Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)

Навчально-науковий інститут комп'ютерних наук та інформаційних технологій
(повне найменування інституту, назва факультету(відділення))

Інформаційних систем та комп'ютерного моделювання
(повна назва кафедри (предметної циклової комісії))

Пояснювальна записка

до дипломної роботи

перший (бакалаврський)
(рівень вищої освіти)

на тему: Розроблення Back-end частини вебсайту Студентського наукового товариства

Виконав студент 4 курсу, групи ІСТ-41
спеціальності: 126 «Інформаційні системи та технології»
(шифр і назва напрямку підготовки, спеціальності)

Сенишин Сергій Васильович
(прізвище, ініціали)

Керівник Бекас Б.О., Флуд Л.О.
(прізвище, ініціали)

Рецензент Карашецький В.П.
(прізвище, ініціали)

Львів-2025

Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)

ННІ комп'ютерних наук та інформаційних технологій
Кафедра інформаційних систем та комп'ютерного моделювання
Рівень вищої освіти перший (бакаравський)
Спеціальність 126 "Інформаційні системи та технології"

ЗАТВЕРДЖУЮ:
Завідувач кафедри ІСКМ
_____ Сторожук О.Л.
"15" 11 2024 року

ЗАВДАННЯ
НА ДИПЛОМНУ РОБОТУ СТУДЕНТУ

Сенишину Сергію Васильовичу
(прізвище, ім'я, по батькові)

1.Тема бакалаврської роботи: Розроблення Back-end частини вебсайту Студентського наукового товариства
керівники роботи Бекас Богдан Олексійович старший викладач,
Флуд Любомир Олегович к.т.н., доц. кафедри ІСКМ
затверджені наказом вищого навчального закладу від "15" 11 2024 р. № С-884
2. Термін подання студентом роботи 12 червня 2025р.
3. Вихідні дані до роботи Розробити систему серверної частини сайту СНТ. Дана система має бути призначена для спільної роботи в реальному часі.
4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)
Вступ
Стан проблемної області
Інформаційне та математичне забезпечення
Програмне та технічне забезпечення
Висновки
Список використаних джерел
Додатки
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Підготовка матеріалу до доповіді.
6. Дата видачі завдання 18 листопада 2024р.

Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)

ННІ комп'ютерних наук та інформаційних технологій
Кафедра інформаційних систем та комп'ютерного моделювання
Рівень вищої освіти перший (бакаравський)
Спеціальність 126 "Інформаційні системи та технології"

ЗАТВЕРДЖУЮ:
Завідувач кафедри ІСКМ
_____ Сторожук О.Л.
"15" 11 2024 року

ЗАВДАННЯ
НА ДИПЛОМНУ РОБОТУ СТУДЕНТУ

Сенишину Сергію Васильовичу
(прізвище, ім'я, по батькові)

1.Тема бакалаврської роботи: Розроблення Back-end частини вебсайту Студентського наукового товариства
керівники роботи Бекас Богдан Олексійович старший викладач,
Флуд Любомир Олегович к.т.н., доц. кафедри ІСКМ
затверджені наказом вищого навчального закладу від "15" 11 2024 р. № С-884
2. Термін подання студентом роботи 12 червня 2025р.
3. Вихідні дані до роботи Розробити систему серверної частини сайту СНТ. Дана система має бути призначена для спільної роботи в реальному часі.
4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)
Вступ
Стан проблемної області
Інформаційне та математичне забезпечення
Програмне та технічне забезпечення
Висновки
Список використаних джерел
Додатки
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Підготовка матеріалу до доповіді.
6. Дата видачі завдання 18 листопада 2024р.

КАЛЕНДАРНИЙ ПЛАН

№, з/п	Етапи бакалаврської роботи	Термін виконання етапів роботи	Примітка
1.	Огляд літератури згідно досліджуваної теми. Збір необхідних матеріалів.	18.11.2024-20.12.2024	Виконано
2.	Постановка задачі і її формалізація	21.12.2024-25.01.2025	Виконано
3.	Виконання вхідного етапу технології	26.02.2025-05.03.2025	Виконано
4.	Реалізація головних функцій проєкту	06.03.2025-09.04.2025	Виконано
5.	Виконання етапу відлагодження проєкту	10.04.2025-14.04.2025	Виконано
6.	Виконання етапу впровадження та випуску бета-версії.	15.04.2025-05.05.2025	Виконано
7.	Оформлення записки до дипломного проєкту.	06.05.2025-10.06.2025	Виконано

Студент _____ Сенишин С.В.
(підпис) (прізвище та ініціали)

Керівники роботи _____ Бекас Б.О.
(підпис) (прізвище та ініціали)

_____ Флуд Л.О.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Дипломна робота містить 51 сторінки пояснювальної записки, 20 рисунок, 9 таблиць, 15 джерел, 2 додатки.

У цій роботі представлено процес розробки серверної частини вебсайту Студентського наукового товариства (СНТ). Система реалізована як динамічний вебресурс, що забезпечує зберігання, обробку та керування контентом, зокрема подіями, новинами, науковими матеріалами. Особливу увагу приділено гнучкості структури сайту, розмежуванню прав доступу, зручності адміністрування і безпеці обробки даних. Створено зручний інтерфейс для адміністратора, реалізовано форму авторизації, систему перегляду й редагування контенту, а також механізми роботи з файлами. Результатом проєкту є повнофункціональна backend-частина сайту СНТ, адаптована до потреб академічного середовища та готова до подальшої інтеграції з сучасним фронтендом.

Ключові слова: *студентське наукове товариство, вебсайт, бекенд, PHP, інформаційна система.*

ABSTRACT

The thesis contains 51 pages of explanatory note, 20 figures, 9 tables, 15 sources, 2 appendix.

This paper presents the development process of the server-side part of the Student Scientific Society (SSS) website. The system is implemented as a dynamic web resource that provides storage, processing, and management of content, including events, news, and scientific materials. Special attention is given to the flexibility of the site structure, access rights management, administrative convenience, and data processing security. A user-friendly administrator interface has been created, including an authorization form, content viewing and editing tools, as well as file management mechanisms. The result of the project is a fully functional backend part of the SSS website, tailored to the needs of the academic environment and ready for integration with a modern frontend.

Keywords: *Student Scientific Society, website, backend, PHP, information system.*

ТЕХНІЧНЕ ЗАВДАННЯ

Розробити серверної (Back-end) частини веборієнтованої системи вебсайту Студентського наукового товариства Національного лісотехнічного університету України. Розробити функціональну, безпечну і масштабовану серверну частину порталу СНТ, який забезпечуватиме обробку запитів від фронтенду, управління користувачами, подіями, заявками, контентом та зберіганням даних.

Реалізувати основні функції Back-end частини:

- Реєстрація та авторизація користувачів.
- Додавання, редагування та видалення новин, подій, документів (адмінпанель).
- Реалізація особистого кабінету адміністратора.

Для реалізації використати мову програмування: PHP 8.x, сервер: Apache або Nginx, інструменти: Postman, XAMPP/Laragon

Оформити пояснювальну записку відповідно до вимог.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

БД – база даних;

ІС – інформаційна система;

НЛТУ –Національний лісотехнічний університет України;

ОС – операційна система;

ПЗ – програмне забезпечення;

ПК – персональний комп’ютер;

СНТ – студентське наукове товариство;

IDE – комп'ютерна програмне середовище;

UI – інтерфейс користувача;

Unit-тестування – модульне тестування.

ВСТУП

У сучасному інформаційному середовищі вебінтерфейс є ключовим елементом взаємодії користувача з цифровими ресурсами, особливо у сфері освіти. Вебсайт студентського наукового товариства (СНТ) виступає не лише інформаційною платформою, а й середовищем активної участі студентів у науковій діяльності, подіях, конкурсах і дослідженнях. Саме тому важливою складовою є розробка інтуїтивного, адаптивного та функціонального фронтенду, який забезпечує зручний доступ до всіх можливостей системи.

Основне завдання фронтенд частини – реалізація сучасного, естетично привабливого та зручного користувацького інтерфейсу. Інтерфейс повинен бути адаптований для роботи на різних пристроях – комп'ютерах, планшетах і смартфонах – із дотриманням принципів UX/UI. Навігація сайтом має бути логічною та зрозумілою, що дає змогу користувачам швидко знаходити потрібну інформацію, реєструватися на заходи, подавати заявки чи переглядати наукові матеріали.

Фронтенд побудовано з використанням HTML, CSS і JavaScript, із можливістю інтеграції з бекенд через REST API. Реалізовано окремі розділи для новин, заходів, документів, а також сторінки «Про СНТ», кабінет користувача та форму зворотного зв'язку. Значну увагу приділено візуальній структуризації контенту, відображенню даних у таблицях і картках, використанню інтерактивних елементів і форм.

Результатом є гнучкий, масштабований фронтенд, який не лише відповідає технічним вимогам, а й враховує потреби студентів, наукових керівників та адміністрації. Такий інтерфейс значно підвищує ефективність комунікації в межах СНТ і сприяє формуванню цифрової культури в університетському середовищі.

Актуальність дослідження. У сучасних умовах цифрової трансформації закладів вищої освіти важливо забезпечити ефективну взаємодію між студентами та науковими осередками. Створення повнофункціонального сайту Студентського наукового товариства (СНТ) з ефективною Back-end частиною дозволяє автоматизувати обробку наукових подій, заявок, керування контентом,

що сприяє покращенню комунікації та прозорості наукової діяльності в університетському середовищі.

Об’єкт дослідження. Інформаційна система вебсайту студентського наукового товариства.

Предмет дослідження. Програмна реалізація серверної (Back-end) частини вебпорталу СНТ з використанням PHP та взаємодією з базою даних.

Мета роботи. Розробити ефективну Back-end частину сайту СНТ, яка забезпечуватиме управління контентом, користувачами та інформаційними потоками між клієнтською частиною сайту й базою даних.

Практична значимість. Результати роботи можуть бути безпосередньо впроваджені в освітній процес і наукову діяльність університету, слугувати основою для автоматизованого супроводу діяльності СНТ та використовуватися для розробки аналогічних порталів інших студентських організацій.

Сайт має бути інтегрований в наявні інформаційні ресурси університету.

РОЗДІЛ 1. СТАН ПРОБЛЕМНОЇ ОБЛАСТІ

1.1 Аналіз рішень у сфері фронтенд-розробки освітніх платформ

У цьому підрозділі розглядаються сучасні підходи до розробки користувацьких інтерфейсів для освітніх вебресурсів, зокрема студентських порталів і сайтів наукових товариств. Проаналізовано популярні JavaScript-фреймворки, такі як React, Vue та Next.js, а також CSS-бібліотеки (Tailwind, Material UI, Shadcn), що використовуються для побудови адаптивного та доступного інтерфейсу[2].

React, Vue та Next.js є найпопулярнішими JavaScript-фреймворками, які широко використовуються для створення сучасних вебінтерфейсів. У проєкті розробки сайту Студентського наукового товариства було проведено порівняльний аналіз цих технологій з метою вибору оптимального рішення для фронтенду. React.js – це компонентно-орієнтований фреймворк, розроблений компанією Meta, який дозволяє створювати гнучкі, масштабовані інтерфейси завдяки використанню віртуального DOM і JSX-синтаксису. Він підтримує широкий екосистемний набір рішень, таких як Redux, React Router, і активно розвивається спільнотою. React надає високу продуктивність та гнучкість, проте потребує ретельної конфігурації і досвіду роботи з супутніми бібліотеками.

Vue.js, на відміну від React, простіший у вивченні, має зрозумілий синтаксис і чудово підходить для проєктів середнього масштабу. Він забезпечує двосторонню прив’язку даних (two-way binding), вбудовану реактивність і можливість швидкої інтеграції в існуючий HTML-код. Vue дозволяє розділяти компоненти за шаблонами, логікою і стилями у межах одного `.vue` файлу, що робить розробку прозорою та зручною. Однак у порівнянні з React Vue менш масштабований у великих проєктах і має меншу екосистему.

Next.js – це фреймворк на базі React, який додає підтримку серверного рендерингу (SSR), статичної генерації та маршрутизації «з коробки». Він ідеально підходить для SEO-оптимізованих сайтів і дозволяє створювати динамічні сторінки з високою швидкістю завантаження. Для навчального сайту СНТ,

орієнтованого на інтерактивність і внутрішній функціонал, використання Next.js було б доцільним у випадку потреби в повноцінному SSR або статичному рендерингу. Однак його конфігурація є складнішою, ніж у базового React, що може бути зайвим для навчального проєкту.

Щодо стилізації, порівнювалися кілька CSS-бібліотек, зокрема Tailwind CSS, Material UI та Shadcn UI. Tailwind є утилітарною бібліотекою, яка дозволяє швидко створювати адаптивні інтерфейси без написання класичних CSS-файлів. Вона забезпечує високий рівень кастомізації, однак потребує звикання до великої кількості класів безпосередньо в HTML. Material UI орієнтована на дотримання стандартів дизайну Google та містить готові компоненти з розширеними функціями. Вона більше підходить для складних додатків з корпоративним стилем. Shadcn – це новітнє рішення, яке поєднує філософію Tailwind із зручністю готових компонентів.

Для серверної частини проєкту було обрано мову програмування PHP [12-14], яка забезпечує обробку запитів, роботу з базою даних, автентифікацію та керування контентом. PHP залишається однією з найбільш використовуваних мов для розробки вебзастосунків завдяки простоті використання, широкому розповсюдженню і сумісності з різними системами управління базами даних (зокрема, MySQL) [4], [5]. Для реалізації адміністративної панелі, керування подіями та обробки форм PHP забезпечує достатній рівень стабільності і масштабованості. Крім того, завдяки підтримці шаблонізаторів і MVC-фреймворків (наприклад, Laravel), PHP легко адаптується до сучасних підходів розробки [13-14].

