Міністерство освіти і науки України
Криворізький національний університет
Кафедра моделювання та програмного забезпечення

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти бакалавра

за напрямом підготовки 121 – Інженерія програмного забезпечення

На тему: Розробка програмного додатку для оптимізації навчального процесу

Засвідчую, що в цій
кваліфікаційній роботі немає
запозичень із праць інших авторів
без відповідних посилань.
Студентка гр. ІПЗ-20-1
_____/М. С. Шиян /

Керівник
кваліфікаційної роботи / А. А. Трачук /

Завідувач кафедри / А. М. Стрюк/

Кривий Ріг

2024

Криворізький національний університет

Факультет: Інформаційних технологій

Кафедра: Моделювання та програмного забезпечення

Ступінь вищої освіти: бакалавр

Спеціальність: 121 – Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:

Зав. кафедри

_____ А. М. Стрюк

«___»_____20____р.

ЗАВДАННЯ

на кваліфікаційну роботу

студентці групи ІПЗ-20-1 Шиян Марині Сергіївні

1. Тема: Розробка програмного додатку для оптимізації навчального процесу

Затверджено наказом по КНУ № 275с від «15» квітня 2024р.

2. Термін подання студентом закінченої роботи: «11» квітня 2024р.
3. Вихідні дані по роботі: система, що надає можливість для оптимізації навчального процес, управління контентом сайту в залежності від ролі користувача.
4. Зміст пояснювальної записки (перелік питань, що їх треба розробити): здійснити аналіз існуючих аналогів розроблюваної системи, визначити вимоги до функціоналу системи, здійснити програмну реалізацію розробленої системи.
5. Перелік ілюстративного матеріалу: структурна схема системи, функціональна діаграма системи, діаграма варіантів використання, блок-схеми функціональних модулів, макети інтерфейсу користувача.

Календарний план:

№	Найменування етапів кваліфікаційної роботи	Терміни виконання етапів роботи
1	Огляд літератури за тематикою та збір даних	15.04.2024–17.04.2024
2	Пошук та аналіз існуючих аналогів розроблюваної системи	18.04.2024–20.04.2024
3	Підготовка матеріалів першого розділу роботи	21.04.2024–25.04.2024
4	Формування функціональних вимог системи, проектування бази даних	25.04.2024–30.04.2024
5	Підготовка матеріалів другого розділу роботи	1.05.2024–5.05.2024
6	Розробка макетів інтерфейсу користувача, вибір технологій реалізації системи та реалізація користувацького інтерфейсу	6.05.2024–16.05.2024
7	Реалізація серверної частини системи	17.04.2024–24.05.2024
8	Налагодження клієнтської та серверної частини системи	25.05.2024–30.05.2024
9	Підготовка матеріалів третьої частини роботи	1.06.2024–04.06.2024
10	Оформлення пояснювальної записки	05.06.2024–10.06.2024

Дата видачі завдання: «15» квітня 2024 р.

Студентка _____/ М. С. Шиян /

Керівник роботи _____/ А. А. Трачук /

РЕФЕРАТ

НАВЧАННЯ, ОСВІТНІЙ ПРОЦЕС, ОПТИМІЗАЦІЯ НАВЧАЛЬНОГО ПРОЦЕСУ, АТОМАТИЗОВАНА СИСТЕМА.

Пояснювальна записка: 73с., 4 табл., 16 рис., 1 дод., 6 джерел.

Метою кваліфікаційної роботи є розробка програмного додатку для оптимізації навчального процесу.

У роботі проведено аналіз існуючих аналогів розроблюваного додатку. Виконано їх порівняння з приводу ефективності та простоти використання. Для розв’язання даної задачі в роботі, було використано архітектуру модульного проектування, яка забезпечить розширення системи за потребою, додаючи нові функції. Розроблено клієнтські та серверні частини додатку. Виконано тестування розробленої системи.

ABSTRACT

LEARNING, EDUCATIONAL PROCESS, OPTIMIZATION OF THE EDUCATIONAL PROCESS, AUTOMATED SYSTEM.

Thesis in: 73 p., 4 tab., 16 fig., 1 app., 6 references.

The purpose of the qualification work is the development of a software application to optimize the educational process.

The paper analyzes the existing analogues of the developed application. They were compared in terms of efficiency and ease of use. To solve this problem in the work, a modular design architecture was used, which will ensure the expansion of the system as needed, adding new functions. The client and server parts of the application have been developed. The developed system was tested.

ЗМІСТ

ВСТУП

У зв’язку з ситуацією в країні останні декілька років, весь навчальний процес студентів зосереджений у дистанційному форматі. Навчальний процес як студентів, так викладачів стикається з певними проблемами, які пов’язані з організацією та ефективністю навчання. Як студенти, так і викладачі стикаються з труднощами з управління розкладом, відстеженням домашніх завдань, практичних та лекційних завдань, доступом до начальних матеріалів та комунікацією. Звичайні методи організації навчання, такі як паперовий розклад, ручні записи завдань, комунікація між студентами та викладачами тільки в університеті, можуть призводити до неефективності, особливо в наших реаліях життя, таких як швидка зміна умов та великий обсяг інформації. Це призведе до зниження продуктивності навчання, пропусків дедлайнів та інше.

Таким чином, виникає необхідність інтегрування сучасних технологічних рішень, які оптимізують навчальний процес. Розробка додатку, що забезпечить функції для управління розкладом, відстеження завдань, доступ до матеріалів та комунікацію, може покращити процес навчання. Цей додаток повинен бути інтуїтивно зрозумілим, легким у використанні та забезпечувати конфіденційність та безпеку даних.

Метою даної дипломної роботи є створення програмного додатку, що допоможе студентам та викладачам ефективно управляти навчальним процесом, забезпечувати всі необхідні функції в одному додатку. При розробці даного додатку, будемо враховувати специфіку навчального процесу та індивідуальні потреби користувачів.

РОЗДІЛ 1. АНАЛІЗ ПРОБЛЕМИ ТА ПОСТАВНОКА ЗАДАЧІ

1.1. Аналіз професійної області проблеми

Організація навчального процесу є складним завданням, яке вимагає часу та сконцентрованості. У даній професійній області є певні проблеми, які впливають на ефективність навчання. Розглянемо основні проблеми та можливості їх вирішення:

- Управління розкладом

  Проблеми: основною причиною даної проблеми є часта зміна розкладу. Наприклад, перенесення практичних або лекційних занять, іспитів, заліків тощо.

  Рішення:

  1) Розробка додатку, який дозволяє оновлювати та синхронізувати розклад в режимі реального часу.
  2) Введення системи сповіщень про зміни в розкладі.

- Відстеження домашніх завдань

  Проблеми:

  1) Кількість завдань: зазвичай у студентів дуже насичений розклад з багатьма різними предметами, що може ускладнювати відстеження всіх домашніх завдань.
  2) Дедлайни: відсутність інструментів для нагадування про дедлайни завдань, може призвести до пропусків термінів виконання завдання, що може вплинути на оцінювання.

  Рішення:

  1) У додатку мають бути функції створення запису про домашнє завдання, встановлення термінів здачі та можливість сповіщення про термін здачі.

2) Можливість зазначення пріоритету завдання для ефективного розподілення часу студента.

- Доступ до навчальних матеріалів

Проблеми:

1) Матеріали можуть знаходитися на різних платформах або лише у друкованому вигляді, що може ускладнювати їх використання.
2) Проблеми з доступом до матеріалів поза університетом або без доступу до Інтернету.

Рішення:

1) Додаток має містити функції зберігання всіх матеріалів в одному місці.
2) Можливість завантаження матеріалів та використання їх в офлайн-режимі.

При вивчені даних проблем можна прийти висновку, що потрібно розробити комплексний додаток для оптимізації навчального процесу. У додатку мають бути такі функції, як управління розкладом, відстеження домашніх завдань, сповіщення про зміни в розкладі та дедлайни домашніх завдань, доступ до навчальних матеріалів.

1.2. **Аналіз існуючих аналогів**

Для того щоб почати роботу над розробкою додатку, проаналізуємо вже існуючі аналоги для оцінки переваг та недоліків, які потрібно буде врахувати при розробці. Для аналізу було вибрано два додатки: «My Study Life» та «Todolist».

На рисунку 1.1 зображено інтерфейс додатку «My Study Life». Розглянемо основні функції, недоліки та переваги даного додатку. У додатку є мобільна версія, яку підтримують такі операційні системи, як Android та iOS, також є веб-версія у вигляді сайту.

Основними функціями даного додатку є управління розкладом, відстеження завдань та іспитів, нагадування про дедлайни та заняття. При відстежування завдання можна зазначити, яку частина завдання було виконано у відсотках.

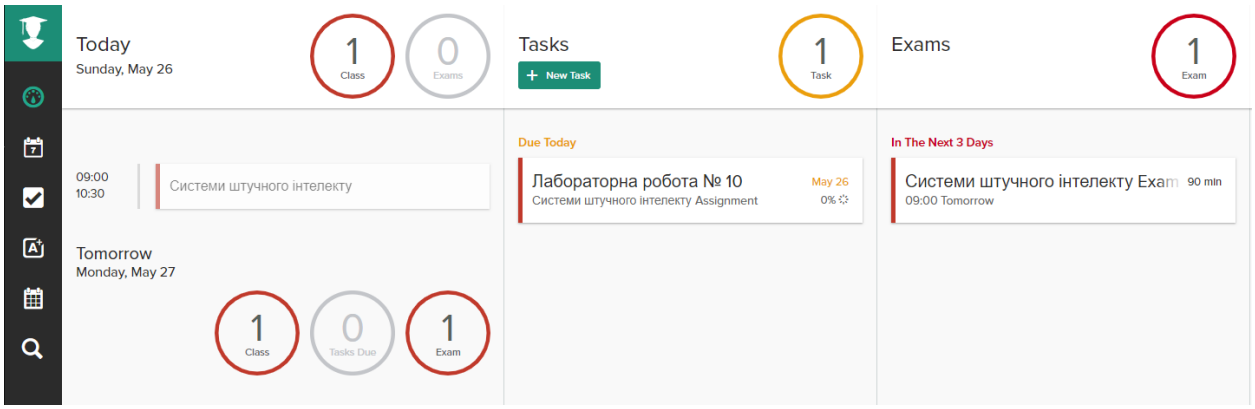

Рисунок Error: Reference source not found1 Інтерфейс додатку «My Study Life»

Інтерфейс додатку лаконічний та інтуїтивно зрозумілий. Розклад можна відобразити у вигляді календарю на місяць або тиждень. При створенні нового завдання можна зазначити такі дані: предмет, дату дедлайну, назву завдання та деталі завдання. При створенні екзамену можна зазначити такі дані: предмет, час початку, кабінет проведення іспиту, дату та тривалість іспиту.

Після аналізу додатку «My Study Life» визначили такі переваги:

1) Інтуїтивний та зручний інтерфейс.
2) Синхронізація даних на різних пристроях.
3) Безкоштовний доступ до базових функцій.

Також були виявлені недоліки даного додатку:

1) Відсутність можливості встановлення пріоритету завдання.
2) Відсутність можливості завантажувати та зберігати навчальні матеріали.

На рисунку 1.2 зображено інтерфейс додатку «Todolist». Розглянемо основні функції, переваги та недоліки даного додатку. У додатку є мобільна

версія, яку підтримують такі операційні системи, як Android та iOS, також є веб-версія у вигляді сайту.

Інтерфейс додатку досить простий та інтуїтивно зрозумілий. При додаванні нової задачі можна зазначити назву, пріоритет та дедлайн завдання. Є можливість фільтрування завдань за пріоритетом та дедлайном. Є декілька режимів відображення завдань: списком, дошкою та календарем. У додатку є платні функції, такі як, можливість нагадування та відображення завдань у вигляді календарю. Також є можливість створювати проекти, в які можна додавати завдання та ділитися ними з іншими користувачами.

