримують інформацію онлайн, використовуючи портативні гаджети.

Загалом, реалізація цього проекту сприятиме модернізації навчального процесу, підвищенню його ефективності та створенню комфортного цифрового середовища для всіх учасників освітнього процесу. Веб-сайт стане не лише джерелом інформації, а й потужним інструментом для оптимізації навчання, зворотного зв’язку та адміністративного управління, що відповідає сучасним тенденціям цифрової освіти.

1. ОСНОВНА ЧАСТИНА
   1. Теоретична частина з предмету

**Модульний дизайн:** Веб-розробка сайту ліцею також орієнтується на модульний підхід, де кожен компонент відповідає за окрему функцію, що дозволяє покращити зручність обслуговування та підтримки сайту. Кожен модуль обробляє конкретне завдання, наприклад, відображення розкладу занять, новин, що сприяє чіткій організації коду. Відокремлення HTML, CSS та JavaScript дозволяє модульно управляти кодом. Це допомагає уникнути забруднення глобальної області та сприяє кращій підтримці проекту.

**Карти імпорту:** У разі потреби, при підключенні сторонніх бібліотек або скриптів на сайт, можна використовувати карти імпорту для зручності управління залежностями. Наприклад, за допомогою карт імпорту можна додавати бібліотеки для створення інтерфейсів або інтерактивних елементів, таких як відображення розкладу занять або інтерактивні блоки для новин, без необхідності вручну керувати шляхами до ресурсів.

Таким чином, модульний підхід до розробки сайту ліцею дозволяє створити масштабовану та підтримувану архітектуру, що відповідає сучасним вимогам до веб-розробки. Розбиття на компоненти та чітке визначення інтерфейсів між ними забезпечує зручність як для розробників, так і для користувачів сайту.

* 1. Теоретична частина з інформатики

2. 2. 1. Засоби та технології

* **HTML**

HTML визначає структуру веб-сторінки, описує, що повинно бути на сторінці і як елементи повинні бути взаємодіяти. Кожен елемент на сторінці (текст, зображення, кнопки, посилання, тощо) визначається через відповідні HTML-теги.

**Приклад 1**: Основна структура сторінки в HTML.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>ДНЛІТ</title>

<link rel="stylesheet" href="style.css">

<link rel="stylesheet" href="light-theme.css">

</head>

<body>

<header>

<nav>

<ul>

<li><a href="index.html">Головна</a></li>

<li><a href="about\_us.html">Про нас</a></li>

<li><a href="contacts.html">Контакти</a></li>

</ul>

</nav>

<img src="icons/logoLIT.png" alt="Логотип" class="logo">

</header>

<div id="intro">

</div>

<footer>

<p>© 2025 ДНЛІТ</p>

</footer>

<button class="light-theme-btn" id="light-theme-btn">

<svg class="light-theme-btn\_\_icon light-theme-btn\_\_icon\_sun"></svg>

<svgclass="light-theme-btn\_\_icon light-theme-btn\_\_icon\_moon"></svg>

</button>

<script src="app.js"></script>

</body>

</html>

* **Теги <header>** та **<footer>** містять інформацію про навігацію та контактні дані, відповідно.
* **Теги <nav>** і **<ul>** описують навігаційне меню, що дозволяє користувачеві переміщатися по сторінках.
* **Тег <div id="intro">** використовується для анімації (Vanta.js) внизу сторінки.
* **CSS**

CSS стилізує елементи, задані в HTML. Він визначає кольори, шрифти, позиціонування елементів, а також додає анімації або переходи для покращення взаємодії з користувачем.

**Приклад 2**: Стилізація теми (світла і темна).

body {

background-color: #22222e;

color: white;

font-family: Arial, sans-serif;

}

button {

background-color: #2c3e50;

color: white;

border: none;

padding: 10px 20px;

border-radius: 5px;

}

button:hover {

background-color: #34495e;

}

body.light-theme {

background-color: #a4ac88;

color: black;

}

button.light-theme-btn {

background-color: rgba(223, 211, 80, 0.5);

}

button.light-theme-btn:hover {

background-color: rgba(245, 238, 98, 0.6);

}

* **Темна тема** має чорний фон і білий текст, що знижує навантаження на очі в умовах поганого освітлення.
* **Світла тема** змінює фон на світлий з білим текстом, а також кнопки отримують жовто-зелену прозору заливку.

