МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД

«УЖГОРОДСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ»

ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМ

Реєстраційний №\_\_\_\_\_\_

Дата \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**КУРСОВА РОБОТА**

з дисципліни «Розробка інтернет клієнт-серверних систем»

спеціальність 121 Інженерія програмного забезпечення

Тема: «Розробка клієнт-серверної системи для організації дистанційного навчання»

Рекомендована до захисту

“\_\_\_\_” \_\_\_\_\_\_\_\_\_\_\_\_ 2022р.

Робота захищена

“\_\_\_\_” \_\_\_\_\_\_\_\_\_\_\_\_ 2022р.

з оцінкою

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Підписи членів комісії

Виконав

Студент 3-го курсу

Денної форми навчання

Групи 2 СТ

Чума Дмитро Іванович

Науковий керівник :

Проф. Мулеса О.Ю.

Ужгород 2022

**ЗМІСТ**

[ВСТУП 3](#_Toc73312819)

[1 ТЕОРЕТИЧНА ЧАСТИНА 4](#_Toc73312820)

[1.1 Загальні відомості про клієнт-серверну архітектуру 4](#_Toc73312826)

[1.2 Опис предметної області та постановка задачі 5](#_Toc73312826)

[1.3 Огляд засобів розробки 6](#_Toc73312826)

[1.3.1 Скриптова мова програмування PHP 6](#_Toc73312826)

[1.3.2 Мова програмування JavaScript 7](#_Toc73312826)

[1.3.3 Платформа Node.js 8](#_Toc73312826)

[1.3.4 Технологія AJAX 8](#_Toc73312826)

[1.3.5 Фреймворк jQuery 9](#_Toc73312826)

[1.3.6 Система керування базами даних MySQL 9](#_Toc73312826)

[1.4 Огляд аналогів 10](#_Toc73312826)

[2 ПРАКТИЧНА ЧАСТИНА 12](#_Toc73312825)

[2.1 Інформаційна структура 12](#_Toc73312826)

[2.2 Функціональні частини програми 13](#_Toc73312826)

[2.3 Опис алгоритму 14](#_Toc73312826)

[2.4 Інтерфейс та керівництво користувача 15](#_Toc73312826)

[ВИСНОВКИ](#_Toc73312831) 40

[ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ 41](#_Toc73312832)

[Додатки 42](#_Toc73312832)

# ВСТУП

Дистанційне навчання – це сукупність технологій та методик викладання, що дають можливість здобуття освіти без фізичної присутності у навчальному закладі. Дистанційне навчання має багато переваг, але також має і недоліки. Перевагами є те, що навчання можна продовжити в складних випадках, вирішує проблему з переміщенням до навчального закладу. Недоліками є потреба у надійному технічному оснащенні.

Подібне програмне забезпечення дозволяє продовжити процес навчання в складні моменти не перериваючи його. Такі продукти є дуже корисними адже дозволяють користувачам дистанційно виконувати певні завдання. Зазвичай викладач може створювати теоретичні матеріали, завдання та тести. Також він може контролювати весь процес проходження певного матеріалу учнями та оцінювати їх роботи. Самі учні мають можливість переглядати свої результати, залишати коментарі чи запитання та здавати роботу та можливість прикріплювати файли.

Зазвичай такі проекти мають простий та зрозумілий інтерфейс, який не повинен викликати складності у людей для виконання певної операції.

Темою курсового проекту є «Розробка клієнт-серверної системи для організації дистанційного навчання».

Програма має такі функції:

* авторизація та реєстрація;
* перегляд доступних курсів;
* створення, редагування та видалення нових тем, завдань, тестів;
* виконання завдань та прикріплення файлів;
* перевірка виконаних робіт.

Зважаючи на тему курсового проекту цільовою аудиторією будуть викладачі та учні.

# 1 ТЕОРЕТИЧНА ЧАСТИНА

## 1.1 Загальні відомості про клієнт-серверну архітектуру

Архітектура клієнт-сервер є домінуючою концепцією у створенні розподілених мережевих застосунків, що надає можливість взаємодії та обміну даними між ними.

Архітектура передбачає такі основні компоненти:

* набір серверів, які надають інформацію та інші послуги;
* клієнти, які використовують сервіси;
* мережа, яка забезпечує взаємодію між серверами та клієнтами.

Загальноприйнятим є положення, що клієнти та сервери – це програмні модулі. Найчастіше вони знаходяться на різних комп’ютерах, але бувають ситуації, коли обидві програми фізично розміщуються на одному комп’ютері. В таких випадках сервер називається локальним.

Модель клієнт-серверної взаємодії визначається розподілом обов’язків між сервером та клієнтом. Можна відокремити три рівні операцій:

* рівень представлення даних;
* прикладний рівень;
* рівень управління даними.

Дворівнева клієнт-серверна архітектура передбачає взаємодію двох програмних модулів. В залежності від того, як розподіляються операції, розрізняють:

* модель тонкого клієнта – вся логіка застосунку та управління даними зосереджена на сервері;
* модель товстого клієнта – це модель в якій сервер тільки керує даними, а обробка інформації зосереджені на стороні клієнта.

Трирівнева клієнт-серверна архітектура передбачає відділення прикладного рівня від управління даними. Відокремлюється окремий програмний рівень, на якому зосереджується прикладна логіка застосунку.

На сучасному етапі для програмування модулів проміжного рівня використовується PHP, а для управління даними MySQL.

## 1.2 Опис предметної області та постановка задачі

Головним завданням курсового проекту є створення «Клієнт-серверної системи для організації дистанційного навчання». Виходячи із аналізу теоретичного матеріалу було вирішено розбити функціонал на декілька частин:

* авторизація та реєстрація користувачів;
* завантаження та вибір курсів користувача;
* фільтр курсів користувача;
* створення нового курсу;
* приєднання до курсу;
* завантаження головної стрічки курсу;
* навігація по курсу;
* налаштування курсу (тільки власник);
* перегляд користувачів, що приєднались до курсу;
* перегляд результатів;
* створення постів, тем, завдань та тестів;
* редагування та видалення тем, завдань та тестів (тільки викладач)
* прикріплення файлів;
* залишення коментарів до постів, тем та завдань;
* оцінювання робіт учнів (тільки викладач);
* чат між користувачами;
* налаштування акаунту користувача;
* вихід.

Проект повинен мати приємний та зрозумілий інтерфейс, зручне меню до сприйняття людиною будь-якого віку.

## 1.3 Огляд засобів розробки

## 1.3.1 Скриптова мова програмування PHP

# PHP (Hypertext Preprocessor) – це скриптова мова програмування, що була створена для генерації HTML-сторінок на стороні вебсервера. Ця мова є однією з найпопулярніших мов, що використовуються для веброзробок. Більшіть хостинг-провайдерів підтримують PHP.

PHP інтерпретується вебсервером у HTML-код, який потім передається на сторону клієнта. Користувач не бачить коду PHP на відміну від JavaScript, тому що браузер отримує вже готовий html-код. Це погіршує інтерактивність сторінок, але є перевагою у безпеці.

PHP – це мова код, якої можна вбудовувати безпосередньо у html-код сторінок. Обробник PHP починає виконання коду після тегу «<?php» і продовжує до тегу «?>».

У PHP є вбудовані бібліотеки для роботи з MySQL, PostgreSQL, SQLite, mSQL, Oracle, dbm, Hyperware, Informix, InterBase, Sybase.

Завдяки стандарту ODBC можна підключатися до всіх БД, до яких існує драйвер.

Мова поєднує в собі переваги Perl і C та спеціально спрямована для роботи в інтернеті. Багато конструкцій у мові запозичені з Perl та C.

PHP є інтерпретованою мовою, що дозволяє оброблювати сценарії з достатньо високою швідкістю. Продуктивність PHP є достатньою для створення серйозних вебзастосунків.

До недоліків можна віднести:

* всі змінні починаються символом $;
* складны назви поширених функцій;
* не підтримується Unicode до версії 6.0;
* непередбачуваність нових версій.

## 1.3.2 Мова програмування JavaScript

# JavaScript – це динамічна, об’єктно-орієнтована прототипна мова програмування. Найчастіше цю мову використовують для створення сторінок, які надають можливість на стороні клієнта взаємодіяти з користувачем, керувати браузером, обмінюватися даними з сервером та змінювати структуру і зовнішній вигляд вебсторінки.

JavaScript класифікується як прототипна та скриптова мова програмування з динамічною типізацією. Також мова підтримує інші парадигми програмування і деякі архітектурні властивості. Наприклад динамічна та слабка типізація, керування пам’яттю, прототипне наслідування, функції як об’єкти першого класу.

JavaScript використовують для:

* надання інтерактивності вебсторінкам;
* програмування на стороні сервера;
* створення односторінкових та прогресивних вебзастосунків;
* створення стаціонарних та мобільних застосунків.

JavaScript не слід плутати з мовою Java, бо вони є двома різними мовами, хоча і мають схожі риси та правила іменування. Синтаксис мови отриманий у спадок від C, але семантика та дизайн від мов Self та Scheme.

JavaScript стала однією з найпопулярніших мов програмування. Спочатку більшість програмістив ставилися до мови скептично, але поява AJAX змінила ситуацію. Подільші модифікації мови внесли багато корисних можливостей, яких не вистачало для ефективного програмування.

JavaScript має властивості об’єктно-орієнтованої мови, але концепція прототипів відрізняють її від традиційних мов ООП. Також мова має деякі властивості притаманні функціональним мовам, що додає додаткову гнучкість мові.

Мова має C-подібний синтаксис та має корінні відмінності в порівнянні з мовою C, а саме:

* автоматичне приведення типів;
* автоматичне збирання сміття;
* анонімні та стрілочні функції;
* динамічна зміна типу через механізм прототипів.

## 1.3.3 Платформа Node.js

Node.js – це платформа з відкритим кодом для виконання високопродуктивних мережевих застосунків, написаних мовою JavaScript. Засновником цієї платформи є Раян Дал. Node.js надає можливість виконувати JavaScript на сервері та відправляти їх результат користувачеві. Платформа перетворила JavaScript на мову загального використання.

Node.js має наступні властивості:

* асинхронна одно-нитева модель виконання запитів;
* неблокуючий ввід/вивід;
* система модулів CommonJS;
* рушій JavaScript Google V8.

Для обробки великої кількості паралельних запитів використовується асинхронна модель запуску коду. Вона заснована на обробці подій в неблокуючому режимі та на визначенні обробників зворотніх викликів.

Стандартна постановка Node.js включає в себе десятки модулів. В них реалізовані операції для взаємодії з операційною системою, файловою системою, мережею і протоколами, утиліти для обробки даних. Також є доступні багато модулів від незалежних розробників.

## 1.3.4 Технологія AJAX

AJAX (Asynchronous JavaScript And XML) – це підхід до побудови інтерфейсів вебзастосунків, за яких вебсторінка не перезавантажується і у фоновому режимі надсилає на сервер запити і довантажує потрібні користувачу дані.

AJAX – це не самостійна технологія, а швидше використання декількох суміжних технологій. AJAX комбінує кілька основних методів і прийомів:

* використання DHTML для динамічного змінення змісту сторінки;
* використання XMLHttpRequest для звернення до сервера, не перезавантажуючи сторінку;
* динамічно створює дочірні фрейми.

Використання AJAX дозволяє стоврити набагато зручніший інтерфейс для вебзастосунку. Він динамічний, а це означає, що користувач може далі переглядати сторінку поки сервер оброблює запит. AJAX став популярним після того, як Google активно використовувала його при створенні свої сайтів.

## 1.3.5 Бібліотека jQuery

jQuery – це популярна JavaScript бібліотека з відкритим кодом. Бібліотека була представлена 2006 року у січні у BarCamp NYC Джоном Ресіґом. jQuery є найпопулярнішою бібліотекою JavaScript, що використовується на сьогоднішній день.

Синтаксис jQuery розроблений, щоб зробити зручнішим орієнтування у навігації завдяки вибору елементів DOM, обробки подій, розробки AJAX-застосунків та створенню анімації. Використовуючи ці об'єкти, розробники можуть створювати абстракції для низькорівневої взаємодії та анімацію для ефектів високого рівня. Це дозволяє створити потужні і динамічні вебсторінки.

Ця бібліотека є файлом JavaScript, що включає весь його DOM, події, ефекти. Вона може бути додана посиланням на локальний файл, або на одну з копій, що розміщені на публічному сервері.

## 1.3.6 Система керування базами даних MySQL

MySQL – це вільна система керування реляційними базами даних, щ обула розроблена компанією «TcX» з метою пришвидшення обробки великих баз даних. Ця система розроблювалася як альтернатива комерційним системам. З часом вона розширювалася і зараз MySQL є однією з найпопулярніших систем керування базами даних. Її використовують для створення динамічних вебсторінок, оскільки вона підтримується багатьма мовами програмування.

MySQL має подвійне ліцензування, вона може розповсюджуватися відповідно до умов GPL, та комерційна ліцензія компанії Oracle.

Можливості сервера MySQL:

* простота у використанні;
* підтримується необмежена кількість користувачів;
* висока швидкість виконання;
* наявність ефективної системи безпеки.

## 1.4 Огляд аналогів

Аналогом курсового проекту може бути Google Classroom. Це безкоштовний вебсервіс, створений Google для навчальних закладів з метою пришвидшити процес поширення файлів між педагогами та учнями.

## How-To: Taking Attendance and Remote Teaching Using Google Classroom

Рисунок 1.1 Google Classroom

Переваги:

* безкоштовний;
* зручний інтерфейс.

Недоліки:

* учень не може переглядати свої результати;
* не можна створити тестування.

Другим аналогом може бути сайт www.classtime.com. Це онлайн-платформа, що дає можливість викладачу провести онлайн оцінювання та оцінити прогрес класу і кожного учня.

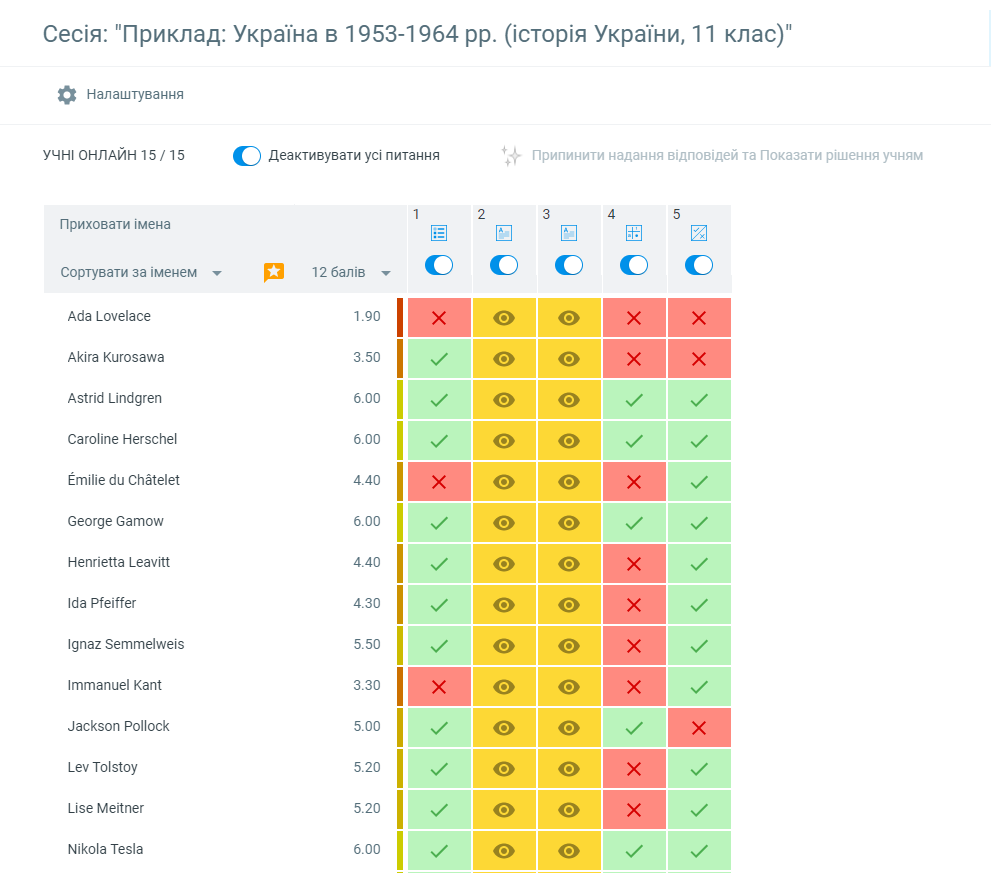


Рисунок 1.2 classtime.com

Переваги:

* зручний за зрозумілий інтерфейс;
* гнучкі налаштування при створенні тесту.

Недоліки:

* можна створити тільки тести;
* учень не бачить своїх результатів.

# 2 ПРАКТИЧНА ЧАСТИНА

# 2.1 Інформаційна структура

При розробці курсового проекту використовувалися графічні файли форматів «png», «jpg», також використовувалися файли векторної графіки формату «svg» та файли шрифтів формату «ttf» та «otf». Також було використано текстовий формат обіну даними «json».

В курсовому проекті користувачі поділені на три типи. Це учні, викладачі та власники курсів.

Учні мають такі можливості:

* переглядати головну стрічку;
* створювати свої пости у головній стрічці та прикріплювати до них файли;
* переглядати теми та завдання;
* залишати до постів, тем та завдань коментарі;
* прикріплювати до відповіді на завдання файли;
* проходити тести;
* переглядати свої результати;
* написати викладачу.

Викладачі мають такі можливості:

* створювати нові пости та прикріплювати до них файли;
* залишати коментарі;
* створювати теми, завдання та тести для учнів;
* редагувати та видаляти теми, завдання та тести;
* переглядати результати всіх учнів;
* оцінювати роботи учнів;
* можуть написати будь якому користувачу присутньому в курсі.

Власник курсу має всі ті самі можливості що і викладач та може повністю налаштовувати курс, а саме:

* видаляти коментарі та пости інших користувачів;
* надавати роль викладача;
* вилучити користувача зі списку викладачів;
* виганяти користувача з курсу;
* передати право власності курсом;
* запрошувати до курсу як нових учнів так і нових викладачів;
* налаштовувати курс, а саме: змінити назву, предмет та згенерувати новий код.

Також всі користувачі можуть створювати свої курси та приєднуватися до інших.

# 2.2 Функціональні частини проекту

Даний розділ містить матеріал, необхідний для повного розуміння внутрішньої архітектури проекту. Проект можна умовно поділити на декілька функціональних частин:

* login.php – сторінка авторизації, реєстрації та форма відновлення паролю;
* registration.php – реєстрація нового користувача;
* logout.php – вихід з акаунту;
* index.php – головна сторінка де завантажуються курси користувача;
* header.php – хедер сайту;
* loadCourses.php – завантаження курсів відповідно до обраних фільтрів;
* invite.php – додавання користувача до курсу за кодом запрошення;
* db/dbConnector.php – підключення до БД;
* db/dbHelper.php – клас, в якому реалізовано функції для полегшення роботи з БД;
* email/emailSender.php – відправлення електроного листа;
* course/course.php – завантаження інформації про вибраний курс;
* course/loadMainTape.php – завантаження головної стрічки курсу;
* course/loadMenu.php – завантаження меню курсу;
* course/loadComment.php – завантаження коментарів;
* course/loadResult.php – завантаження результатів;
* course/loatTasks.php – завантаження завдань та тестів;
* course/loatThemes.php – завантаження теоретичного матеріалу.

Структуру проекту в середовищі Sublime text 3 подано на рисунку 2.1.

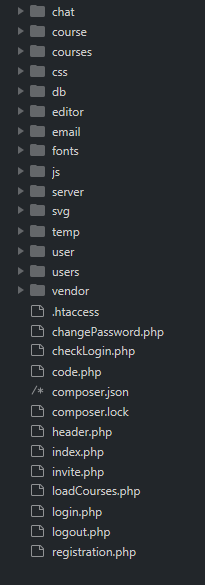


Рисунок 2.1 Внутрішня структура проекту

# 2.3 Опис алгоритму

Загальний опис алгоритму гри із точки зору користувача:

* авторизація або реєстрація;
* створення нового курсу або приєднання до існуючого;
* вибір курсу;
* перегляд головної стрічки;
* перегляд тем, завдань та тестів;
* виконання завдань та тестів;
* стоврення постів та прикріплення файлів;
* залишання коментарів;
* перегляд результатів;
* створення, редагування, видалення тем, завдань та тестів;
* чат з іншими користувачами.

Блок-схема алгоритму подана на рисунку 2.2.