Таблиця 1.1 – Порівняння основних характеристик JavaScript-фреймворків

Характеристика	**React**	**Vue**	**Next.js**
Крива навчання	Середня	Низька	Висока
Продуктивність	Висока	Висока	Висока
SSR підтримка	Через Next.js	Через Nuxt.js	Так
SEO-оптимізація	Складно реалізувати	Обмежено	Проста реалізація
Гнучкість	Висока	Середня	Середня
Розмір спільноти	Дуже велика	Велика	Велика

Для реалізації фронтенд частини проєкту СНТ було доцільно обрати React як основу деяких функцій, CSS для гнучкої стилізації, а PHP для бекенд-обробки логіки, що забезпечило ефективну взаємодію між частинами сайту, масштабованість і зручність обслуговування.

Особливу увагу приділено структурам освітніх сайтів на прикладі університетів України та Європи. Структура освітніх сайтів університетів як в Україні, так і в Європі, зазвичай будується за схожими принципами, орієнтованими на зручність доступу до навчальної, наукової, адміністративної та довідкової інформації для студентів, абітурієнтів, викладачів і зовнішніх відвідувачів [9].

Сучасні європейські сайти часто мають інтеграцію з внутрішніми інформаційними системами університету – електронними журналами, бібліотеками, обліковими кабінетами, системами подання заявок. Вони мають адаптивну верстку, багатомовну підтримку та інтеграцію з соціальними мережами.

Технологічно фронтенд таких сайтів реалізовують на:

React.js – часто використовується в Європі для проєктів з динамічними компонентами (реєстраційні системи, особисті кабінети) [3];

Vue.js – популярний серед освітніх закладів Франції, Іспанії, Бельгії для портальних рішень із меншою складністю;

Next.js – використовується в Німеччині, Нідерландах для SEO-оптимізованих академічних платформ;

HTML + CSS + Bootstrap/Tailwind – поширене поєднання для базових інформативних сторінок [1; 2; 11];

CMS-платформи (Drupal, WordPress) – на них базуються багато українських сайтів (наприклад, НУ "Львівська політехніка" або КНУ ім. Шевченка), де інтерфейс спрощено реалізовано через шаблони, плагіни та PHP.

У більшості українських університетів фронтенд – це статичні сторінки з HTML/CSS та невеликою часткою JavaScript, іноді застарілі шаблони з Joomla/WordPress або самописні рішення на PHP. У європейських – дедалі більше

переходять на JAMstack-архітектуру (JavaScript + API + Markup) для гнучкості, швидкості та безпеки.

Таким чином, структура сайтів схожа за змістом, однак технічне втілення значно відрізняється: європейські рішення зазвичай більш динамічні, інтегровані, з високим рівнем UX/UI-дизайну, тоді як в Україні ще зберігаються багато сайтів тенденція яких до оновлення активно набирає обертів.

1.2 Аналіз потреб користувачів та функціональних вимог до інтерфейсу

Розгорнутий аналіз потреб користувачів та функціональних вимог до інтерфейсу сайту Студентського наукового товариства (СНТ) має ключове значення для побудови ефективної та зручної взаємодії між платформою й її відвідувачами. У межах даного дослідження було здійснено ідентифікацію основних типів користувачів системи: студенти (як учасники та потенційні члени СНТ), викладачі (як наукові керівники), адміністративний персонал (редактори, адміністратори контенту), а також сторонні користувачі (гості сайту, абітурієнти, представники інших установ). Кожна з цих категорій має власні цілі, сценарії використання та очікування щодо взаємодії з інтерфейсом, тому їхні потреби були враховані при формуванні функціональних блоків і UX/UI-рішень [6].

Для студентів критично важливо мати простий і швидкий доступ до таких функцій, як перегляд новин і анонсів заходів, можливість подати заявку на участь, ознайомитися з поточними науковими групами, завантажити необхідні документи (положення, протоколи, інструкції), а також залишити запитання чи зворотний зв’язок. Враховуючи те, що більшість цільової аудиторії користується мобільними пристроями, було висунуто вимогу до повної адаптивності інтерфейсу з можливістю комфортного перегляду і взаємодії зі всіма елементами – навіть на екранах із діагоналлю менше 6 дюймів [10].

Викладачі очікують зручної навігації у розділі "Наукова діяльність", де вони можуть переглядати інформацію про студентські роботи, контактувати з командами, завантажувати результати досліджень та модерувати певні елементи в своїх підрозділах. Для них також важливо, щоб система зберігала структуру

публікацій, дозволяла відображати авторські матеріали та подавати інформацію для звітів і конференцій.

Адміністрація СНТ має особливі потреби у можливості редагування контенту через адміністративну панель, зокрема додавання новин, створення подій, модерування форм зворотного зв’язку та перегляду аналітики користувацької активності. Система повинна бути інтуїтивно зрозумілою, захищеною від випадкових змін та мати рівні доступу для різних типів адміністраторів [7].

Ключові функціональні елементи інтерфейсу включають:

- **Горизонтальне меню** з основними розділами;
- **Панель пошуку**, яка працює як локально по сторінці, так і по загальній структурі сайту;
- **Картки подій та новин**, оформлені у вигляді блоків із коротким описом, датою, кнопкою реєстрації;
- **Форми зворотного зв’язку і заявок**, реалізовані за допомогою адаптивних компонентів із перевіркою введених даних;
- **Інтерактивні вкладки**, що дозволяють структурувати великий обсяг інформації (наприклад, у розділі "Документи").

Інтерфейс відповідає сучасним принципам UX-дизайну: усі важливі елементи знаходяться у межах одного-двох кліків, контрастність шрифтів та кольорова палітра дотримуються норм доступності для людей з вадами зору, навігація побудована за принципом мінімального часу досягнення цілі.

Окремо було проаналізовано відповідність стандартам WCAG 2.1 для забезпечення базової доступності сайту. Система протестована для взаємодії з клавіатурою, підтримки скринрідерів, масштабування шрифтів, що робить її придатною до використання різними категоріями користувачів. У контексті мобільних пристроїв, забезпечено логічну зміну розміщення блоків та адаптивну зміну розміру шрифтів та кнопок при зменшенні екрану.

Функціональні вимоги сформовано на основі аналізу інших університетських сайтів, опитування студентів і бенчмаркінгу з ресурсами

провідних європейських ВНЗ. У результаті отримано перелік функціональних модулів, які можуть бути представлені у вигляді таблиці відповідності ролей користувача та функцій, яку доцільно вставити як "Таблиця 1 – Відповідність функцій інтерфейсу до ролей користувачів" [8].

Такий всебічний аналіз дає змогу чітко сформулювати обґрунтовані технічні вимоги до фронтенду та визначити логіку побудови його структури на наступних етапах проєктування та розробки.

1.3 Проблеми існуючих фронтенд-інтерфейсів навчальних вебсайтів

У підрозділі виявлено типові проблеми, притаманні багатьом студентським чи науковим сайтам, зокрема:

– невпорядкована або надто складна структура навігації, що ускладнює доступ до потрібного контенту;

– відсутність адаптивного дизайну, що призводить до некоректного відображення на мобільних пристроях;

– обмежена інтерактивність, що робить інтерфейс застарілим або незручним для користувачів;

– відсутність чіткої візуальної ієрархії та стандартів UI/UX-дизайну.

Ці проблеми стають перепоною на шляху ефективної взаємодії користувача з інформаційною системою та потребують сучасного підходу до фронтенд-розробки на основі гнучких фреймворків і принципів адаптивного дизайну.

Проблеми, які виникають у фронтенд-інтерфейсах навчальних вебсайтів, зокрема сторінок студентських наукових товариств (СНТ), пов’язані насамперед із застарілим дизайном, відсутністю адаптивності, фрагментованістю інформації та низьким рівнем інтерактивності. Більшість таких сторінок розташовані у вигляді простих статичних підрозділів на основному сайті університету без окремого функціонального середовища, що обмежує зручність і ефективність користування.

Серед найбільш поширених проблем можна виділити: перевантаження текстом без чіткої візуальної ієрархії; відсутність інтерактивних елементів

(реєстрацій, форм, динамічного контенту); погано реалізовану навігацію або її повну відсутність; несумісність із мобільними пристроями; відсутність пошуку; низьку продуктивність у слабких мережах; а також відсутність персоналізації або авторизації користувачів. Це знижує залучення аудиторії та не сприяє активній роботі в межах СНТ.

Представимо порівняльну таблицю з аналізом інтерфейсів сторінок СНТ у кількох ЗВО України (табл. 1.2).

Основні узагальнені проблеми:

- **Формальність** сторінок СНТ: у більшості випадків вони сприймаються лише як «інформаційна довідка» без інтерактиву;
- **Слабкий UX/UI**: дизайн не відповідає сучасним очікуванням молодої аудиторії, відсутня адаптивна верстка;
- **Брак динаміки**: не використовується JavaScript або використовується мінімально – без динамічного завантаження, інтеграції з формами, без кабінету користувача;
- **Недостатня підтримка мобільних пристроїв**, що є критично важливо в умовах зростання мобільного трафіку;
- **Немає централізованого механізму взаємодії** – як от особистих кабінетів, систем реєстрації, онлайн-архівів, автоматизованих подій.

Це свідчить про необхідність розробки нового вебінтерфейсу СНТ на сучасному технологічному стеку, що забезпечить повноцінний функціонал, адаптивність, модульність та UX-орієнтованість. Такий підхід дозволить значно підвищити ефективність комунікації з аудиторією, зменшити інформаційний шум і створити повноцінне онлайн-середовище для наукової діяльності студентів.

Таблиця 1.2 – Аналіз інтерфейсів сторінок СНТ у кількох ЗВО України

Університет / СНТ	Адаптивність	Структура контенту	Наявність інтерактиву	Дизайн / UX	Окрема система чи сторінка	Проблеми
БДМУ (snt.bsmu.edu.ua)	+/-	Є логічна структура, окремі блоки	Присутні форми, але без повного особистого кабінету	Застарілий, без мобільної оптимізації	Окремий сайт	Дрібні шрифти, низький контраст, відсутність анімації
НМУ (nmu.ua/studentske-naukove-tovarystvo)	-	Сторінка з текстом і розділами	Лише email-контакти	Мінімалістичний, статичний	Вкладка на головному сайті	Відсутність динаміки, адаптивності, пошуку
НЛТУ (nltu.edu.ua/.../studentske-naukove-tovarystvo)	-	Одна статична сторінка	Інтерактив відсутній	Недоступний для мобільних	Частина головного сайту	Слабка структурованість, застарілий інтерфейс
Київський політех (sn.kpi.ua)	+	Чітке меню, сторінки, секції	Авторизація, завантаження файлів	Сучасний, зрозумілий	Окрема система	Деяка перевантаженість контентом, довгі сторінки

Огляд наукових публікацій, що досліджують проблеми фронтенд-інтерфейсів та юзабіліті університетських сайтів. У праці [4] «результати показали, що спостереження було найкращим методом, порівняно з двома іншими, для виявлення великої кількості основних та незначних проблем зручності використання на вебсайтах університетів. Результати також показали, що використання якісних даних з анкети після тестування було корисним додатковим методом, оскільки це виявило додаткові проблеми зручності використання, які не були виявлені методом спостереження. Однак результати показали, що кількісні дані з анкети після тестування були неточними та неефективними з точки зору виявлення проблем зручності використання на таких вебсайтах.» Це свідчить про те, що реальна поведінка користувачів дозволяє виявити як критичні, так і дрібні, але значущі помилки у фронтенді цих сайтів. Reem Alsaeed & Mohammad Mahdi Hassan у своїй роботі [5] вказують, що «користувачі університетських вебсайтів стикаються з різними проблемами зручності використання під час навігації та пошуку інформації. Вивчення цих проблем є критично важливим для успіху зростаючого ринку вищої освіти. Головною метою цієї роботи є визначення проблем зручності використання, з якими стикаються користувачі університетських вебсайтів». Це показує, що наявні інтерфейси часто не відповідають стандартам зрозумілості та ефективності навігації. Layla Hasan у своєму дослідженні [6] запевняє, що «У цьому дослідженні було розглянуто відносну важливість конкретних критеріїв дизайну, розроблених для цілей цього дослідження, в оцінці зручності використання освітніх вебсайтів з точки зору студентів; потім було оцінено зручність використання дев'яти освітніх вебсайтів на основі уподобань студентів. Результати показали, що контент та навігація були першою та другою бажаними категоріями дизайну, які слід враховувати під час оцінки зручності використання освітніх вебсайтів, тоді як організація/архітектура була найменш важливою категорією. Також результати показали, що існувала статистично значуща різниця між чоловіками та жінками лише щодо однієї категорії: контенту. Жінки вважали цю категорію найважливішою, тоді як чоловіки вважали її другою за важливістю.

Натомість, результати показали, що не було статистично значущих відмінностей між студентами двох вибраних факультетів (факультету інформаційних технологій та наук та факультету економіки та адміністративних наук) щодо відносної важливості розроблених критеріїв на основі їхніх спеціальностей/спеціалізацій. Загалом, результати показали, що більшість студентів були задоволені зручністю використання вебсайтів йорданських університетів. Зокрема, результати показали, що студенти були задоволені контентом та навігацією (зручністю використання) протестованих вебсайтів, але незадоволені дизайном вебсайтів.» А її висновки підкреслюють: «що хоча структура важлива, студенти часто незадоволені самим дизайном – графікою, розташуванням елементів, візуальною привабливістю».

Виходячи з оглянутого можна виокремити основні виявлені проблеми на сайтах ВНЗ:

1. Нечітка або заплутана навігаційна структура.
2. Відсутність адаптивності та невідповідність стандартам WCAG.
3. Відсутність інтерактиву та форм, що обмежує залучення студентів.
4. Повільне завантаження, зрушення макета (CLS, LCP проблеми) – спостерігаються у квитках продуктивності.