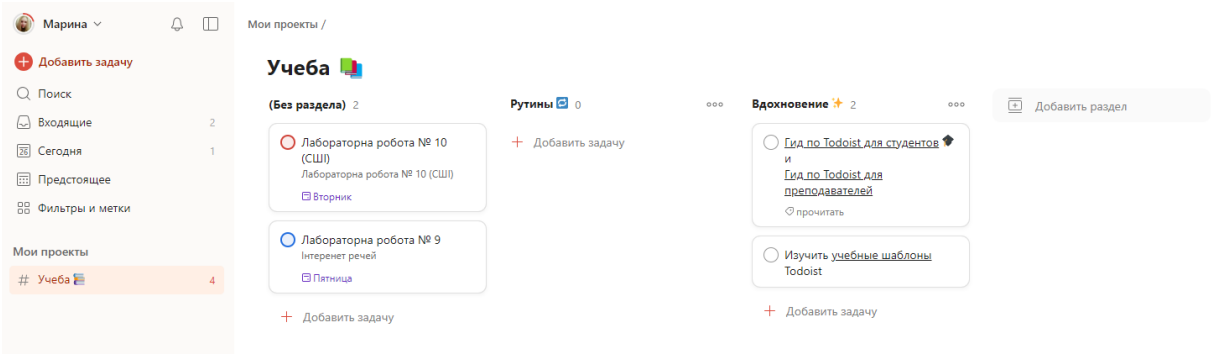

Рисунок Error: Reference source not found Інтерфейс додатку «Todolist»

Під час аналізу додатку «Todolist» було виявлено такі переваги:

1) Синхронізація між різними пристроями.
2) Простий та інтуїтивно зрозумілий дизайн.
3) Можливість роботи у команді.

Також було виявлено такі недоліки:

1) Платний доступ до розширених функцій.
2) Відсутність спеціальних функцій для навчального процесу. Наприклад, розклад занять.

Отже при аналізі існуючих аналогів, визначили їх переваги та недоліки. При розробці власного додатку, скористаємося цим аналізом для забезпечення комплексності додатку.

1.3. Формування актуальності і задач

На рисунку 1.3 зображено діаграму варіантів використання розроблюваного додатку (Use Case diagram). Дана діаграма показує функціональні можливості розроблюваного програмного додатку для оптимізації навчального процесу, а також взаємодію між типами користувачів та системою.

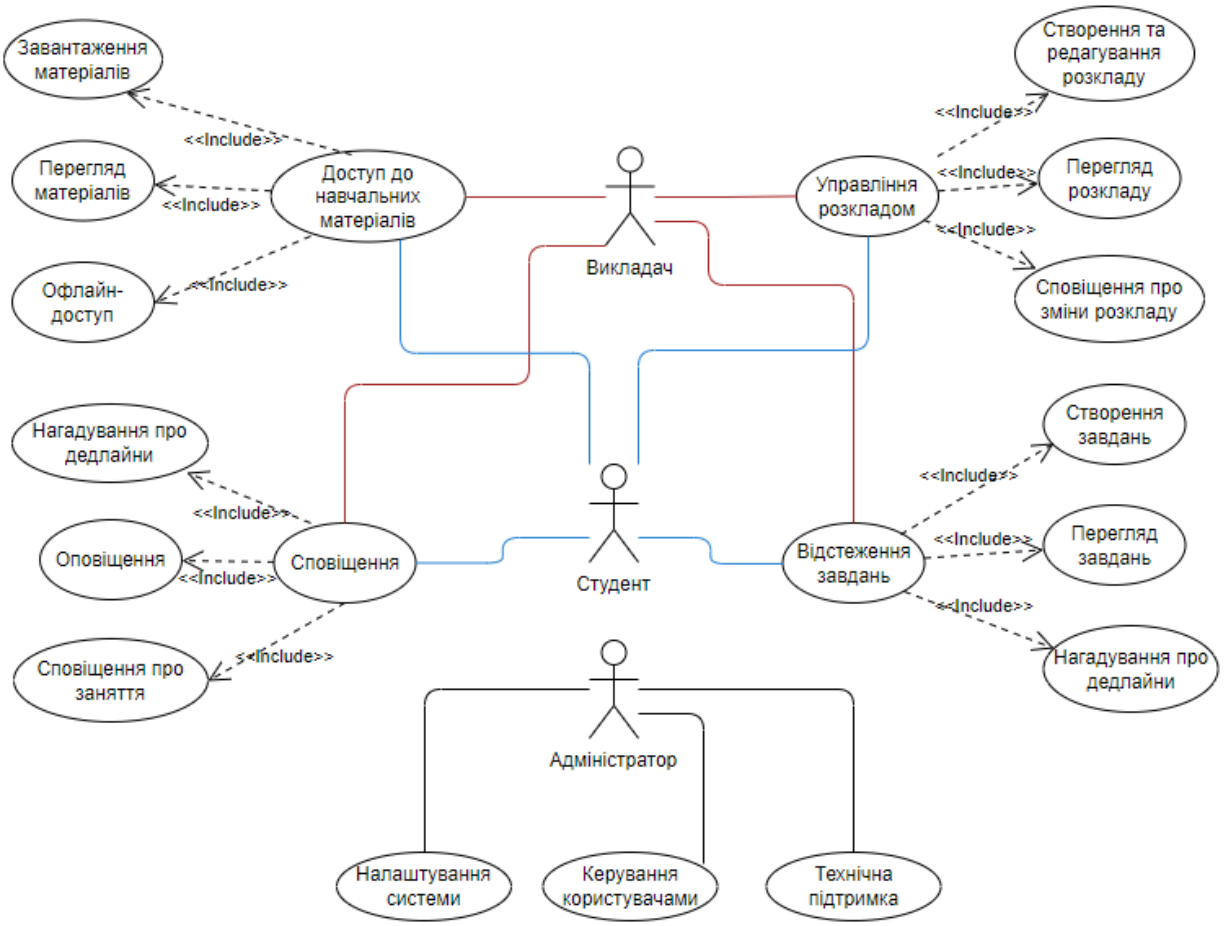

Рисунок Error: Reference source not found Діаграма варіантів використання (Use case diagram)

Для досягнення цілей кваліфікаційної роботи, тобто розробки додатку для оптимізації навчального процесу, визначимо функції які будуть реалізовані:

- Реєстрація та авторизація користувача (студента та викладача).
- Створення та редагування розкладу.
- Перегляд розкладу.
- Сповіщення про зміни в розкладі.
- Створення, редагування та перегляд завдань.
- Нагадування про дедлайни.
- Завантаження та перегляд навчальних матеріалів.
- Можливість офлайн-доступу до навчальних матеріалів.

Даний додаток буде розроблений у середі розробки Microsoft Visual Studio Code. Та буде розроблений такими мовами програмування: HTML5, CSS3, PHP, JavaScript, PHP та СУБД MySql.

1.4. **Розробка візуального інтерфейсу**

При написанні даної кваліфікаційної роботи було розроблено візуальний інтерфейс основних сторінок додатку на платформі Figma, результати розробки графічного інтерфейсу буде представлено далі у вигляді скріншотів.

На рисунку 1.4 зображено сторінку авторизації у додатку. На сторінці авторизації розташовані два поля для введення: логін і пароль, дві кнопки: «Увійти» та «Реєстрація». Якщо користувач (студент або викладач) зареєстрований у системі, вводимо логін (електронну адресу, яка була зазначена при реєстрації) та пароль, потім натискаємо кнопку «Увійти».

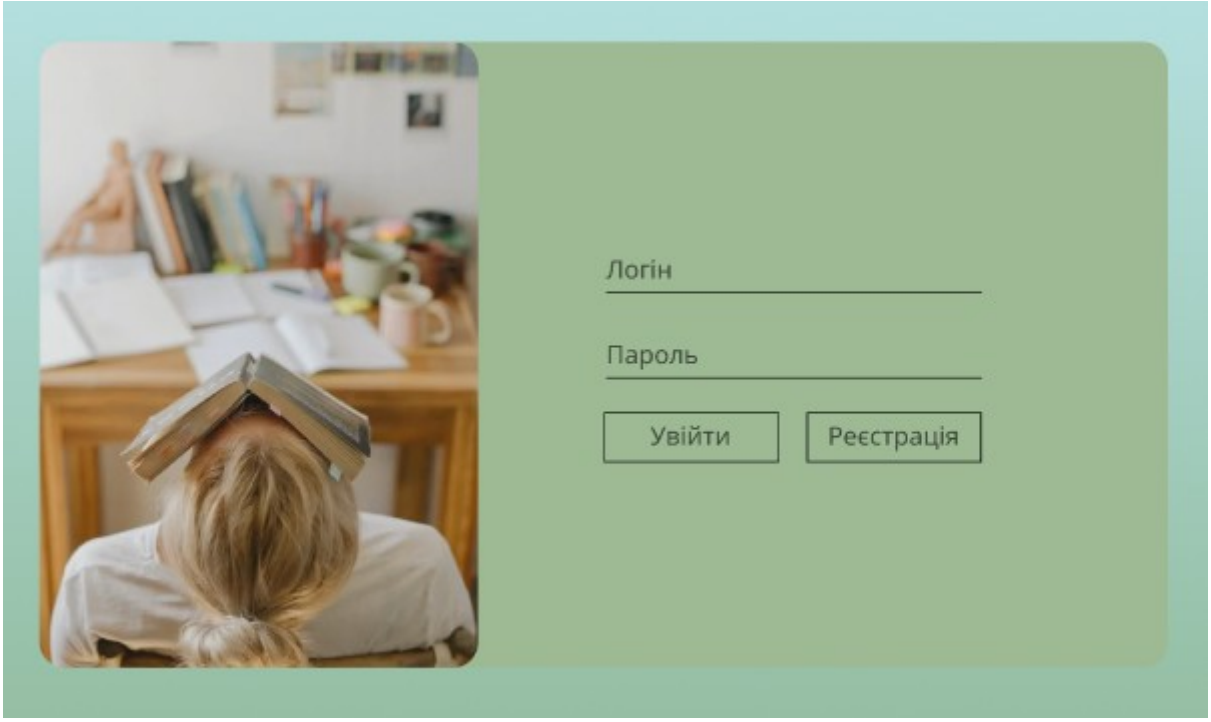

Рисунок Error: Reference source not found Макет форми авторизації

Якщо користувач не зареєстрований у системі, натискаємо кнопку «Реєстрація» та переходимо на сторінку реєстрації (Рисунок 1.5). На формі реєстрації розташовані 5 полів для введення: прізвище, ім’я, група, e-mail (електронна адреса), пароль. Також група радіо-кнопок для вибору ролі (студент або викладач), дві кнопки «Увійти» та «Реєстрація». Якщо користувач вже зареєстрований у системі, то натискаємо кнопку «Увійти» та переходимо на сторінку авторизації, якщо ні, то заповнюємо поля та натискаємо кнопку «Реєстрація».