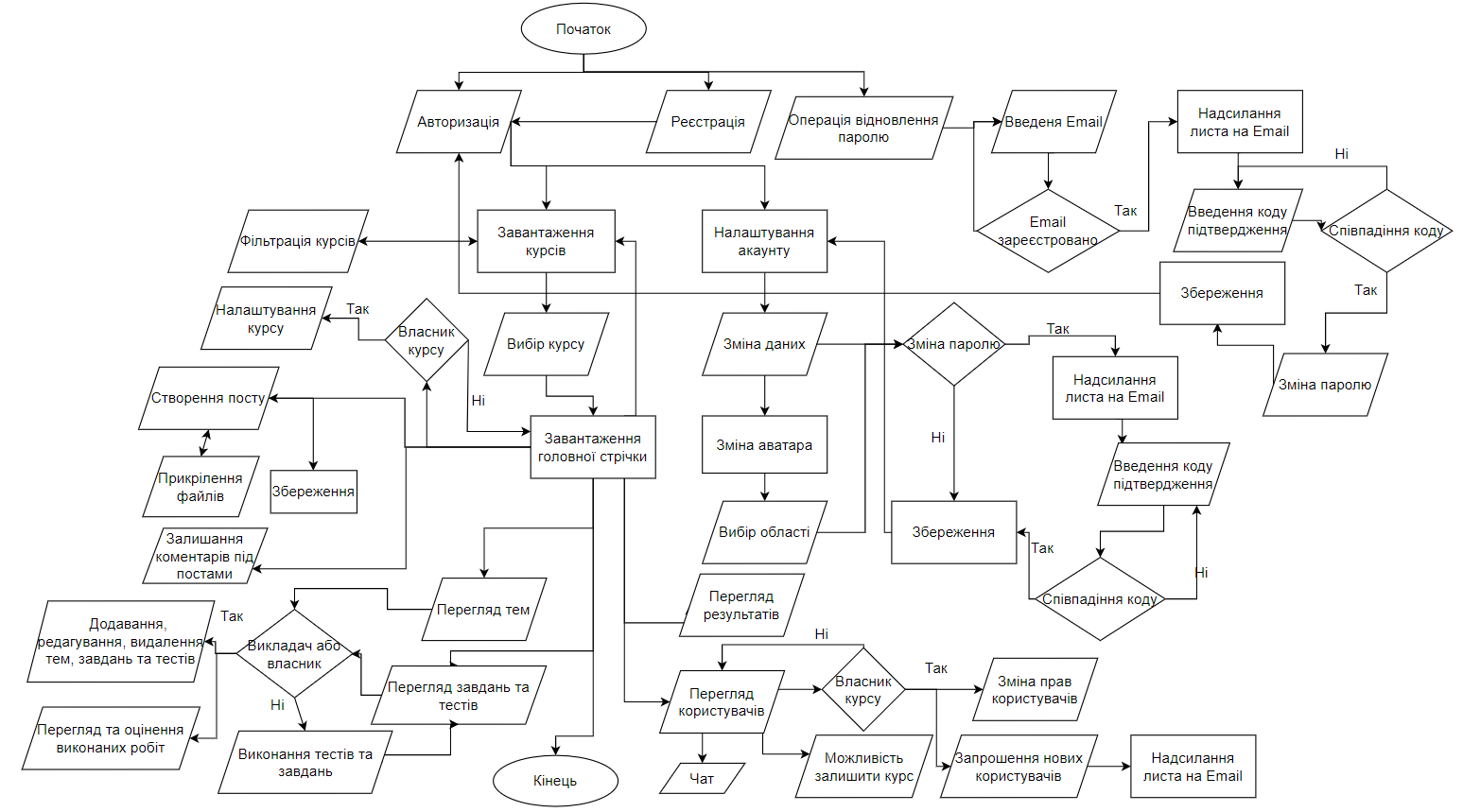


Рисунок 2.2 Блок-схема

# 2.4 Інтерфейс та керівництво користувача

При запуску сайту відкриється сторінка з завантаженими курсами користувача, але якщо користувач не зареєстрований то буде перехід на сторінку авторизації (див. рисунок 2.3).



Рисунок 2.3 Сторінка авторизації

На цій сторінці також можна зареєструвати нового користувача, або змінити пароль якщо ви його забули. Щоб зареєструвати нового потрібно натиснути кнопка «Реєстрація» відбудеться анімований перехід на сторінку реєстрації, де потрібно буде заповнити всі поля для того щоб зареєструвати нового користувача (див. рисунок 2.4).

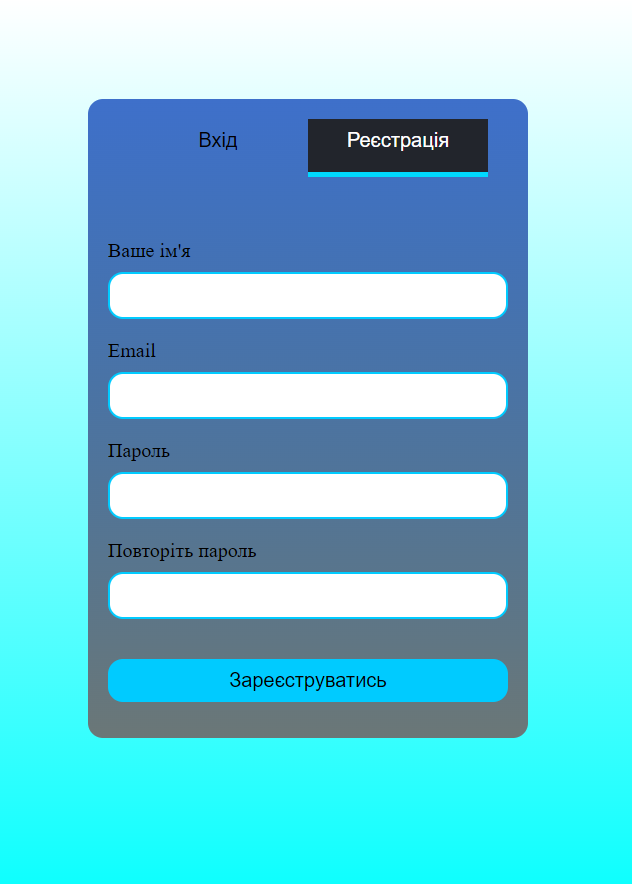


Рисунок 2.4 Реєстрація

Якщо під час заповнення буде допущено помилку про це буде виведено повідомлення (див. рисунок 2.5).

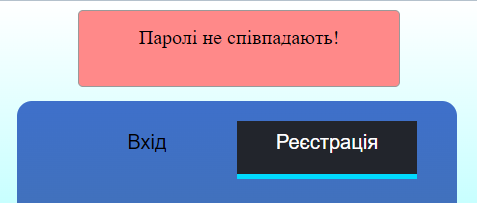


Рисунок 2.5 Повідомлення про помилку

Коли всі поля будуть заповнені без помилок то відбудеться перехід на сторінку авторизації з вже заповненими полями. На цій сторінці користувач може натиснути кнопку «Запам’ятати мене». Це дозволить йому робити вхід без авторизації 30 днів. Також на цій сторінці є функція відновлення паролю якщо користувач його забув (див. рисунок 2.6).

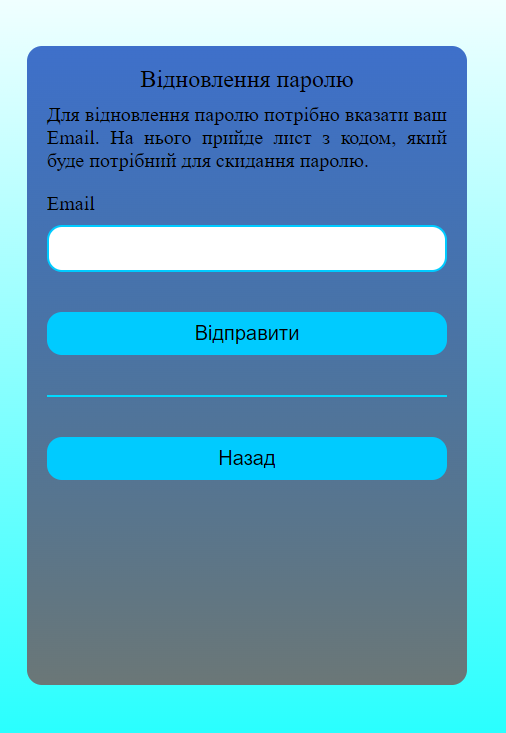


Рисунок 2.6 Відновлення паролю

Для відновлення паролю користувачу потрібно вказати Email, на який буде надіслано листа з кодом підтвердження (див. рисунки 2.7 – 2.8)

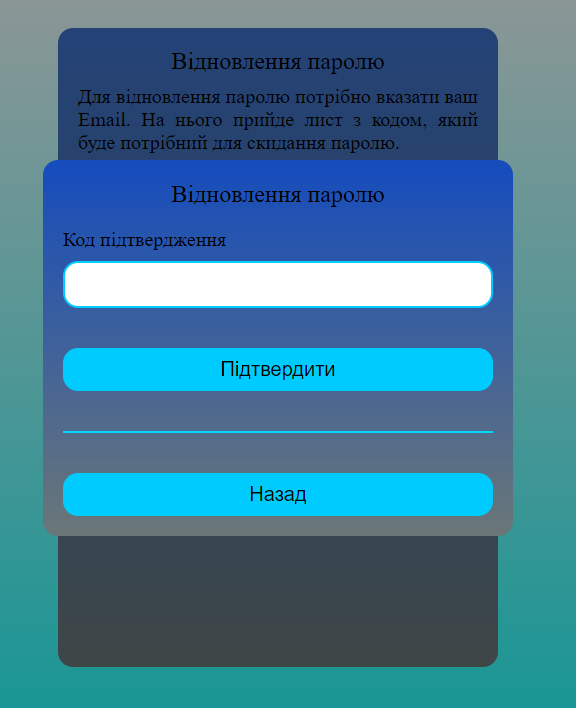


Рисунок 2.7 Вікно підтвердження коду

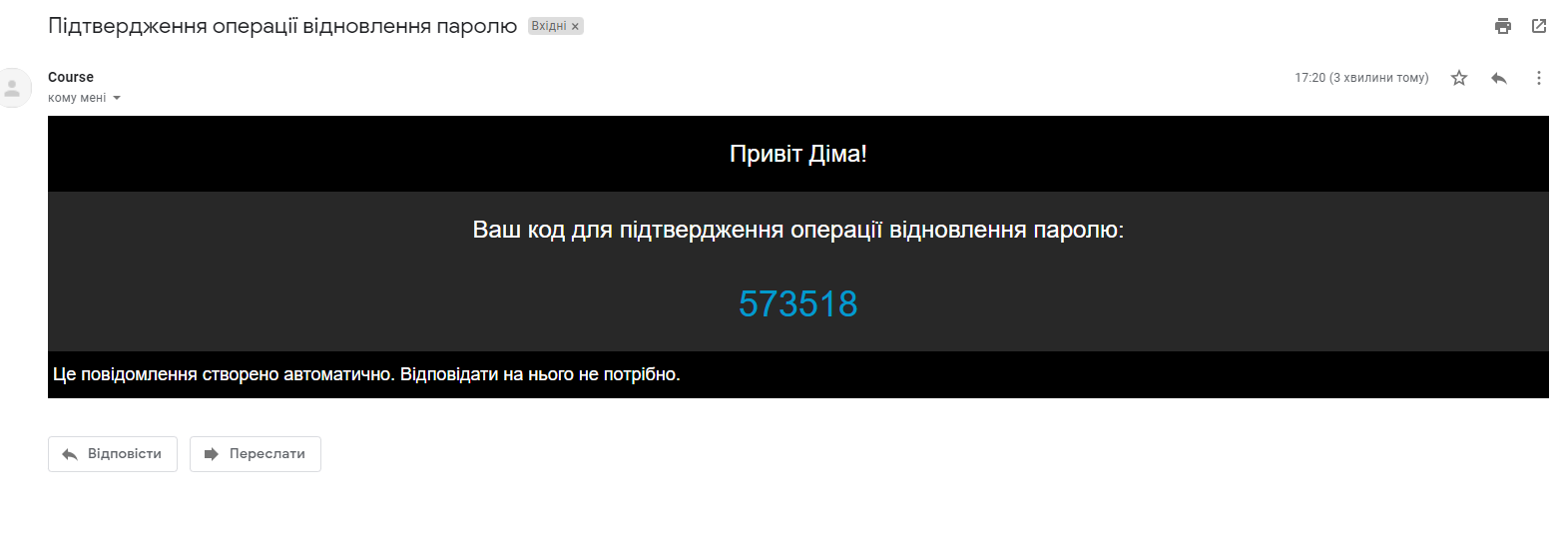


Рисунок 2.8 Лист з кодом підтвердження

Коли користувач введе правильний код, то він перейде на сторінку зміни паролю (див. рисунок 2.9).

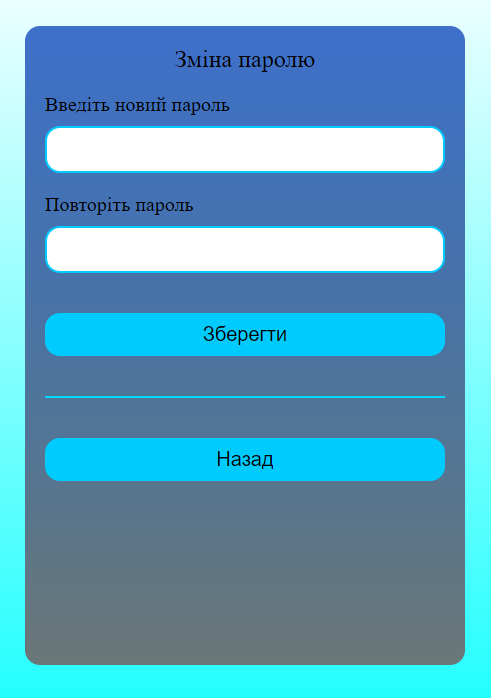


Рисунок 2.9 Зміна паролю

Після того як користувач авторизується він перейде на сторінку де завантажаться всі курси в яких він навчає або навчається. Якщо курсів немає про це буде виведено повідомлення. Також на цій сторінці та на всіх інших сторінках надалі користувач може створити новий курс або приєднатися до курсу, перейти в налаштування акаунту або в чат та обрати світлу або темну тему в будь який час (див. рисунки 2.10 – 2.11).

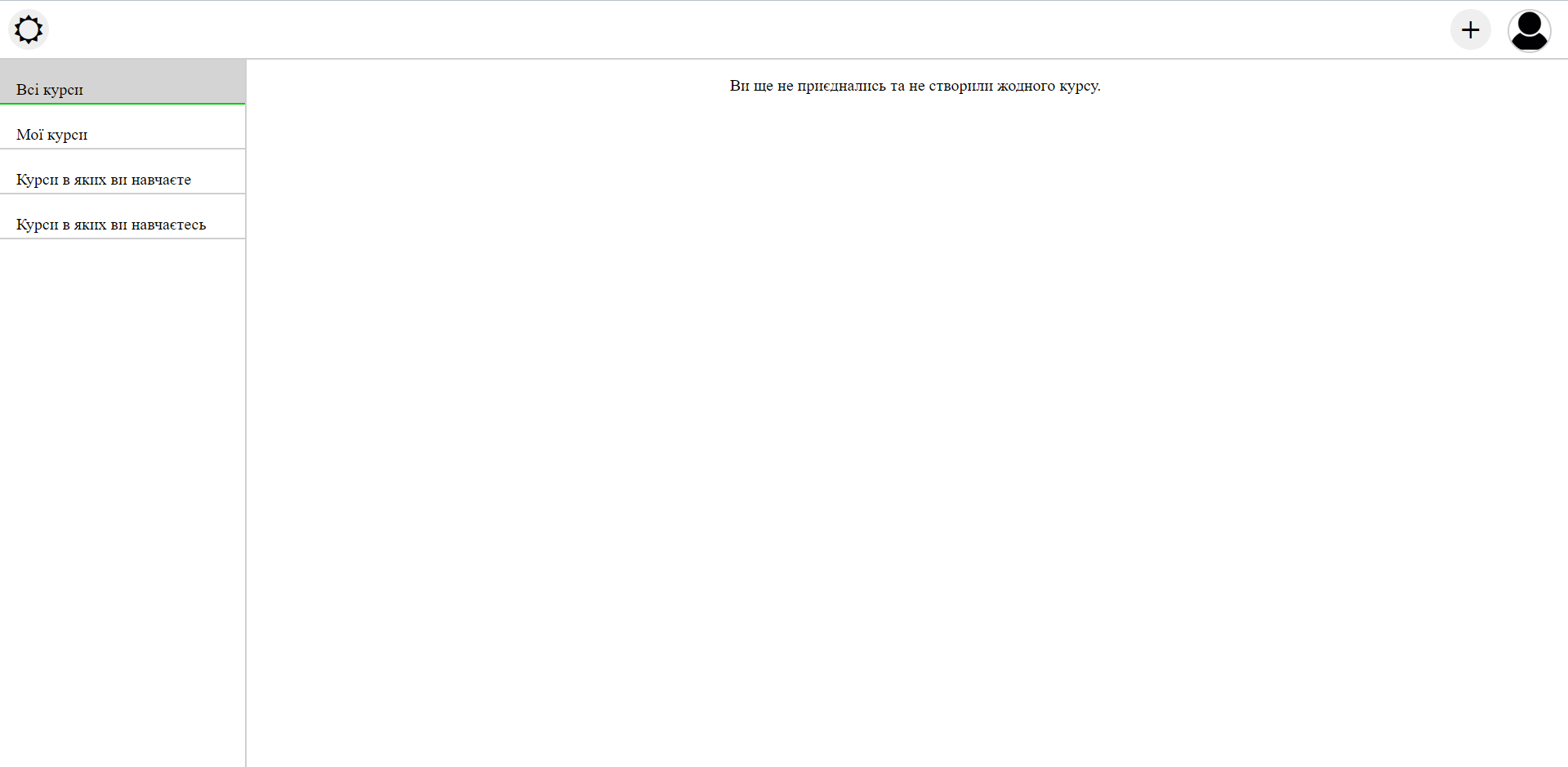


Рисунок 2.10 Світла тема та повідомлення, що користувач немає курсів

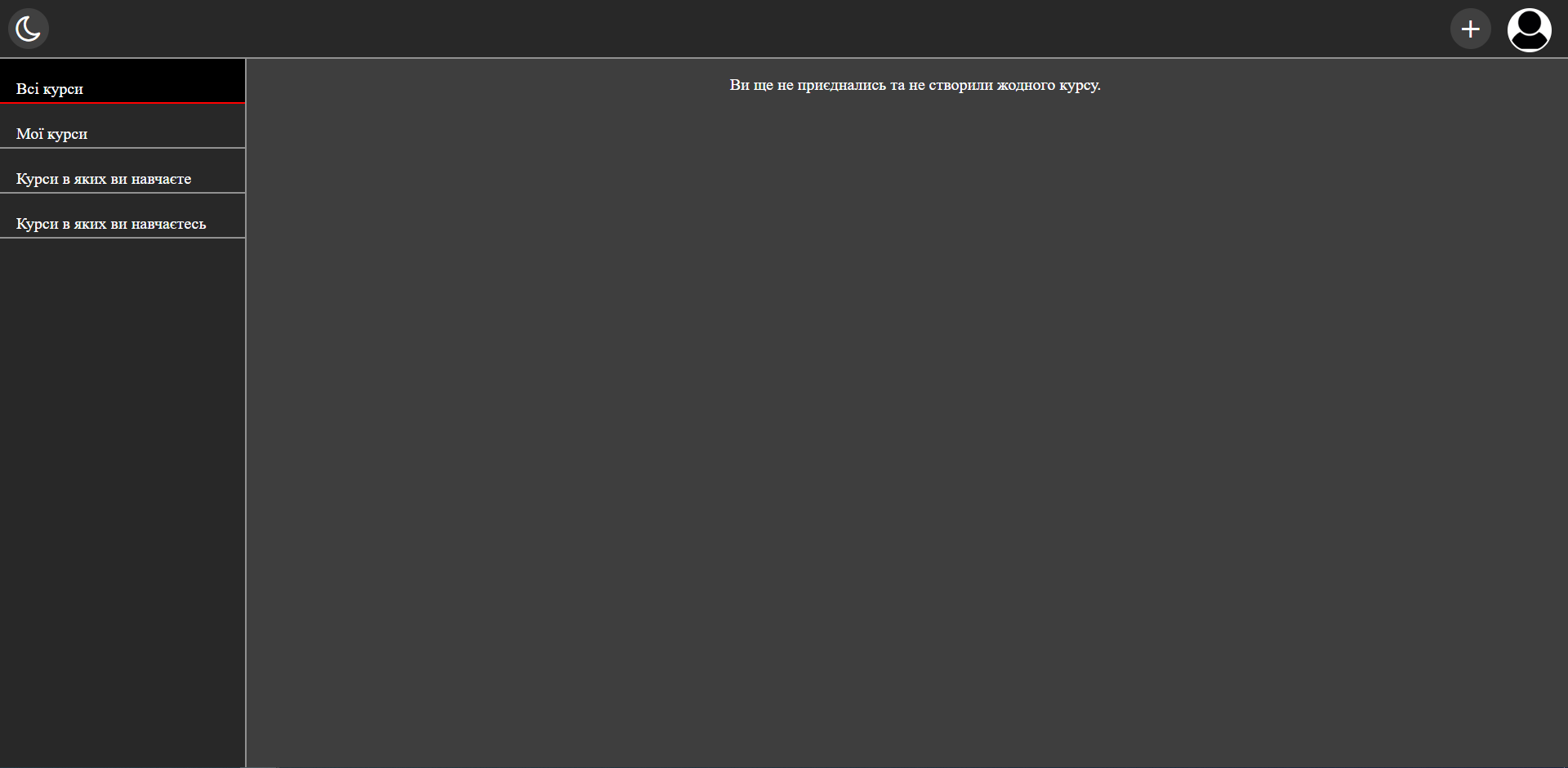


Рисунок 2.11 Темна тема

Щоб створити курс або приєднатися до курсу потрібно натиснути на кнопку «+», яка розташована в правому верхньому кутку. Після того як користувач на неї натисне відкриєтсья меню вибору: стоврити новий курс, приєднатися до курсу (див. рисунок 2.12).