**Приклад 3**: Стилізація кнопки для перемикання теми.

.light-theme-btn {

position: fixed;

right: 15px;

top: 15%;

width: 60px;

height: 60px;

border-radius: 8px;

background-color: rgba(66, 165, 198, 0.5);

cursor: pointer;

z-index: 999;

}

.light-theme-btn:hover {

filter: brightness(120%);

}

.light-theme-btn:active {

filter: brightness(80%);

}

* Кнопка позиціонується у правому нижньому куті і змінює свою яскравість при наведенні або натисканні.
* **JavaScript**

JavaScript додає інтерактивність на сторінку, дозволяє обробляти події, взаємодіяти з DOM і змінювати вигляд елементів сторінки в реальному часі.

**Приклад 4**: Перемикання теми за допомогою JavaScript.

document.addEventListener("DOMContentLoaded", function () {

const theme = localStorage.getItem("theme");

if (theme) {

document.body.classList.add(theme);

}

document.getElementById("light-theme-btn").addEventListener("click", function () {

document.body.classList.toggle("light-theme");

if (document.body.classList.contains("light-theme")) {

localStorage.setItem("theme", "light-theme");

} else {

localStorage.removeItem("theme");

}

});

});

* **localStorage.getItem("theme")** дозволяє зберігати стан вибраної теми, щоб після перезавантаження сторінки тема не змінювалась.
* **document.body.classList.toggle("light-theme")** перемикає між світлою та темною темою.

1. 2. 2. Засоби та технології

* **Алгоритм перемикання теми**

Початкова перевірка: при завантаженні сторінки перевіряється наявність збереженого в localStorage значення для теми.

Подія кліку на кнопку: коли користувач натискає кнопку перемикання теми, на елемент body додається або видаляється клас light-theme, що змінює вигляд сторінки.

Збереження теми в localStorage: після перемикання теми стан (світла або темна) зберігається в localStorage, що дозволяє відновити тему при наступному відвідуванні сайту.

**Приклад 5:** Функція збереження теми:

if (document.body.classList.contains("light-theme")) {

localStorage.setItem("theme", "light-theme");}

else {

localStorage.removeItem("theme");

}

* **Алгоритм ініціалізації анімації з ефектом глобуса**

Цей алгоритм створює 3D-анімацію (глобус) за допомогою бібліотеки Vanta.js. Залежно від обраної теми, глобус змінює свій колір і фон.

**Приклад 6:** Ініціалізація ефекту Vanta.js:

let vantaEffect = null;

function initVanta() {

const isLight = document.body.classList.contains("light-theme");

if (!vantaEffect) {

vantaEffect = VANTA.GLOBE({

el: "#intro",

color: isLight ? 0xe1d143 : 0x843854

backgroundColor: isLight ? 0x98a283 : 0x21212f,

});

}

else {

vantaEffect.setOptions({

color: isLight ? 0xe1d143 : 0x843854,

backgroundColor: isLight ? 0x98a283 : 0x21212f,

});

}

}

**VANTA.GLOBE** ініціалізує анімацію глобуса на сторінці.

Кольори глобуса та фону змінюються залежно від теми.

* **Алгоритм оновлення кольору анімації**

Анімація повинна змінювати кольори залежно від вибору теми. Це забезпечується через метод setOptions бібліотеки Vanta.js.

**Приклад 7:** Оновлення кольорів анімації:

vantaEffect.setOptions({

color: isLight ? 0xe1d143 : 0x843854,

backgroundColor: isLight ? 0x98a283 : 0x21212f,

});

Цей фрагмент змінює кольори, коли користувач перемикає тему.

1. 2. 3. Елементи управління та їх властивості

Елементи управління — це компоненти веб-інтерфейсу, які взаємодіють з користувачем, дозволяючи змінювати стан інтерфейсу або виконувати певні дії. Вони включають кнопки, меню, форми та інші інтерактивні елементи.

* **Кнопка перемикання теми**

У моєму проекті основною інтерактивною кнопкою є кнопка перемикання теми, що дозволяє користувачу змінювати тему сайту між світлою та темною. Ця кнопка містить іконки для сонця і місяця, що відповідають світлій та темній темам відповідно.

**Приклад 1:** HTML-код для кнопки перемикання теми:

<button class="light-theme-btn" id="light-theme-btn">

<svg class="light-theme-btn\_\_icon light-theme-btn\_\_icon\_sun" viewBox="0 0 64 64">

<circle fill-rule="evenodd" clip-rule="evenodd" fill="#231F20" cx="32.003" cy="32.005" r="16.001" />

<-- Далі йде код для іконки сонця -->

</svg>

<svg class="light-theme-btn\_\_icon light-theme-btn\_\_icon\_moon" viewBox="0 0 24 24">

<path d="M21.5287 15.9294C21.3687 15.6594 20.9187 15.2394 19.7987 15.4394C19.1787 15.5494 ..." />

<-- Далі йде код для іконки місяця -->

</svg>

</button>

**SVG-іконки:** Для створення ікон сонця і місяця використовуються SVG-елементи, що дозволяє відображати векторні зображення без втрати якості при масштабуванні.