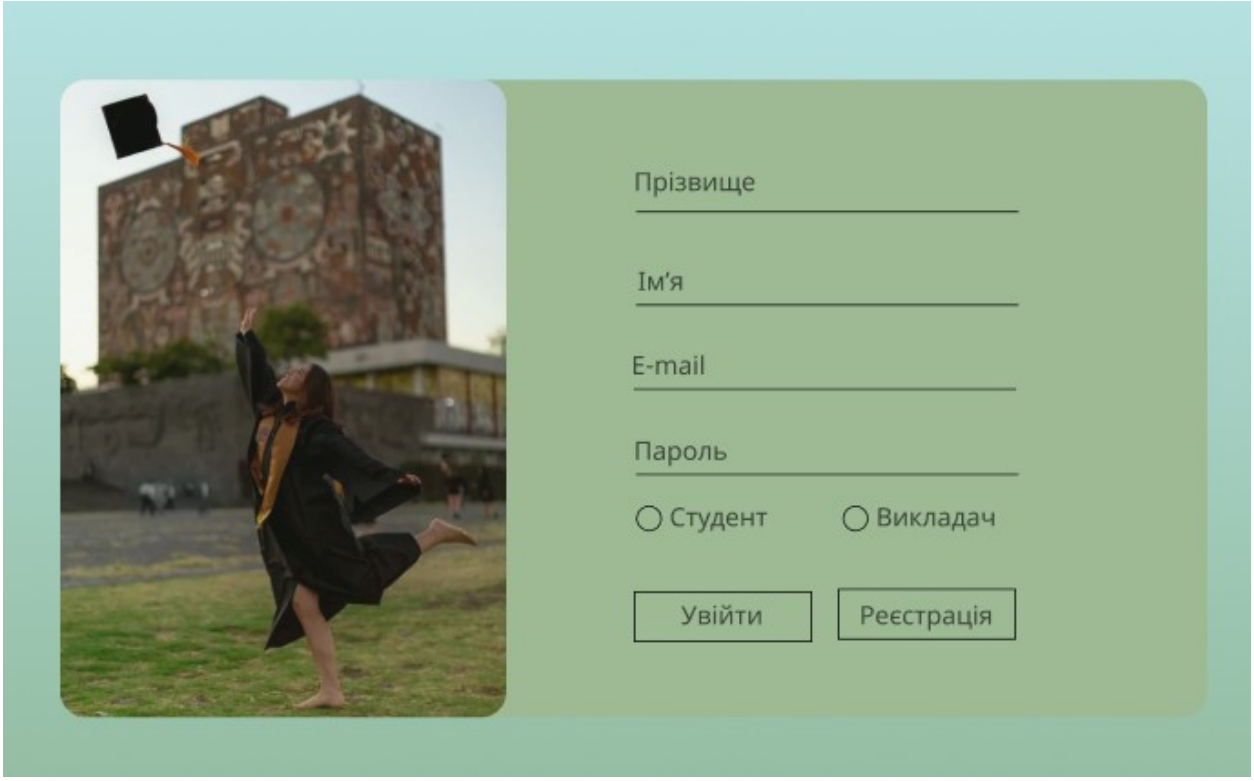

Рисунок 1.5 Макет форми реєстрації

Після реєстрації або авторизації користувача відриється сторінка «Профіль» (Рисунок 1.6). На сторінці профілю є можливість переглянути та змінити дані, які були внесені при реєстрації користувача.

Звернемо увагу на верхнє меню сторінки. У меню присутні такі розділи: «Розклад», «Завдання» та «Навчальні матеріали». Розглянемо детальніше кожен розділ.

На рисунку 1.7 зображено макет розділу «Розклад». На даній сторінці буде відображатися розклад студента, який увійшов до системи. Є можливість перегляду розкладу у двох варіантах: розклад на тиждень та місяць. У правому верхньому кутку є кнопки перемикання режиму відображення. Також можна вибрати за який період треба відобразити розклад.

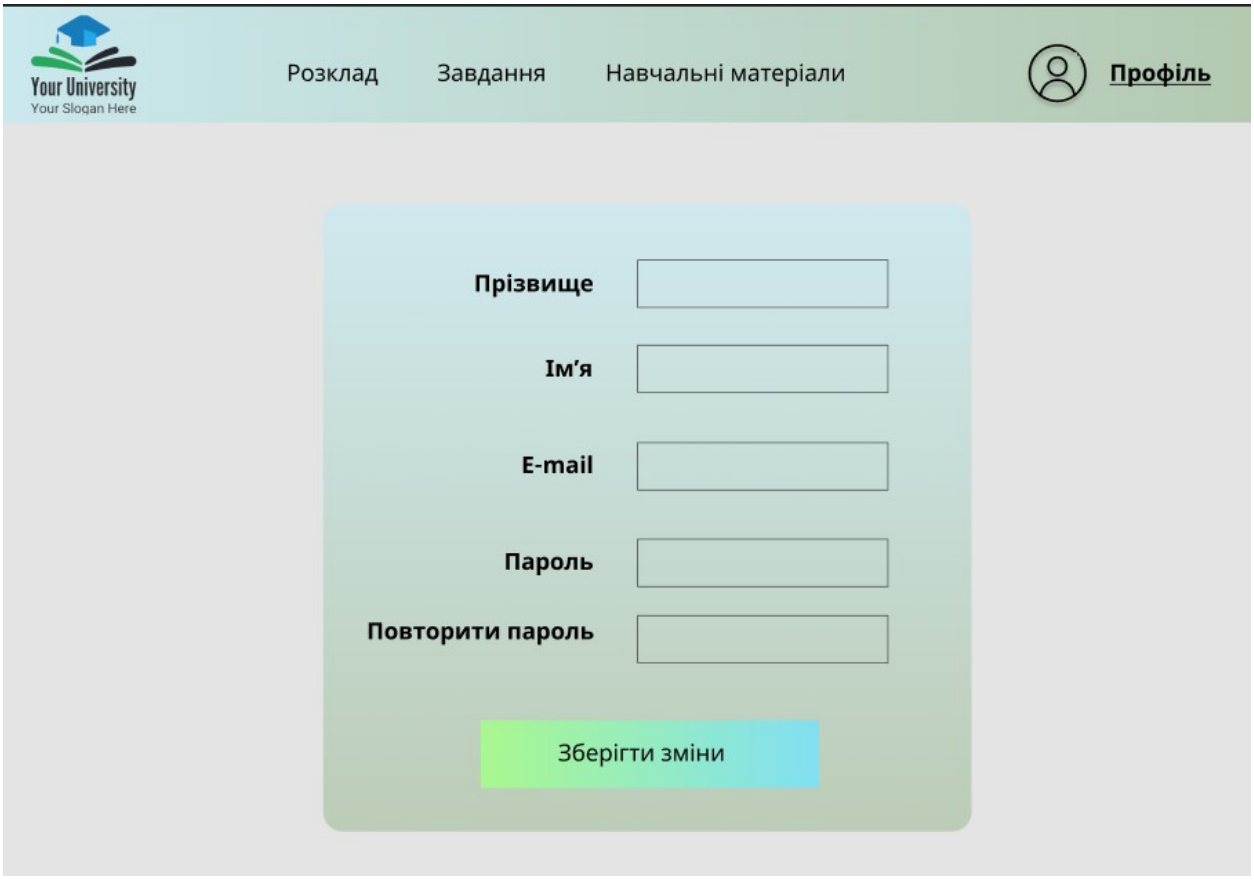

Рисунок 1.6 Макет сторінки «Профіль»

Розглянемо детальніше макет сторінки «Завдання» (Рисунок 1.8). На даній сторінці студент може переглянути всі заплановані завдання та додати нове завдання. Завдання можна помітити як виконане, видалити та редагувати. Дані які потрібно щоб створити завдання: назва завдання, предмет та дедлайн.

На рисунку 1.9 зображено макет сторінки «Навчальні матеріали». У розділі «Доступні матеріали» знаходять всі навчальні матеріали, які викладачі могли завантажити у систему. Також є можливість завантаження матеріалів з вибором місця збереження користувачем.

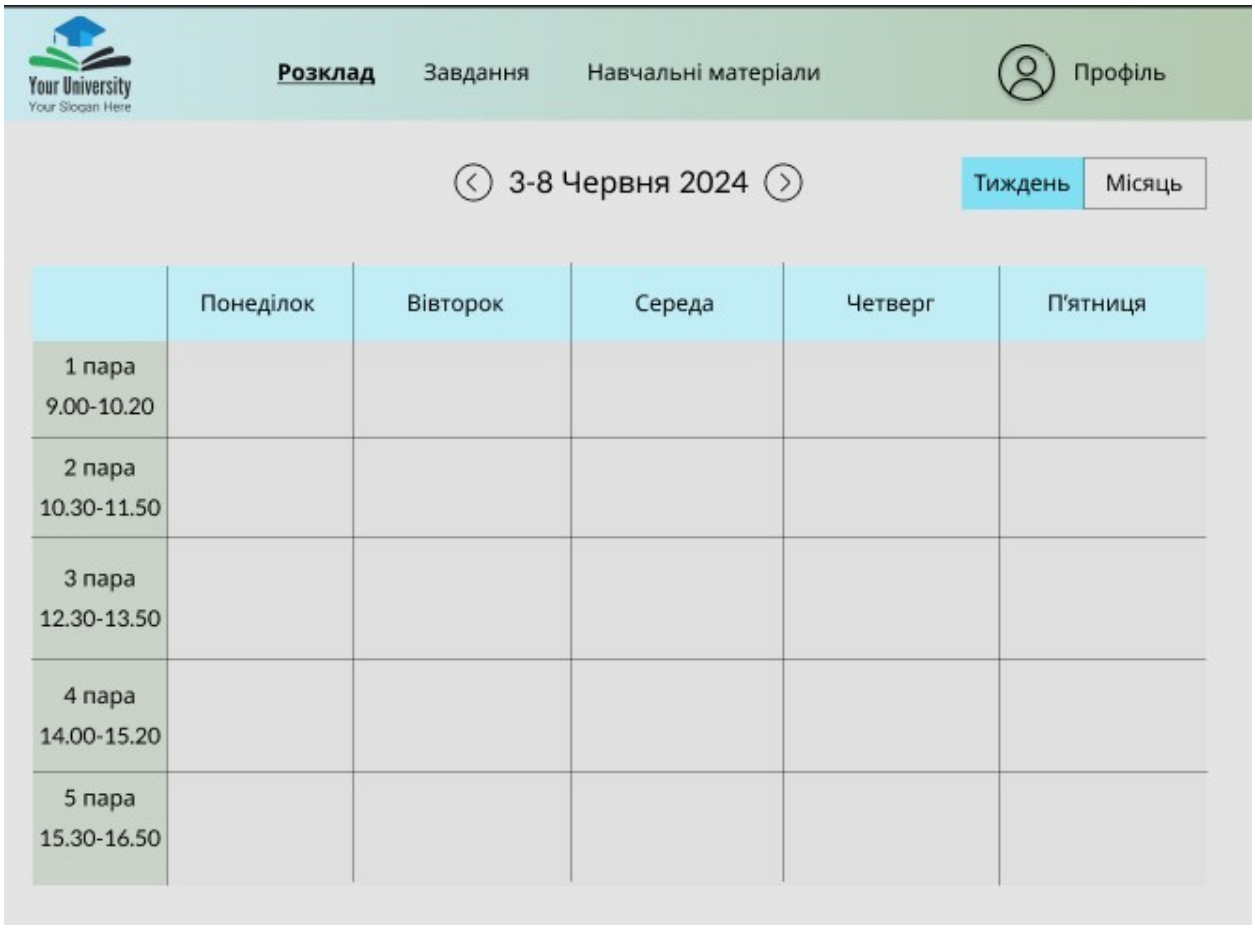

Рисунок 1.7 Макет сторінки «Розклад»

Рисунок 1.8 Макет сторінки «Завдання»