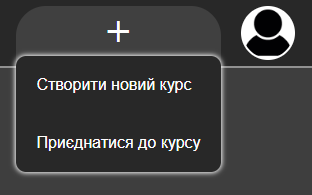


Рисунок 2.12 Меню вибору

Якщо користувач натисне на кнопку «Створити новий курс», то відкриється вікно де потрібно вказати назву та предмет курсу. Якщо він натисне на «Приєднатися до курсу», то відкриється вікно де потрібно вказати код курсу, який бачать тільки викладачі та власник курсу (див. рисунки 2.13 – 2.14)

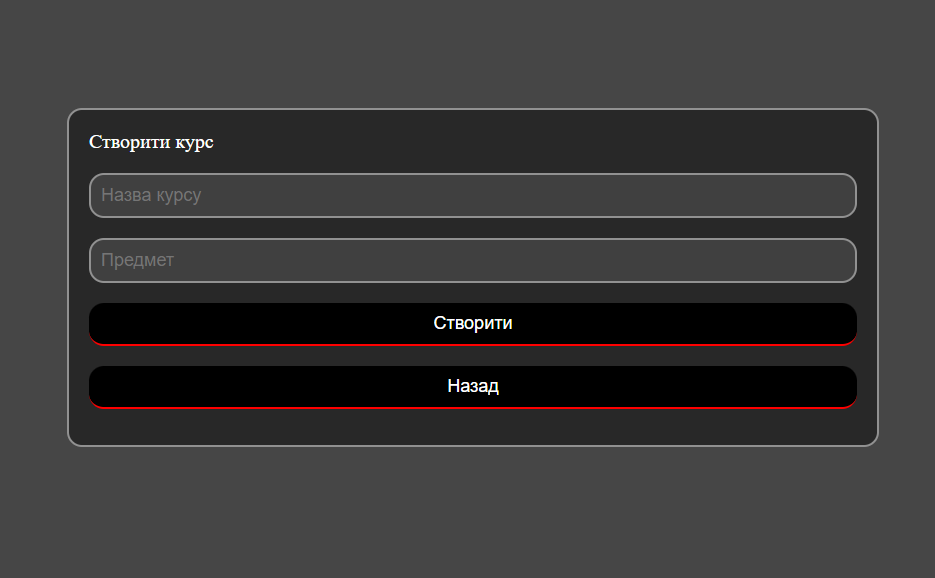


Рисунок 2.13 Вікно створення курсу

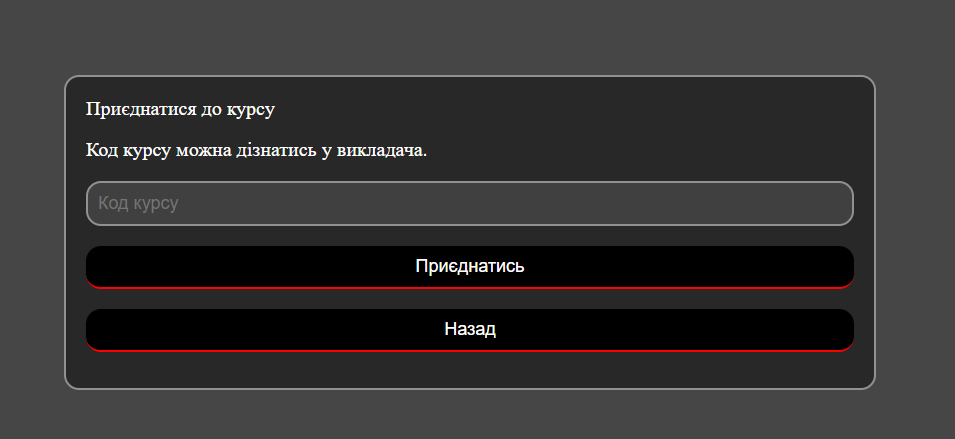


Рисунок 2.14 Вікно приєднання до курсу

Якщо користувач створить або приєднається до курсу то він відразу перейде на сторінку де буде завантажена головна стрічка курсу. При створенні нового курсу його код автоматично згенерується з десяти символів (див. рисунок 2.15).

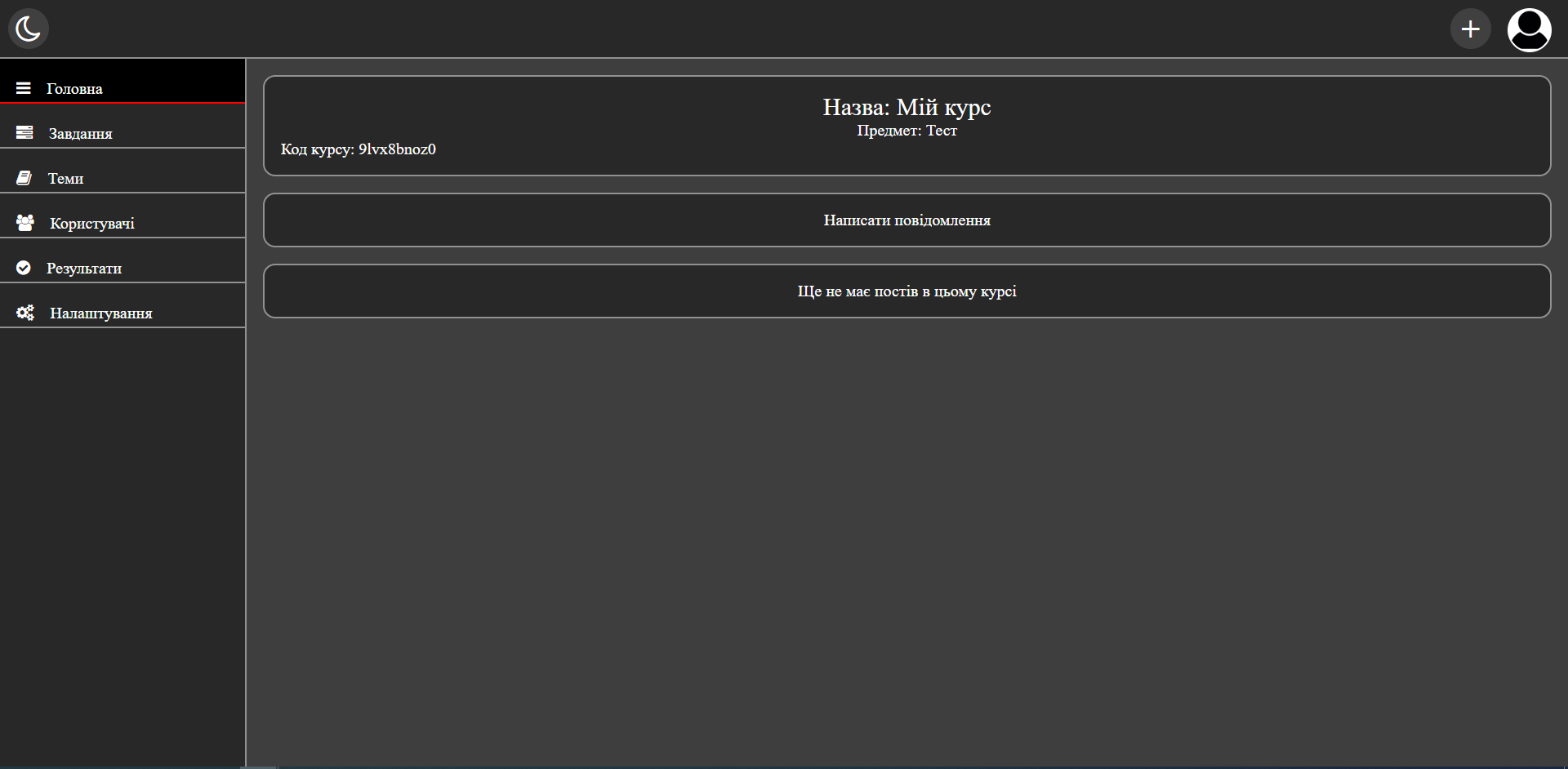


Рисунок 2.15 Головна стрічка курсу

На головній стрічці можна побачити назву, предмет та код курсу якщо ви його власник або викладач. Можна залишити повідомлення та переглянути повідомлення інших користувачів. Також на головній стрічці будуть виводитися повідомлення про створення нової теми, завдання чи тесту.

Щоб повернутися до вибору курсів потрібно натиснути на свій аватар, який розміщений у правому верхньому кутку (див. рисунок 2.16).

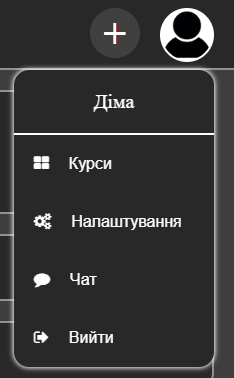


Рисунок 2.16 Випадаюче меню

У меню буде можливість повернутися до вибору курсів, перейти в налаштування акаунту, перейти до чату та вийти.

Якщо користувач натисне на кнопку «Налаштування», то він перейде на сторінку де зможе змінити ім’я, пароль, аватар (див. рисунок 2.17).

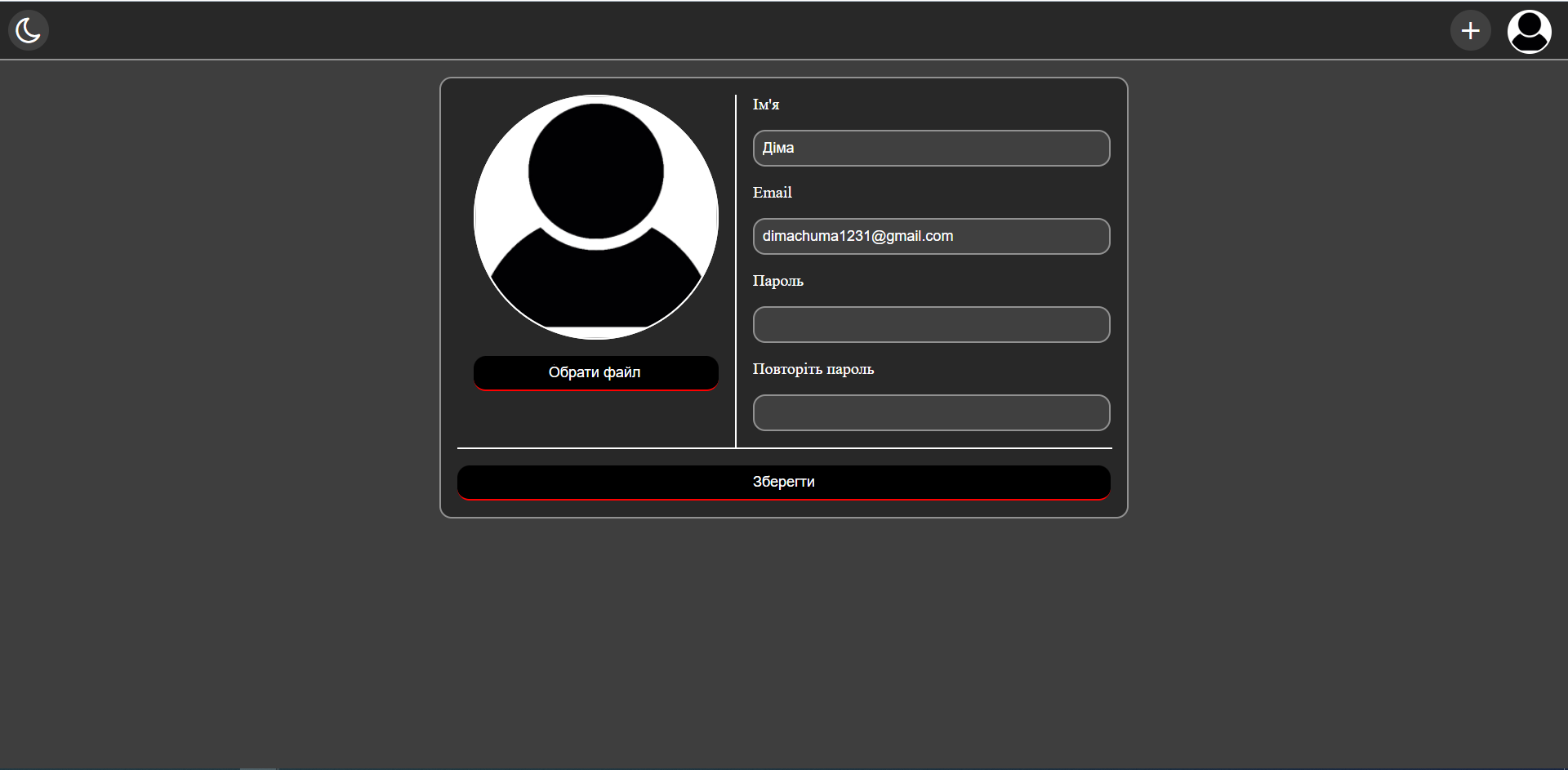


Рисунок 2.17 Налаштування акаунту

Для зміни аватару потрібно обрати файл, після того як він буде обраний відкриється редактор, де потрібно обрати область, яку буде вирізано під аватар (див. рисунок 2.18).

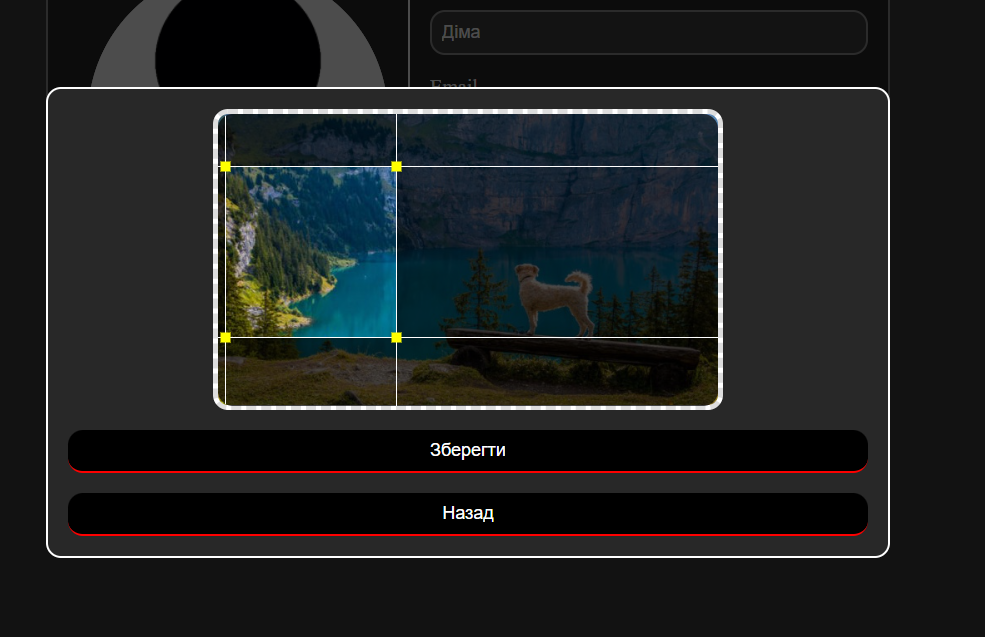


Рисунок 2.18 Вибір області

Якщо користувач не буде змінювати пароль то всі зміни збережуться після натискання кнопки «Зберегти». Якщо користувач захоче змінити пароль то перед збереженням даних потрібно буде ввести код підтвердження який буде відправлено на його Email. Також після зміни аватару з'явиться кнопка, яка дозволить його видалити та повернутися до стандартного аватару.

Коли користувач повернеться до курсів і він вже створив або приєднався до них, то вони будуть завантажені на сторінці і також буде виведено коротку інформацію про них, а саме: назва курса, предмет та власник (див. рисунок 2.19).

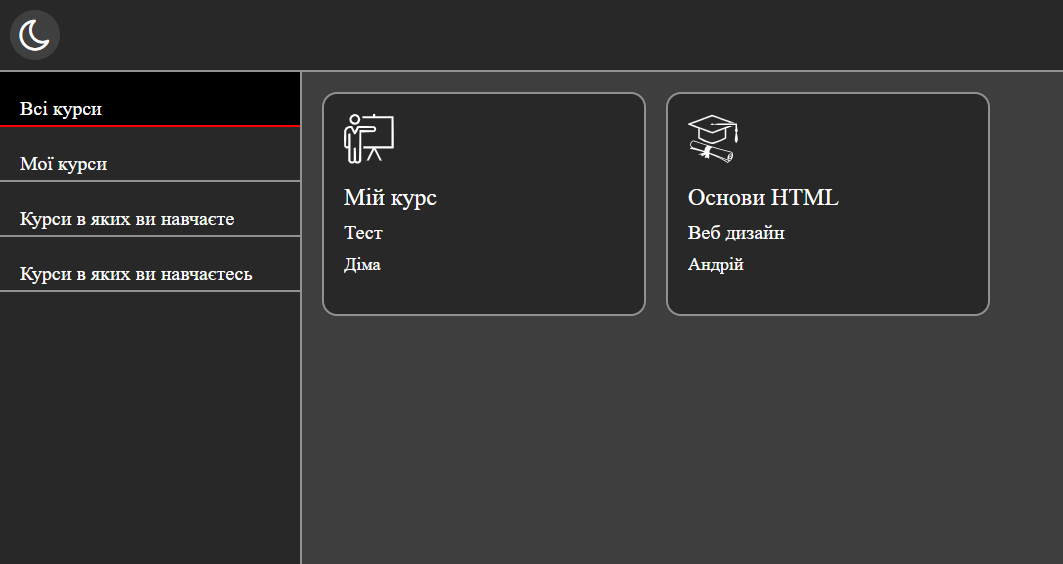


Рисунок 2.19 Курси користувача

Також можна побачити різницю в іконках між курсами в яких ви власник або викладач і курсах де ви навчаєтесь.

В головній стрічці курсу також можна залишати коментарі під постами та переходити до тем, завдань чи тестів. В кожному пості буде показано хто його додав, дату коли пост було створено та повідомлення або тип посту (див. рисунок 2.20).

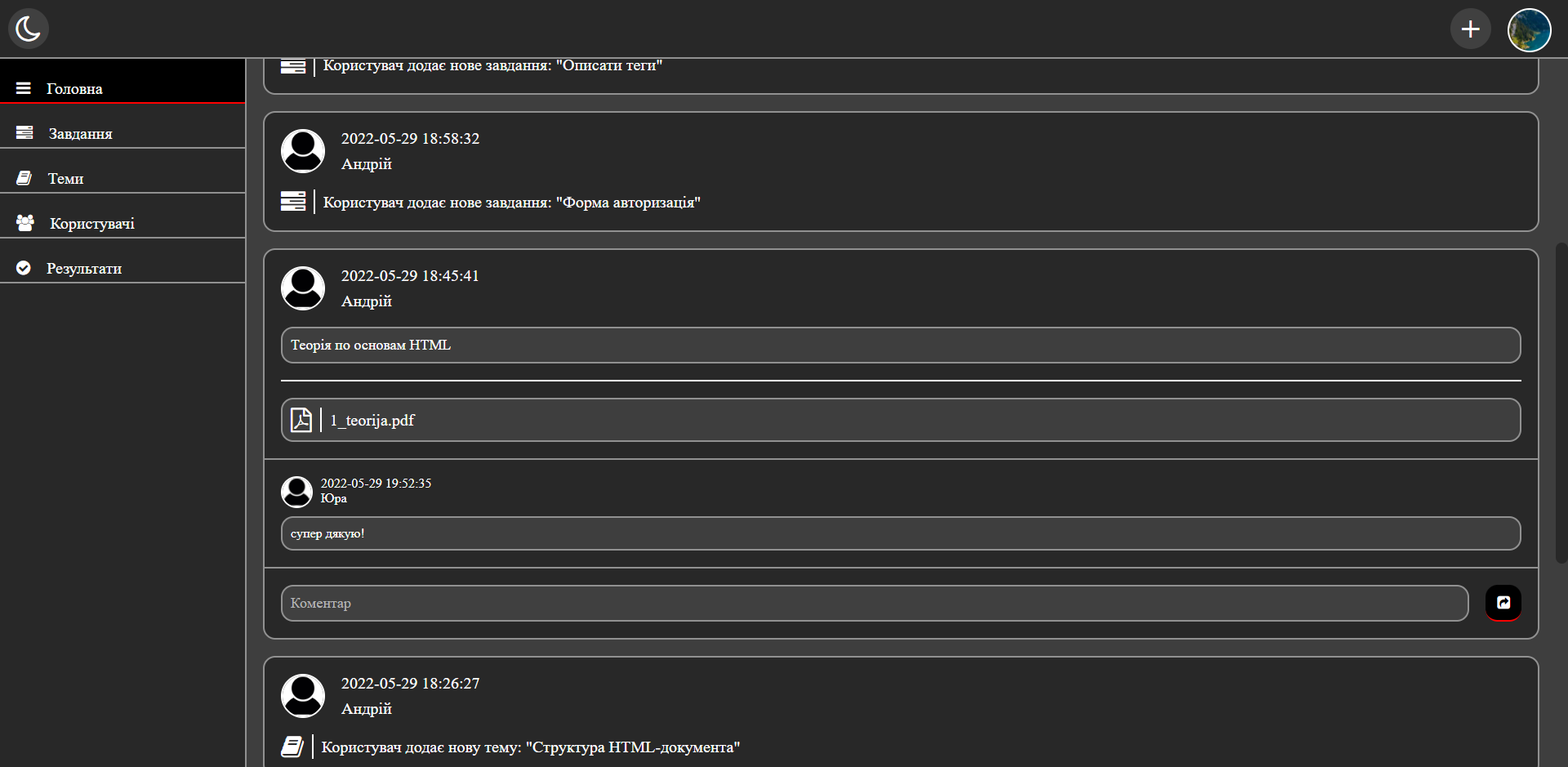


Рисунок 2.20 Головна стрічка курсу

Щоб залишити повідомлення потрібно натиснути кнопку «Написати повідомлення», тоді завантажиться редактор де можна прикріпити картинку, відео чи інші файли (див. рисунок 2.21).