Наукові результати підтверджують, що університетські сайти часто не відповідають сучасним вимогам UX/UI та доступності. Методології тестування – від простих опитувань до eye-tracking – рекомендують фокус на чіткій меню-структурі, адаптивності, швидкості та інтуїтивності. Для фронтенд-розробки сайту СНТ важливо врахувати всі ці аспекти: забезпечити зручну навігацію, коректне відображення на різних пристроях, мінімум CLS, оптимізацію швидкості та інтерактивні елементи. Це дозволить створити сучасну, доступну і ефективну платформу для студентської аудиторії.

РОЗДІЛ 2. ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ

2.1 Структура інформаційних потоків у системі

Фронтенд-частина сайту Студентського наукового товариства (СНТ) – це інтерфейсна оболонка, через яку користувач взаємодіє із системою. Вона забезпечує зручне та візуально структуроване подання інформації, а також ініціює обмін даними між клієнтом та сервером. Ключову роль у цьому відіграють **інформаційні потоки**, які реалізуються через запити до бекенду, обробку відповідей та відображення динамічного контенту.

Класифікація інформаційних потоків у фронтенді. У межах фронтенд-архітектури інформаційні потоки поділяються на:

- **Вхідні потоки** – дані, які надходять від користувача (введення у форми, натискання кнопок, вибір зі списків);
- **Вихідні потоки** – дані, отримані з бекенду та виведені на екран у вигляді HTML-елементів;
- **Запити до бекенду** – асинхронні запити (AJAX / fetch / axios) до PHP API, які формуються фронтендом;
- **Події інтерфейсу** – локальні реакції на дії користувача (випадаючі меню, модальні вікна, адаптація під екран).

Приклади реалізації потоків у розділах сайту

- **Головна сторінка:**
  - Отримання списку останніх новин та подій через GET-запити.
  - Виведення контенту за допомогою динамічної генерації компонентів React/Vue.
  - Інформаційний потік: `Користувач → Запит → Сервер → JSON → Відображення`.
- **Адмін панель:**
  - Після авторизації виконується запит користувацьких даних → Виводиться персоналізований інтерфейс.
  - Потоки: `JWT токен → Авторизація → Запит → Дані → Відображення`.

Схема інформаційних потоків нашої системи відповідає класичній схемі, що представлена на рисунку 2.1.

Типова структура потоку запиту і відповіді (табл. 2.1).

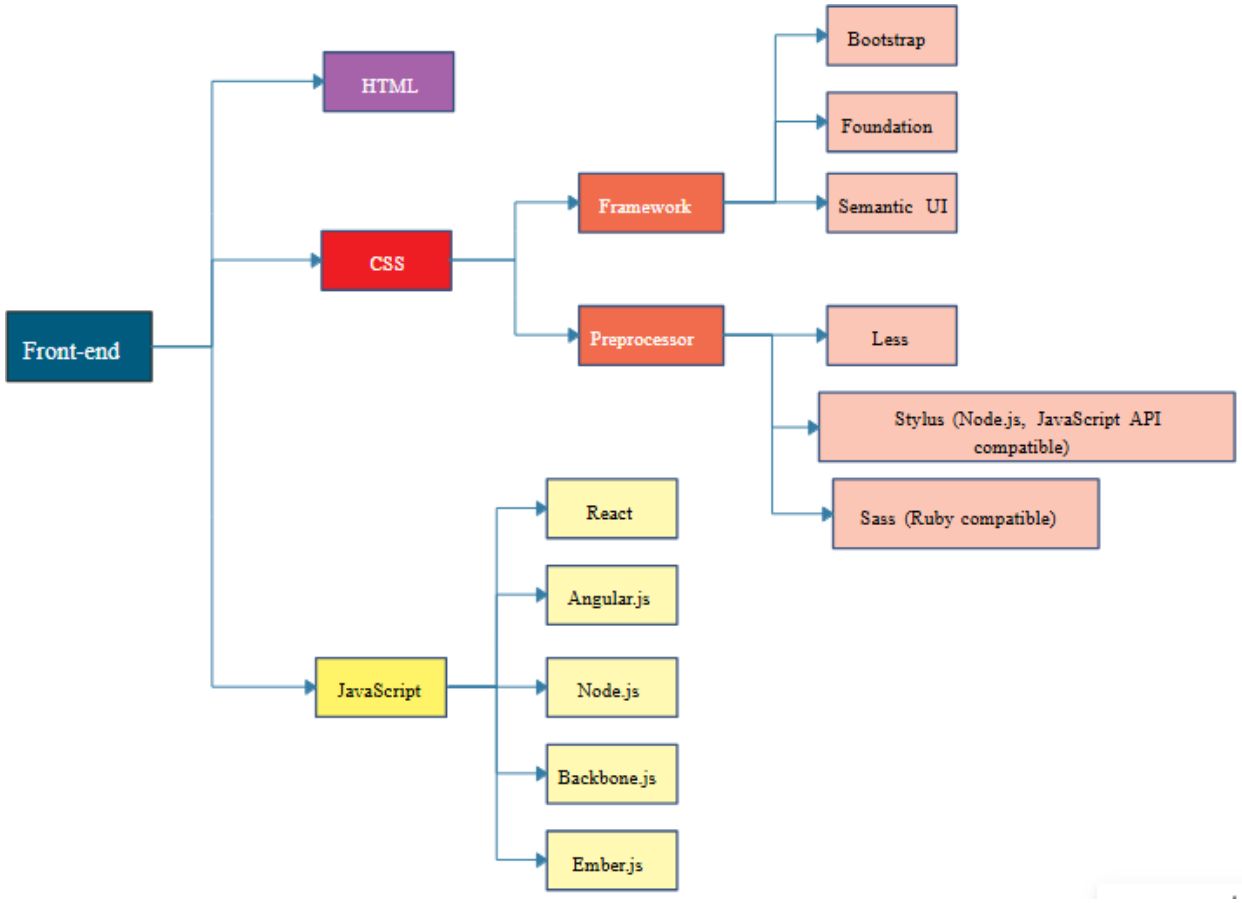

Рисунок 2.1 – Класична схема інформаційних потоків (front-end)

До технологій реалізації інформаційного обміну належать:

- **React/Vue** – забезпечує реактивну побудову інтерфейсу;
- **Fetch API або Axios** – для HTTP-запитів до бекенду;
- **JSON** – формат передачі даних між фронтендом і сервером;
- **CSS/Media Queries** – відповідають за адаптивну реакцію на тип пристрою.

Таблиця 2.1 – Типова структура потоку запиту і відповіді

Етап	Тип даних	Формат	Призначення
Користувач вводить	Текст/дані	HTML Form	Ініціація дії
Запит	JSON	POST / GET	Відправка запиту до API
Обробка	PHP Script	SQL + PHP	Обробка та звернення до БД
Відповідь	JSON	Status + Data	Повернення результату
Відображення	HTML/JSX	DOM Rendering	Показ результату на екрані користувача

Основні блоки, що формують інформаційні потоки:

- **Компоненти відображення:** картки новин, таблиці заходів, банери;
- **Форми введення:** зворотний зв'язок, заявка на участь, реєстрація;
- **Сервіси взаємодії:** API-обгортки, класи для взаємодії з бекендом;
- **Контейнери стану:** глобальний стан (Context/Store) для збереження стану сесії, користувача, навігації.

Інформаційна система вебсайту Студентського наукового товариства (СНТ) є середовищем для збору, обробки, зберігання та представлення інформації різним групам користувачів. Основу інформаційної моделі складають інформаційні потоки, що формуються між користувачем, клієнтським інтерфейсом, серверною логікою та базою даних.

У структурі СНТ-сайту можна виділити три основні типи учасників інформаційного обміну:

Адміністратор (керує сайтом, додає новини, обробляє заявки);

Студент / учасник СНТ (реєструється на заходи, переглядає матеріали, подає заявки);

Гість сайту (переглядає відкриту інформацію, може звернутись через форму зворотного зв’язку).

Інформаційні потоки мають двосторонній характер: з одного боку – надходження даних у систему, з іншого – передача підготовленої інформації користувачеві (табл. 2.2).

Таблиця 2.2 – Узагальнення інформаційних потоків

Джерело / Споживач	**Тип інформації**	**Напрям руху**	**Засіб передачі**
Адміністратор	Новини, заходи, документи	У систему	Панель керування
Студент	Заявки, коментарі, звернення	У систему	Форма введення, реєстрація
Система	Новини, події, відповіді	До користувача	Вебінтерфейс, e-mail-сповіщення
База даних	Дані користувачів, подій, заявок	До системи	PHP-запити через сервер
Гість	Перегляд відкритої інформації	До користувача	Вебінтерфейс

У системі реалізовано клієнт-серверну модель, де дані обробляються на сервері (PHP), а результати передаються клієнту (браузеру) через API-запити або безпосередню генерацію HTML. Користувач взаємодіє з інтерфейсом, який, у свою чергу, надсилає запити на сервер, наприклад: подання заявки, вхід до акаунта, реєстрація на подію.

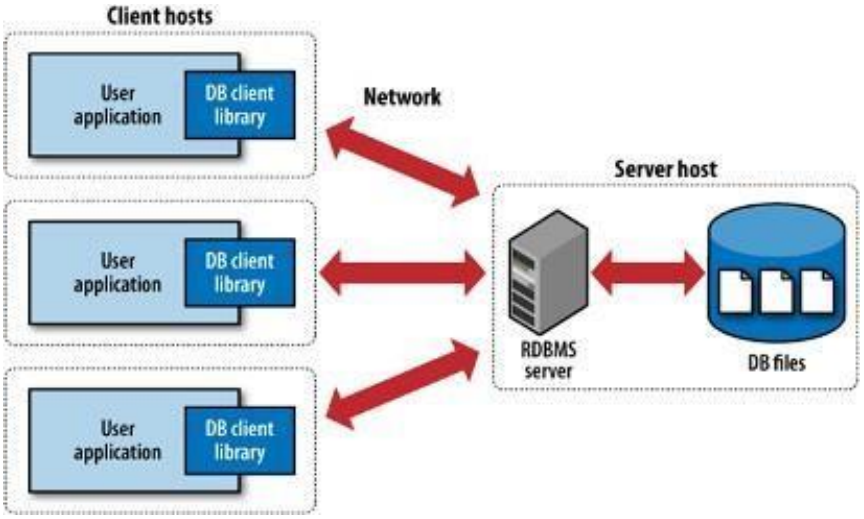

Рисунок 2.2 – Структурна схема обміну даними: Користувач – Сервер – БД

Для зручності адміністрування передбачено окремий інформаційний потік**,** що передає дані з форми керування контентом до бази даних, забезпечуючи оновлення структури сайту в режимі реального часу.

Загалом, побудова ефективної моделі інформаційного обміну дозволяє реалізувати злагоджену роботу системи з можливістю масштабування та підтримки різних ролей користувачів.

Фронтенд-система не лише відображає дані, а й формує динамічну взаємодію, що забезпечує інтуїтивну навігацію, швидку реакцію на дії користувача та чіткий зворотний зв’язок. Такий підхід гарантує ефективність та масштабованість вебдодатку СНТ.

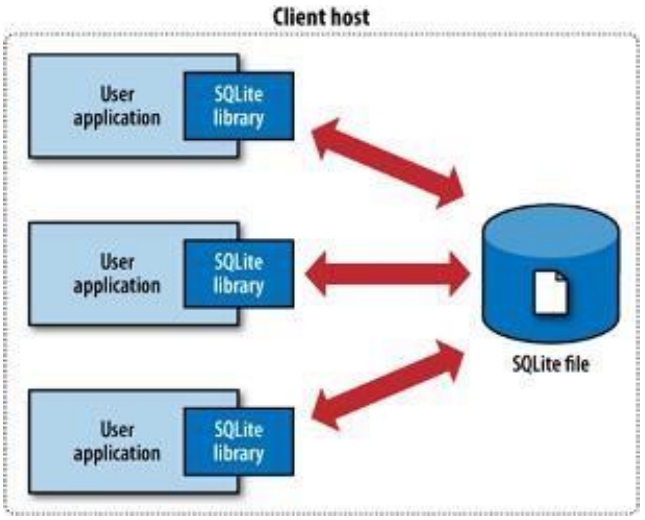

Рисунок 2.3 – Нова структурна схема обміну даними: Користувач – Сервер – БД

2.2 Дерево проблем і дерево рішень

2.2.1 Дерево проблем

Дерево проблем (рис. 2.4), що стосується розробки фронтенду сайту Студентського наукового товариства, виявляє три ключові труднощі. По-перше, відсутність гнучкої адміністративної панелі ускладнює керування контентом і зміну шаблонів без залучення програміста. По-друге, є проблеми з взаємодією між клієнтською та серверною частинами – нестандартизовані запити, слабка обробка помилок та загрози безпеки. По-третє, інтерфейс побудований без уніфікованих компонентів, що ускладнює масштабування та повторне використання коду. Ці проблеми обмежують зручність роботи для адміністрації та функціональність для користувачів. Також знижується якість підтримки сайту у майбутньому.

Головна проблема:

- неефективна та фрагментарна реалізація фронтенд-частини сайту СНТ, яка ускладнює користування, адміністрування і масштабування.