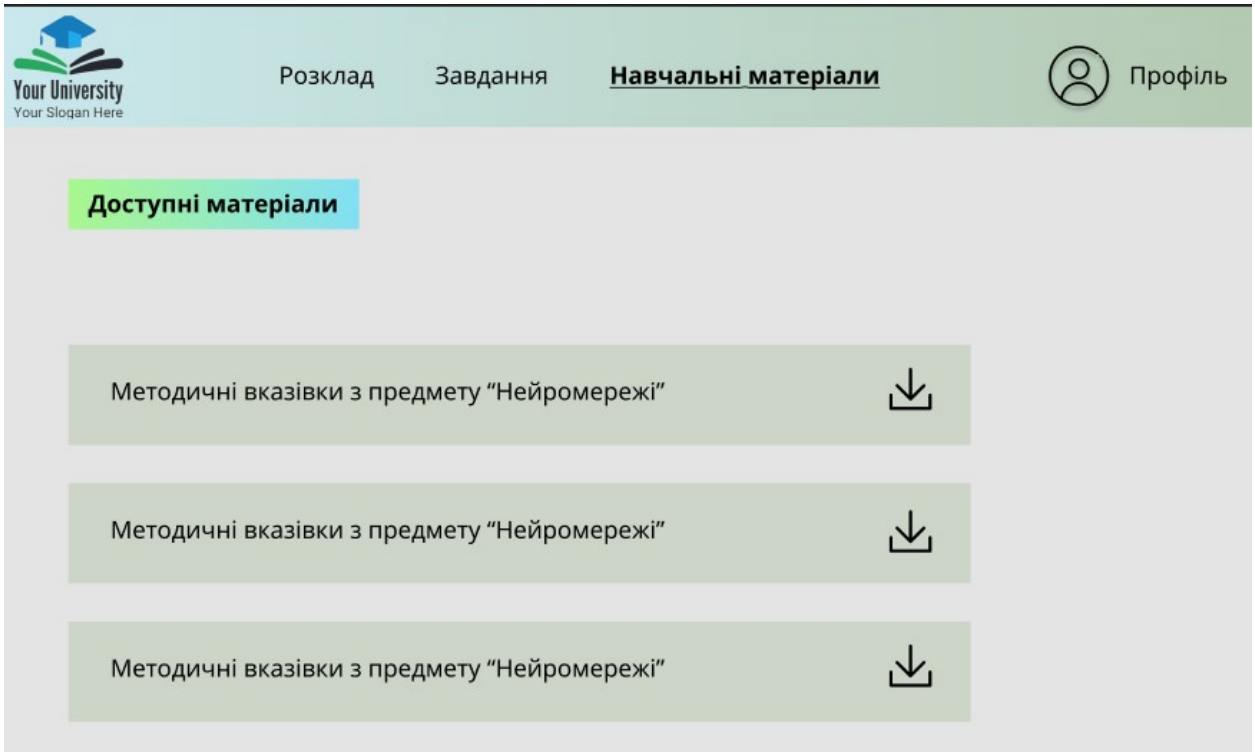

Рисунок 1.9 Макет сторінки «Навчальні матеріали»

РОЗДІЛ 2. ТЕОРЕТИЧНІ ОСНОВИ ВИРІШЕННЯ ЗАВДАНЬ РОБОТИ

1.1. Розробка структурних та функціональних схем

Проектування системи починається з розробки структурної схеми. Структурна схема програмного додатку для оптимізації навчального процесу візуалізує основні компоненти системи та взаємодію між ними. На рисунку 2.10 зображено структурну схему, яка відображає основні модулі та їх зв’язки.

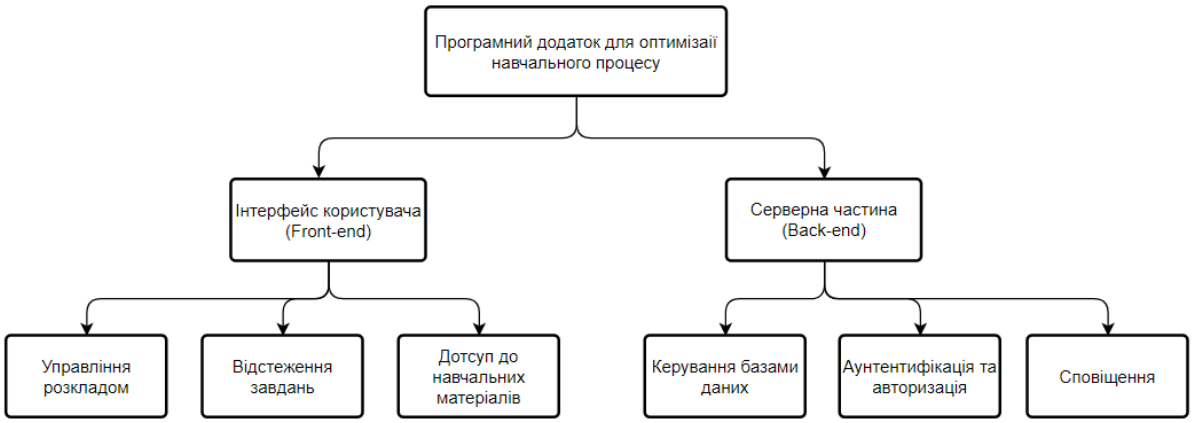

Рисунок 2.10 Структурна схема

Функціональна діаграма дозволяє візуалізувати основні процеси системи та їх взаємодію. У минулому розділі було визначено набір функцій які будуть реалізовані у програмному додатку. На рисунку 2.11 зображена контекстна функціональна діаграма, що відображає основну функцію розроблюваної системи та її взаємодію з зовнішніми елементами.

Звернемо увагу на стрілку що розташовується зверху блоку, яка означає управління системою. Політика безпеки має на увазі такі компоненти:

- Політика реєстрації.
- Політика доступу.

- Правила створення розкладу.
- Політика сповіщень.
- Правила створення завдань.
- Політика доступу до навчальних матеріалів.
- Політика доступу до офлайн-матеріалів.

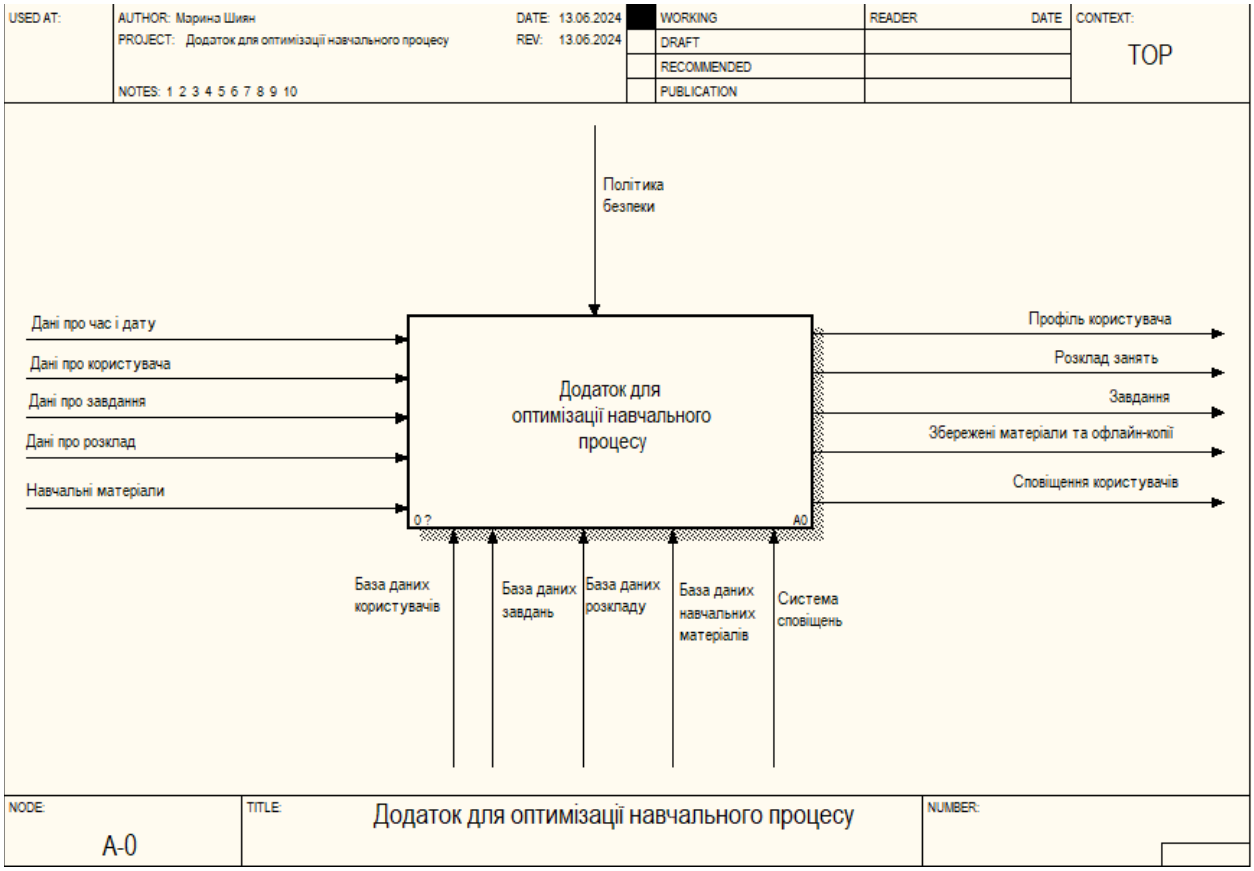

Рисунок 2.11 Контекстна функціональна діаграма

Декомпозуємо контекстну діаграму на більш детальну схему, яка покаже основні функції розроблюваного додатку для оптимізації навчального процесу (Рисунок 2.12). На схемі показано 5 основних блоків функцій системи які були виділені при створені архітектури додатку. Отже, основними блоками функцій системи є: авторизація та реєстрація користувачів, управління розкладом, управління завданнями, управління навчальними

матеріалами та управління сповіщеннями. Кожен з цих блоків можна декомпозувати на функції та зв’язок між ними.