Рисунок 2.21 Написання повідомлення

Для того щоб прикріпити файли потрібно натиснути кнопку в лівому нижньому кутку під редактором. Відкриється вікно куди можна перетягнути файли або обрати (див. рисунок 2.22).

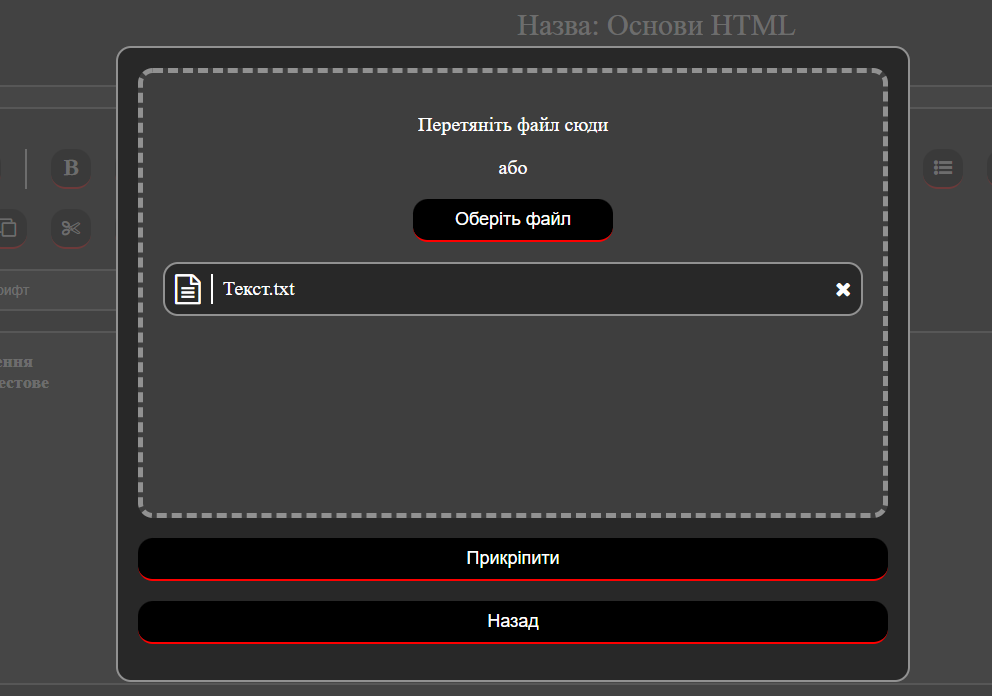


Рисунок 2.22 Прикріплення файлу

Якщо користувач додав зайвий файл його можна видалити. Після натиснення кнопки «Прикріпити» ці файли будуть виведені під редактором.

Щоб зберегти повідомлення потрібно натиснути кнопку «Відправити», кнопка «Відмінити» видалить повідомлення та всі додані файли.

Свої повідомлення можна редагувати та видаляти, для цього потрібно натиснути на кнопку в правому кутку після чого відкриється меню де потрібно обрати дію (див. рисунок 2.23).

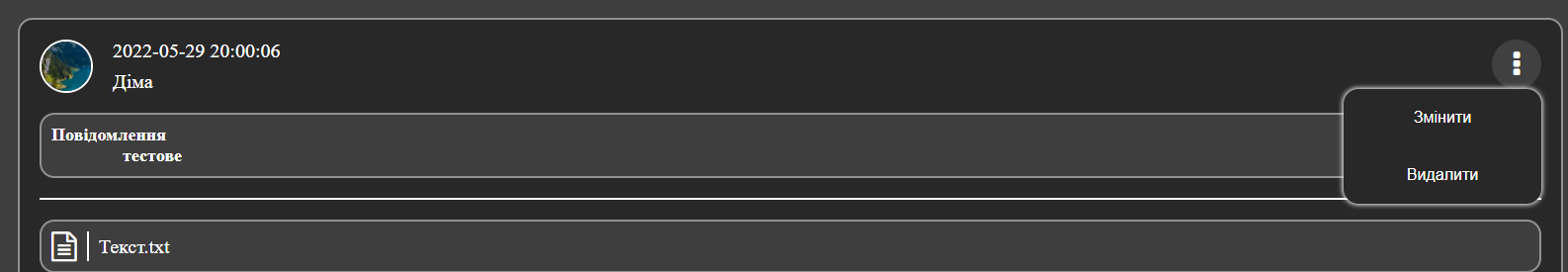


Рисунок 2.23 Меню вибору операції

При натисканні кнопки «Видалити» повідомлення та прикріплені файли будуть видалені. При натисканні кнопки «Змінити» відкриється редактор де можна змінити текст повідомлення та прикріплені файли, або додати інші, або видалити зайві.

Власник курсу може видаляти повідомлення всіх користувачів.

Якщо натиснути кнопку «Теми» в меню курсу то будуть відфільтровані всі записи та виведені тільки теми, також вони будуть поділені по розділах (див. рисунок 2.24).

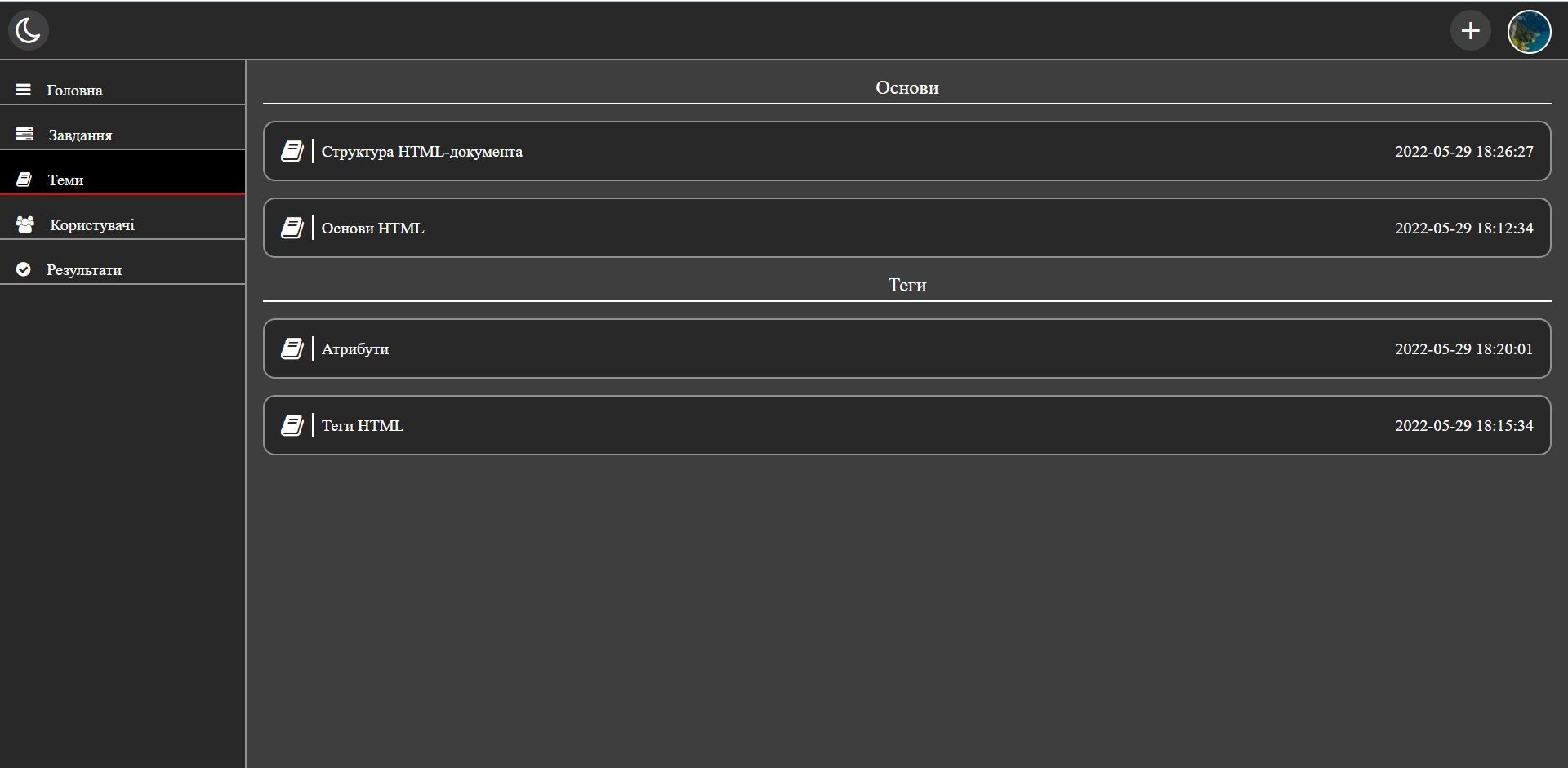


Рисунок 2.24 Теми

При натискані на тему буде перехід на нову сторінку, де буде завантажено текст, картинки та прикріплені файли до теми. Також там можна залишити коментар (див. рисунок 2.25).

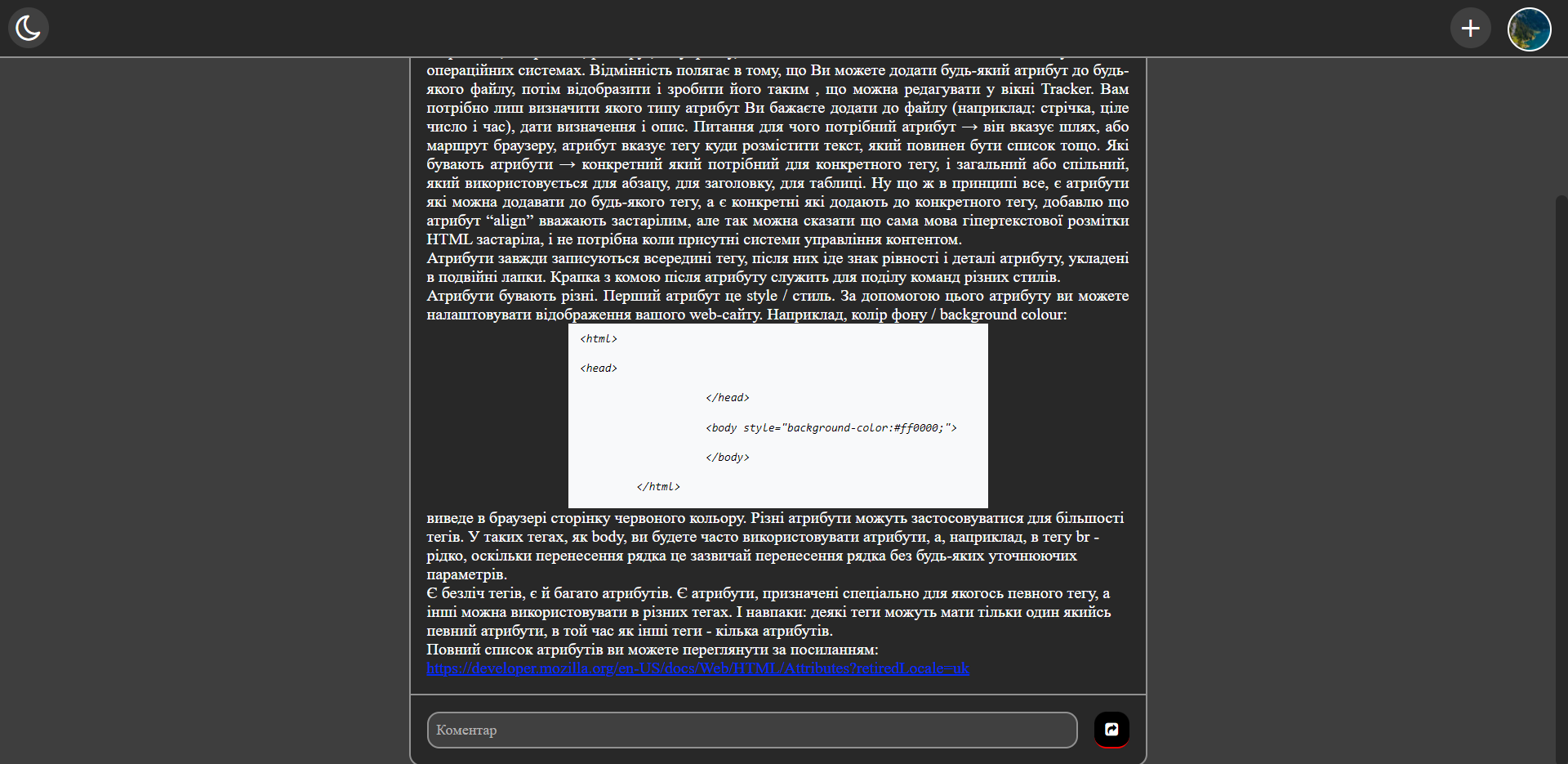


Рисунок 2.25 Вибрана тема

Всі прикріплені файли можна скачати.

Щоб залишити коментар потрібно ввести його у поле «Коментар» та натиснути кнопку «Відправити». Свої коментарі можна змінювати та видаляти.

Якщо користувач вибере «Завдання» у головному меню, то будуть відфільтровані самі завдання та тести. При натисканні на них буде здійснено перехід на сторінку з інформацією про завдання або тест (див. рисунок 2.26).

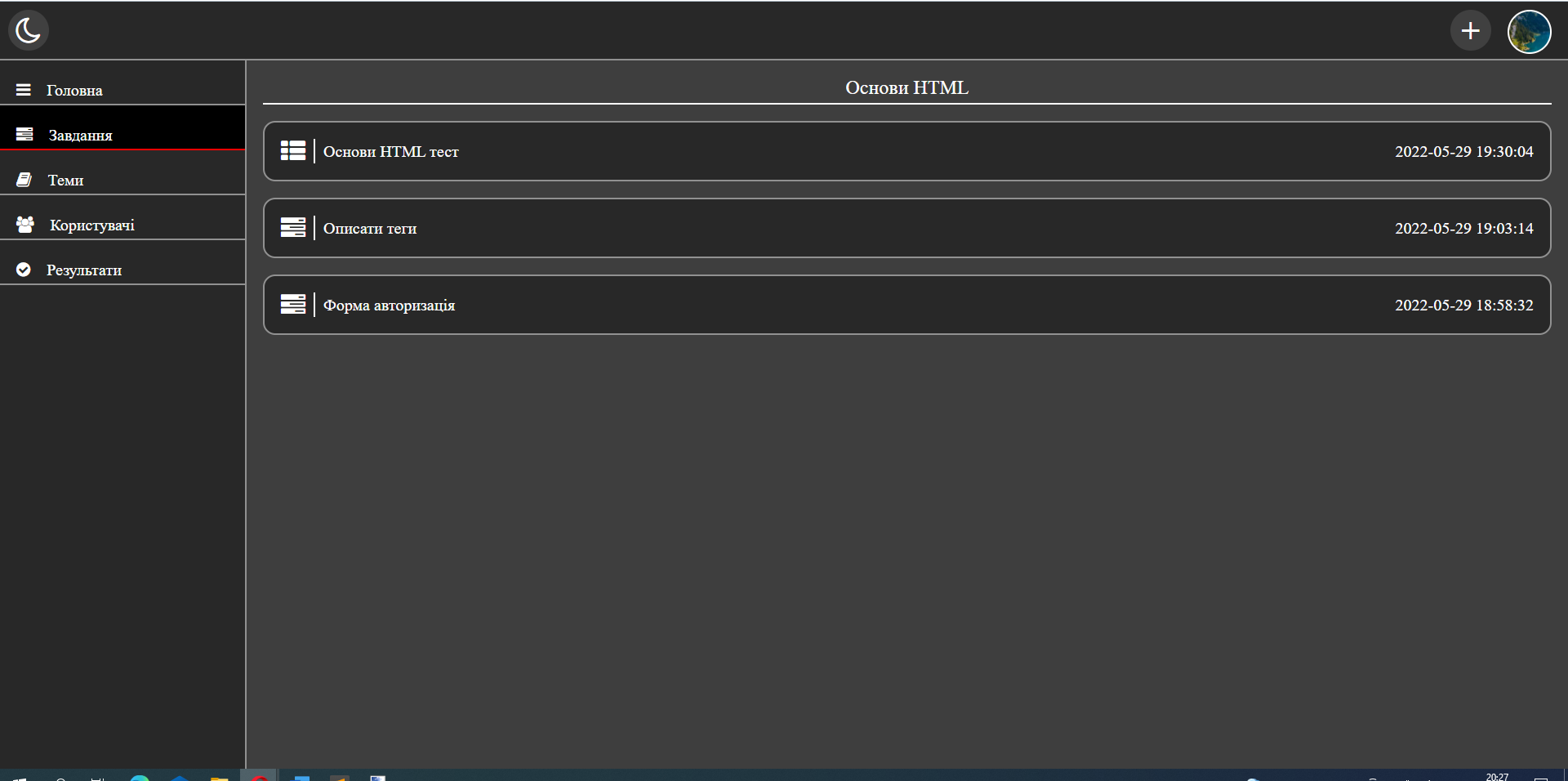


Рисунок 2.26 Завдання

Коли користувач натисне на завдання, то відкриється інформація про завдання, а саме: дата до якої потрібно здати, максимальна кількість балів та чи здано завдання. Потім виводиться текст завдання та прикріплені файли, а потім коментарі. Праворуч буде форма де можна написати відповідь та прикріпити файли. Писати відповідь не обов’язково так само як і прикріплювати файли, достатньо зробити щось одне (див. рисунок 2.27).

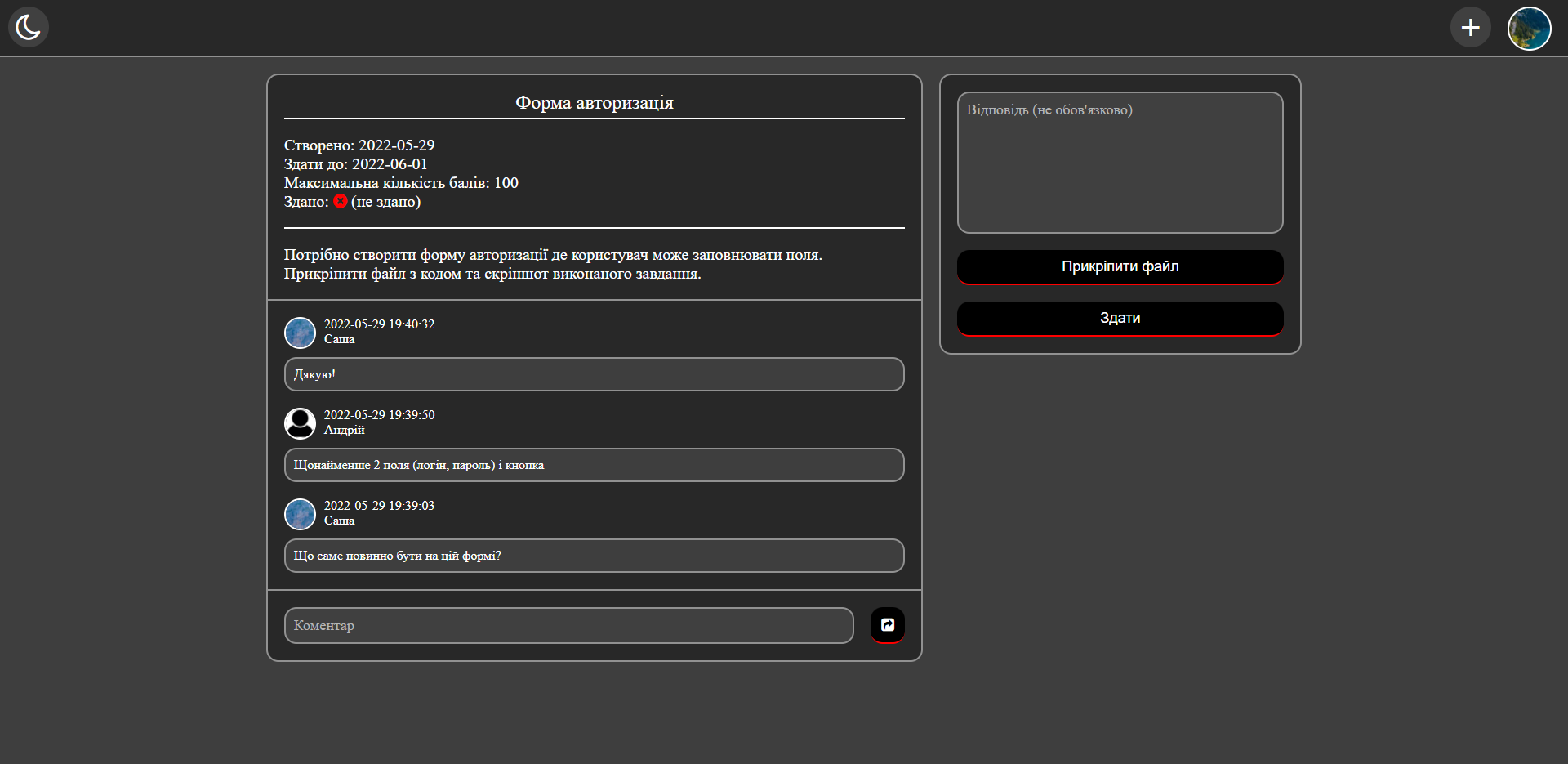


Рисунок 2.27 Інформація про завдання

Щоб здати роботу потрібно або написати відповідь або прикріпити файл та натиснути кнопку «Здати». Користувач завжди може редагувати свою відповідь (див. рисунок 2.28).