**Кнопка:** Вона позиціонується у нижньому правому куті за допомогою CSS і має округлені краї та прозорий фон, що робить її зручною для натискання.

**CSS:**

.light-theme-btn {

position: fixed;

right: 15px;

top: 93%;

width: 60px;

height: 60px;

border-radius: 8px;

background-color: rgba(66, 165, 198, 0.5);

cursor: pointer;

z-index: 9999;

}

.light-theme-btn:hover {

filter: brightness(120%);

}

.light-theme-btn:active {

filter: brightness(80%); }

**position: fixed:** Кнопка закріплена у правому нижньоиу куті сторінки.

**hover та active ефекти:** Кнопка змінює яскравість при наведенні та натисканні, щоб дати користувачу зрозуміти, що вона є інтерактивною.

* **Випадаюче меню**

Випадаюче меню є важливим елементом навігації, дозволяючи користувачам переміщатися між різними розділами сайту.

**Приклад 2:** HTML для випадаючого меню:

<ul class="menu">

<li class="menu-item">

<a href="#">Меню 1</a>

<ul class="submenu">

<li><a href="#">Підменю 1</a></li>

<li><a href="#">Підменю 2</a></li>

</ul>

</li>

<li><a href="#">Меню 2</a></li>

</ul>

**<ul>** визначає список основних елементів меню.

**<li>** використовується для кожного пункту меню.

**<a>** створює посилання для переходу на інші сторінки.

Випадаюче підменю **(<ul class="submenu">)** з'являється при наведенні на основний елемент меню.

CSS для випадаючого меню:

.menu {

list-style: none;

padding: 0;

margin: 0;

display: flex;

}

.menu-item {

position: relative;

}

.submenu {

display: none;

position: absolute;

top: 100%;

left: 0;

background-color: rgba(0, 0, 0, 0.8);

color: white;

padding: 10px;

border-radius: 8px;

}

.menu-item:hover .submenu {

display: block; }

**position: relative** для батьківського елементу дозволяє розташувати підменю абсолютно відносно цього елемента.

**display: none** приховує підменю за замовчуванням, а при наведенні на пункт меню (:hover) воно стає видимим.

* + 1. Цікаві фрагменти програмного коду

Цікаві фрагменти коду є тими частинами, які демонструють, як вирішуються складні задачі або додаються інтерактивні ефекти в проект.

* **Ініціалізація ефекту Vanta.js**

Vanta.js — це бібліотека, що дозволяє створювати захоплюючі анімації на фоні вашої сторінки, як наприклад анімація глобуса. Вона дозволяє додавати цікаві ефекти, які взаємодіють з користувачем.

**Приклад 3:** Ініціалізація ефекту глобуса з Vanta.js:

let vantaEffect = null;

function initVanta() {

const isLight = document.body.classList.contains("light-theme");

if (!vantaEffect) {

vantaEffect = VANTA.GLOBE({

el: "#intro", // Вказуємо елемент, на якому буде анімація

mouseControls: true,

touchControls: true,

gyroControls: false,

minHeight: 200,

minWidth: 200,

scale: 1.00,

scaleMobile: 1.00,

color: isLight ? 0xe1d143 : 0x843854,

backgroundColor: isLight ? 0x98a283 : 0x21212f

});

} else {

vantaEffect.setOptions({

color: isLight ? 0xe1d143 : 0x843854,

backgroundColor: isLight ? 0x98a283 : 0x21212f,

});

}

}

**el: "#intro"** — анімація буде застосовуватися до елемента з id intro.

Параметри **color** та **backgroundColor** змінюються в залежності від теми, щоб ефекти відповідали вибраному кольору.

* **Перемикання теми та збереження налаштувань**

Це простий і ефективний спосіб зберегти вибір користувача на сторінці та застосувати його на наступних відвідуваннях, використовуючи localStorage.

**Приклад 4:** Додавання та видалення класу для перемикання теми:

document.addEventListener("DOMContentLoaded", function () {

const theme = localStorage.getItem("theme");

if (theme) {

document.body.classList.add(theme);

}

document.getElementById("light-theme-btn").addEventListener("click", function ()

{

document.body.classList.toggle("light-theme");

if (document.body.classList.contains("light-theme")) {

localStorage.setItem("theme", "light-theme");

} else {

localStorage.removeItem("theme");

}

initVanta(); });

});

**localStorage.setItem("theme", "light-theme")** зберігає вибір користувача в браузері, щоб відновити його при наступному відвідуванні.

* **Ініціалізація 3D-анімації з Three.js**

Three.js — це потужна бібліотека для рендерингу 3D-графіки прямо в браузері. Вона використовується для створення складних тривимірних сцен.