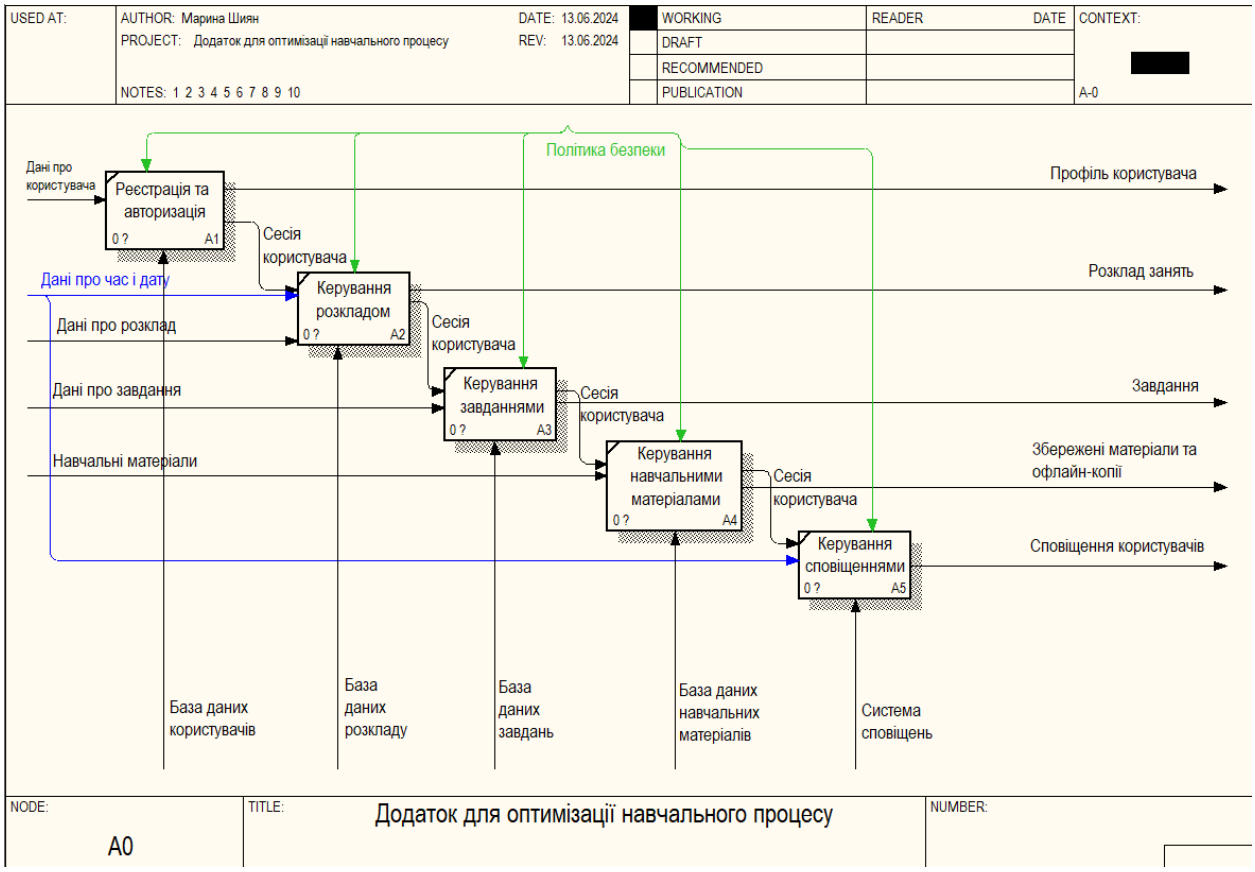

Рисунок 2.12 Декопомпозиція контекстної функціональної діаграми

1.2. Розробка блок-схем основних алгоритмів програмного комплексу

У підрозділі 2.2 ми визначили основні функції розроблюваного програмного комплексу, які були представлені у вигляді функціональної діаграми. Розглянемо кожну функцію більш детально у вигляді блок-схем. На рисунку 2.13 зображено блок-схему, що описує авторизацію та реєстрацію користувача.

Дана блок-схема реалізує логіку входу у систему користувачем. Для початку перевіряємо чи зареєстрований користувач у системі (блок №1), якщо «Ні» то переходимо у блок №2 «Реєстрація». Вводимо дані про користувача, дані автоматично заносяться у базу даних після реєстрації. Якщо

у блоці №1 відповідь «Так», то переходимо у блок №5 «Авторизація». Вводимо електрону адресу та пароль. Потім введені дані перевіряються, якщо дані були введенні правильно виконується вхід у систему. Незалежно чи була виконана реєстрація або авторизація, після успішного виконання одною з цих підпрограм, користувач потрапляє на домашню сторінку.

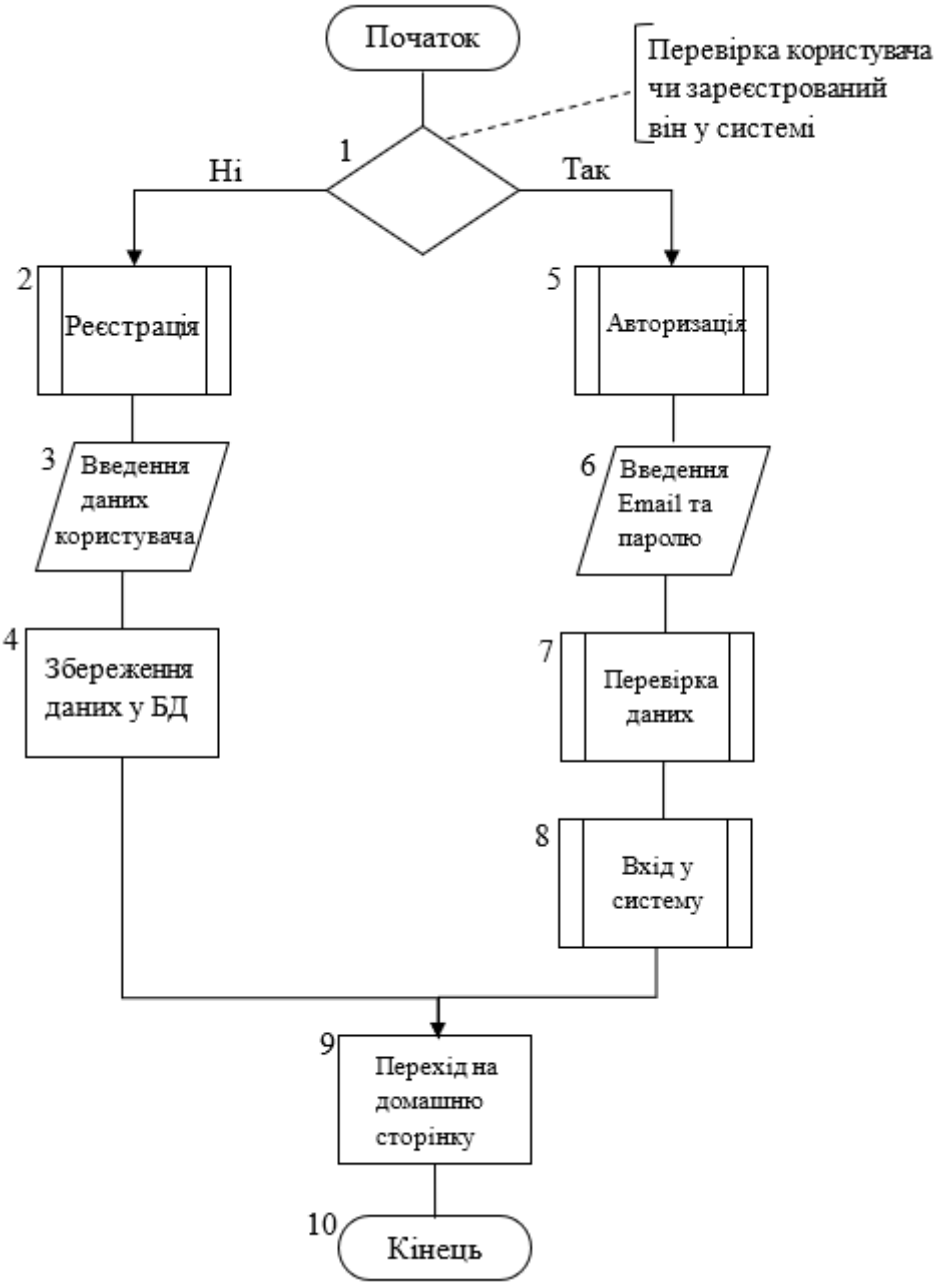

Рисунок 2.12 Блок-схема функції авторизації та реєстрації користувача

Розглянемо блок-схему функції «Керування розкладом» (рисунок 2.13). Ми визначили, що у системі є два типи користувачів: студенти та викладачі. Студенти мають доступ лише до перегляду розкладу, тоді ж як у викладачів є доступ до змінення розкладу.

У блоці № 1 перевіряємо право доступу до управління розкладом. Якщо це викладач, переходимо у блок №2 та вводимо дані про розклад. Дані автоматично зберігаються у БД. Якщо зміни були внесені, користувачам приходить сповіщення про зміни у розкладі (блок №4). Якщо у блоці №1 відповідь «Студент» то переходимо у блок №5, так як у студентів є лише право перегляду розкладу.

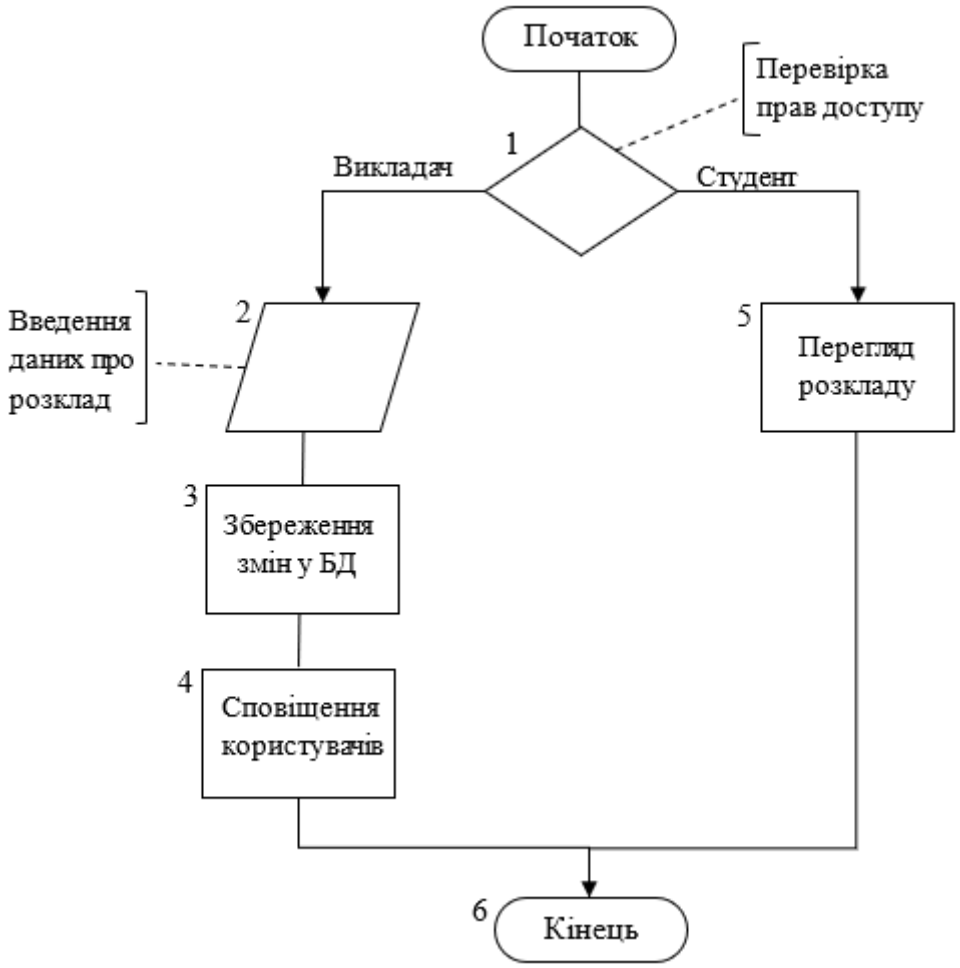

Рисунок 2.13 Блок-схема функції «Управління розкладом»

Розглянемо блок-схему функції «Керування сповіщеннями» на рисунку 2.14. Сповіщення відправляються у двох випадках, коли є зміни у розкладі або час дедлайну завдання закінчується. У блоці №1 перевіряємо чи були внесені зміни у розклад, якщо зміни були внесені переходимо до блоку №2 та формуємо сповіщення. Після його формування, відправляємо сповіщення користувачам (блок №3). Якщо у блоці №1 відповідь «Немає змін», переходимо до блоку №4. У цьому блоці перевіряємо дедлайни завдань. Якщо час дедлайнів підходить до кінця, переходимо у блок №5 та формуємо сповіщення. Після формування сповіщення про закінчення дедлайну завдання, відправляє сформоване сповіщення користувачам (блок №6).