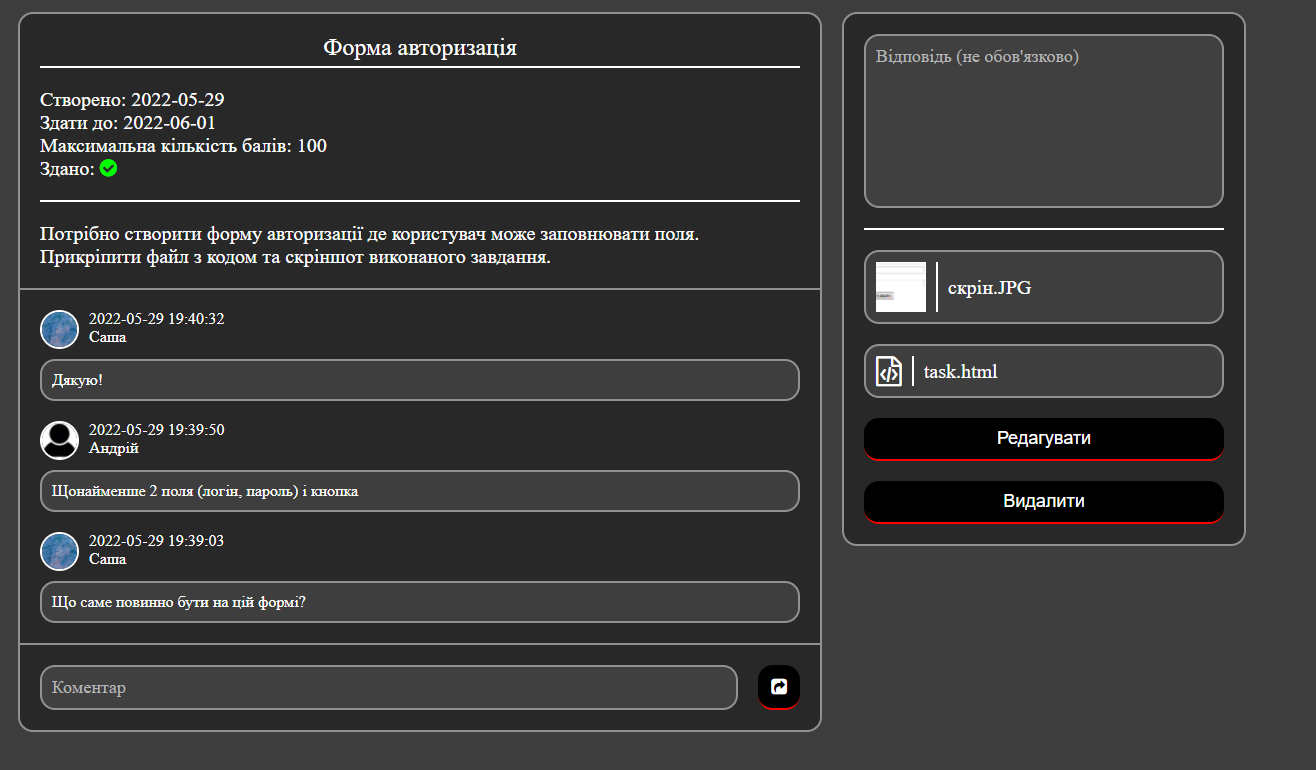


Рисунок 2.28 Здане завдання

Якщо користувач натисне на тест, то відкриється сторінка з інформацією про тест. Буде виведено дату початку та кінця тесту, кількість хвилин на тест, кількість питань, максимальний бал за тест, мінімальний прохідний, кількість спроб та метод оцінювання (див. рисунок 2.29).

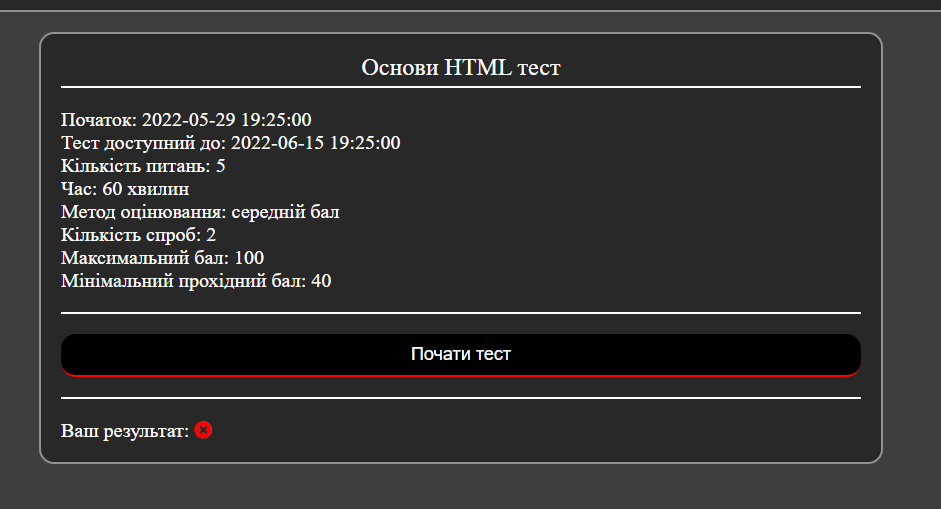


Рисунок 2.29 Інформація про тест

Якщо тест доступний то буде виведена кнопка інакше повідомлення чому тест недоступний. Якщо користувач почав тест та не закінчив його, то почати інший він не зможе поки не закінчить попередній.

Внизу сторінки буде виведено результат якщо користувач вже його проходив.

Після натискання кнопки «Почати тест», відкриється сторінка з тестом, на ній можна буде здійснити перехід між питаннями та буде показано які питання вже пройдені (підсвічені зеленим кольором) і яке питання зараз вибране (підсвічене жовтим кольором) всі інші будуть підсвічені червоним кольором. Також буде показано бал за запитання та час, що залишився до кінця тесту (див. рисунок 2.30).

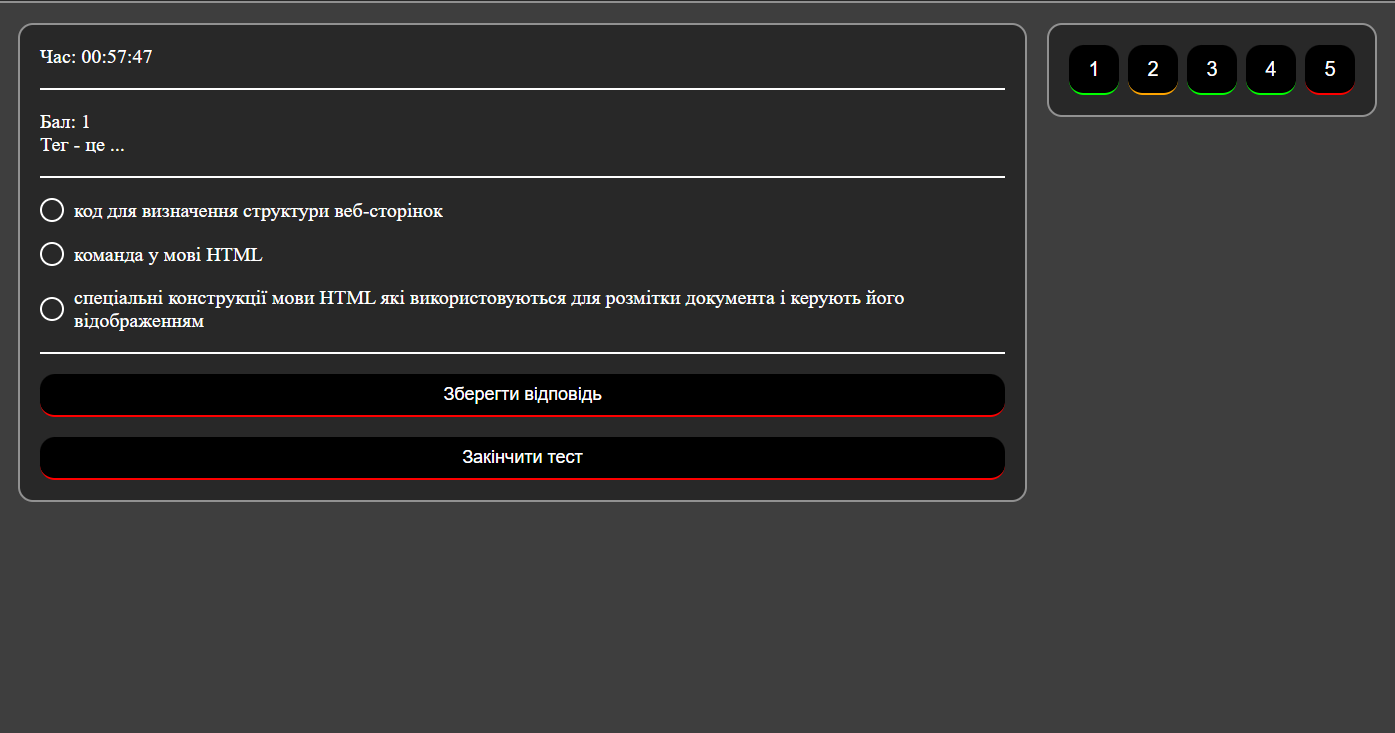


Рисунок 2.30 Тест

Щоб завершити тест потрібно натиснути кнопку «Закінчити тест» та підтвердити операцію. Закінчити тест тест можна в будь-який момент.

Після закінччення тесту буде виведено результати. В них буде показано відповіді користувача та виділено червоним якщо вона не правильна та зеленим якщо правильна. Якщо на тест не відповісти то буде виведено повідомлення про те, що він пропущений. Внизу сторінки буде виведена оцінка за тест. Якщо в тесті були питання де потрібно розгорнуту відповідь то буде вивеена лише попередня оцінка. Остаточку виставляє викладач (див. рисунок 2.31).



Рисунок 2.31 Результат проходження тесту

Якщо користувач в меню вибере «Результати», то він побачить всі свої результати по всіх завданнях курсу (див. рисунок 2.32).

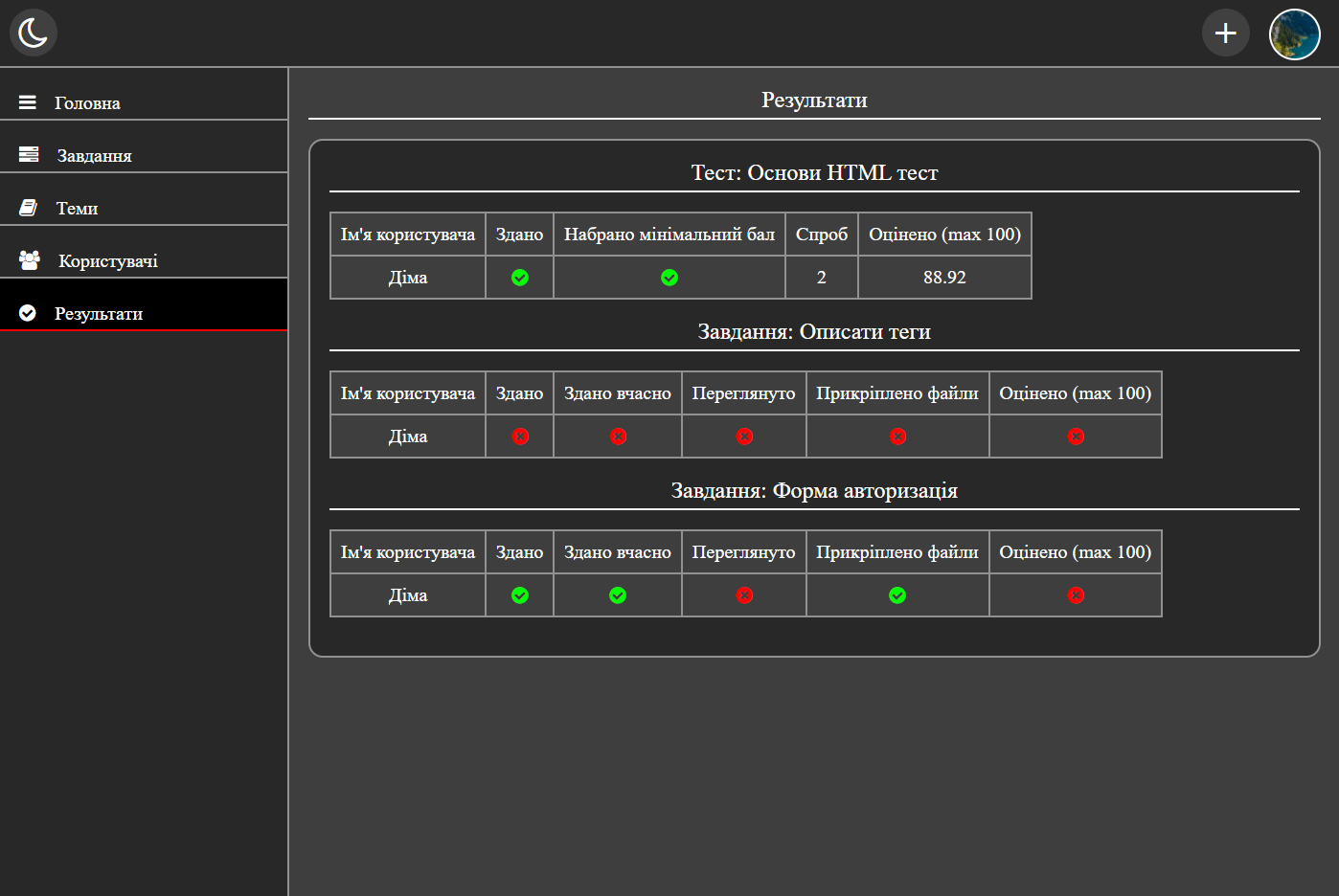


Рисунок 2.32 Результати користувача

Якщо користувач вибере «Користувачі», то буде виведено всіх учнів та викладачів на курсі. Учень може писати тільки викладачу, а також покинути курс за власним бажанням (див. рисунок 2.33).

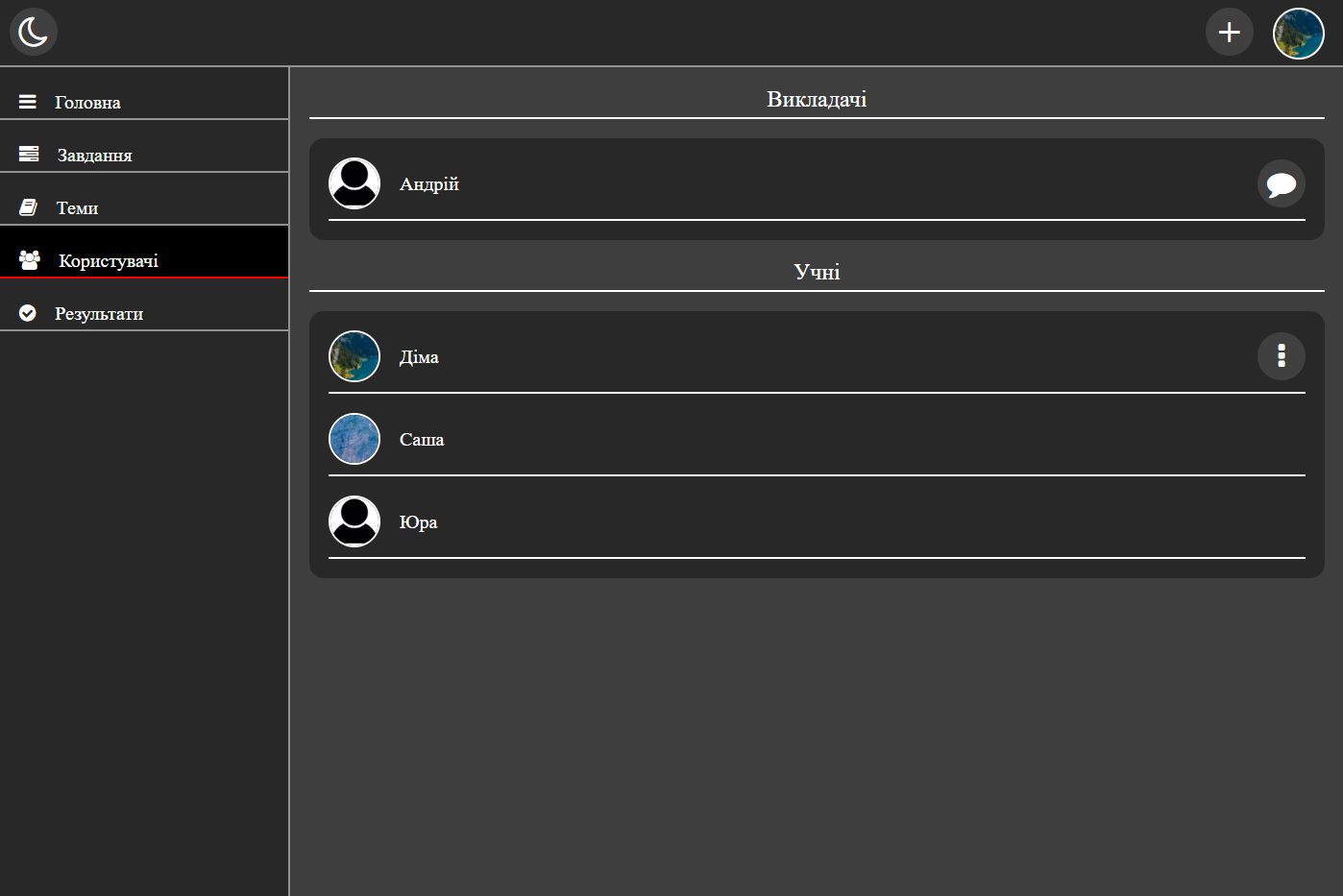


Рисунок 2.33 Користувачі курсу

Щоб написати викладачу потрібно натиснути кнопку по правому краю. Відкриється нова сторінка, де будуть завантажені всі інші користувачі, з якими переписувався користувач а також історія чату з обраним користувачем (див рисунок 2.34).