**Приклад 5:** Ініціалізація 3D-сцени за допомогою Three.js:

var scene = new THREE.Scene();

var camera = new THREE.PerspectiveCamera(75, window.innerWidth / window.innerHeight, 0.1, 1000);

var renderer = new THREE.WebGLRenderer();

renderer.setSize(window.innerWidth, window.innerHeight);

document.body.appendChild(renderer.domElement);

var geometry = new THREE.BoxGeometry();

var material = new THREE.MeshBasicMaterial({ color: 0x00ff00 });

var cube = new THREE.Mesh(geometry, material);

scene.add(cube);

camera.position.z = 5;

function animate() {

requestAnimationFrame(animate);

cube.rotation.x += 0.01;

cube.rotation.y += 0.01;

renderer.render(scene, camera);

}

animate();

**THREE.Scene()** створює сцену.

**THREE.PerspectiveCamera()** визначає перспективу камери для коректного відображення 3D-об'єктів.

**THREE.BoxGeometry()** створює 3D-об'єкт (куб).

* **Анімація при зміні теми**

Якщо ви хочете додавати анімації на фоні при зміні теми, можна використовувати плавні переходи для зміни кольорів.

**Приклад 6:** CSS-перехід для зміни кольору фону при перемиканні теми:

body {

transition: background-color 0.5s ease-in-out;

}

body.light-theme {

background-color: #f4f4f4;

}

body.dark-theme {

background-color: #333;

}

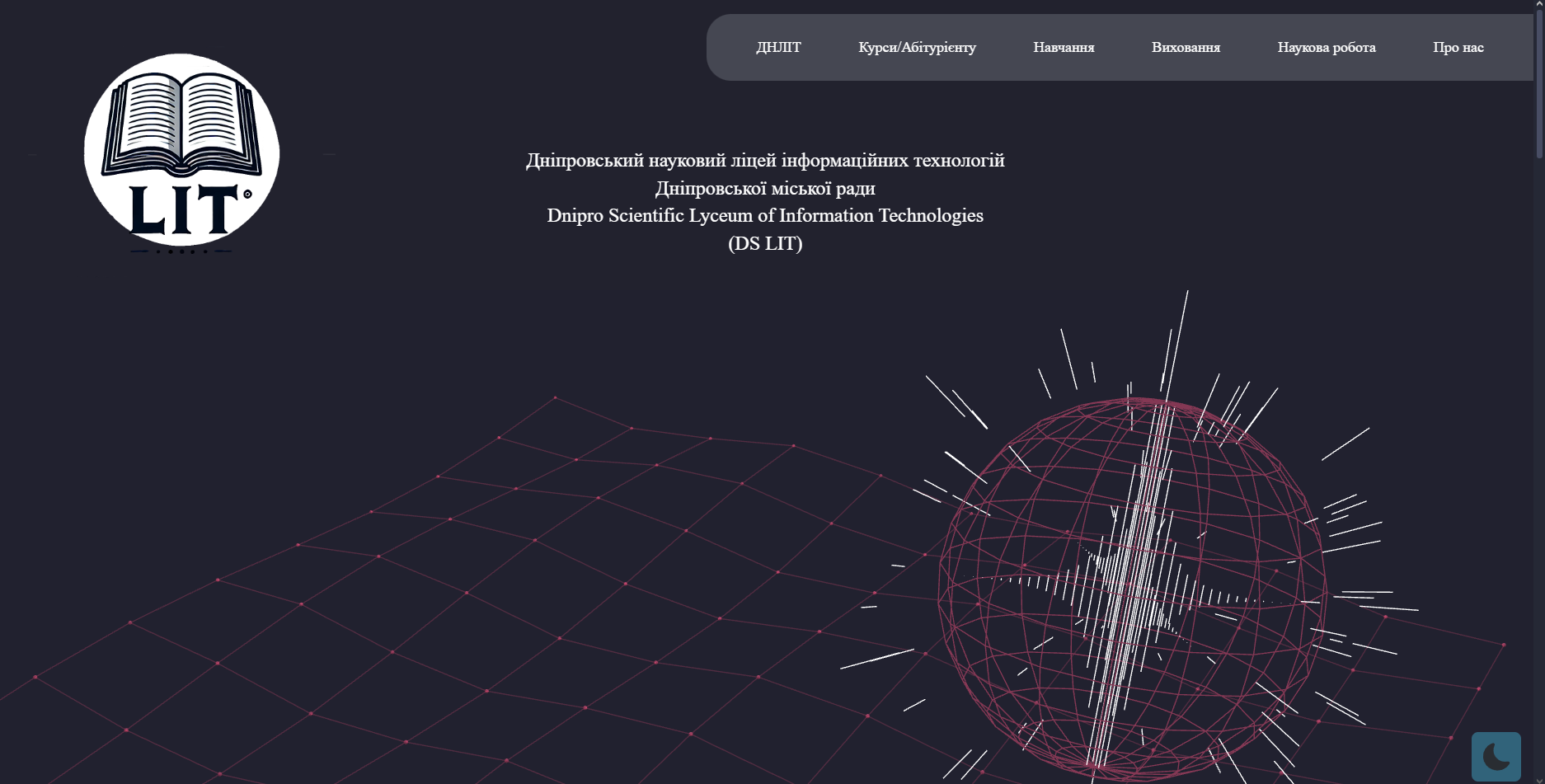
**transition: background-color 0.5s ease-in-out;** дозволяє плавно змінювати фон при перемиканні теми.