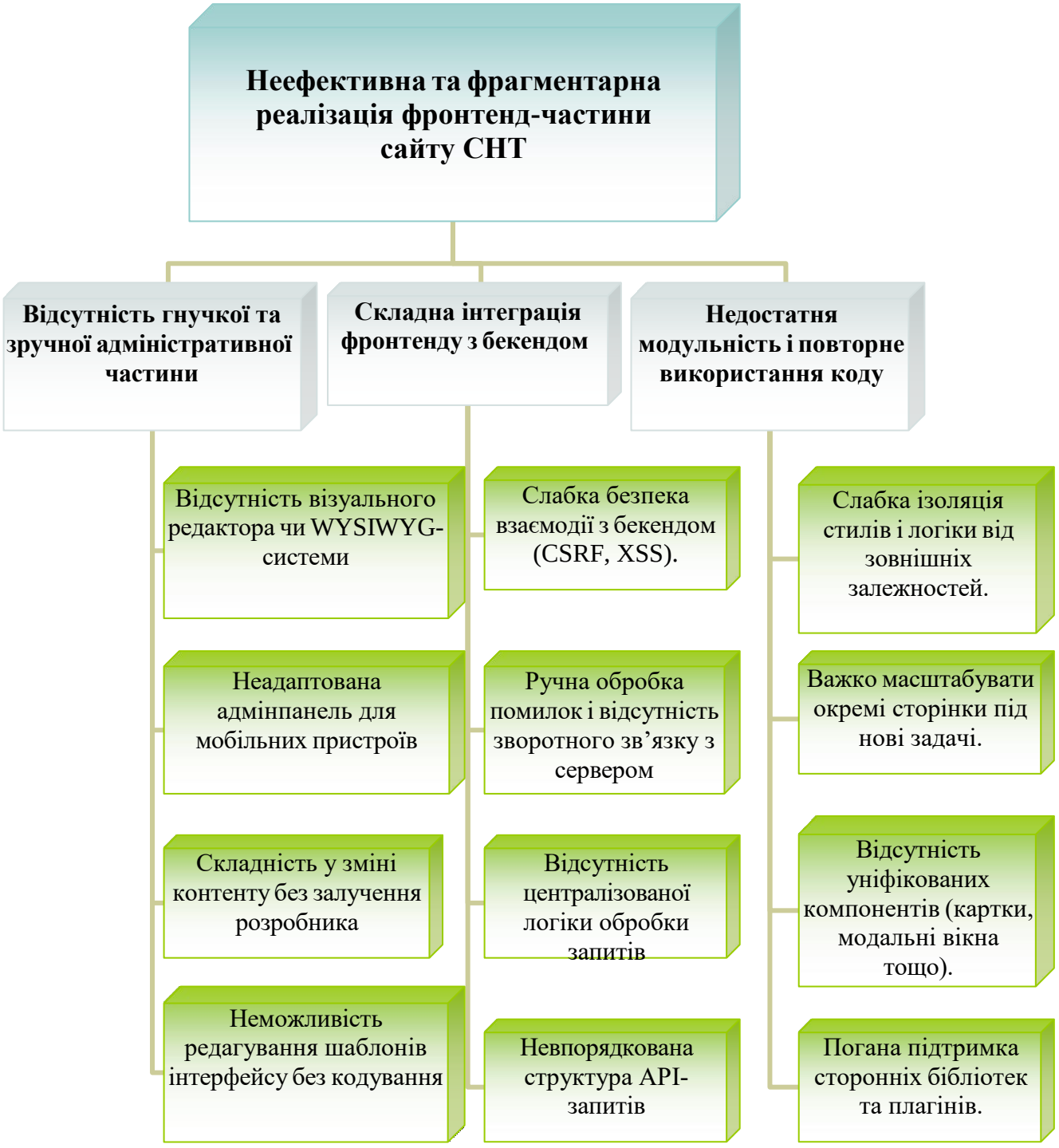

Рисунок 2.4 – Дерево проблем

Основна проблема 1: Відсутність гнучкої та зручної адміністративної частини

- **Проблема 1.1:** Складність у зміні контенту без залучення розробника.
- **Проблема 1.2:** Відсутність візуального редактора чи WYSIWYG-системи.
- **Проблема 1.3:** Неадаптована адмінпанель для мобільних пристроїв.
- **Проблема 1.4:** Неможливість редагування шаблонів інтерфейсу без кодування.

Основна проблема 2: Складна інтеграція фронтенду з бекендом

- **Проблема 2.1:** Невпорядкована структура API-запитів.
- **Проблема 2.2:** Відсутність централізованої логіки обробки запитів.
- **Проблема 2.3:** Ручна обробка помилок і відсутність зворотного зв'язку з сервером.
- **Проблема 2.4:** Слабка безпека взаємодії з бекендом (CSRF, XSS).

Основна проблема 3: Недостатня модульність і повторне використання коду

- **Проблема 3.1:** Відсутність уніфікованих компонентів (картки, модальні вікна тощо).
- **Проблема 3.2:** Погана підтримка сторонніх бібліотек та плагінів.
- **Проблема 3.3:** Важко масштабувати окремі сторінки під нові задачі.
- **Проблема 3.4:** Слабка ізоляція стилів і логіки від зовнішніх залежностей.

2.2.2 Дерево рішень

Дерево рішень (рис. 2.5) пропонує комплексний підхід для усунення виявлених недоліків. Запровадження сучасної адаптивної адмін-панелі дозволить редагувати контент, події, шаблони без складного кодування. Уніфікація API-запитів, використання JSON і обробка помилок на фронтенді підвищать стабільність та безпеку. Компонентний підхід у фреймворках типу React або Vue дозволить створювати гнучкі та масштабовані модулі інтерфейсу. Також передбачається використання сучасних CSS-рішень для стилізації та сторонніх плагінів для форм, календарів, каруселей. Загальна мета – створити зручний, надійний і сучасний вебінтерфейс, який відповідатиме потребам студентської спільноти та адміністрації СНТ.

Головна мета:

- Розробити зручний, адаптивний та масштабований фронтенд сайту СНТ із гнучким адмініструванням, надійною серверною інтеграцією та підтримкою повторно використовуваних компонентів (рис. 2.5).

Рисунок 2.5 – Дерево рішень

Ціль 1: Розробити інтуїтивно зрозумілу адміністративну панель

- **Завдання 1.1:** Впровадити легку адмін-систему з WYSIWYG-редактором (наприклад, TinyMCE, CKEditor).
- **Завдання 1.2:** Додати адаптивну верстку для мобільних пристроїв.
- **Завдання 1.3:** Створити окремий модуль редагування шаблонів (CSS/JS).

- **Завдання 1.4:** Передбачити можливість керування структурою сайту без кодування.

Ціль 2: Забезпечити надійну та стандартизовану інтеграцію з бекендом

- **Завдання 2.1:** Розробити чітку REST API структуру (із назвами, версіями і т.п.).
- **Завдання 2.2:** Впровадити глобальні методи fetch/axios з обробкою помилок.
- **Завдання 2.3:** Використовувати токенізацію для захисту API та інших приватних даних.
- **Завдання 2.4:** Застосувати валідацію на клієнтському рівні до відправки запиту.

Ціль 3: Створити модульну архітектуру фронтенду

- **Завдання 3.1:** Використати фреймворк (React/Vue) для компонентного підходу.
- **Завдання 3.2:** Винести спільні блоки в окремі компоненти (Header, Footer, Modal).
- **Завдання 3.3:** Використовувати Tailwind або CSS Modules для ізоляції стилів.
- **Завдання 3.4:** Інтегрувати популярні плагіни (форми, каруселі, календарі) з мінімальною прив’язкою.

2.3 Формалізація функцій та модулів сайту

Для реалізації функціональних можливостей сайту СНТ було проведено формалізацію основних дій, які мають виконуватись у межах системи. Це дозволяє спроєктувати структуру модулів вебдодатку таким чином, щоб вона була логічною, розширюваною та узгодженою з потребами кінцевого користувача або одміністротора сайту.

Основні функціональні блоки сайту розподілено на фронтенд-модулі (інтерфейс користувача) та бекенд-модулі (логіка обробки даних) представлено у таблиці 2.3.

Таблиця 2.3 – Основні функціональні блоки фронтенд частини сайту

Назва	**Тип**	**Опис функціоналу**
Головна сторінка	Статична сторінка	Вивід новин, подій, інтро про СНТ, посилання на важливі розділи
Сторінка “Про нас”	Фронтенд	Історія СНТ, опис команд, наукові керівники
Новини і заходи	Фронтенд/Бекенд	Список новин і подій, перегляд, подання заявки на участь
Сторінка “Документи”	Фронтенд/Бекенд	Доступ до положень, протоколів, статуту, завантаження PDF
Особистий кабінет	Фронтенд/Бекенд	Авторизація, перегляд особистих подій, управління заявками
Панель адміністратора	Бекенд	Додавання/редагування новин, подій, документів, керування користувачами

Основу математичного забезпечення складає система класифікації ролей користувачів **та** доступу до ресурсів. Для цього використовується концепція ACL (Access Control List), яка реалізується через рівні прав доступу. Наприклад:

- **Гість:** доступ лише до перегляду публічної інформації.
- **Зареєстрований користувач:** доступ до заявок, подій, персонального кабінету.
- **Адміністратор:** повний контроль над контентом та користувачами.

Умовні залежності між модулями та їхніми функціями можна представити у вигляді графа функціональних переходів, який демонструє, як користувач переходить між модулями системи, рис. 2.5.

Також формалізація включає опис структури запитів до сервера, зокрема RESTful API:

- GET /events – отримати список подій
- POST /register – надіслати заявку
- PUT /event/:id – оновити подію (адміністратор)

Формалізація дозволяє не лише чітко розмежувати функції сайту, але й забезпечити ефективну інтеграцію фронтенду з бекендом через стандартизовані запити і структури відповідей.

Карта сайту (sitemap) – це структурована схема, яка відображає ієрархію сторінок вебсайту та їхні зв’язки між собою (рис. 2.6). Вона використовується для організації контенту та полегшення навігації як для користувачів, так і для пошукових систем. Основна мета карти сайту – забезпечити зручний огляд усіх доступних розділів ресурсу, особливо якщо сайт містить велику кількість сторінок, категорій або підкатегорій.

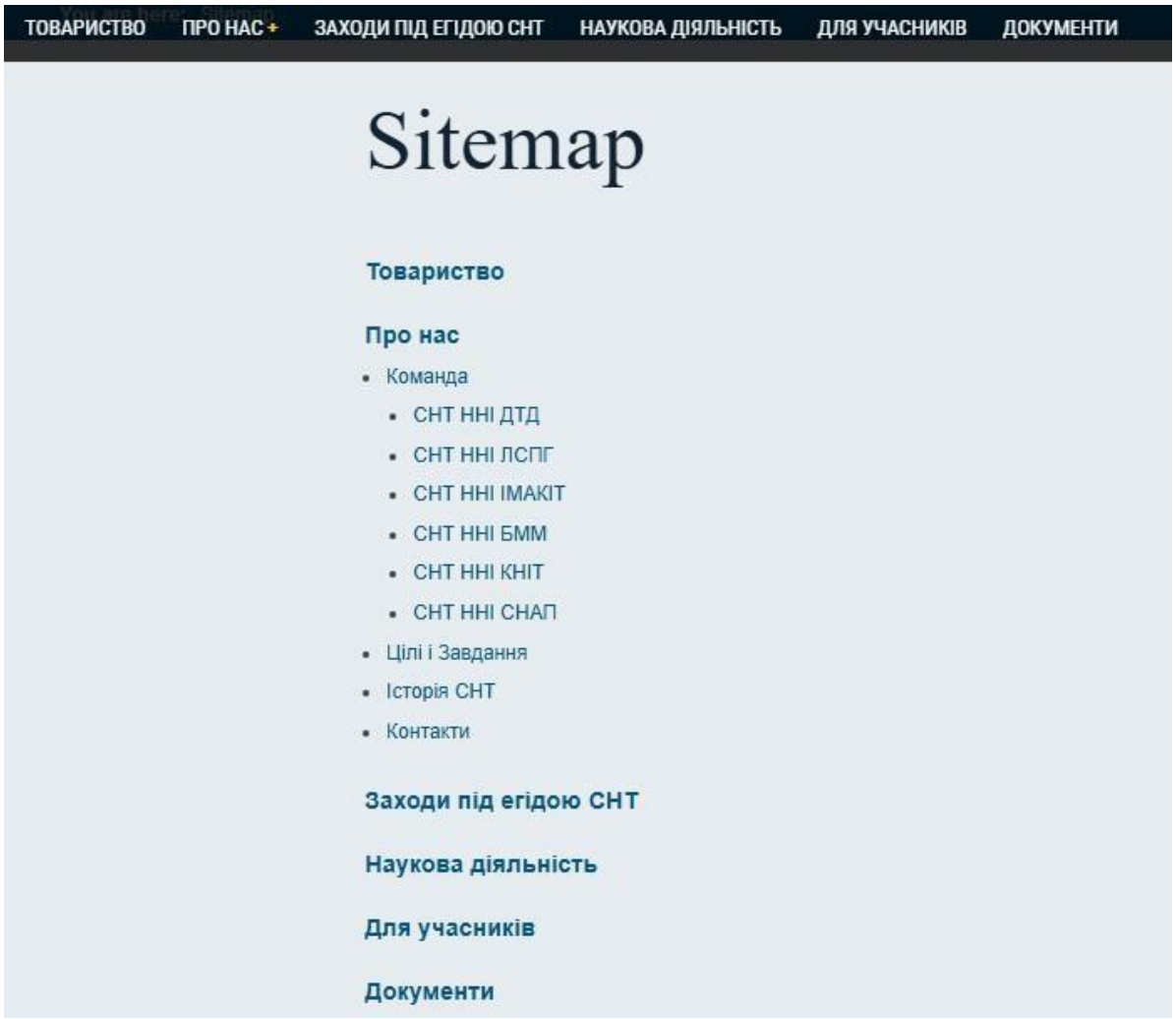

Рисунок 2.6 – Мапа сайту

Існує два основні типи sitemap: візуальна (структурна) – для планування й дизайну сайту, та XML-карта – спеціальний файл, який читається пошуковими роботами. XML sitemap полегшує індексацію контенту, дозволяючи Google, Bing та іншим пошуковикам швидше й точніше знаходити нові або оновлені сторінки.

Візуальна карта сайту використовується на етапі проектування для розробників, дизайнерів та клієнтів, щоб узгодити структуру перед реалізацією.