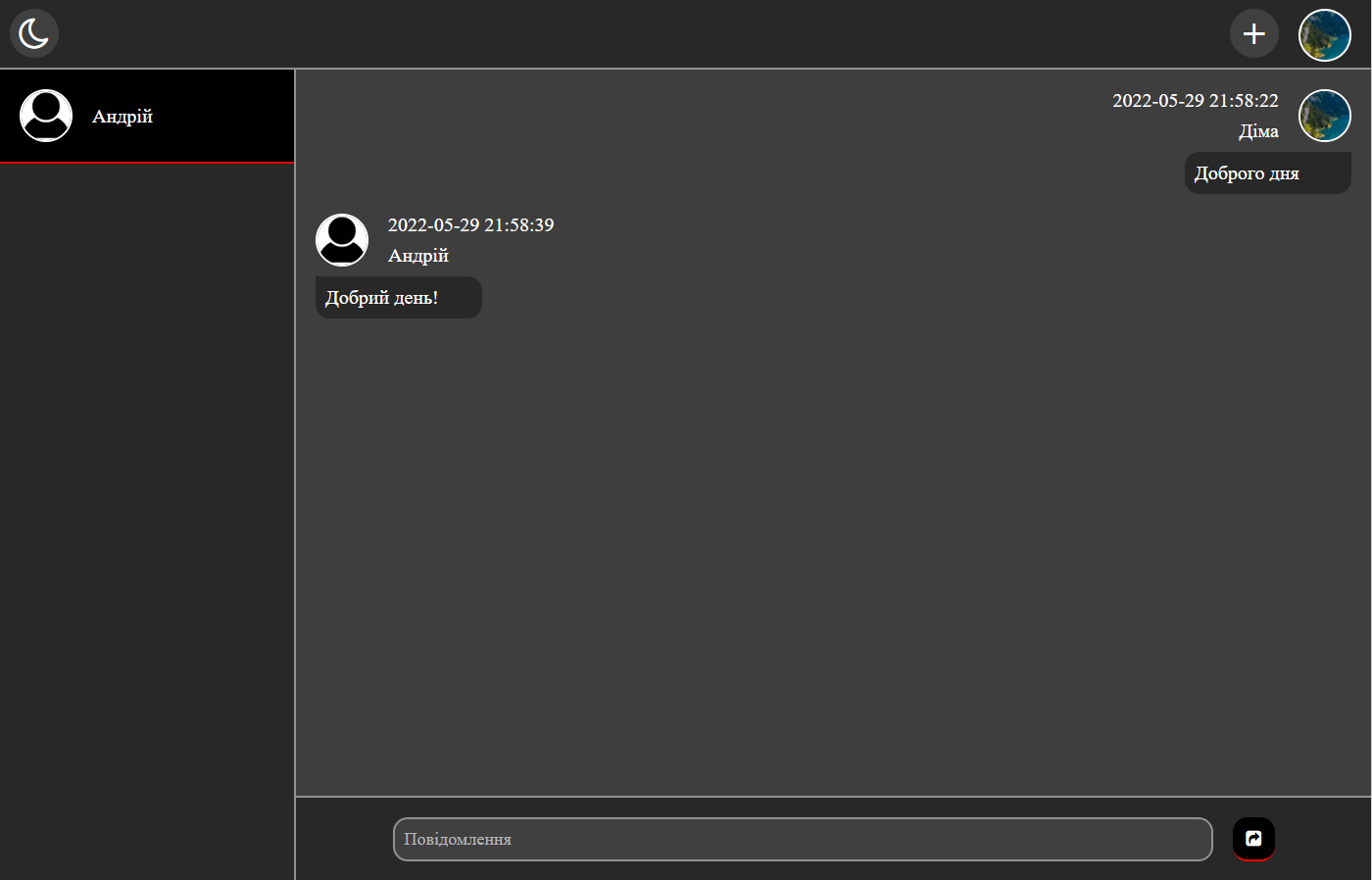


Рисунок 2.34 Чат між користувачами

Свої повідомлення можна змінювати та видаляти для цього потрібно на ньому натисну правою кнопкою миші та обрати операцію з меню (див. рисунок 2.35).

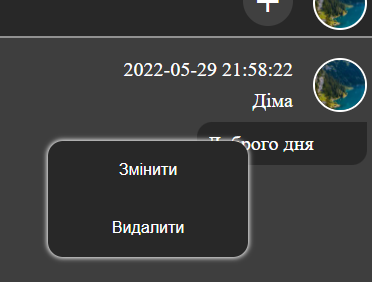


Рисунок 2.35 Вибір операції

Викладачі та власник курсу можуть додавати теми, завдання та тести. Щоб додати тему потрібно обрати «Теми». Буде виведено кнопку «Додати нову тему» при натисканні якої відкриється редактор (див. рисунок 2.36).

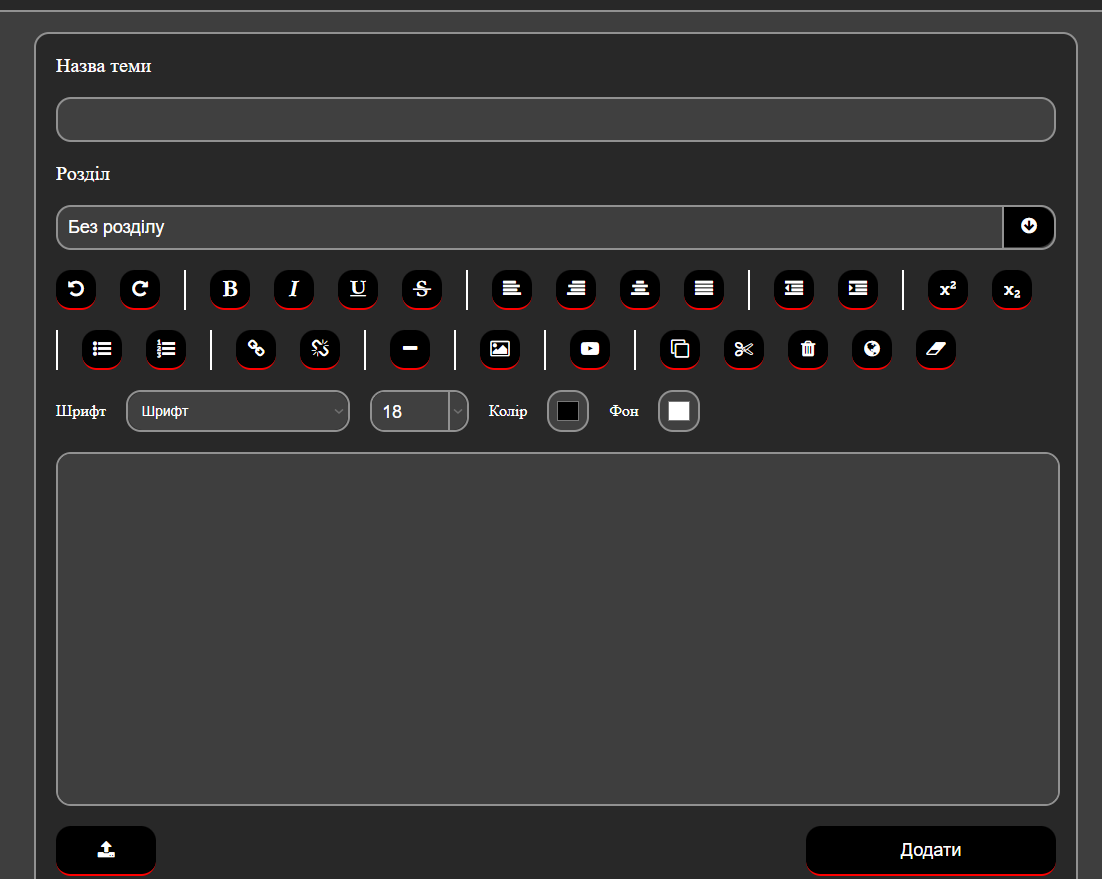


Рисунок 2.36 Редактор тем

Щоб додати завдання потрібно перейти до завдань. Буде дві кнопки, а саме: додати нове завдання та додати новий тест (див. рисунки 2.37 – 2.38).

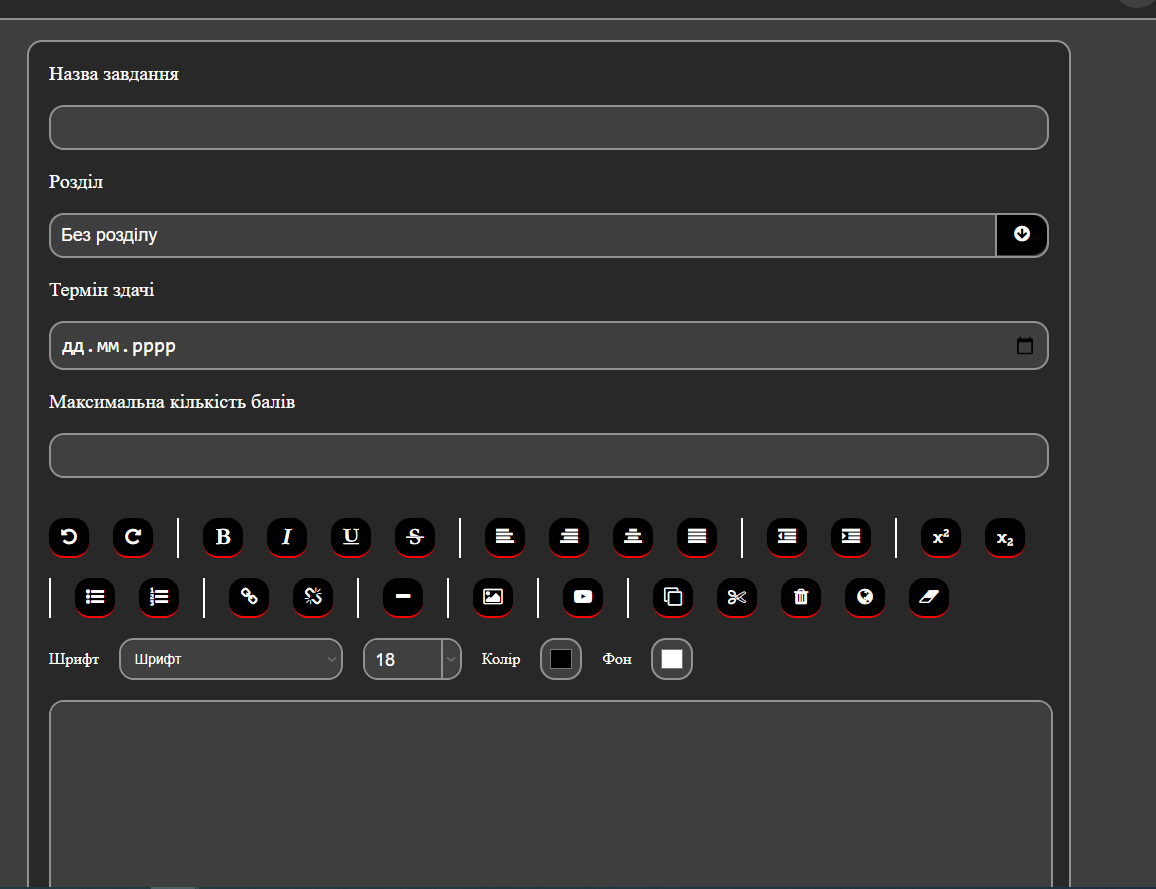


Рисунок 2.37 Редактор завдань

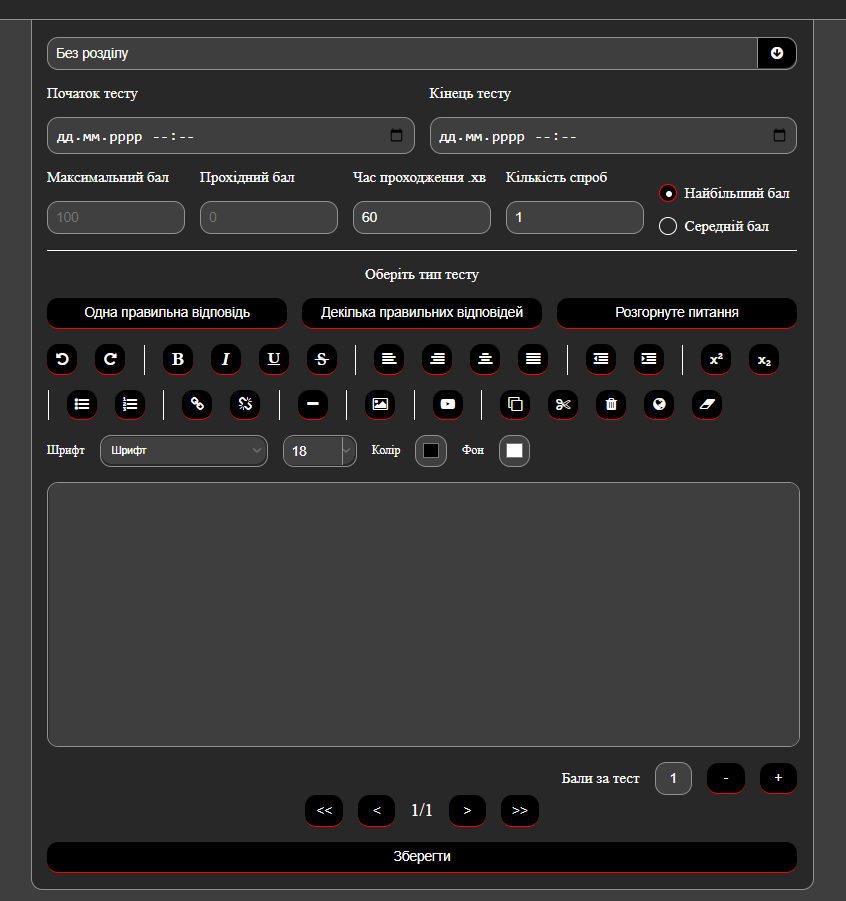


Рисунок 2.38 Редактор тестів

В редакторі тестів можна обрати три типи питання, а саме: питання з одною правильною відповіддю, питання де декілька відповідей можуть бути правильні та розгорнуте (див. рисунки 2.39 – 2.40).

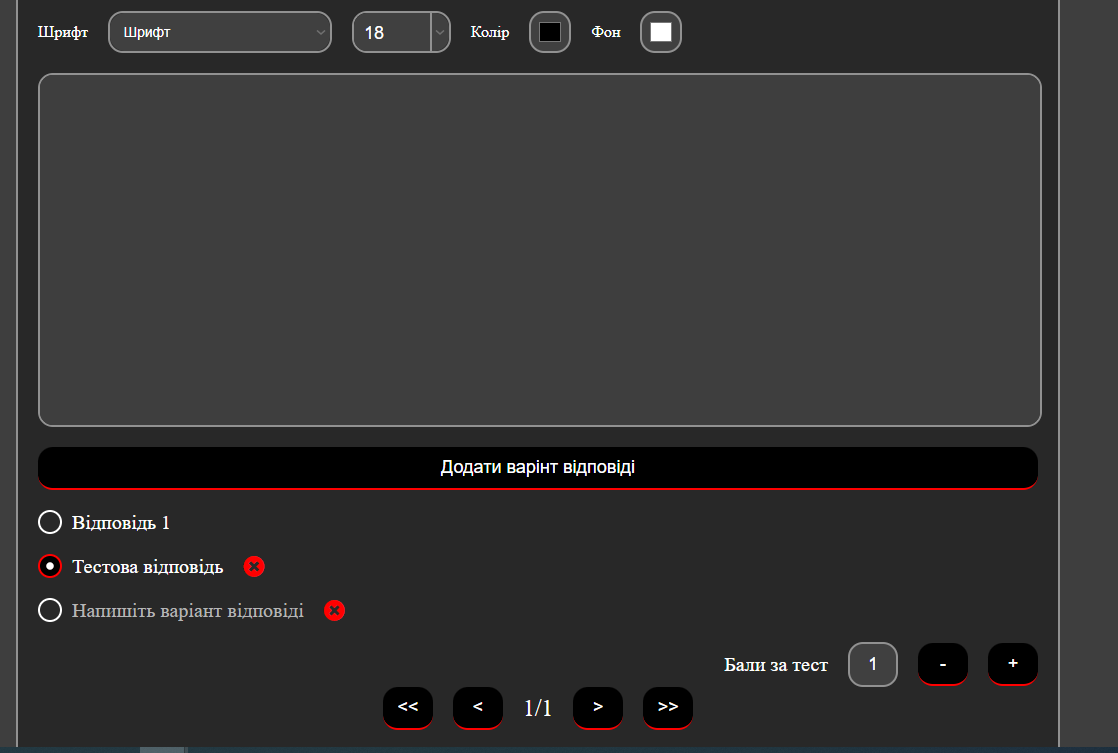


Рисунок 2.39 Тест з одною правильною відповіддю

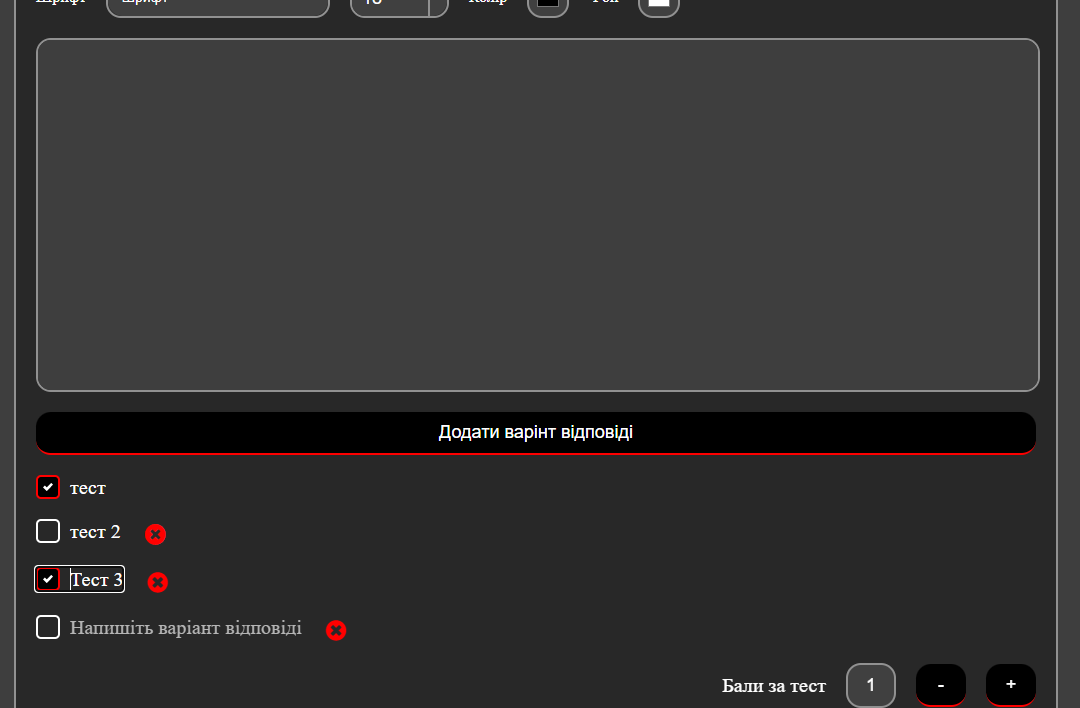


Рисунок 2.40 Тест з декількома правильними відповідями

При створенні нової теми, завдання чи тесту всім користувачам відправляється лист на Email про це.

Якщо викладач або власник перейде на завдання, то він може переглянути роботи учнів якщо вони їх здавали (див. рисунок 2.41 – 2.42).

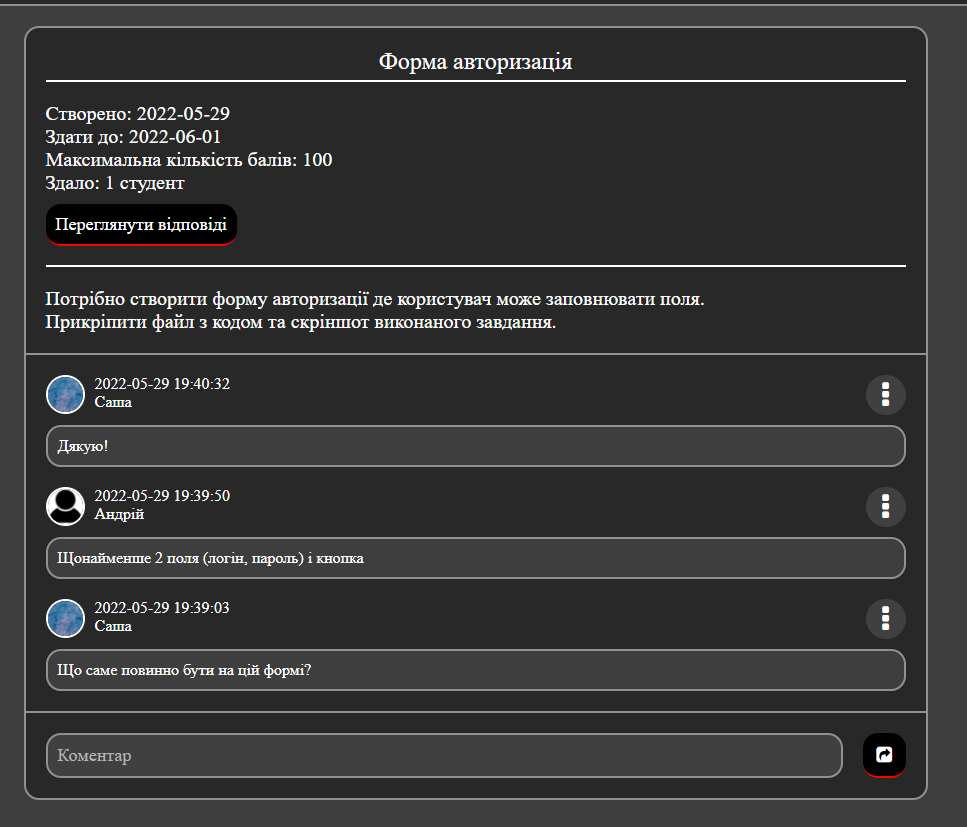


Рисунок 2.41 Здані завдання

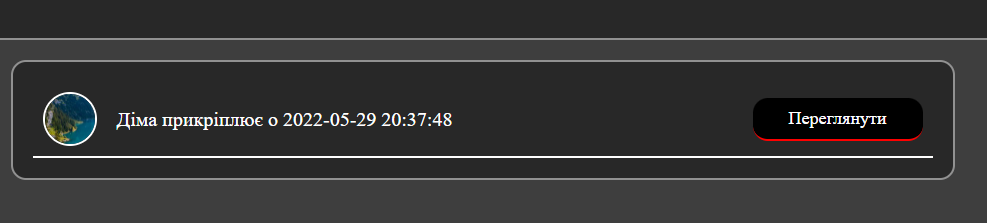


Рисунок 2.42 Вибір зданої роботи

Щоб переглянути відповідь потрібно натиснути кнопку «Переглянути». Відкриється сторінка з відповідями учня (див. рисунок 2.43).