* 1. Опис роботи
     1. Головна сторінка (index.html)

**Призначення:** Головна сторінка є першою, яку бачить користувач після завантаження сайту. Вона містить основну навігацію та загальний вітальний контент, що вводить у курс справи.

**Ключові елементи:**

* Логотип та заголовок сайту.
* Основне меню з посиланнями на інші сторінки.
* Кнопка для перемикання між світлою та темною темами.



Головна сторінка сайту

* + 1. Про нас (about\_us.html)

**Призначення:** Ця сторінка надає користувачам більше інформації про авторів сайту. Вона може включає історію, цінності, ключових осіб та іншу корисну інформацію.

**Ключові елементи:**

* Опис про авторів сайту
* Фото авторів.
* Список досягнень або ключових подій.



Сторінка про творців сайту

2. 3. 3. Контакти (contacts.html)

**Призначення:** Сторінка контактної інформації містить дані для зв'язку з організацією. Це може бути адреса, телефон, електронна пошта або інші способи зв'язку.

**Ключові елементи:**

* Адреса організації.
* Контактні номери телефонів.
  + 1. Кнопка перемикання теми (light-theme-btn)



Сторінка про зворотній зв’язок за ЛІТом

**Призначення:** Кнопка перемикання теми дозволяє користувачеві змінювати вигляд сайту між світлою та темною темами. Це дозволяє підлаштувати сайт під переваги користувача, знижуючи навантаження на очі в умовах слабкого освітлення.

**Ключові елементи:**

* Кнопка перемикання теми на сайті.
* SVG-іконки для сонця (світла тема) та місяця (темна тема).
* Логіка збереження вибору в localStorage, щоб при наступному відвідуванні сайт автоматично відображав останню обрану тему.



## Приклад JavaScript-коду:



## Приклад HTML-коду:

1. 3. 5. Анімаційний фон за допомогою Vanta.js

**Призначення**: Цей розділ відповідає за створення інтерактивного фонового ефекту на сторінці, що підвищує естетичну привабливість сайту. Бібліотека **Vanta.js** дозволяє додавати динамічні фонові анімації, такі як глобуси.

**Ключові елементи:**

* Вибір фону за допомогою Vanta.js.
* Зміна кольору фону залежно від обраної теми.
* Реалізація анімації, яка реагує на взаємодію користувача (наприклад, рух миші).



## Приклад JavaScript-коду для Vanta.js:

1. 4. Посібник користувача
      1. Зміна теми сайту (світла/темна)

**Призначення:**

Ця функція дозволяє користувачу перемикати між світлою та темною темою сайту, що робить сайт зручнішим для роботи в різних умовах освітлення.

**Як користуватись:**

1. Знайдіть кнопку перемикання теми в правому нижньому куті (кнопка з іконками сонця або місяця).
2. Натискайте на кнопку для перемикання між світлою та темною темами.

**Примітка:** Після натискання на кнопку ваш вибір буде збережено в браузері за допомогою localStorage, тому при наступному завантаженні сайту буде автоматично застосована обрана тема.

2. 4. 2. Навігація по сайту

**Призначення:**

Цей розділ дозволяє користувачам переміщатися між основними сторінками сайту.