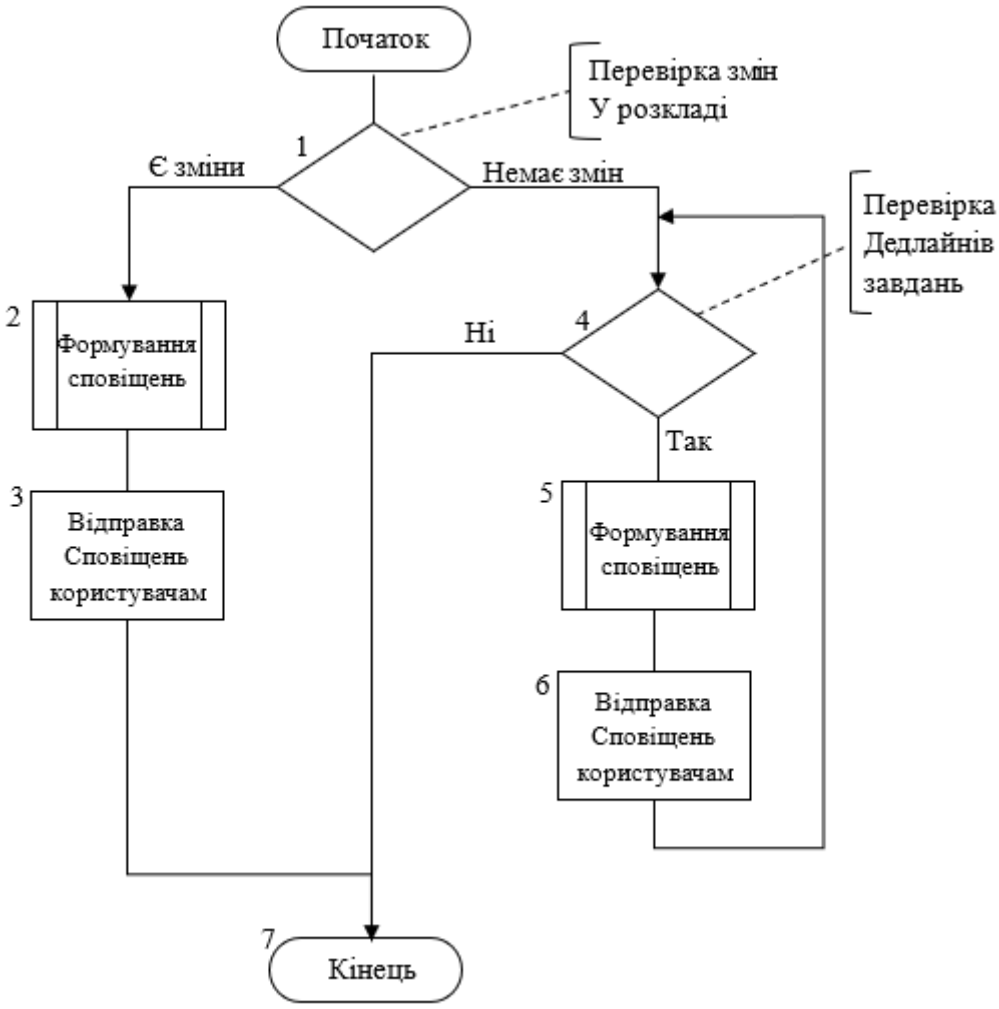

Рисунок 2.14 Блок-схема функції «Керування сповіщеннями»

На рисунку 2.15 зображено блок-схему функції «Управління завданнями». У розроблювані системі, що у студентів, що у викладачів є можливість створювати завдання. Створенні завдання будуть доступні для перегляду та редагуванню лише користувачу, який створив це завдання. Завдання може мати будь-який зміст та можливість встановлення дедлайну. Редагування створеного завдання має аналогічні кроки. У блоці №1 вводимо дані про завдання, введенні дані зберігаються у базу даних (блок №2). У блоці №3, формується сповіщення для користувачів про завдання.

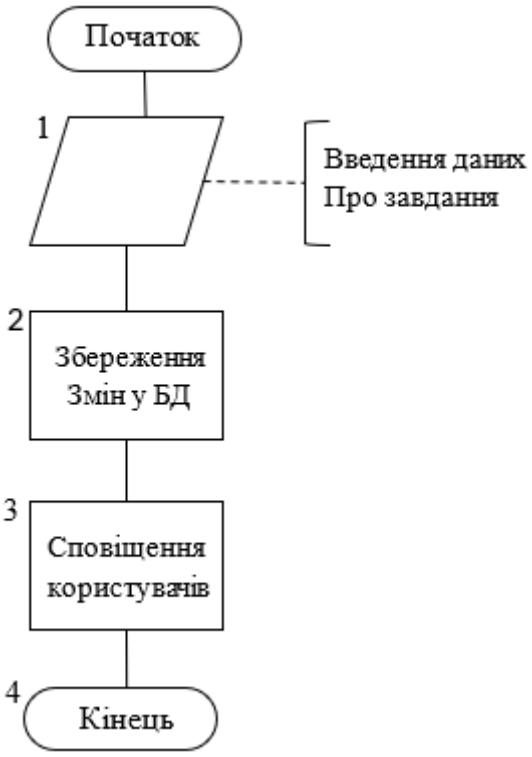

Рисунок 2.15 Блок-схема функції «Керування завданнями»

Останньою основною функцією у системі є функція «Керування навчальними матеріалами». У додатку є два рівні доступу: студенти та викладачі. У викладачів є доступ до завантаження навчальних матеріалів у систему, студенти мають можливість переглядати або завантажити матеріали для офлайн-доступу до них. На рисунку 2.16 зображено блок-схему яка представляє логіку роботи даної функції.

Рисунок 2.16 Блок-схема функції «Керування навчальними матеріалами»

У блоці №1 перевіряємо права доступу до навчальних матеріалів. Якщо у цьому блоці відповідь «Викладач» переходимо у блок №2 та вводимо дані про навчальний матеріал. Після переходимо у блок №3 та завантажуємо файли. Після введення даних та завантаження файлів, у блоці №4 ці дані зберігаються у базі даних. Якщо у блоці №1 відповідь «Студент», пе6рехдимо у блок №5. Відображається список матеріалів які доступні для студента. Студент може вибрати матеріал який його цікавить (блок №6). Після вибору матеріалу є можливість відобразити матеріал у онлайн доступі, або завантажити матеріал для можливості доступу офлайн (блок №7).

1.3. Розробка структур таблиць бази даних та вибір їх типів

Розроблювана система передбачає наявність бази даних, яка складається з 4 таблиць: користувачі, завдання, навчальні матеріали та розклад.

Таблиця «Users» (користувачі) зберігає інформацію про користувачів системи. У таблиці 2.1 представлено інформацію про поля таблиці, тип даних полів та опис.

Таблиця 2.1

Поле	Тип даних	Опис
User_id	INT (ціле число)	Унікальний ідентифікатор користувача
email	VARCHAR (короткий текст)	Електронна пошті користувача
password	VARCHAR (короткий текст)	Пароль користувача
First_name	VARCHAR (короткий текст)	Ім’я користувача
Last_name	VARCHAR (короткий текст)	Прізвище користувача
role	ENUM (перелік)	Роль користувача (студент/ викладач)

Таблиця «Schedules» (розклад) зберігає інформацію про розклад. У таблиці 2.2 представлено інформацію про поля таблиці, тип даних полів та опис.

Таблиця 2.2

Поле	Тип даних	Опис
Schedules_id	INT (ціле число)	Унікальний ідентифікатор розкладу
User_id	INT (ціле число)	Ідентифікатор користувача
Teacher_id	INT (ціле число)	Ідентифікатор викладача
date	DATE (дата)	Дата заняття
Start_time	TIME (час)	Час початку заняття
End_time	TIME (час)	Час закінчення заняття
subject	VARCHAR (короткий текст)	Назва предмету
teachers	VARCHAR (короткий текст)	Викладач заняття

Таблиця «Assignments» (завдання) зберігає інформацію про завдання. У таблиці 2.3 представлено інформацію про поля таблиці, тип даних полів та опис.

Поле	**Тип даних**	**Опис**
Assignment_id	INT (ціле число)	Унікальний ідентифікатор завдання
User_id	INT (ціле число)	Ідентифікатор користувача
Teacher_id	INT (ціле число)	Ідентифікатор викладача
title	VARCHAR (короткий текст)	Назва завдання
subject	VARCHAR (короткий текст)	Назва предмету
Due_date	DATE (дата)	Дата дедлайну
status	ENUM (перелік)	Статус завдання (виконано, не виконано)

Таблиця «Materials» (навчальні матеріали) зберігає інформацію про навчальні матеріали. У таблиці 2.4 представлено інформацію про поля таблиці, тип даних полів та опис.

Таблиця 2.4

Поле	**Тип даних**	**Опис**
Material_id	INT (ціле число)	Унікальний ідентифікатор матеріалу
title	VARCHAR (короткий текст)	Назва навчального матеріалу
File_path	VARCHAR (короткий текст)	Шлях до файлу

Розглянемо відношення між таблицями створеної бази даних.

- «Користувачі» та «Розклади»: один-до-багатьох. Один користувач може мати багато записів у розкладі.
- «Користувачі» та «Завдання»: один-до-багатьох. Один користувач може мати багато завдань.
- «Користувачі» та «Матеріали»: один-до-багатьох. Один користувач може завантажувати багато матеріалів.

1.4. Стратегія тестування та тестові дані

Стратегія тестування – план або набір правил, які використовуються для визначення того, як тестування програмного продукту буде виконуватися. Вона включає в себе визначення того, як тести будуть виконанні, в якому порядку вони будуть виконанні, як будуть збиратися тестові дані та як будуть оцінюватися результати тестування.

Тестування розроблюваного додатку буде зроблено за допомогою стратегії функціонального тестування. Дана стратегія тестування являє собою процес перевірки того, чи працює програмний продукт відповідно до вимог та очікувань користувачів.

Функції будуть перевірятися за допомогою тест-кейсів. Визначимо які тест-кейси будуть перевірятися:

- Реєстрація користувача з валідними даними та невалідними даними.
- Авторизація користувача з валідними даними та невалідними даними.
- Створення та редагування розкладу (створення нового розкладу, редагування існуючого розкладу, видалення запису з розкладу).
- Перегляд розкладу (перегляд розкладу користувача, перегляд розкладу на тиждень/місяць, перегляд деталей заняття).
- Сповіщення про зміни в розкладі (отримання сповіщень про зміни в розкладі, налаштування отримання сповіщень).
- Створення, редагування та перегляд завдань (створення нового завдання, редагування існуючого завдання, видалення завдання, перегляд списку завдань, зміна статусу завдання).
- Нагадування про дедлайни (отримання сповіщень про дедлайни).
- Завантаження та перегляд навчальних матеріалів (завантаження нового навчального матеріалу, перегляд списку доступних матеріалів, завантаження матеріалів з додатку, видалення матеріалів).

- Доступ до завантажених матеріалів в офлайн-режимі.

Для кожної з цих функцій будуть створені тест-кейси, які включатимуть наступні дані:

- Назва: унікальний ідентифікатор тест-кейсу.
- Опис: опис тест-кейсу та те, що потрібно перевірити.
- Попередні умови: стан веб-сайту, який має бути досягнений перед початком тестування.
- Кроки: послідовність дій, які потрібно виконати для перевірки функції.
- Очікуваний результат: опис того, що має статися після виконання кроків.
- Фактичний результат: результат виконання тесту.

Дані тест-кейси допоможуть перевірити коректність роботи кожної функції і виявити можливі помилки або недоліки в реалізації розроблюваної системи.

РОЗДІЛ 3. РОЗРОБКА ПРОГРАМНИХ МОДУЛІВ ПРОЕКТУ

1.1. Розробка основних форм програмних модулів

У даній кваліфікаційній роботі додаток реалізується у вигляді веб-сайту, який являє собою окремі сторінки пов’язані посиланнями. Додаток має два рівні взаємодії: перший – студентський рівень, другий – рівень викладача. На сайті є форми авторизації та реєстрації користувача. Поділ між рівнями доступу відбувається при реєстрації користувача.

Форма реєстрації користувача має наступні поля:

- Прізвище.
- Ім’я.
- Електронна адреса.
- Пароль.
- Вибір рівню користувача: студент або викладач.
- Кнопка «Реєстрація».
- Кнопка «Увійти» (для того щоб перейти на сторінку авторизації, якщо користувач вже зареєстрований у системі).