Рисунок 2.43 Відповіді учня

Також викладачі бачать результати всіх учнів на курсі.

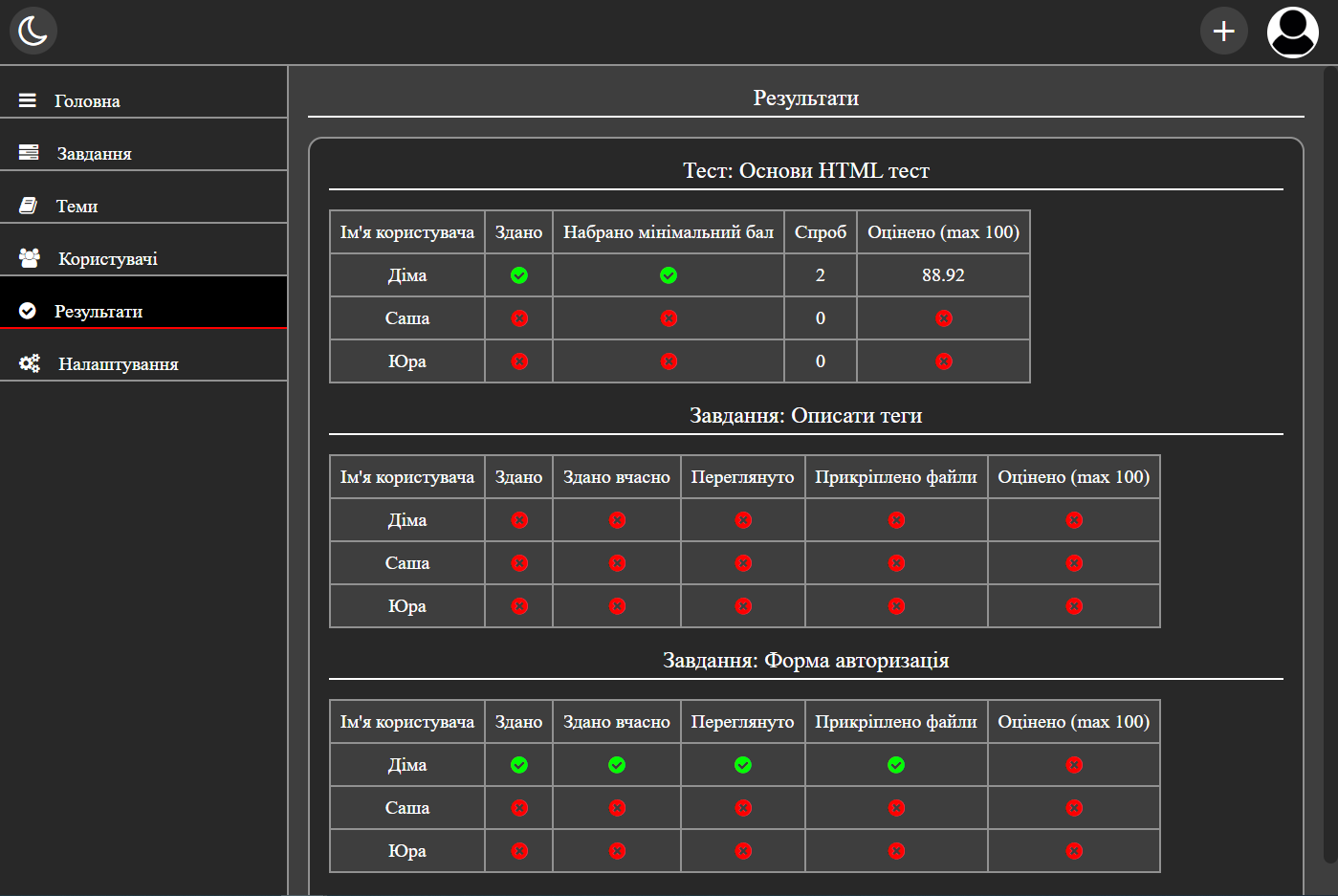


Рисунок 2.44 Результати учнів

Власник може налаштовувати курс, для цього йому потрібно обрати пункт «Налаштування» в меню (див. рисунок 2.45).

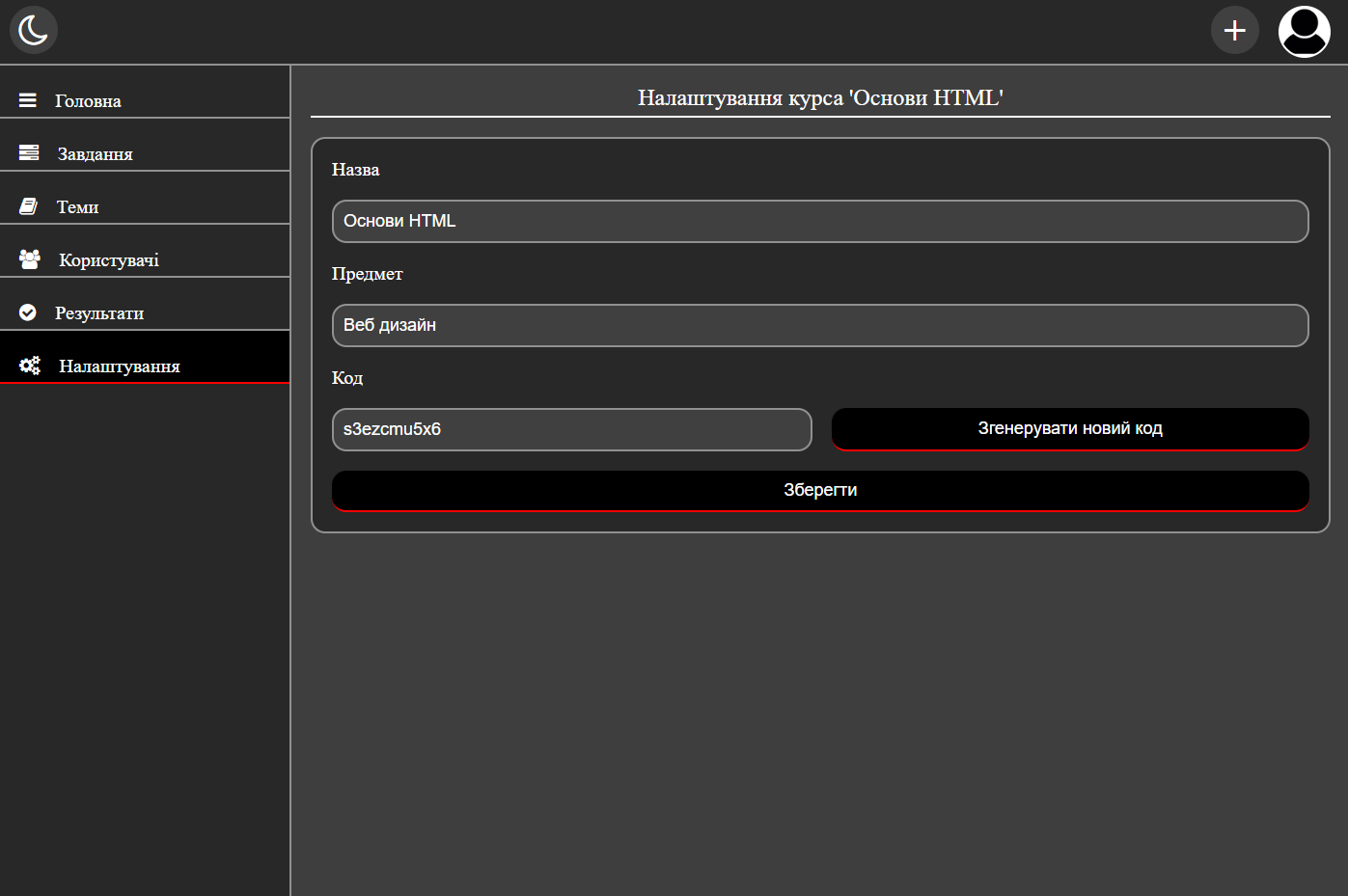


Рисунок 2.45 Налаштування курса

Він може змінювати назву, предмет та згенерувати новий код.

Також власник може надавати права викладача та виключати користувачів з курсу (див. рисунки 2.46 – 2.47).

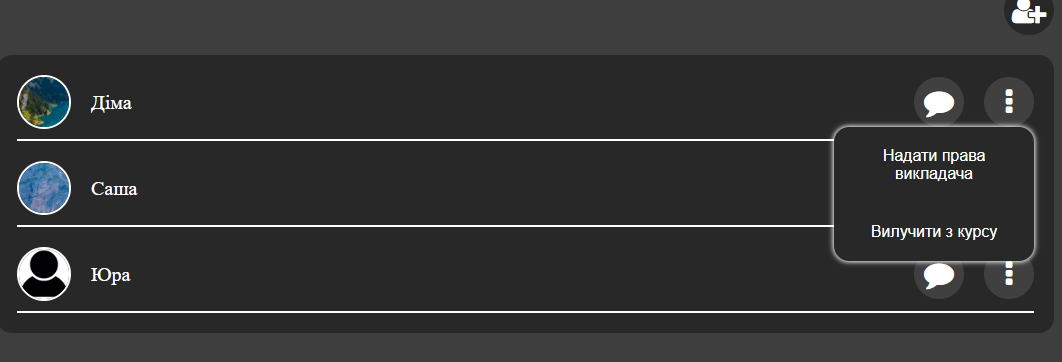


Рисунок 2.46 Вибір операції для учня



Рисунок 2.47 Вибір операції для викладача

Також тільки власник курсу може запрошувати нових учнів та викладачів. Для цього потрібно натиснути кнопку «Запросити викладача» або «Запросити учня». Після цього потрібно вказати Email (див. рисунки 2.48 – 2.49).

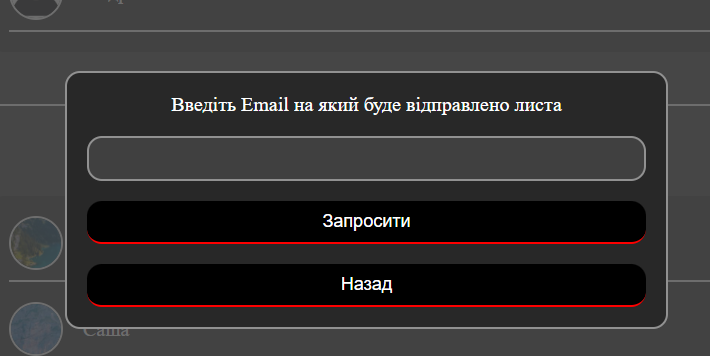


Рисунок 2.48 Запрошення користувача

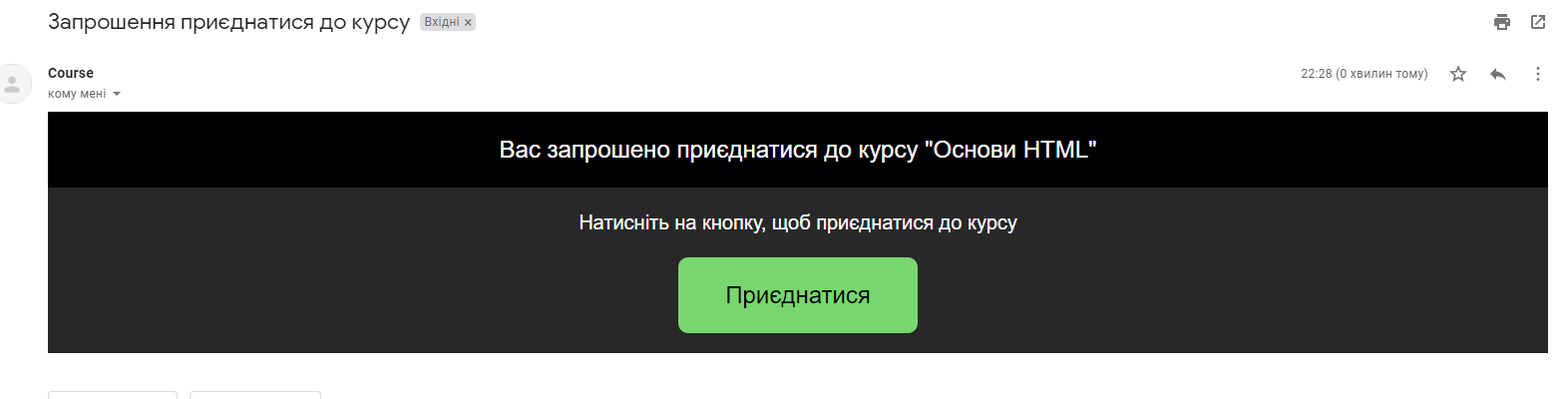


Рисунок 2.49 Запрошення

Якщо користувач авторизований, то він автоматично приєднається до курсу. Якщо не авторизований то він буде перенаправлений на сторінку авторизації, а потім вже приєднається до курсу.

# ВИСНОВКИ

Темою курсового проекту була «Розробка клієнт-серверної системи для організації дистанційного навчання». В результаті було створено сайт з зрозумілим інтерфейсом, що надає можливість створювати курси. У курсах може бути теоретичний матеріал, завдання та тести, а також можна створювати повідомлення прикріплювати файли та залишати коментарі. Реалізовано чат між користувачами.

В курсах є три типи користувачів, це учні, викладачі та власник.

Викладачі та власник можуть створювати теоретичні матеріали, завдання та тести, а такою оцінювати роботи учнів та переглядати їхні результати.

Власник може налаштовувати курс, тобто змінювати його назву, предмет та згенерувати новий код. Також він може надавати права користувачам. Власник може запрошувати до курсу нових користувачів, як учнів так і викладачів. Одночасно може бути лише один власник курса.

Учні можуть переглядати свої результати, залишати коментарі та повідомлення в головній стрічці. Також виконувати завдання та проходити тести.

Всі користувачі можуть створювати свої курси та приєднуватись до курсів інших користувачів.

Також надсилаються електронні листи на Email при спробі змінення паролю, або коли створюється нова тема, завдання чи тест. Також листи надсилаються коли власник запрошує користувача на курс.

У курсу є унікальний код, який бачать тільки викладачі та власник, власник може його змінити. Саме по цьому коду можна приєднатись до курсу.

# ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. [Електронний ресурс] – PHP – https://uk.wikipedia.org/ wiki/PHP.
2. [Електронний ресурс] – JavaScript – https://uk.wikipedia/ wiki/JavaScript
3. [Електронний ресурс] – Node.js – https://uk.wikipedia.org/wiki/Node.js
4. [Електронний ресурс] – AJAX – https://uk.wikipedia.org/wiki/AJAX
5. [Електронний ресурс] – jQuery – https://uk.wikipedia.org/wiki/jQuery
6. [Електронний ресурс] – MySQL – https://uk.wikipedia.org/wiki/MySQL
7. [Електронний ресурс] – MySQL – https://www.oracle.com/ru/mysql/
8. [Електронний ресурс] – stackoverflow – https://stackoverflow.com/
9. Джон Дакетт «Javascript і jQuery. Інтерактивна веб-розробка»
10. [Електронний ресурс] – JavaScript – https://developer.mozilla.org/en-US/docs/Web/JavaScript?retiredLocale=uk
11. [Електронний ресурс] – JavaScript AJAX – https://www.tutorialrepublic.com/javascript-tutorial/javascript-ajax.php
12. [Електронний ресурс] – jQuery – https://jquery.com
13. [Електронний ресурс] – jQuery tutorial – https://www.w3schools.com/jquery/default.asp
14. [Електронний ресурс] – WebSocket – https://developer.mozilla.org/en-US/docs/Web/API/WebSocket?retiredLocale=uk
15. [Електронний ресурс] Node.js з нуля – https://www.udemy.com/course/nodejs-start/
16. PHP & MySQL (Люк Веллінг, 2001)

# Додатки

Лістинг файлу index.php

<html>

<head>

<meta charset="utf-8">

</head>

<?php

session\_start();

if (!isset($\_SESSION['email'])) {

header("Location: login");

}

if (isset($\_SESSION['invite'])) {

$id = $\_SESSION['invite'];

unset($\_SESSION['invite']);

header("Location: invite?id=".$id);

}

$dir = "";

$courseDIR = "course/";

include("header.php");

?>

<div class='main-container'>

<div class='menu'>

<div id='all-course' class='menuButton active'>Всі курси

</div>

<div id='my-course' class='menuButton'>Мої курси

</div>

<div id='teach-course' class='menuButton'>Курси в яких ви навчаєте

</div>

<div id='courses' class='menuButton'>Курси в яких ви навчаєтесь

</div>

</div>

<div class='courses-container' id='courses-container'>

</div>

</div>

<script src='js/menu.js'></script>

<script type="text/javascript">

$(document).ready(function() {

$.ajax({

type: "POST",

url: "loadCourses.php",

async: false,

data: {"type":1},

success: function(data) {

document.getElementById("courses-container").innerHTML = data

}

});

});

function loadCourse(type) {

$.ajax({

type: "POST",

url: "loadCourses.php",

async: false,

data: {"type":type},

success: function(data) {

document.getElementById("courses-container").innerHTML = data

}

});

}

$(document).on('click', '#all-course', function(){

setActive();

document.getElementById("all-course").classList.add("active");

loadCourse(1);

});

$(document).on('click', '#my-course', function(){

setActive();

document.getElementById("my-course").classList.add("active");

loadCourse(2);

});

$(document).on('click', '#courses', function(){

setActive();

document.getElementById("courses").classList.add("active");

loadCourse(3);

});

$(document).on('click', '#teach-course', function(){

setActive();

document.getElementById("teach-course").classList.add("active");

loadCourse(4);

});

jQuery(document).ready(function($) {

window.realAlert=window.alert

window.alert=function(s){

customAlert(s)

}

function customAlert(s) {

if($('#customalert')[0]) {

if($('#customalert').hasClass('on')) {

$('#customalert').find('.alerttext').html(s);

}else{

$('#customalert').addClass('on').find('.alerttext').html(s);

}

}else{

$('body').append('<div id="customalert" class="on"><span class="alerttext">' + s + '</span></div>');

}

$('#customalert').fadeIn(1);

}

$(document).on('click', function(e) {

$('#customalert').removeClass('on').fadeOut();

});

});

</script>

<html>

Лістинг файлу login.php

<html>

<head>

<meta name="viewport" content="width=device-width, initial-scale=1">

<meta charset="utf-8">

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

<link rel="stylesheet" href="css/font-awesome.css">

<script src="https://code.jquery.com/jquery-3.3.1.js" integrity="sha256-2Kok7MbOyxpgUVvAk/HJ2jigOSYS2auK4Pfzbm7uH60=" crossorigin="anonymous"></script>

</head>

<div class='fone' >

<div class='enter-password' style='z-index: 2; display: block;'>

<div style='text-align: center;margin-top: 20px;margin-bottom: 20px;'>

<label style='font-size:24px;'>Відновлення паролю</label>

</div>

<form id='codeConfirm' method='post'>

<label for="passwordforgot">Код підтвердження</label>

<input class='field' id='codeField' type='text' name='passwordforgot' required />

<!--<label style='color:red;'>2</label> -->

<button type='submit' id="codeButton" name='codeButton' class='sign'>Підтвердити</button>

<!--<input type='button' id="codeButton" name='codeButton' class='sign' value="Підтвердити" /> -->

<div class='separator' style='margin-top:20px;'></div>

<button class='sign' type='button' onclick="back()">Назад</button>

</form>

</div>

</div>

<!-- контейнер форм логін, реєстрація, забули пароль -->

<div class='container'>

<div class='forgot-form' >

<!-- введення нового паролю -->

<div class='changePassword' style="">

<div style='text-align: center; margin-top: 20px;margin-bottom: 20px;'>

<label style='font-size:24px;'>Зміна паролю</label>

</div>

<form class='changePasswordForm' id='changePasswordForm' method='post'>

<input type='hidden' id='emailHidden' name='email' />

<label for="password">Введіть новий пароль</label>

<input class='field' id='newPass' type='password' name='password' required />

<label for="Cpassword">Повторіть пароль</label>

<input class='field' id='cNewPass' type='password' name='Cpassword' required />

<input class='sign' name='newPassword' type='submit' value='Зберегти'/>

<div class='separator' style='margin-top:20px;'></div>

<button class='sign' type='button' onclick="changeForgotPassword()">Назад</button>

</form>

</div>

<div class='fg'>

<div style='text-align: center; margin-top: 20px;margin-bottom: 20px;'><label style='font-size:24px;'>Відновлення паролю</label><div style='text-align: justify; margin-top:10px;'>

<label >Для відновлення паролю потрібно вказати ваш Email. На нього прийде лист з кодом, який буде потрібний для скидання паролю.</label></div></div>