**Як користуватись:**

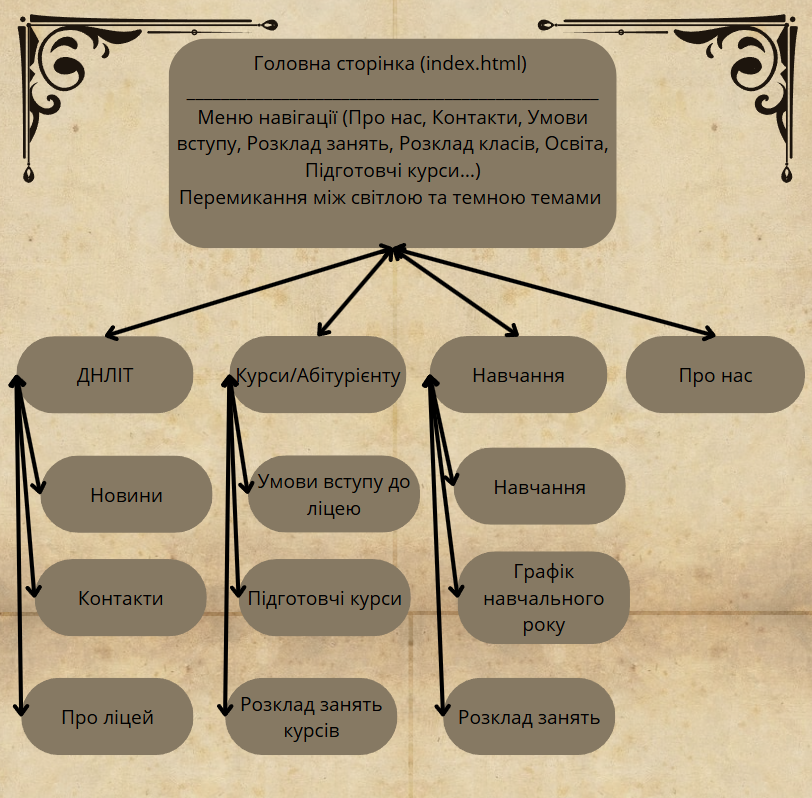
1. Знайдіть кнопку перемикання для навігації у верхній правій частині сторінки.
2. Перейдіть по одному з наступних посилань:
3. Для переміщення на інші сторінки просто натискайте на відповідні пункти меню.
5. 4. 3. Контакти та зворотний зв'язок

**Призначення:**

Ця функція дозволяє користувачам звертатися до організації за допомогою контактних даних.

**Як користуватись:**

1. Перейдіть на сторінку Контакти.
2. Ви побачите контактну інформацію, зокрема номер телефону, електронну пошту.
   1. Структурна схема програми



## Структура програми

3. 6. Напрямки, де можна використати роботу

**Освітні установи та навчальні заклади**

Розроблений сайт з такими функціями і структурою може бути використаний освітніми установами, такими як школи, університети або курси, для надання інформації про навчальні програми, курси, розклад, умови вступу, контактні дані, а також для покращення взаємодії з учнями або студентами.

**Приклади використання:**

* Школи: Інформація про розклад занять, розклад класів, підготовчі курси.
* Університети: Інформація про освітні програми, курси, умови вступу, розклад на навчальний рік.
* Курси та тренінги: Опис підготовчих курсів або спеціалізованих програм.

**Місцеві адміністрації або державні установи**

Державні установи можуть використовувати цей сайт для надання інформації громадянам про різні освітні або соціальні послуги. Це може включати розклад державних заходів, умови вступу до різних програм або шкіл, контактні дані для зв'язку з органами місцевого самоврядування.

**Онлайн-курси та платформи для навчання**

Для платформ, які пропонують онлайн-освіту, даний сайт стане чудовим варіантом для організації і надання структурованої інформації про курси, умови навчання, розклади, а також контактні дані для зворотного зв'язку.

**Інтерактивні інформаційні портали для студентів або учнів**

Сайт може бути використаний як інтерактивний портал для студентів або учнів, що дозволяє їм отримувати доступ до розкладів, умов вступу, зворотного зв'язку, а також переглядати інформацію про доступні курси.

**Сайти для інформаційних і культурних заходів**

Інформація про культурні та освітні події (лекції, семінари, курси) також може бути структурована на такому сайті. Це може бути корисно для культурних або освітніх центрів, музеїв, бібліотек, які хочуть організувати і публікувати розклад заходів.

* 1. Програмно-апаратні вимоги

**Операційна система:**

Сайт працюватиме на всіх основних операційних системах, таких як:

* Windows (версії Windows 10 та новіші).
* macOS (версії 10.14 і новіші).
* Linux (сучасні версії дистрибутивів Ubuntu, Fedora, Debian та т.і.).

**Веб-браузери:**

* Google Chrome.
* Mozilla Firefox.
* Safari.
* Microsoft Edge.

**Ці браузери повинні підтримувати:**

* HTML5, CSS3 та JavaScript.
* Анімації та взаємодії через Vanta.js (сучасні браузери підтримують WebGL, який використовується для анімацій).

**Програмне забезпечення для розробки (для локального тестування):**

* Текстовий редактор / IDE: Sublime Text, Visual Studio Code, або будь-який інший редактор для роботи з HTML, CSS та JavaScript.
* JavaScript-бібліотеки та фреймворки:
* Vanta.js (для анімацій).