У навчальних проєктах, зокрема на прикладі сайту студентського наукового товариства, карта сайту допомагає чітко визначити, які сторінки будуть статичними, а які – динамічними, які дані змінюються через CMS або панель адміністратора. Вона також є підґрунтям для реалізації навігації, маршрутизації (у фреймворках на зразок React чи Vue) та формування шаблонів. Sitemap може бути корисною не лише на початковому етапі розробки, а й при тестуванні чи SEO-оптимізації. У сучасному веброзробленні її часто створюють автоматично або у вигляді JSON-файлу.

Карта сайту має особливу важливість для вебресурсу Студентського наукового товариства (СНТ), оскільки вона дозволяє систематизувати велику кількість інформаційних розділів, які стосуються діяльності, наукової роботи, заходів, документів і взаємодії з користувачами. У межах такого сайту передбачається як статичний, так і динамічний контент, тому наявність чітко побудованої карти допомагає як студентам, так і адміністраторам швидко орієнтуватися в структурі.

Для студентів карта сайту є основою зручної навігації: вона дає змогу без зусиль знаходити потрібні розділи, реєструватися на заходи, подавати заявки або переглядати архіви наукових робіт. Для адміністратора – це логічний каркас для управління контентом, який охоплює новини, події, документи, заявки тощо. Крім того, така карта слугує підґрунтям для побудови маршрутизації в SPA-додатках або в CMS-системі, що спрощує реалізацію функціоналу.

З технічної точки зору, XML-карта сайту дозволяє покращити індексацію ресурсу в пошукових системах, що є важливим для публічності діяльності СНТ та залучення нових учасників. Візуальна ж карта використовується на етапі проєктування – щоб погодити структуру сайту з реальними потребами цільової аудиторії та відповідністю стандартам UX/UI. У сайті СНТ, що об’єднує різні підрозділи й напрями наукової діяльності, така карта дозволяє уникнути

дублювання сторінок, надлишковості, а також покращує логіку внутрішніх переходів.

Отже, карта сайту є не просто додатковим інструментом, а критично важливим елементом для ефективного функціонування, підтримки та масштабування проєкту СНТ.

2.4 Модульність та повторне використання коду

Модульність у веброзробці – це принцип побудови програмного коду у вигляді окремих частин (модулів), кожен з яких виконує конкретну функцію. Наприклад, окремий модуль відповідає за форму реєстрації, інший – за виведення новин, ще один – за шапку сайту. Завдяки цьому можна легко змінювати або вдосконалювати частину сайту, не зачіпаючи всю систему.

Повторне використання коду – це можливість застосовувати вже створені модулі або компоненти в різних частинах сайту або навіть у різних проєктах без необхідності переписування. Це значно скорочує час розробки, зменшує кількість помилок і покращує підтримку проєкту. У фреймворках типу React або Vue компоненти – це стандартна форма модульності: один компонент кнопки, форми або картки можна використати десятки разів (табл. 2.4).

Таблиця 2.4 – Порівняння ефективності при повторному використанні коду

Параметр	**Без повторного використання**	**З повторним використанням**
Час на розробку (1 компонент)	2 години	2 години
Час на повторне використання	2 години × 5 = 10 годин	0.5 години × 5 = 2.5 годин
Ризик помилки при зміні	Високий	Низький (зміни в 1 місці)
Підтримка та оновлення	Важка	Легка
Гнучкість дизайну	Низька	Висока
Загальний час на 5 сторінок	~12 годин	~4.5 години

Повторне використання коду дозволяє працювати швидше, ефективніше та підтримувати сайт без зайвих витрат часу й зусиль. Це особливо важливо у проєктах з великою кількістю повторюваних елементів, як-от сайт СНТ.

Доцільність повторного використання коду має ґрунтовне наукове та інженерне обґрунтування, яке розглядається у багатьох дослідженнях, стандартах програмної інженерії та освітніх курсах з розробки ПЗ. Кілька ключових підтверджень:

1. IEEE Standard for Software Reuse (IEEE Std 1517-2010). Цей стандарт описує підходи до повторного використання коду як до пріоритетної стратегії у розробці програмного забезпечення. Він зазначає, що повторне використання зменшує витрати, покращує якість та скорочує час виходу на ринок.

2. Дослідження в галузі Software Engineering. Наукові роботи [3] доводять, що gовторне використання коду знижує середню кількість помилок на рядок коду. Програмні системи з високим ступенем повторного використання мають кращу масштабованість і підтримуваність. Приклад цитати з дослідження [3] “Software reuse can significantly improve software productivity, quality, and reduce cost of development and maintenance”.

3. Підтримка в фреймворках. Сучасні фреймворки, такі як React, Angular, Vue, архітектурно орієнтовані на компонентність та повторне використання. Це не просто тренд, а найкраща інженерна практика, яка інтегрована в саму структуру цих систем.

4. Метрики якості ПЗ. Повторне використання прямо впливає на такі метрики:

- Coupling/Cohesion**:** компоненти з високим рівнем повторного використання мають низьке зв’язування та високу згуртованість.
- MTTR (Mean Time To Repair)**:** зменшується, бо помилки локалізуються в одному місці.
- Cyclomatic Complexity**:** загалом менша у добре організованому повторно використовуваному коді.

5. Підтвердження в освіті та індустрії. Навіть у навчальних програмах (Coursera, edX, CS50, CS231n тощо) повторне використання коду подається як базовий принцип програмної інженерії. А у великих компаніях (Google, Meta, Microsoft) внутрішні бібліотеки й модулі створюються саме з метою повторного використання.

Повторне використання коду не тільки науково обґрунтоване, а й практично необхідне для будь-якої масштабованої, якісної та ефективної розробки. Це зменшує дублювання, помилки, витрати часу і спрощує командну роботу. Для студентських проєктів, зокрема створення сайту СНТ, цей підхід забезпечує легку підтримку і швидкий розвиток функціоналу.

РОЗДІЛ 3. ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1 Архітектура серверної частини

Систему управління сайтом студентського наукового товариства важко формально ділити на жорсткі шари, проте її компоненти можна представити як шари:

1. Presentation layer (шаблони HTML + CSS + JS)
2. Application layer (логіка PHP, Front Controller)
3. Data layer (зберігання flat files)

Це відповідає класичним принципам багаторівневих архітектур (Layered Architecture) (рис. 3.1).

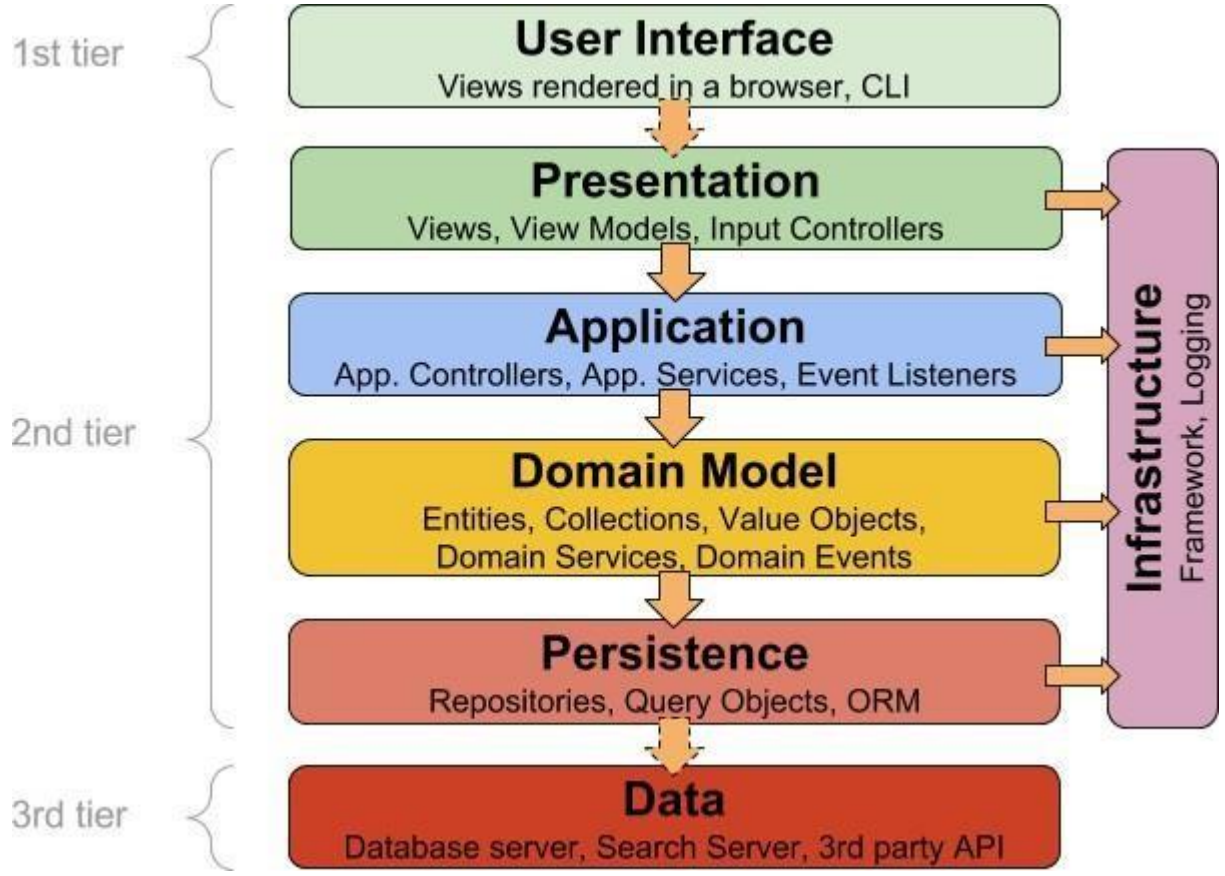

Рисунок 3.1 – Багаторівнева архітектура

Архітектура серверної частини базується на класичній моделі «один файл – одна сторінка» з використанням PHP-скрипта, який діє як фронт-контролер [15]. Головний файл, зазвичай index.php, приймає всі запити від користувача і,

залежно

від URL або параметрів, визначає, який контент слід відобразити. Вміст сайту зберігається у вигляді статичних HTML-файлів або текстових файлів у певних папках, до яких система звертається безпосередньо. Після отримання потрібного файлу PHP зчитує його вміст і обробляє, вставляючи у нього динамічні елементи – меню, заголовки, плагіни та інші компоненти (рис. 3.2).

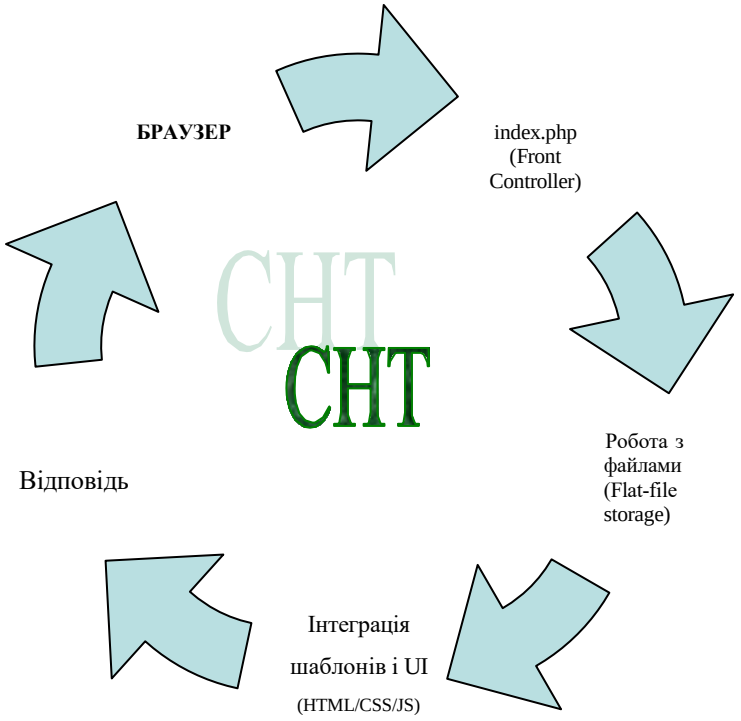

Рисунок 3.2 – Схема представлення контенту

Серверна логіка мінімальна і включає в себе механізми маршрутизації, авторизації адміністратора, обробки форм та інтеграції плагінів. Плагіни додають функціональність без необхідності змінювати ядро, вони підключаються через спеціальні хук-точки в коді. Усі налаштування зберігаються у конфігураційних файлах PHP або простих текстових файлах, що спрощує редагування та перенесення сайту. Обробка сторінок відбувається безпосередньо з файлової системи, тому швидкість відповіді сервера зазвичай висока.

Адміністративна панель працює теж на PHP і містить форму авторизації, інтерфейс для редагування сторінок через WYSIWYG-редактор, управління плагінами та налаштуваннями сайту. Використання PHP-сесій забезпечує безпечний доступ до адмінки. Серверна архітектура проста, проте гнучка, що робить систему легкою для розгортання на будь-якому PHP-сумісному хостингу.

Відсутність складних залежностей робить систему надійною і мало вразливою. Загалом, серверна частина виконує роль координатора між файловою системою, логікою сайту та клієнтським інтерфейсом.

3.2 Опис використаного ПЗ (фреймворки, бібліотеки, утиліти)

Back-end чатсина сайту CHT – це по суті система управління вмістом, яка має вирізнятися своєю простотою завдяки використанню лише найнеобхідніших вебтехнологій. Основною мовою програмування, на якій побудована, є PHP. Вона використовується для обробки запитів, генерування сторінок на сервері та динамічного відображення контенту. Завдяки PHP система працює без потреби в базі даних, що забезпечує високу швидкість та легкість обслуговування.

Уся візуальна структура та оформлення створюється за допомогою HTML і CSS. HTML відповідає за розмітку сторінок, заголовків, абзаців, таблиць та інших блоків вмісту, тоді як CSS використовується для стилізації – задає кольори, шрифти, відступи, позиціювання елементів. Розробник або користувач може легко змінювати вигляд сайту, редагуючи шаблони HTML та таблиці стилів CSS у відповідних файлах теми.