<form class='codeSend' id='codeSend' method='post'>

<label for="email">Email</label>

<input class='field' id='forgotEmail' type='text' pattern="[^@\s]+@[^@\s]+\.[^@\s]+" title="email@example.com" name='email' required />

<input class='sign' name='forgot' type='submit' value='Відправити'/>

</form>

<div class='separator' style='margin-top:20px;'></div>

<button class='sign' type='button' onclick="changeForgot()">Назад</button>

</div>

</div>

<div class='login-registration'>

<!-- кнопка переходу з логіна на реєстрацію -->

<div class='nav'>

<div class='buttonDiv' onClick='changeSignIn()' >

<button class='change active' >Вхід</button>

<br>

<label class='labelChange active'> </label>

</div>

<div class='buttonDiv' onClick='changeSignIn()'>

<button class='change' >Реєстрація</button>

<br>

<label class='labelChange'> </label>

</div>

</div>

<div class='panel'>

<div class='sign-in'>

<form id='Userlogin' method='post'>

<label for="email">Email</label>

<input class='field' pattern="[^@\s]+@[^@\s]+\.[^@\s]+" title="email@example.com" type='text' name='email' id='logEmail' required />

<label for="password">Пароль</label>

<input class='field' id='logPassword' type='password' name='password' required/>

<label class="checkcontainer">Запам'ятати мене

<input type="checkbox" name='check'>

<span class="checkmark"></span>

</label>

<input class='sign' type='submit' value='Ввійти' />

<br>

<div class='separator'></div>

<div style='text-align: center;'>

<a class='forgot' onClick='changeForgot()' >Забули пароль?</a>

</div>

</form>

</div>

<div class='sign-up'>

<form id='registration' method='post'>

<label for="name">Ваше ім'я</label>

<input class='field' type='text' name='name' required/>

<label for="email">Email</label>

<input class='field' pattern="[^@\s]+@[^@\s]+\.[^@\s]+" title="email@example.com" type='text' name='email' id='regEmail' required/>

<label for="password">Пароль</label>

<input class='field' id='regPassword' type='password' name='password' required/>

<label for="confirn-password">Повторіть пароль</label>

<input class='field' id='conRegPass' type='password' name='confirn-password' required/>

<input class='sign' type='submit' value='Зареєструватись'/>

</form>

</div>

</div>

</div>

</div>

<script src='js/login.js'></script>

<script type="text/javascript">

$(document).ready(function() {

$('#codeSend').submit(function(e) {

e.preventDefault();

console.log(document.getElementById("forgotEmail").value);

$.ajax({

type: "POST",

url: 'email/emailSender.php',

data: {

'email':document.getElementById("forgotEmail").value,

'subject':'Підтвердження операції відновлення паролю',

'type': 1

}

,

success: function(response)

{

console.log(response.split("Message sent!"));

if($('#customalert').hasClass('on')) {

$('#customalert').removeClass('on').fadeOut();

}

if (response.split("Message sent!")[1] == '1') {

document.querySelector('.fone').classList.toggle("show");

document.querySelector('.enter-password').classList.toggle("show");

return;

}

else

{

alert("Email не зареєтровано!");

}

}

});

});

$('#codeConfirm').submit(function(e) {

e.preventDefault();

$.ajax({

type: "POST",

url: 'email/checkCode.php',

data: $(this).serialize(),

success: function(response)

{

var text = document.getElementById("codeField");

if($('#customalert').hasClass('on')) {

$('#customalert').removeClass('on').fadeOut();

}

var jsonData = JSON.parse(response);

if (jsonData.success == "1")

{

document.querySelector('.fone').classList.toggle("show");

document.querySelector('.enter-password').classList.toggle("show");

document.querySelector('.changePassword').classList.toggle("down");

document.querySelector('.fg').classList.toggle("down");

text.value = "";

document.getElementById('emailHidden').value = document.getElementById('forgotEmail').value;

}

else {

alert("Не правильний код!");

}

}

});

});

$('#Userlogin').submit(function(e) {

e.preventDefault();

$.ajax({

type: "POST",

url: 'checkLogin.php',

data: $(this).serialize(),

success: function(response)

{

if($('#customalert').hasClass('on')) {

$('#customalert').removeClass('on').fadeOut();

}

var jsonData = JSON.parse(response);

if (jsonData.success == "0") {

alert("Не правильний Email або пароль!");

}

else {

window.location.href = "index";

}

}

});

});

$('#changePasswordForm').submit(function(e) {

e.preventDefault();

$.ajax({

type: "POST",

url: 'changePassword.php',

data: $(this).serialize(),

success: function(response)

{

if($('#customalert').hasClass('on')) {

$('#customalert').removeClass('on').fadeOut();

}

var jsonData = JSON.parse(response);

if (jsonData.success == "0") {

alert("Паролі не співпадають!");

}

else if (jsonData.success == "2") {

alert("Пароль повинен бути щонайменше 6 символів!");

}

else {

greenMessage("Пароль змінено!");

changeForgotPassword();

document.getElementById('newPass').value = "";

document.getElementById('cNewPass').value = "";

}

}

});

});

$('#registration').submit(function(e) {

e.preventDefault();

$.ajax({

type: "POST",

url: 'registration.php',

data: $(this).serialize(),

success: function(response)

{

if($('#customalert').hasClass('on')) {

$('#customalert').removeClass('on').fadeOut();

}

var jsonData = JSON.parse(response);

if (jsonData.success == "1")

{

greenMessage("Зареєстровано!");

changeSignIn();

document.getElementById('logEmail').value = document.getElementById('regEmail').value;

document.getElementById('logPassword').value = document.getElementById('regPassword').value;

}

else if (jsonData.success == "0") {

alert("Такий Email уже зареєстровано!");

}

else if (jsonData.success == "2") {

alert("Паролі не співпадають!");

}

else if (jsonData.success == "3") {

alert("Пароль повинен бути щонайменше 6 символів!");

}

}

});

});

});

function greenMessage(message) {

if($('#greenAlertText')[0]) {

}else{

$('body').append('<div id="greenAlertText" class="on"><span class="greenAlertText">' + message + '</span></div>');

}

$('#greenAlertText').fadeIn(1);

$(document).on('click', '#greenAlertText', function(){

$(this).removeClass('on').fadeOut();

});

}

jQuery(document).ready(function($) {

window.realAlert=window.alert

window.alert=function(s){

customAlert(s)

}

function customAlert(s) {

if($('#customalert')[0]) {

if($('#customalert').hasClass('on')) {

$('#customalert').find('.alerttext').html(s);

}else{

$('#customalert').addClass('on').find('.alerttext').html(s);

}

}else{

$('body').append('<div id="customalert" class="on"><span class="alerttext">' + s + '</span></div>');

}

$('#customalert').fadeIn(1);

}

$(document).on('click', '#customalert', function(){

$(this).removeClass('on').fadeOut();

});

});

</script>

<html>

Лістинг файлу registration.php

<?php

include("db/dbHelper.php");

$db = new dbHelper();

if ($db->checkEmail($\_POST['email'])) {

echo json\_encode(array('success' => 0));

}

else if (strlen($\_POST['password']) < 6) {

echo json\_encode(array('success' => 3));

}

else if ($\_POST['password'] !== $\_POST['confirn-password']) {

echo json\_encode(array('success' => 2));

}

else {

$db->insertIntoDB('INSERT INTO users (name, email, password, token) VALUES ("'.$\_POST['name'].'", "'.$\_POST['email'].'", "'.password\_hash($\_POST['password'], PASSWORD\_DEFAULT).'", "")');

echo json\_encode(array('success' => 1));

}

?>

Лістинг файлу logout.php

<?php

session\_start();

require\_once("db/dbConnector.php");

if(isset($\_SESSION["email"])){

$update\_password\_cookie\_token = $mysqli->query("UPDATE users SET token = '' WHERE email = '".$\_SESSION["email"]."'");

if(!$update\_password\_cookie\_token){

echo "Помилка ".$mysqli->error();

}else{

setcookie("password\_cookie\_token", "", time() - 3600);

}

session\_destroy();

header("Location: login");

exit();

return;

}

?>

Лістинг файлу logout.php

<html>

<head >

<div style='display: none;' id='theme' class='theme'></div>

<meta charset="utf-8">

<?php

echo "<link id='style-index' rel='stylesheet' href='".$dir."css/index.css'>

<link id='style' rel='stylesheet' href='".$dir."css/header.css'>

<link rel='stylesheet' href='".$dir."css/font-awesome.css'>";

?>

<script src="https://code.jquery.com/jquery-3.3.1.js" integrity="sha256-2Kok7MbOyxpgUVvAk/HJ2jigOSYS2auK4Pfzbm7uH60=" crossorigin="anonymous"></script>

<script>

var dir = "<?php echo $dir; ?>";

var id = "<?php echo $\_SESSION['userId']; ?>"

var email = "<?php echo $\_SESSION['email']; ?>"

if (localStorage.getItem(id) == 1) {

$('head').append("<link id='style-index' rel='stylesheet' href='"+ dir + "css/index-dark.css'>"+

"<link id='style' rel='stylesheet' href='" + dir + "css/header-dark.css'>");

document.querySelector('.theme').classList.toggle("dark");

}

else {

$('head').append("<link id='style-index' rel='stylesheet' href='"+ dir + "css/index-light.css'>"+

"<link id='style' rel='stylesheet' href='" + dir + "css/header-light.css'>");

}

</script>

</head>

<body id='menu-body' class='menu-body' style='margin:0px;'>

<div class='header'>

<div id='themeButton' class='themeButton'>

</div>

<!-- div user -->

<?php

require\_once("db/dbHelper.php");

$db = new dbHelper();

$id = $db->insertIntoDB("SELECT \* FROM users WHERE email ='".$\_SESSION['email']."'");

$id = mysqli\_fetch\_array($id);

echo "<div class='user' id='user' style='background: url(".$dir."users/".$id['image']."?hash=".random\_int(1, 100).") 100% 100% no-repeat; background-size: cover; '>";

echo "<input type='hidden' id='userId' name='userId' value='".$id['id']."'/>";

?>

<div class='dropdownUser' id='dropdownUser'>

<label class='userName'><?php echo $id['name']; ?></label>

<button id='return-course' style="width: 100%;" class='dropdown-button'>Курси</button>

<?php echo "<form style='margin-block-end: 0em;' action='".$dir."user/userSettings' method='post'>"; ?>

<button id='settingUD' style="width: 100%;" class='dropdown-button'>Налаштування</button>

</form >

<button onclick='location.href="<?php echo $dir;?>chat?id=0"' id='chatUD' class='dropdown-button'>Чат</button>

<form style='margin-block-end: 0em;' action='<?php echo $dir."logout.php"; ?>'>

<button id='exitUD' style="width: 100%;" type='submit' class='dropdown-button exit'>Вийти</button>

</form>

</div>

</div>

<div id='add' class='add'><div style='width:50px; display: flex;height: 50px;align-items: center;justify-content: center;align-content: center;' title='Створити або приєднатись до курсу'>+</div>

<div class='dropdown'>

<button id='create' class='dropdown-button'>Створити новий курс</button>

<button id='join' class='dropdown-button'>Приєднатися до курсу</button>

</div>

</div>

</div>

</body>

<script>

var userEmail = '<?php echo $\_SESSION['email']; ?>'

var courseDIR = '<?php echo $courseDIR; ?>';

var dir = "<?php echo $dir; ?>";

var theme = document.getElementById('themeButton');

var add = document.getElementById('add');

document.getElementById('user').onclick = function() {

if ($('#dropdownUser').hasClass('show')) {

document.getElementById('dropdownUser').classList.remove("show");

document.getElementById('dropdownUser').classList.add("hide");

}

else {

document.getElementById('dropdownUser').classList.add("show");

document.getElementById('dropdownUser').classList.remove("hide");

}

};

$(document).on('click', '#return-course', function(){

document.location.href = dir + "index";

});

theme.onclick = function() {

if($('#theme').hasClass('dark')) {

document.querySelector('.theme').classList.toggle("dark");

document.getElementById("style").href = dir + "css/header-light.css";

document.getElementById("style-index").href = dir + "css/index-light.css";

localStorage.setItem(document.getElementById("userId").value, 0);

}

else{

document.querySelector('.theme').classList.toggle("dark");

document.getElementById("style").href = dir + "css/header-dark.css";

document.getElementById("style-index").href = dir + "css/index-dark.css";

localStorage.setItem(document.getElementById("userId").value, 1);

}

}

add.onclick = function() {

document.querySelector('.add').classList.toggle("show");

document.querySelector('.dropdown').classList.toggle("show");

}

</script>

<script src='<?php echo $dir; ?>js/createOrJoin.js'></script>

</html>

Лістинг файлу course.php

<html>

<head>

</head>

<?php

session\_start();

if (!isset($\_SESSION['email'])) {

header("Location: ../login");

}

require\_once("../db/dbHelper.php");

$db = new dbHelper();

if ( $\_GET['id'] == "" || !$db->userContainsInCourse($\_GET['id']) ){

header("Location: ../index");

}

$dir = "../";

$courseDIR = "";

include("../header.php");

$cl = "menuButton";

?>

<div class='main-container'>

<div class='menu' id='mCont'>

<?php

include('loadMenu.php');

?>

</div>

<div class='tape-container' id='tape-container'>

<?php

if (isset($\_GET['users'])) {

include('loadUsers.php');

}

else if (isset($\_GET['task'])) {

include('loadTasks.php');

}

else if (isset($\_GET['themes'])) {

include ('loadThemes.php');

}

else if (isset($\_GET['results'])) {

include ('loadResults.php');

}

else if (isset($\_GET['settings'])) {

include ('settings.php');

}

else {

include('loadMainTape.php');

}

?>

</div>

</div>

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

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

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

<script src='../js/post.js'></script>

</html>

Лістинг файлу loadMainTape.php

<?php

if (isset($\_REQUEST['start'])) {

session\_start();

}

$dir = '../';

require\_once('../db/dbConnector.php');

require\_once('loadComment.php');

?>

<div class='courseInfo'>

<?php

$result = mysqli\_query($mysqli,'SELECT \* FROM course WHERE id = '.$\_REQUEST['id']);

$res = mysqli\_fetch\_array($result);

$own = $res['owner'];

echo '<div class="infoText" style="font-size:30px;"><label>Назва: '.$res['name'].'</label></div>';

echo '<div class="infoText"><label>Предмет: '.$res['subject'].'</label></div>';

if ($\_SESSION['userId'] == $res['owner']) {

echo '<div class="infoText"><label style="float:left">Код курсу: '.$res['code'].'</label></div>';

}

?>

</div>

<div id='sendMessage' class='userMessage' >

<div id='sendMessageDiv' class='send-text' >

Написати повідомлення

<?php

if (!isset($\_REQUEST['start'])) {

?>

<script src='../js/mainTape.js'></script>

<?php

}

?>

</div>

</div>

<div id='post' class='post'>

<?php

$result = mysqli\_query($mysqli,'SELECT users.name, users.image, themes.section, themes.name as themename, task.name as taskname, post.date, post.text, post.type, post.file, post.id as postId, themes.id as themeId, task.id as taskId, test.id as testId, test.name as testname, post.user FROM users

LEFT JOIN post on post.user = users.id

LEFT JOIN themes on themes.id = post.text

LEFT JOIN task on task.id = post.text

LEFT JOIN test on test.id = post.text

WHERE post.courseId = '.$\_REQUEST['id'].' AND users.id = post.user ORDER BY post.date DESC');

$comments = mysqli\_query($mysqli, 'SELECT comment.id as cId, comment.userId as uId, comment.postId, comment.text, comment.date, users.name, users.image, post.date as pDate, post.courseId FROM comment

LEFT JOIN post ON post.id = comment.postId

LEFT JOIN users ON users.id = comment.userId

WHERE post.courseId = '.$\_REQUEST['id'].'

ORDER BY post.date DESC, comment.date DESC');

$records = array();

while($r = mysqli\_fetch\_array($comments)) {

$records[] = $r;

}

if (mysqli\_num\_rows($result) == 0) {

echo '<div style="margin-bottom:20px;" class="postmessage" >Ще не має постів в цьому курсі</div>';

}

else {

while ($res = mysqli\_fetch\_array($result)) {

if ($res['type'] == 1) {

?>

<div id='<?php echo $res["postId"]; ?>' class='userPostMessage'>

<?php

}

else if ($res['type'] == 2){

?>

<a href='task?id=<?php echo $\_REQUEST['id'];?>&task=<?php echo $res["taskId"]; ?>' id='<?php echo $res["taskId"]; ?>' class='userPostMessage link'>

<?php

}

else if ($res['type'] == 3) {

?>

<a href='theme?id=<?php echo $\_REQUEST['id'];?>&theme=<?php echo $res["themeId"]; ?>' id='<?php echo $res["themeId"]; ?>' class='userPostMessage link'>

<?php

}

else if ($res['type'] == 4) {

?>

<a href='test?id=<?php echo $\_REQUEST['id'];?>&test=<?php echo $res["testId"]; ?>' id='<?php echo $res["testId"]; ?>' class='userPostMessage link'>

<?php

}

?>

<div class='userPostInfo'>

<div class='uImg'><img class='UsImg' src='../users/<?php echo $res["image"] ?>'></div>

<div class='infoPostText'>

<div class='postText' ><?php echo $res['date']; ?></div>

<div class='postText' ><?php echo $res['name']; ?></div>

</div>

<?php

if ($\_SESSION['userId'] == $own || $\_SESSION['userId'] == $res['user']) {

?>

<div class='postSett' >

<div id='' class='postMenu <?php echo ($res["type"] != 1) ? "link" : ""; ?>'>

</div>

</div>

<?php

}

?>

</div>

<?php

if ($res['type'] == 1) {

?>

<div class='MessagePost'><?php echo $res['text']; ?></div>

<?php

if ($res['file'] != '') {

?> <div class='sepWhite' ></div> <?php

}

?>

<div class='attachedFilesPost'>

<?php

if ($res['file'] != '') {

$imgArr = ['jpg','png','gif','jpeg','bmp'];

$archiveArr = ["rar", "zip"];

$textArr = ["txt"];

$wordArr = ['doc', 'docx','docm','rtf'];

$pdf = ['pdf'];

$codeArr = ['php','html','js','css','java','cpp','bat','c','pl','py','rb','sh','cs','class','pyo'];

$files = explode("\n", $res['file']);

for ($i = 0; $i < sizeof($files);$i++) {

$mime = explode(".", $files[$i]);

$type = '';

?>

<div <?php if($i == 0) {echo "style='margin-top:0px;'";}?> id='idAttach' class='attachFile attachLight'>

<div class='infoAttach'>

<div class='imgAttach'>

<?php

if (in\_array(strtolower($mime[sizeof($mime)-1]), $imgArr)) {

echo "<img class='uploadImg' src='".$dir."courses/".$\_REQUEST['id']."/".$files[$i]."' />";

}

else if (in\_array(strtolower($mime[sizeof($mime)-1]), $archiveArr)){

$type = 'archiveType';

}

else if (in\_array(strtolower($mime[sizeof($mime)-1]), $textArr)) {

$type = 'textType';

}

else if (in\_array(strtolower($mime[sizeof($mime)-1]), $wordArr)) {

$type = 'wordType';

}

else if (in\_array(strtolower($mime[sizeof($mime)-1]), $pdf)) {

$type = 'pdfType';

}

else if (in\_array(strtolower($mime[sizeof($mime)-1]), $codeArr)) {

$type = 'codeType';

}

else {

$type = 'fileType';

}

if ($type != '') {

echo '<div class="'.$type.' typeImg"></div>';

}

?>

</div>

<div class='nameAttach'>

<a class='nameAttach' download href="<?php echo $dir."courses/".$\_REQUEST['id']."/".$files[$i]; ?>"><?php echo $files[$i]; ?></a>

</div>

</div>

</div>

<?php

}

}

?>

</div>

<?php

loadComment($records, $res["postId"]);

?>

<div class='sepLine'></div>

<div class='sendCommentCont'>

<div style='width: auto;margin: 20 0 0 0;' id='messUpdate' class='messUpdate mhide'></div>

<div style='width: auto;display: flex;'>

<div id='postCommentArea' contenteditable="plaintext-only" class='field commentDiv' name='comment' data-placeholder='Коментар' ></div>

<button id='sendComment' class='button sendComment' ></button>

</div>

</div>

<?php

}

else if ($res['type'] == 3) { //theme

?>

<div class='typePost theme'>Користувач додає нову тему: "<?php echo $res['themename'];?>"</div>

<?php

}

else if ($res['type'] == 2) { //task

?>

<div class='typePost task'>Користувач додає нове завдання: "<?php echo $res['taskname'];?>"</div>

<?php

}

else if ($res['type'] == 4) { //test

?>

<div class='typePost test'>Користувач додає новий тест: "<?php echo $res['testname'];?>"</div>

<?php

}

if ($res['type'] == 1) {

?>

</div>

<?php

}

else {

?>

</a>

<?php

}

}

}

?>

</div>

<script >

window.addEventListener('beforeunload', function (e) {

$.ajax({

type: "POST",

url: 'reloadHandler.php',

data: {

},

success: function(response)

{

console.log(response);

}

});

}, false);

</script>