При натисканні на кнопку «Реєстрація», дана форма оброблюється php скриптом на серверному рівні додатку. Даний скрипт записує дані, які ввів користувач у поля на формі, у таблицю «Users» у базі даних. Для захисту пароля, скрипт здійснює шифрування пароля, за допомогою функції password_hash() та у зашифрованому вигляді записує його у базу даних.

Лістинг 3.1 – Скрипт реєстрації користувача

```
<?php
header('Content-Type: application/json');

$servername = "127.127.126.25";
$username = "root";
$password = "";
$dbname = "DB";
$port = 3306;

$conn = new mysqli($servername, $username, $password, $dbname,
$port);
```

```
if ($conn->connect_error) {
    echo json_encode(["status" => "error", "message" => "Помилка
з'єднання: " . $conn->connect_error]);
    exit();
}

$last_name = $_POST['last_name'];
$first_name = $_POST['first_name'];
$email = $_POST['email'];
$password = password_hash($_POST['password'], PASSWORD_DEFAULT);
$role = $_POST['role'];

$checkEmailSql = "SELECT * FROM Users WHERE email = ?";
$checkEmailStmt = $conn->prepare($checkEmailSql);
$checkEmailStmt->bind_param("s", $email);
$checkEmailStmt->execute();
$checkEmailResult = $checkEmailStmt->get_result();

if ($checkEmailResult->num_rows > 0) {
    echo json_encode(["status" => "error", "message" => "Цей
email вже зареєстрований! Увійдіть в акаунт."]);
    $checkEmailStmt->close();
    $conn->close();
    exit();
}
$checkEmailStmt->close();

$sql = "INSERT INTO Users (last_name, first_name, email,
password, role) VALUES (?, ?, ?, ?, ?)";
$stmt = $conn->prepare($sql);
$stmt->bind_param("sssss", $last_name, $first_name, $email,
$password, $role);

if ($stmt->execute()) {
    echo json_encode(["status" => "success", "message" =>
"Реєстрація успішна! Ввійдіть у систему."]);
} else {
    echo json_encode(["status" => "error", "message" =>
"Помилка: " . $stmt->error]);
}

$stmt->close();
$conn->close();
?>
```

Розглянемо форму авторизації користувача. Форма має наступні поля:

- Логін (електронна адреса).
- Пароль.

- Кнопка «Увійти».
- Кнопка «Реєстрація» (для переходу до форми реєстрації, якщо користувач ще не зареєстрований у системі).

Форма обробляється php скриптом на серверному рівні додатку. Так як пароль у базі даних у зашифрованому вигляді, щоб перевірити правильність введеного паролю використовується функція password_verify(). Авторизація користувача здійснюється за електронною адресою що була використана при реєстрації.

Лістинг 3.2 – php скрипт форми авторизації

```
<?php
session_start();
header('Content-Type: application/json');

$servername = "127.127.126.25";
$username = "root";
$password = "";
$dbname = "DB";
$port = 3306;

$conn = new mysqli($servername, $username, $password, $dbname,
$port);

if ($conn->connect_error) {
    echo json_encode(["status" => "error", "message" => "Помилка
з'єднання: " . $conn->connect_error]);
    exit();
}

$email = $_POST['email'];
$password = $_POST['password'];

$sql = "SELECT * FROM Users WHERE email = ?";
$stmt = $conn->prepare($sql);
$stmt->bind_param("s", $email);
$stmt->execute();
$result = $stmt->get_result();

if ($result->num_rows > 0) {
    $user = $result->fetch_assoc();
    if (password_verify($password, $user['password'])) {
        $_SESSION['user_email'] = $email;
        $_SESSION['user_id'] = $user['user_id'];

        if ($user['role'] === 'teacher') {
```

```
            echo json_encode(["status" => "success", "message"
=> "Успішний вхід!", "redirect" =>
"../pages/profile_teachers.html"]);
        } else {
            echo json_encode(["status" => "success", "message"
=> "Успішний вхід!", "redirect" => "../pages/profile.html"]);
        }
    } else {
        echo json_encode(["status" => "error", "message" =>
"Неправильний пароль."]);
    }
} else {
    echo json_encode(["status" => "error", "message" =>
"Користувача з таким email не знайдено."]);
}

$stmt->close();
$conn->close();
?>
```

На сторінці «Профіль» після успішної авторизації користувача, розташована форма з даними користувача, який увійшов у систему. Форма має наступні поля:

- Прізвище.
- Ім’я.
- Електронна адреса.
- Пароль.
- Повторити пароль.
- Кнопка «Зберегти зміни».

Поля прізвище, ім’я та електронна адреса заповнюються автоматично. Php скрипт автоматично оброблює запит та заповняє поля відповідною інформаціє з бази даних при завантажені сторінки.

Лістинг 3.3 – завантаження даних користувача

```
<?php
session_start();
header('Content-Type: application/json');

if (!isset($_SESSION['user_email'])) {
    echo json_encode(["status" => "error", "message" =>
"Користувач не авторизований."]);
    exit();
```

```
}

$servername = "127.127.126.25";
$username = "root";
$password = "";
$dbname = "DB";
$port = 3306;

$conn = new mysqli($servername, $username, $password, $dbname,
$port);

if ($conn->connect_error) {
    echo json_encode(["status" => "error", "message" => "Помилка
з'єднання: " . $conn->connect_error]);
    exit();
}

$email = $_SESSION['user_email'];

$sql = "SELECT first_name, last_name, email FROM Users WHERE
email = ?";
$stmt = $conn->prepare($sql);
$stmt->bind_param("s", $email);

if ($stmt->execute()) {
    $result = $stmt->get_result();
    if ($result->num_rows > 0) {
        $user_data = $result->fetch_assoc();
        echo json_encode(["status" => "success", "data" =>
$user_data]);
    } else {
        echo json_encode(["status" => "error", "message" =>
"Користувача не знайдено."]);
    }
} else {
    echo json_encode(["status" => "error", "message" =>
"Помилка: " . $stmt->error]);
}

$stmt->close();
$conn->close();
?>
```

На даній сторінці є можливість редагування інформації про користувача. Для того щоб редагувати дані, потрібно заповнити поля новою інформацією та натиснути кнопку «Зберегти зміни». При натисканні на кнопку php скрипт, створює запит на оновлення інформації у базі даних. Зверніть увагу, що поле електронна адреса не змінюється.

Лістинг 3.4 – php скрипт редагування інформації про користувача

```
<?php
session_start();
header('Content-Type: application/json');

$servername = "127.127.126.25";
$username = "root";
$password = "";
$dbname = "DB";
$port = 3306;

$conn = new mysqli($servername, $username, $password, $dbname,
$port);

if ($conn->connect_error) {
    echo json_encode(["status" => "error", "message" => "Помилка
з'єднання: " . $conn->connect_error]);
    exit();
}

$last_name = $_POST['last_name'];
$first_name = $_POST['first_name'];
$new_password = $_POST['new_password'];
$email = $_SESSION['user_email'];

if (!empty($new_password)) {
    $new_password_hash = password_hash($new_password,
PASSWORD_DEFAULT);
    $update_password_sql = ", password = '$new_password_hash'";
} else {
    $update_password_sql = "";
}

$sql = "UPDATE Users SET last_name = ?, first_name = ?" .
$update_password_sql . " WHERE email = ?";
$stmt = $conn->prepare($sql);
$stmt->bind_param("sss", $last_name, $first_name, $email);

if ($stmt->execute()) {
    echo json_encode(["status" => "success", "message" => "Дані
успішно змінені!"]);
} else {
    echo json_encode(["status" => "error", "message" =>
"Помилка: " . $stmt->error]);
}

$stmt->close();
$conn->close();
?>
```

Розглянемо детальніше сторінку «Навчальні матеріали». На даній сторінці при завантажені сторінки, відображаються всі доступні навчальні матеріали. Відображення даних на клієнтській стороні забезпечує скрипт який написаний на мові JavaScript (лістинг 3.5), на серверній – php скрипт (лістинг 3.6). JavaScrpipt викликає php скрипт, який повертає список матеріалів. Якщо список який повертає скрипт не порожній, автоматично створюється блок який містить такі поля: назва навчального матеріалу, кнопка «Завантажити матеріал» у вигляді відповідної іконки.

Якщо користувач, який увійшов у систему є викладачем, на сторінці відображається форма додавання нового матеріалу, з такими полями: назва матеріалу, кнопка для завантаження файлу та кнопка «Додати матеріал». У блоці для відображення вже існуючого матеріалу додається кнопка для виділення матеріалу у вигляді відповідної іконки. Форму додавання матеріалу обробляє php скрипт на серверній частині додатку (лістинг 3.7).

Лістинг 3.5 – скрипт відображення матеріалів на сторінці

```
<script>
        document.addEventListener('DOMContentLoaded', function
() {
            const materialsSection =
document.querySelector('.assigments__section');

            fetch('../php/get_material.php')
                .then(response => response.json())
                .then(data => {
                    data.forEach(material => {
                        const materialElement =
createMaterialElement(material);

materialsSection.appendChild(materialElement);
                    });
                })
                .catch(error => console.error('Error:', error));

            materialsSection.addEventListener('click', function
(event) {
                const target = event.target;
                const materialElement =
target.closest('.assigment__element');
```

```
                if (target.classList.contains('download-icon'))
{
                    const materialId =
materialElement.dataset.id;
                    const fileUrl =
`../php/download_material.php?id=${materialId}`;
                    window.open(fileUrl, '_blank');
                }
            });
        });

        function createMaterialElement(material) {
            const materialElement =
document.createElement('div');
            materialElement.classList.add('assigment__element');
            materialElement.dataset.id = material.id;

            const titleElement = document.createElement('div');
            titleElement.classList.add('asssigment__names');
            titleElement.textContent = material.title;
            materialElement.appendChild(titleElement);


            const downloadIcon = document.createElement('img');
            downloadIcon.src = '../images/download.svg';
            downloadIcon.alt = 'Download';
            downloadIcon.classList.add('material__icon');
            downloadIcon.classList.add('download-icon');
            materialElement.appendChild(downloadIcon);

            return materialElement;
        }

    </script>
```

Лістинг 3.6 – скрипт завантаження даних про матеріли з бази даних

```
<?php

$servername = "127.127.126.25";
$username = "root";
$password = "";
$dbname = "DB";

$conn = new mysqli($servername, $username, $password, $dbname);

if ($conn->connect_error) {
    die("Помилка під'єднання до бази даних: " . $conn-
>connect_error);
}
```

```
$sql = "SELECT material_id, title FROM materials";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    $materials = array();

    while($row = $result->fetch_assoc()) {
        $material = array(
            "id" => $row["material_id"],
            "title" => $row["title"]
        );
        array_push($materials, $material);
    }
    header('Content-Type: application/json');
    echo json_encode($materials);
} else {
    echo "Немає доступних матеріалів";
}

$conn->close();
?>
```

Лістинг 3.7 – php скрипт для додавання нового матеріалу

```
<?php
session_start();

if ($_SERVER['REQUEST_METHOD'] === 'POST' &&
isset($_FILES['file'])) {
    $title = $_POST['title'];
    $file_name = $_FILES['file']['name'];
    $file_tmp = $_FILES['file']['tmp_name'];
    $file_type = $_FILES['file']['type'];
    $file_size = $_FILES['file']['size'];

    if ($file_tmp !== '') {
        $file_path = '../material/' . $file_name;
        move_uploaded_file($file_tmp, $file_path);

        $servername = "127.127.126.25";
        $username = "root";
        $password = "";
        $dbname = "DB";

        $conn = new mysqli($servername, $username, $password,
$dbname);

        if ($conn->connect_error) {
            die("Помилка під'єднання до бази даних: " . $conn-
>connect_error);
        }
```

```
        $sql = "INSERT INTO Materials (title, file_path) VALUES
('$title', '$file_path')";

        if ($conn->query($sql) === TRUE) {
            echo "Матеріал успішно завантажений та збережений у
базі даних.";
        } else {
            echo "Помилка: " . $sql . "<br>" . $conn->error;
        }

        $conn->close();
    } else {
        echo "Помилка при завантажені файлу.";
    }
} else {
    echo "Файл не було надіслано.";
}
?>
```

Розглянемо сторінку «Завдання». При завантажені сторінки, завдання які є у користувача який увійшов у систему автоматично завантажуються та відображаються у вигляді блоку, яка містить таку інформацію:

- Назва завдання.
- Предмет.
- Дата дедлайну.
- Кнопка видалення, у вигляді відповідної іконки.
- Кнопка редагування, у вигляді відповідної іконки.
- Кнопка типу checkbox, яка визначає статус завдання. Якщо кнопка активна, то завдання позначено як виповнене, якщо ні – невиповнене.