Взаємодію з користувачем на клієнтському боці підтримує мова JavaScript. Вона використовується для реалізації таких функцій, як динамічне меню, підказки, а також інтеграція візуальних редакторів. Іноді JavaScript застосовується для обробки подій на сторінці, наприклад, для відкриття або згортання блоків інформації, щоб покращити зручність користування сайтом.

Замість використання традиційної реляційної бази даних, ми використовуємо файлову систему для збереження вмісту. Уся інформація про сторінки, включно з текстом, зберігається у вигляді звичайних .htm або .txt файлів, які зчитуються та обробляються під час кожного запиту. Такий підхід дозволяє легко створювати резервні копії, переміщувати сайт між серверами, а також уникати налаштування баз даних.

У редакторському інтерфейсі використаємо WYSIWYG-редактор, тобто візуальний редактор, у якому користувач може редагувати сторінки «як є», без знань HTML наприклад TinyMCE.

Щоб зобразити, як саме використовуються ці технології, розглянемо таблицю 3.1 з узагальненням їхніх ролей у системі:

Таблиця 3.1 – Основні технології

Технології та засоби	**Основне призначення**
PHP	Обробка запитів, логіка CMS, генерація HTML
HTML + CSS	Структура сторінки, оформлення, створення шаблонів
JavaScript	Динаміка UI, меню, інтеграція редакторів
Flat-file system	Зберігання контенту у вигляді файлів
WYSIWYG-редактор	Візуальне редагування тексту без знання коду

Використання PHP

Реалізована система побудована на мові програмування PHP, яка виконується на сервері. Вона відповідає за обробку запитів користувача, зокрема – за те, яку сторінку потрібно відобразити. PHP виконує роль логічного ядра системи, забезпечуючи керування вмістом та шаблонами. Сторінки зберігаються у вигляді HTML-файлів, але PHP динамічно вставляє в них меню, заголовки та інші елементи. При кожному відкритті сторінки PHP генерує остаточну HTML-сторінку, яку бачить користувач. Код написаний таким чином, що навіть новачки можуть зрозуміти базову структуру. Система вимагає фреймворків та сторонніх бібліотек PHP, завдяки чому її легко розгорнути на звичайному хостингу. PHP також забезпечує підтримку плагінів, які розширюють функціональність сайту СНТ. Усі параметри конфігурації та налаштування зчитуються і обробляються за допомогою PHP-функцій.

Завдяки об'єктно-орієнтованим та процедурним можливостям PHP, систему можна буде легко масштабувати. Без потреби у складних SQL-запитах, PHP обслуговує повноцінний сайт з кількома розділами. PHP-скрипти також відповідають за захист адміністративної частини сайту через сесії або паролі.

Використання HTML + CSS

Основу візуального оформлення сайту становлять HTML і CSS. HTML відповідає за розміщення тексту, зображень, таблиць, заголовків та блоків на сторінці. Кожна сторінка створюється у вигляді окремого HTML-файлу, який легко редагується вручну. Шаблони сайту також реалізовані на HTML, що дозволяє кастомізувати структуру сайту без спеціального інтерфейсу. CSS використовується для стилізації вмісту: від кольору фону до розміщення елементів. Розробники тем можуть змінювати вигляд сайту, просто редагуючи файл stylesheet.css. Сторінки можуть мати адаптивний дизайн, якщо CSS шаблону враховує медіа-запити. Можна легко додати свої стилі, навіть без знання PHP, і сайт набуде нового вигляду. HTML-структура дуже прозора. Це дозволяє без труднощів вбудовувати сторонні елементи, як-от форми, мапи чи слайдери. HTML-файли легко індексуються пошуковими системами, що позитивно впливає на SEO. Структура HTML також сприяє швидкому завантаженню сторінки навіть на слабких пристроях.

Використання JavaScript

JavaScript в реалізації серверної частини виконує допоміжну роль, зосереджену переважно на клієнтській взаємодії. Його основне завдання – забезпечення інтерактивних елементів, таких як випадаючі меню або динамічне оновлення блоків. Завдяки JavaScript сайт СНТ може інтегрувати сторонні компоненти без перезавантаження сторінки. Найчастіше JavaScript використовується у візуальних редакторах, таких як TinyMCE. Ці редактори дозволяють редагувати сторінки у вигляді, максимально наближеному до остаточного результату. JavaScript також може керувати вкладками, формами, попереднім переглядом вмісту або підтвердженням дій. Підключення скриптів здійснюється або через шаблони, або через плагіни. Розробник має повний контроль над клієнтськими скриптами – їх можна редагувати або додавати власні. У шаблонах JavaScript відповідає за адаптивну навігацію або зміну тем оформлення. Завдяки цьому сайт виглядає сучасно, незважаючи на свою просту

архітектуру. Користувач може розширити JavaScript-функціональність під конкретні потреби без значного програмування.

Концепція Flat-file system

Однією з найголовніших особливостей системи є відмова від бази даних. Увесь вміст зберігається у вигляді окремих файлів на сервері в HTML-форматі. Це означає, що система не потребує встановлення чи налаштування MySQL або SQLite. Завдяки цьому вона швидко встановлюється: достатньо просто завантажити файли на хостинг. Збереження даних відбувається у вигляді plain text, що спрощує резервне копіювання. Редагувати вміст можна як через адмінку, так і безпосередньо через текстовий редактор. Кожна сторінка – це окремий файл, ім'я якого відповідає URL-адресі. Flat-file система знижує ризики, пов'язані з SQL-ін'єкціями. Оскільки доступ до вмісту відбувається напряму з файлової системи, швидкість завантаження висока. Така структура робить сайт надзвичайно стабільним та простим у перенесенні на інший сервер. Необхідність у технічному обслуговуванні мінімальна, адже немає СУБД, що потребує оптимізації. Недоліком може бути ускладнення роботи з великим обсягом вмісту, проте для малих і середніх сайтів цього цілком достатньо.

WYSIWYG-редактор (візуальне редагування)

Для адміністрування сайту CHT використано підтримку візуального редактора WYSIWYG, який дає змогу редагувати сторінки у звичному графічному інтерфейсі. TinyMCE працює на JavaScript і вставляється у вікно редагування сторінки. Користувач бачить сторінку практично такою, якою вона буде після збереження – без необхідності знати HTML. Можна вставляти зображення, таблиці, посилання, змінювати шрифт, колір і структуру тексту. Такий редактор дозволяє зосередитися на вмісті, а не на коді. Інтерфейс редактора схожий на Word або Google Docs, що полегшує адаптацію новачків. Редактори можна змінювати або оновлювати залежно від уподобань. Плагіни TinyMCE дозволяють розширювати можливості редактора без програмування. Результат зберігається як HTML-файл, але користувач цього не помічає. Редактор включає функції перевірки орфографії, вставки медіа та перегляду вихідного коду.

3.3 Реалізація системи

Ддля того щоб адмініструвати сайт СНТ реалізовано багатокористувацький режим. Власне користувачі поділяються на клієнтів – відвідувачів та адміністраторів. Вхід у адміністративну частину та перевірка доспуту відбувається шляхом верифікації введених даних адміністратора (рис. 3.3).

Рисунок 3.3 – Модальне вікно авторизації на сайті

Спочатку функція loginchek() перевіряє коректність введених даних у формі входу (рис. 3.4), що реалізується функцією loginform().

```
function logincheck()
{

function writelog($m)
{

function loginforms()
{
    global $adm, $cf, $print, $hjs, $tx, $onload, $f, $o, $u, $s;

    if ($f == 'login')
    {
        $cf['meta']['robots'] = "noindex";
        $onload.= "self.focus();document.login.passwd.focus();";
        $f = $tx['menu']['login'];
        $o.= '
<div id="cmsimple_loginformBG" class="cmsimple_loginformBG">
<div id="cmsimple_loginform" class="cmsimple_loginform">
<div class="cmsimple_close"><a href="./?' . $u[$s] . '">X</a></div>
<h1>' . $tx['menu']['login'] . '</h1>
<div>
' . str_replace('<br><br><br><br>','<br><br>', str_replace("\r",'<br>',str_replace("\n",'<br>',$tx['login']['warning']))) . '
</div>
<form id="login" name="login" action="./?' . $u[$s] . '" method="post">
<input type="hidden" name="login" value="true">
<input type="hidden" name="selected" value="' . $u[$s] . '">
' . $tx['login']['user_optional'] . ': <br>
<input type="text" name="user" id="user" value=""><br>
' . $tx['login']['password'] . ':<br>
<input type="password" name="passwd" id="passwd" value=""><br>
<br>
<input type="submit" name="submit" id="submit" class="submit" value="' . $tx['login']['login'] . '">
</form>
</div>
```

Рисунок 3.4 – Реалізація функцій для авторизації на сайті

3.4 Побудова дерева сторінок

Побудова дерева сторінок є ключовим етапом організації структури сайту Студентського наукового товариства (СНТ). Для нашого сайт структура дерева сайту зберігатиметься у вигляді одного HTML-файлу content.htm. Вміст кожної сторінки реалізуємо позначеннями за допомогою спеціальної розмітки заголовків (<h1>, <h2> тощо).

Кожен заголовок першого рівня (<h1>) у файлі content.htm створює окрему сторінку, яка автоматично потрапляє до головного меню. Заголовки другого рівня (<h2>) і нижче формують ієрархію підсторінок. Таким чином формується повноцінне дерево навігації, яке система розпізнає без необхідності додаткового програмування або створення бази даних (рис. 3.5).

```
<h1>Про нас</h1>
Тут розміщується контент розділу.

<h2>Історія товариства</h2>
Опис історії СНТ...

<h2>Команда</h2>
Список учасників, фото тощо.

<h1>Наукова діяльність</h1>
Контент про дослідницькі напрями...
```

Рисунок 3.5 – Дерево сторінок із заголовків

У цьому прикладі автоматично згенерується головне меню з пунктами "Про нас" і "Наукова діяльність", а підпункти “Історія товариства” та “Команда” будуть відображені як вкладені сторінки або доступні через внутрішні посилання.

Особливістю є те, що побудова дерева сторінок поєднана з візуальним розмітковим підходом, де контент одночасно є і даними, і структурою. Це дозволить адміністраторам без спеціальних знань легко змінювати структуру сайту, просто редагуючи заголовки та порядок у файлі контенту через вбудований редактор.

Для сайту СНТ це зручно тим, що кожен розділ – «Новини», «Документи», «Контакти», «Події» тощо - будуть оформлені як окрема сторінка (<h1>) з підрозділами (<h2>), що відповідає інтуїтивній навігації (рис. 3.6) для студентів та гостей сайту.

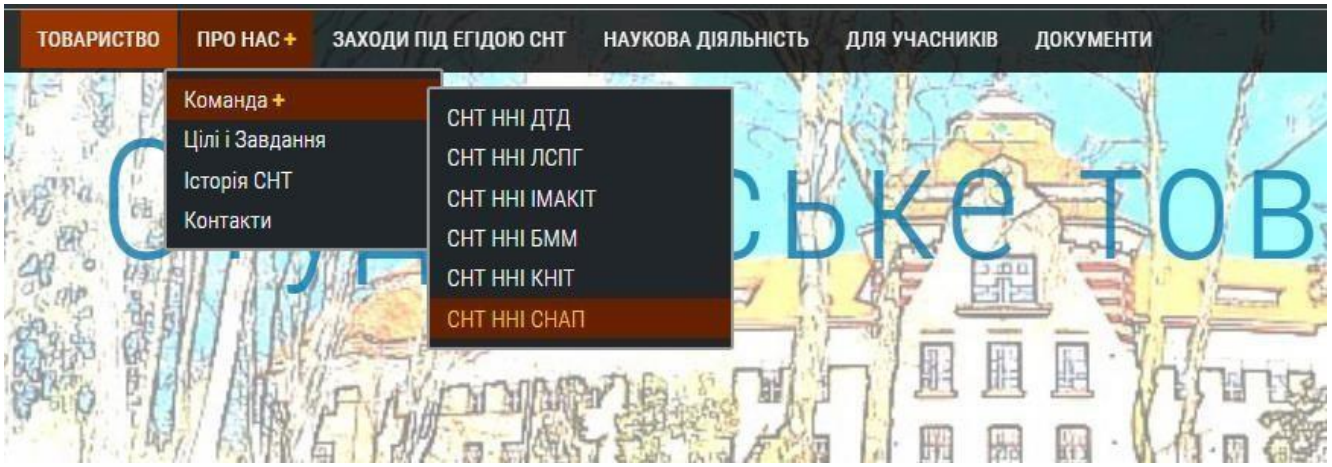

Рисунок 3.6 – Меню сайту

Завдяки цій системі, дерево сторінок (рис. 3.7) водночас є логічною, візуальною та функціональною картою сайту, яка може легко підтримуватись без залучення програмістів.