**Мережеві вимоги:**

* Інтернет-з'єднання для доступу до веб-сайтів, серверів анімацій, бібліотек та інших зовнішніх ресурсів.
* Для локального тестування доступ до інтернету може бути необов'язковим, але для використання анімацій Vanta.js і завантаження ресурсів через CDN потрібне підключення до інтернету.
  1. Програмно-апаратні вимоги
* **HTML-файли:**

Це файли, які визначають структуру веб-сторінок. Вони містять усі елементи, що відображаються на сторінках сайту.

**index.html** — Головна сторінка сайту з основним контентом, меню навігації та анімацією.

**about\_us.html** — Сторінка з інформацією про авторів.

**about\_lyceum.html** — Сторінка з інформацією про ЛІТ.

**contacts.html** — Сторінка з контактною інформацією.

**admission\_conditions.html** — Сторінка, яка містить умови вступу до навчального закладу.

**academic\_year\_schedule.html** — Сторінка з графіком на навчального року.

**class\_schedule.html** — Сторінка з розкладом занять для класів.

**course\_schedule.html** — Сторінка з розкладом курсів.

**education.html** — Сторінка з інформацією про освітні програми та курси.

**preparatory\_courses.html** — Сторінка з описом підготовчих курсів.

* **CSS-файли**:

Ці файли визначають стиль і зовнішній вигляд веб-сторінок.

**style.css** — Загальний стиль для сайту, що визначає кольори, шрифти, відступи та інші параметри для всіх сторінок.

**light-theme.css** — Стилі для світлої теми сайту, що змінюються при перемиканні теми.

**contactscss.css** — Додаткові стилі для сторінки контактів.

**academic\_year\_schedule.css** — Стилі для сторінки графіку на навчальний рік.

**course\_schedule.css** — Стилі для сторінки розкладу курсів.

**preparatory\_courses.css** — Стилі для сторінки підготовчих курсів.

і т.і.

* **JavaScript-файли:**

Файли JavaScript додають інтерактивність і функціональність на сайті, такі як перемикання між темами, анімації та інші динамічні ефекти.

**app.js** — Основний JavaScript-файл, що містить логіку перемикання теми (світлої/темної) та ініціалізації анімацій.

**light-theme.js** — Файл, що відповідає за логіку перемикання теми (світла/темна) на сайті.

**three.min.js** — Бібліотека для створення 3D-ефектів на сайті.

**vanta.globe.min.js** — Бібліотека для створення анімацій з ефектами глобуса на фоні.

* 1. Використані програмні засоби
* **Текстові редактори / IDE:**

Для написання коду веб-сайту були використані текстові редактори та інтегровані середовища розробки (IDE), які допомагають зручно писати і редагувати код.

**Visual Studio Code (VSCode):**

Це популярне середовище для розробки, яке підтримує всі сучасні веб-технології (HTML, CSS, JavaScript). VSCode має зручний інтерфейс, плагіни для автодоповнення коду, а також підтримує Git для контролю версій.

* **Інструменти для тестування і відладки:**

**Live Server** — це плагін для Visual Studio Code (VSCode), який дозволяє запускати локальний сервер для швидкого перегляду веб-сторінок у браузері під час розробки. Він автоматично оновлює сторінку в браузері при збереженні змін у файлах проекту (HTML, CSS, JavaScript), що значно прискорює процес розробки.

* **Figma:**

Інструмент за домогою якого, я перероблював макет у код.

* **Git**:

**Git** — це система контролю версій, яка дозволяє розробникам зберігати історію змін коду і співпрацювати в команді.

1. ВИСНОВКИ
2. **Опанування ключових технологій та методів:**

У процесі виконання роботи було набуте глибоке розуміння та практичний досвід у роботі з основними технологіями веб-розробки, такими як **HTML5**, **CSS3**, та **JavaScript**. Це дозволило створити веб-сайт, який має структуровану основу з використанням HTML, відповідну стилізацію за допомогою CSS і динамічний функціонал завдяки JavaScript.

Особлива увага була приділена використанню **CSS для стилізації** та **адаптивного дизайну**, що дозволяє сайту коректно відображатися на різних пристроях. Важливим аспектом було **створення анімацій** через **Vanta.js**, що забезпечує інтерактивність і візуальну привабливість сайту. Завдяки цим технологіям, я зміг створити сайт з красивими та функціональними елементами, що позитивно впливає на досвід користувача.