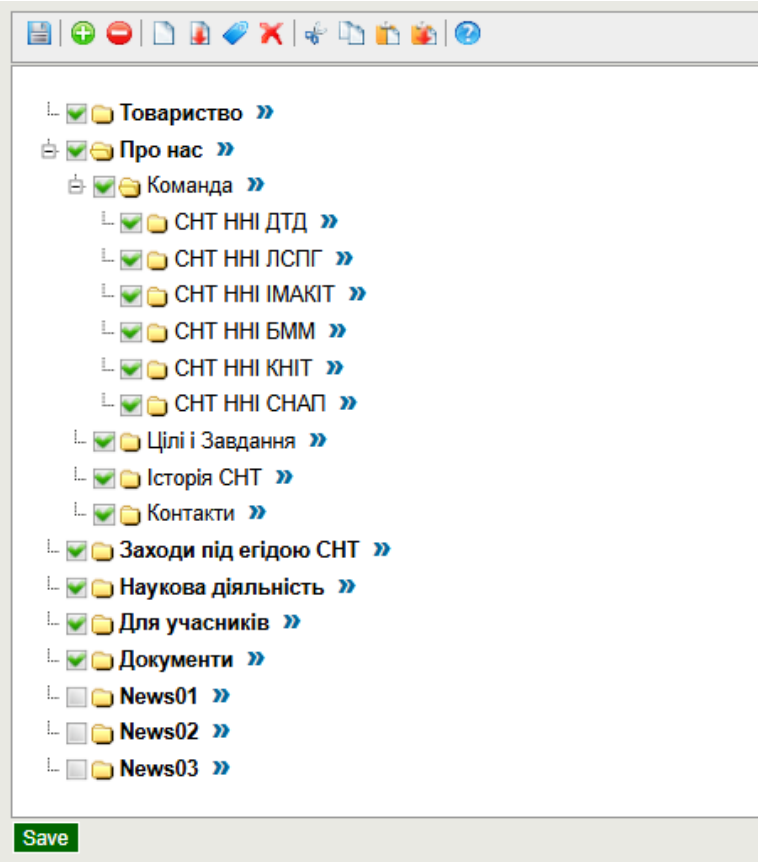

Рисунок 3.7 – Дерево сторінок

3.5 Інтеграція редактораTinyMCE

Для редагування контенту статичних та динамічних сторінок використаємо TinyMCE який вбудовуються у форму редагування. Редактор надає інтерфейс, подібний до текстових редакторів, зі зручними кнопками для форматування, вставки зображень, посилань, списків тощо. Користувач не бачить HTML-коду, але редактор формує сторінку у вигляді чистого HTML. Це особливо корисно для людей без технічних знань, які можуть редагувати сайт без програмування. редактор ініціалізується JavaScript-кодом під час відкриття адмін-панелі. Його функції можна налаштовувати або розширювати плагінами.

Для його підключення слід у<head> нашого HTML-файлу додати посилання на Js-скрипт (рис. 3.5). А саме відображення редактора формується кодом представленим на рисунку 3.8.

```
15      <title>Форма з TinyMCE</title>
16      <!-- Підключення TinyMCE через CDN -->
17      <script src="https://cdn.tiny.cloud/1/no-api-key/tinymce/6/tinymce.min.js" referrerpolicy="origin"></script>
18      <script>
```

Рисунок 3.8 – Підєднання TinyMCE через CDN

Фрагмент коду для інтеграції TinyMCE у проект подано на рисунку 3.9.

```
31  <h2>Редагування контенту (TinyMCE)</h2>
32
33  <form method="post" action="">
34      <textarea id="myeditor" name="myeditor"><?php echo htmlspecialchars($submittedContent); ?></textarea>
35      <br>
36      <button type="submit">Надіслати</button>
37  </form>
38
39  <?php if (!empty($submittedContent)): ?>
40      <h3>Збережений вміст:</h3>
41      <div style="border:1px solid #ccc; padding:10px;">
42          <?php echo $submittedContent; ?>
43      </div>
44  <?php endif; ?>
```

Рисунок 3.9 – Код що реалізує інтеграцію TinyMCE

Після підєднання редактор матиме вигляд і зручний користувацький інтерфейс (рис. 3.10). Користувач вводить текст, і після надсилання він зберігається в змінну та може бути виведений на сторінку або оброблений далі (наприклад, запис у БД).

Меню TinyMCE – це верхня панель редактора, яка дозволяє користувачу отримати доступ до розширених інструментів редагування тексту, аналогічно до меню у текстових процесорах, таких як Microsoft Word або Google Docs.

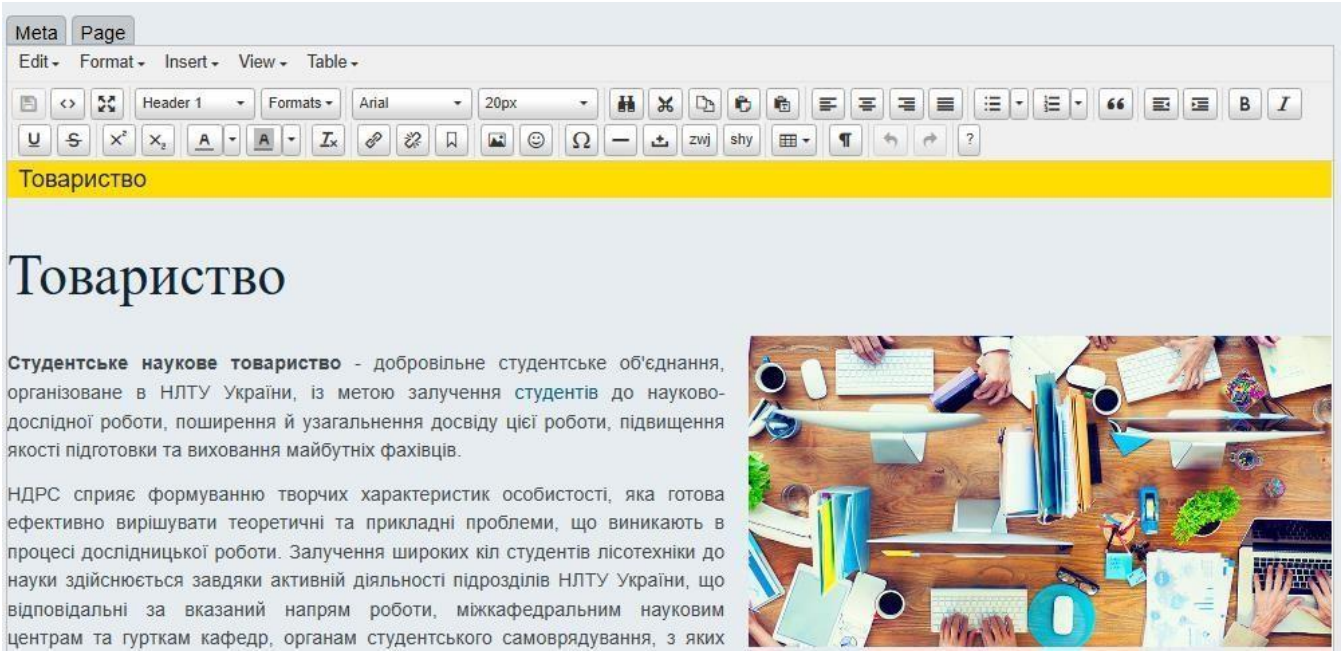

Рисунок 3.10 – Вигляд інтегрованого редактора TinyMCE

Меню TinyMCE – це рядок вкладок у верхній частині редактора, що містить пункти, такі як:

- **Edit** – скасування, повторення, вирізати, копіювати, вставити, пошук і заміна.
- **Format** – стилі абзаців, шрифт, розмір, колір тексту.
- **Insert** – вставлення зображень, посилань, символів тощо.
- **View** – режим перегляду, повноекранний режим, блоки.
- **Table** – вставлення та редагування таблиць.

Після збереження наповненого контенту створюються відповідні сторінки сайту. Вигляд деяких із них представлено на рисунках 3.11–3.12.

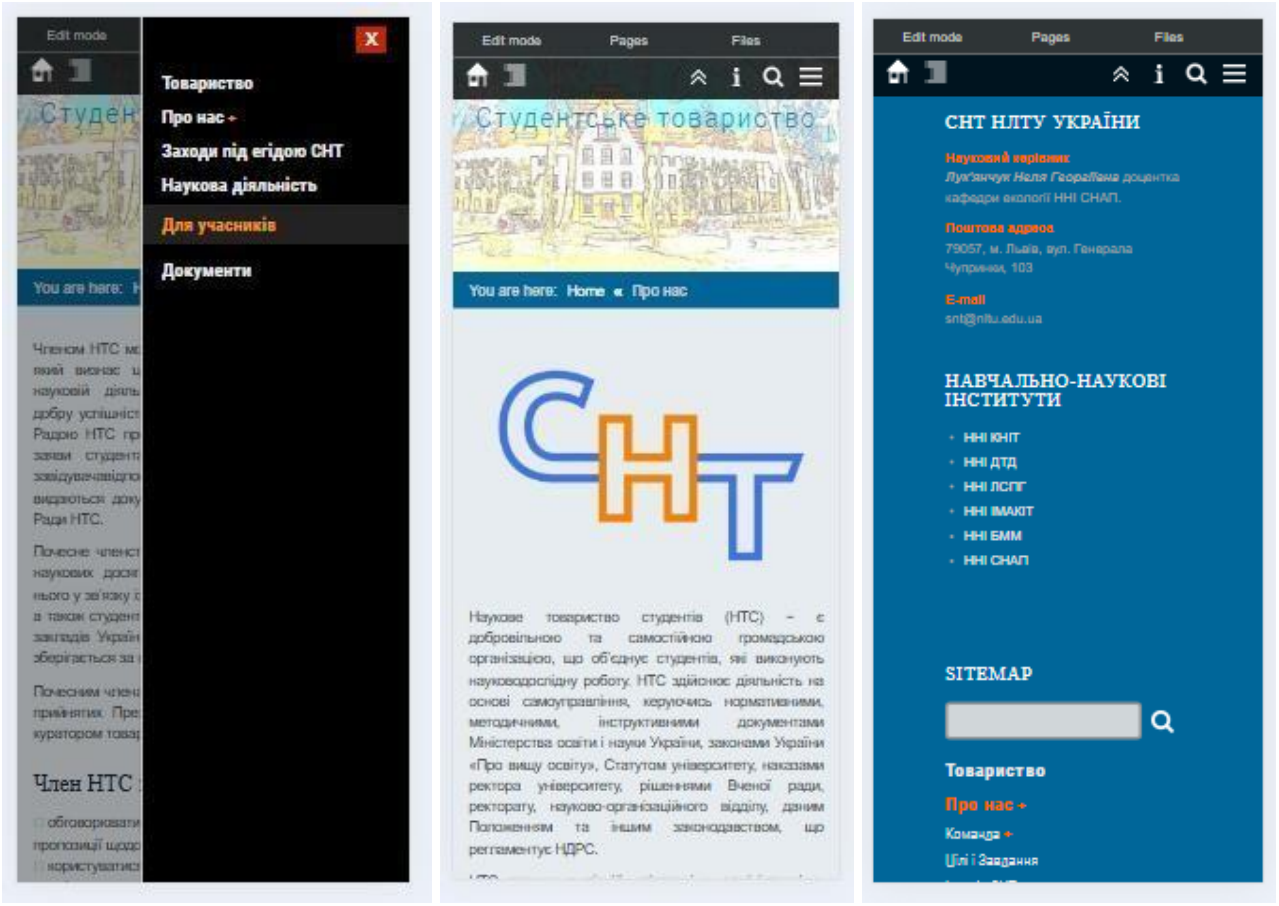

Рисунок 3.11 – Вигляд адаптивних сторінок сайту СНТ

Рисунок 3.12 – Вигляд адаптивних сторінок сайту СНТ

3.6 Пошукова система

Пошук на сайті – це функціональний механізм, який дозволяє користувачам швидко знаходити потрібну інформацію, переглядаючи великі обсяги контенту за ключовими словами чи фразами. Він є важливою частиною будь-якого сучасного вебсайту, особливо освітнього, інформаційного або організаційного, як сайт Студентського наукового товариства (СНТ).

Пошукова система працює за принципом індексації вмісту сторінок і порівняння запиту користувача з текстом, що зберігається у базі даних або файлах сайту. На сайті СТН використаємо файловий пошук через вміст content.htm.

Інтерфейс пошуку включає текстове поле для введення запиту (рис. 3.13), кнопку "Пошук", а також сторінку де буде відображається перелік знайдених сторінок з частиною тексту та посиланням на повний запис.

У проєкті сайту СНТ пошук дозволить швидко знаходити:

- наукові роботи,
- анонси заходів, новини,
- документи товариства.

Таким чином, реалізація якісного пошуку – це не просто технічна опція, а елемент доступності та ефективності взаємодії з сайтом. Програмний код, що реалізований для досягнення ефективного пошуку представлено в ДОДАТКУ Б.

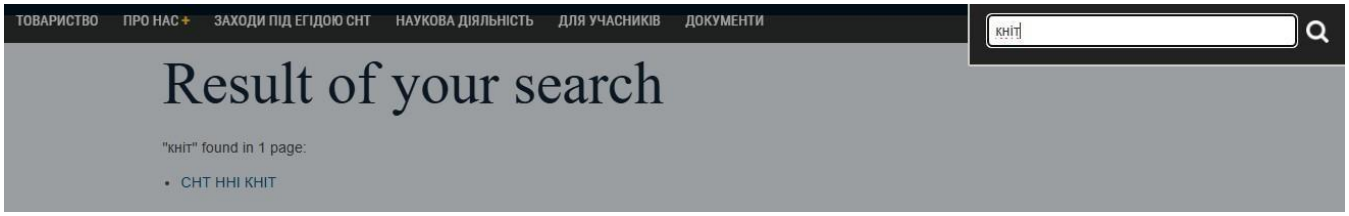

Рисунок 3.13 – Пошукова система

3.7 Технічна реалізація

XAMP – віртуальний сервер

XAMPP - це програмний комплекс, який дозволяє розгорнути повноцінне серверне середовище прямо на локальному комп’ютері. Він є одним із

найпопулярніших засобів для розробки вебдодатків на етапі тестування або створення, коли ще не потрібне оприлюднення в Інтернеті. XAMPP (рис. 3.14) включає всі необхідні компоненти для роботи вебсайту: вебсервер Apache, систему керування базами даних MySQL або MariaDB, інтерпретатор мови PHP та іноді Perl.

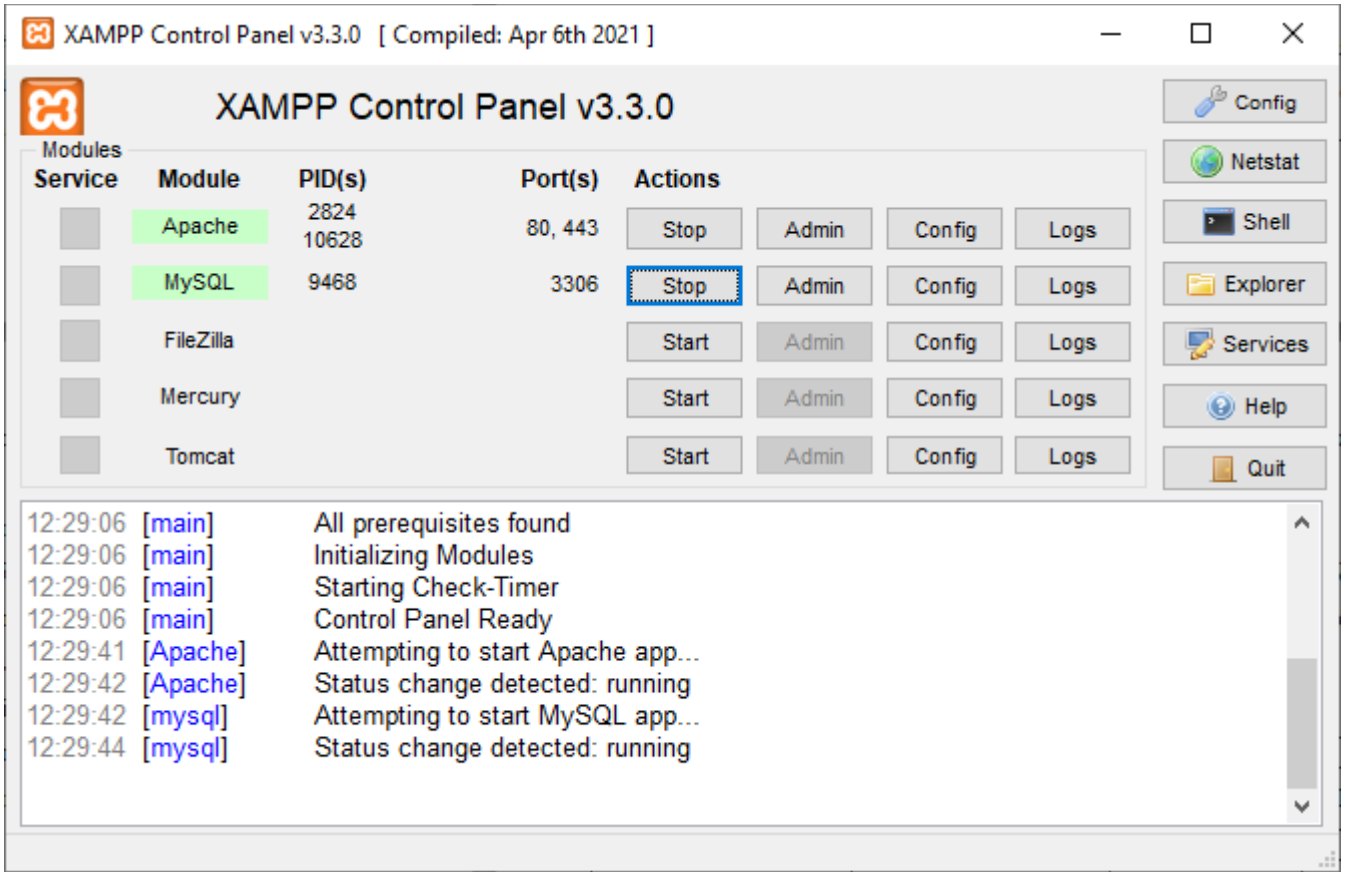

Рисунок 3.14 – Панель управління XAMPP

Завдяки цьому розробник отримує можливість створити локальне середовище, яке імітує справжній вебсервер.

XAMPP має дружній інтерфейс керування, де можна запускати або зупиняти сервери одним кліком. Його можна встановити як на Windows, так і на Linux або macOS, що робить його зручним для кросплатформеної розробки.

Для проєкту сайту Студентського наукового товариства, XAMPP є найбільш зручним та доступним вибором завдяки своїй простоті, функціональності та великій кількості навчальних матеріалів.

Порівняння XAMPP з аналогами подано в таблиці 3.2.

Таблиця 3.2 – Порівняння віртуальних серверів

Параметр	XAMPP	WampServer	Laragon
ОС	Windows, Linux, macOS	Тільки Windows	Тільки Windows
Вебсервер	Apache	Apache	Apache/Nginx
База даних	MySQL / MariaDB	MySQL	MySQL / PostgreSQL
Встановлення	★★★★☆	★★★☆☆	★★★★★
Інтерфейс керування	Графічна панель	Графічна панель	Інтегроване меню
Рівень користувача	Початківець – просунутий	Початківець	Початківець – експерт
PhpMyAdmin	Так	Так	Так
Підтримка SSL, mail тощо	Частково	Обмежено	Повноцінно

Вимоги до сервера хостингу

Щоб коректно працювала система сервер повинен підтримувати мінімальний набір функцій, оскільки ця система не потребує бази даних і побудована на PHP. Основною вимогою є встановлений інтерпретатор PHP, зазвичай версії 5.3 або вище. Сервер повинен бути здатен обробляти запити до файлів .php, а також зчитувати та виводити HTML-сторінки з файлової системи. Також важливо, щоб сервер мав дозволи на читання й запис файлів, оскільки сайт зберігає контент у вигляді текстових файлів. Підтримка роботи з сесіями PHP потрібна для авторизації адміністратора.

Таблиця 3.3 – Технічні вимоги до сервера

Компонент	Вимога
PHP	Від 5.3 до 8.0
Вебсервер	Apache / Nginx / LiteSpeed
База даних	Не потрібна
Права доступу	Запис у файлову систему (chmod 755 / 777 на content/)
Сесії PHP	Увімкнено
mod_rewrite	Рекомендовано (для чистих URL)
Підтримка HTML/CSS	Так (для шаблонів і сторінок)

Система добре працює як на Apache, так і на Nginx. Варто переконатися, що ввімкнено модуль mod_rewrite у випадку Apache, якщо використовуються «чисті» URL. Сервер повинен мати базовий набір прав доступу (табл. 3.3), аби система могла створювати або редагувати файли в папках content, templates, plugins. Загалом, не потребує потужного сервера – її можна успішно розгорнути навіть на бюджетному віртуальному хостингу.

ВИСНОВКИ

У ході реалізації проєкту було розроблено серверну частину вебсайту Студентського наукового товариства (СНТ), яка забезпечує функціональну основу для керування динамічним контентом, взаємодії з базою даних та адміністрування ресурсу. Система дозволяє ефективно зберігати, редагувати й відображати інформацію про наукові заходи, новини, події, користувацькі заявки та інші матеріали. Основна увага була зосереджена на побудові зручної та безпечної архітектури, що дозволяє розмежувати доступи для різних типів користувачів, а також забезпечити стабільну роботу сервісу в умовах змінного навантаження.

Проєкт продемонстрував доцільність застосування класичних інструментів веброзробки, зокрема мови PHP для створення надійного та масштабованого бекенду. Результати реалізації можуть бути адаптовані до подібних інформаційних платформ у сфері освіти та студентського самоврядування. Створена система є основою для подальшої інтеграції з сучасним фронтендом та забезпечення повноцінної роботи сайту СНТ як відкритого інформаційного ресурсу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Міннік К. HTML5 і CSS3 для чайників / Кріс Міннік; пер. з англ. – Київ : Наш Формат, 2020. – 368 с.
2. Фрейн Б. Дизайн на HTML і CSS для будь-яких пристроїв / Бен Фрейн; пер. з англ. – Харків : Фабула, 2019. – 312 с.
3. Фланаган Д. JavaScript. Повний довідник / Девід Фланаган; пер. з англ. – Київ : Діалектика, 2022. – 1152 с.
4. Васильєв О. Програмування мовою PHP: навчальний посібник. / Олексій Васильєв. – Київ: Ліра-К, 2022. – 312 с.
5. Основи PHP: навчальний посібник. / [уклад. викладачі каф. ТК ОНМУ]. – Одеса : Одеський нац. мор. ун-т, 2018. – 128 с.
6. Hasan, Layla. The Usefulness of User Testing Methods in Identifying Problems on University Websites. Journal of Information Systems and Technology Management. 2014. 229-256.
7. Alsaeed R., Hassan M. M. Common Usability Issues on University's Websites // Proceedings of the Fifteenth International Conference on Advances in Computer-Human Interactions (ACHI 2022), 26–30 June 2022, Porto, Portugal. – IARIA, 2022. – P. 67–72.
8. Hasan, Layla. Evaluating the Usability of Educational Websites Based on Students' Preferences of Design Characteristics. The International Arab Journal of e-Technology (IAJeT), 2014, pp. 179-193.
9. КАПЛЮК, Олександр; ПОДУРЕЦЬ, Назар; ЗЛОТКІВСЬКА, Тетяна. Принципи юзабіліті для проектування порталу наукового товариства закладу вищої освіти // Всесвітній конгрес "Авіація в XXI столітті" – "Безпека в авіації та космічні технології". 2023. С. 39–42.
10. Кравченко, О., Матрос, О., Біленко, А. Студентське наукове товариство як осередок підготовки майбутніх фахівців соціальної сфери // Соціальна робота та соціальна освіта. 2022. Вип. 2(7). С. 163–172.

11. HTML [Електронний ресурс] – Режим доступу: https://wikipedia.org/wiki/HTML – Назва з екрана (дата звернення 10.04.2025р).
12. PHP: Hypertext Preprocessor [Електронний ресурс]. – Режим доступу: https://www.php.net/ – Назва з екрана (дата звернення 12.05.2025р).
13. W3Schools. PHP Tutorial [Електронний ресурс]. – Режим доступу: https://www.w3schools.com/php/ – Назва з екрана (дата звернення 15.05.2025р).
14. WebLib – бібліотека для веброзробника [Електронний ресурс]. – Режим доступу: http://weblib.com.ua/projects/2 – Назва з екрана (дата звернення 10.05.2025р).
15. PHP Manual – Filesystem Functions [Електронний ресурс]. – Режим доступу: https://www.php.net/manual/en/ref.filesystem.php
– Назва з екрана (дата звернення 05.05.2025р).

ДОДАТКИ

Додаток А

```
<?php
// Обробка форми після надсилання
$submittedContent = '';
if ($_SERVER["REQUEST_METHOD"] === "POST") {
    if (isset($_POST['myeditor'])) {
        $submittedContent = $_POST['myeditor'];
    }
}
?>

<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <title>Форма з TinyMCE</title>
    <!-- Підключення TinyMCE через CDN -->
    <script src="https://cdn.tiny.cloud/1/no-api-
key/tinymce/6/tinymce.min.js" referrerpolicy="origin"></script>
    <script>
        tinymce.init({
            selector: '#myeditor',
            height: 300,
            menubar: false,
            plugins: 'lists link image code',
            toolbar: 'undo redo | bold italic | bullist numlist | link |
code',
            language: 'uk'
        });
    </script>
</head>
<body>
```

```
<h2>Редагування контенту (TinyMCE)</h2>

<form method="post" action="">
    <textarea id="myeditor" name="myeditor"><?php echo
htmlspecialchars($submittedContent); ?></textarea>
    <br>
    <button type="submit">Надіслати</button>
</form>

<?php if (!empty($submittedContent)): ?>
    <h3>Збережений вміст:</h3>
    <div style="border:1px solid #ccc; padding:10px;">
        <?php echo $submittedContent; ?>
    </div>
<?php endif; ?>

</body>
</html>
```

Додаток Б

```
<?php // utf8-marker = äöü
if (preg_match('/search.php/i', $_SERVER['SCRIPT_NAME']))
    die('Access Denied');

if(!function_exists('mb_strtolower'))
{
    function mb_strtolower($string, $charset = null)
    {
        $string = utf8_decode($string);
        $string = strtolower($string);
        $string = utf8_encode($string);
        return $string;
    }
}

$title = $tx['title']['search'];
$ta = array();
if ($search != '')
{
    $search = mb_strtolower(trim(stsl($search)), 'utf-8');
    $words = explode(' ', $search);

    foreach ($c as $i => $pagexyz)
     {
        if ((!hide($i) || $cf['hidden']['pages_search'] == 'true') &&
!strstr($pagexyz,'_splitToc_'))
            {
             $found  = true;
                 $pagexyz = evaluate_plugincall($pagexyz, TRUE);
             $pagexyz = mb_strtolower(strip_tags($pagexyz), 'utf-8');
             $pagexyz = html_entity_decode($pagexyz, ENT_QUOTES, 'utf-
8');
```

```
                foreach ($words as $word)
                    {
                    if (strpos($pagexyz, trim($word)) === false)
                         {
                        $found = false;
                        break;
                    }
                }
                if (!$found) {continue;}
                $ta[] = $i;
            }
        }

        if(count($ta) > 0){
            $cms_searchresults = "\n" .'<ul class="searchresults">';

          $words = (implode( ",", $words));
            foreach($ta as $i)
               {
                    $cms_searchresults .= "\n\t" . '<li><a href="./?' .
$u[$i] . amp() . 'search=' . urlencode($words) .'">' . $h[$i] .
'</a></li>';
            }
            $cms_searchresults .= "\n" . '</ul>' . "\n";
        }
    }

    $o .= '<h1>' . $tx['search']['result'] . '</h1><p>"' .
htmlspecialchars($search, ENT_COMPAT, 'UTF-8') . '" ';

    if (count($ta) == 0)
    {
        $o .= $tx['search']['notfound'] . '.</p>';
    }
    else
```

```
{
    $o .= $tx['search']['foundin'] . ' ' . count($ta) . ' ';
    if (count($ta) > 1
    )$o .= $tx['search']['pgplural'];
    else
        $o .= $tx['search']['pgsingular'];
    $o .= ':</p>' . $cms_searchresults;
}

?>
```