1. **Практичне застосування теоретичних знань:**

У процесі розробки сайту було застосовано теоретичні знання з області структурування веб-сторінок, стилізації, а також взаємодії з користувачем через JavaScript. На практиці це включало:

* **HTML** для структурування контенту: кожна сторінка сайту була побудована за допомогою HTML, що дозволило чітко визначити структуру сайту та його елементи.
* **CSS** для стилізації та організації макету сторінок: використано різні стилі для адаптивності сайту, розроблені окремі стилі для **світлої** та **темної** теми.
* **JavaScript** для створення динамічних ефектів, таких як зміна теми сайту, а також анімації для фонового контенту через **Vanta.js**. Це дало змогу покращити взаємодію користувача з сайтом, роблячи його більш інтерактивним.

Завдяки такому підходу була успішно реалізована інтерактивність, яка включала перемикання між світлою та темною темою через JavaScript, а також використання анімованих фонів для поліпшення візуального сприйняття сайту.

1. **Опанування принципів модульного дизайну:**

Сайт був розроблений як модульний додаток, де кожна сторінка виконувала свою функцію та мала чітко визначену роль у загальній структурі. Модульність дизайну дозволила зберегти гнучкість і зручність для подальшого розширення проекту. Кожна сторінка сайту є самостійним компонентом, що робить її більш зручною для редагування і підтримки. Наприклад, сторінка Про нас містить лише текстову інформацію про авторів, в той час як Головна сторінка включає форму для зворотного зв'язку, що дозволяє користувачам надсилати запити.

Завдяки використанню модульного підходу сайт можна швидко масштабувати, додаючи нові сторінки чи функціональні можливості без порушення загальної структури. Всі компоненти були розроблені так, щоб їх можна було легко оновлювати та підтримувати.

1. **Опанування принципів модульного дизайну:**

Одним із основних принципів, на якому побудовано сайт, є модульний дизайн, що дозволяє створювати масштабовані і зручні в обслуговуванні веб-застосунки. Кожна частина сайту була розроблена як окремий модуль, що відповідає за конкретну функціональність. Це дозволяє:

* Легко оновлювати компоненти без впливу на інші частини сайту.
* Перезавантажувати тільки потрібні частини без переробки всього сайту.
* Можливість додавати нові модулі чи компоненти без суттєвих змін в існуючому коді.

Наприклад, всі стилі для теми (світла та темна) зберігаються в окремих CSS-файлах, а JavaScript для зміни теми та анімації знаходиться в окремих файлах. Це дозволяє легко адаптувати та змінювати окремі компоненти без порушення роботи інших частин сайту.

1. **Практичне засвоєння сучасних веб-технологій:**

Під час роботи над проектом були використані сучасні веб-технології, що дозволяють створювати динамічні, інтерактивні та адаптивні веб-сайти. Зокрема, було застосовано:

* **HTML5** для структурування контенту, забезпечення доступності та інтеграції мультимедійних елементів.
* **CSS3** для стилізації сторінок, в тому числі для створення адаптивного дизайну та підтримки двох тем (світлої і темної).
* **JavaScript** для реалізації інтерактивних елементів, таких як перемикання теми та додавання анімацій.
* **Vanta.js** — бібліотека для створення захоплюючих анімацій фону, що додає візуальної привабливості сайту.
* **Live Server** для швидкого перегляду та тестування сайту в реальному часі, що дозволяє миттєво бачити зміни при редагуванні коду.

Ці технології дозволяють створювати сайти, які не тільки виглядають сучасно, але й є зручними для користувачів, з можливістю змінювати теми та взаємодіяти з сайтом в реальному часі.

Виконання цієї роботи дозволило отримати практичний досвід у створенні функціонального веб-сайту з використанням сучасних технологій. Програмування з використанням **HTML, CSS, JavaScript** та бібліотек, таких як **Vanta.js**, дало можливість створити інтерактивний та динамічний сайт з функцією перемикання між світлою і темною темами, а також з анімацією фону. Модульний підхід до розробки забезпечив гнучкість і зручність у підтримці та розширенні проекту. Знання і досвід, отримані в процесі виконання цієї роботи, дозволяють використовувати їх для розробки більш складних веб-додатків і для подальшого удосконалення проекту.

1. СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

* Ritwick Dey. Live Server Extension for Visual Studio Code. URL: <https://marketplace.visualstudio.com/items?itemName=ritwickdey.LiveServer> (дата звернення: 24.11.2024).
* W3Schools. JavaScript Tutorial. URL: <https://www.w3schools.com/js/> (дата звернення: 20.11.2024).
* Mozilla Developer Network. JavaScript - MDN Web Docs. URL: <https://developer.mozilla.org/en-US/docs/Web/JavaScript> (дата звернення: 20.11.2024).
* GitHub. GitHub Docs. URL <https://github.com> (дата звернення: 15.03.2024).
* CSS-Tricks. *A Complete Guide to Flexbox*. URL: <https://css-tricks.com/snippets/css/a-guide-to-flexbox/> (дата звернення: 20.12.2024).