Також на сторінці є кнопка «Додати завдання». При натисканні на дану кнопку відображається форма з такими полями:

- Назва.
- Предмет.
- Дата дедлайну.
- Email викладача або студента (у залежності, користувач з якою роллю авторизований у системі).
- Кнопка «Зберегти».

На клієнтському рівні відображення блоків з завданнями забезпечуються скриптом на мові JavaScript (лістинг 3.8), на серверному рівні – php скриптом (лістинг 3.9). Додавання завдання оброблюється php скриптом на серверному рівні додатку (лістинг 3.10).

Лістинг 3.8 – скрипт відображення даних завдання

```
<script>
        document.addEventListener('DOMContentLoaded', function
() {
            fetch('../php/get_user_assignments.php', {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/x-www-form-
urlencoded'
                }
            })
                .then(response => response.json())
                .then(data => {
                    if (data.status === 'success') {
                        const assignmentsSection =
document.querySelector('.assigments__section');
                        data.assignments.forEach(assignment => {
                            const assignmentElement =
document.createElement('div');

assignmentElement.classList.add('assigment__element');

assignmentElement.setAttribute('data-id',
assignment.assignments_id);

                            const checkbox =
document.createElement('input');
                            checkbox.type = 'checkbox';
                            checkbox.checked = assignment.status
=== 'Виконано';
                            checkbox.setAttribute('data-id',
assignment.assignments_id);

assignmentElement.appendChild(checkbox);

                            const assigmentNames =
document.createElement('div');

assigmentNames.classList.add('asssigment__names');
```

```
                            const mainName =
document.createElement('span');

mainName.classList.add('main__name');
                            mainName.textContent =
assignment.title;
                            const decriptionSpan =
document.createElement('span');

decriptionSpan.classList.add('decription__span-assigment');
                            decriptionSpan.textContent =
assignment.subject;

assigmentNames.appendChild(mainName);

assigmentNames.appendChild(decriptionSpan);

assignmentElement.appendChild(assigmentNames);

                            const dateAssigment =
document.createElement('span');

dateAssigment.classList.add('date__assigment');
                            dateAssigment.textContent =
assignment.due_date;

assignmentElement.appendChild(dateAssigment);

                            const deleteIcon =
document.createElement('img');
                            deleteIcon.src =
'../images/backet.svg';
                            deleteIcon.alt = 'Delete';
                            deleteIcon.classList.add('assigment-
img');
                            deleteIcon.addEventListener('click',
function () {
                                const assignmentId =
assignment.assignments_id;

fetch('../php/delete_assignment.php', {
                                    method: 'POST',
                                    headers: {
                                        'Content-Type':
'application/json'
                                    },
                                    body:
JSON.stringify({ assignmentId: assignmentId })
                                })
                                    .then(response =>
response.json())
```

```
                                    .then(data => {
                                        if (data.status ===
'success') {

assignmentElement.remove();
                                        } else {

console.error('Error:', data.message);
                                        }
                                    })
                                    .catch(error =>
console.error('Error:', error));
                            });

assignmentElement.appendChild(deleteIcon);

                            const editIcon =
document.createElement('img');
                            editIcon.src =
'../images/pencil.svg';
                            editIcon.alt = 'Edit';
                            editIcon.classList.add('assigment-
img');
                            editIcon.addEventListener('click',
function () {
                                showEditForm(assignment);
                            });

assignmentElement.appendChild(editIcon);


assignmentsSection.appendChild(assignmentElement);
                        });
                    } else {
                        console.error('Error:', data.message);
                    }
                })
                .catch(error => console.error('Error:', error));
        });
</script>
```

Лістинг 3.9 – php скрипт для відображеня даних про завдання

```
<?php
session_start();

if (!isset($_SESSION['user_id'])) {
    echo json_encode(["status" => "error", "message" =>
"Користувач не увійшов у систему"]);
    exit();
}
```

```
$user_id = $_SESSION['user_id'];

$servername = "127.127.126.25";
$username = "root";
$password = "";
$dbname = "DB";

$conn = new mysqli($servername, $username, $password, $dbname);

if ($conn->connect_error) {
    echo json_encode(["status" => "error", "message" => "Помилка
з'єднання з базоб даних: " . $conn->connect_error]);
    exit();
}

$sql = "SELECT * FROM Assignments WHERE user_id = ?";
$stmt = $conn->prepare($sql);
$stmt->bind_param("i", $user_id);

if ($stmt->execute()) {
    $result = $stmt->get_result();
    $assignments = array();
    while ($row = $result->fetch_assoc()) {
        $assignments[] = $row;
    }
    echo json_encode(["status" => "success", "assignments" =>
$assignments]);
} else {
    echo json_encode(["status" => "error", "message" => "Помилка
виконання запиту: " . $conn->error]);
}

$stmt->close();
$conn->close();
?>
```

Лістинг 3.10 – php скрипт для обробки додавання нового завдання

```
<?php
session_start();
header('Content-Type: application/json');

if ($_SERVER['REQUEST_METHOD'] !== 'POST') {
    echo json_encode(["status" => "error", "message" => "Метод
запиту повинен бути POST"]);
    exit();
}

if (!isset($_SESSION['user_id'])) {
    echo json_encode(["status" => "error", "message" =>
"Користувач не увійшов у систему"]);
```

```
    exit();
}

$data = json_decode(file_get_contents("php://input"), true);

if (!isset($data['title']) || !isset($data['subject']) || !
isset($data['due_date']) || !isset($data['email_teacher'])) {
    echo json_encode(["status" => "error", "message" =>
"Недостатньо даних для додавання завдання"]);
    exit();
}

$title = $data['title'];
$subject = $data['subject'];
$due_date = $data['due_date'];
$email_teacher = $data['email_teacher'];
$status = 'Невиконано';

$user_id = $_SESSION['user_id'];

$servername = "127.127.126.25";
$username = "root";
$password = "";
$dbname = "DB";

$conn = new mysqli($servername, $username, $password, $dbname);

if ($conn->connect_error) {
    $error_message = "Помилка підключення до бази даних: " .
$conn->connect_error;
    error_log($error_message);
    echo json_encode(["status" => "error", "message" =>
$error_message]);
    exit();
}

$stmt_teacher = $conn->prepare("SELECT user_id FROM Users WHERE
email = ?");
$stmt_teacher->bind_param("s", $email_teacher);
$stmt_teacher->execute();
$result_teacher = $stmt_teacher->get_result();

if ($result_teacher->num_rows > 0) {
    $teacher_id_row = $result_teacher->fetch_assoc();
    $teacher_id = $teacher_id_row['user_id'];

    $sql = "INSERT INTO Assignments (title, subject, due_date,
status, user_id, teacher_id) VALUES (?, ?, ?, ?, ?, ?)";
    $stmt = $conn->prepare($sql);
    $stmt->bind_param("ssssii", $title, $subject, $due_date,
$status, $user_id, $teacher_id);
```

```
    if ($stmt->execute()) {
        echo json_encode(["status" => "success", "message" =>
"Завдання успішно додано"]);
    } else {
        $error_message = "Помилка при додаванні завдання: " .
$conn->error;
        error_log($error_message);
        echo json_encode(["status" => "error", "message" =>
$error_message]);
    }
} else {
    echo json_encode(["status" => "error", "message" =>
"Викладач не знайдений"]);
}

$stmt_teacher->close();
$stmt->close();
$conn->close();
?>
```

Розглянемо сторінку «Розклад». На даній сторінці є можливість вибрати режим відображення розкладу: недільний або місячний вигляд, та період часу за який потрібно відобразити розклад. Режим відображення оброблює скрипт на мові JavaScript (лістинг 3.11), дані які він відображає оброблює php скрипт на серверній стороні додатку (лістинг 3.12).

Лістинг 3.11 – скрипт відображення розкладу на сторінці

```
<script>
        document.addEventListener('DOMContentLoaded', function
() {
            const weekInput = document.getElementById('week-
input');
            const monthInput = document.getElementById('month-
input');
            const weeklySchedule =
document.getElementById('weekly-schedule');
            const monthlySchedule =
document.getElementById('monthly-schedule');
            const monthlyScheduleBody =
document.getElementById('monthly-schedule-body');
            const toggleButton =
document.getElementById('toggle-view-btn');

            function formatDate(dateString) {
                const date = new Date(dateString);
                const day = String(date.getDate()).padStart(2,
'0');
```

```
                const month = String(date.getMonth() +
1).padStart(2, '0');
                const year = date.getFullYear();
                return `${day}.${month}.${year}`;
            }

            function loadWeeklySchedule(week) {
                fetch(`../php/get_schedule.php?week=${week}`)
                    .then(response => response.json())
                    .then(data => {
                        const scheduleCells =
document.querySelectorAll('#weekly-schedule td[id^="monday"],
#weekly-schedule td[id^="tuesday"], #weekly-schedule
td[id^="wednesday"], #weekly-schedule td[id^="thursday"],
#weekly-schedule td[id^="friday"]');
                        scheduleCells.forEach(cell =>
cell.textContent = '');

                        data.forEach(assignment => {
                            const date = new
Date(assignment.date);
                            const dayOfWeek = date.getDay();
                            const startTime =
assignment.start_time;
                            const subject = assignment.subject;
                            const teacher = assignment.teachers;

                            let lessonNumber;
                            switch (startTime) {
                                case '09:00:00': lessonNumber =
1; break;
                                case '10:30:00': lessonNumber =
2; break;
                                case '12:30:00': lessonNumber =
3; break;
                                case '14:00:00': lessonNumber =
4; break;
                                case '15:30:00': lessonNumber =
5; break;
                                default: lessonNumber = null;
                            }

                            let cellId;
                            switch (dayOfWeek) {
                                case 1: cellId = 'monday-' +
lessonNumber; break;
                                case 2: cellId = 'tuesday-' +
lessonNumber; break;
                                case 3: cellId = 'wednesday-' +
lessonNumber; break;
```

```
                                case 4: cellId = 'thursday-' +
lessonNumber; break;
                                case 5: cellId = 'friday-' +
lessonNumber; break;
                                default: cellId = null;
                            }

                            if (cellId && lessonNumber) {
                                const cell =
document.getElementById(cellId);
                                if (cell) {
                                    cell.textContent = `$
{assignment.date}: ${subject} (${teacher})`;
                                }
                            }
                        });
                    })
                    .catch(error => console.error('Error:',
error));
            }

            function loadMonthlySchedule(year, month) {
                fetch(`../php/get_schedule.php?year=${year}
&month=${month}`)
                    .then(response => response.json())
                    .then(data => {
                        monthlyScheduleBody.innerHTML = '';
                        const daysInMonth = new Date(year,
month, 0).getDate();
                        for (let i = 1; i <= daysInMonth; i++) {
                            const row =
document.createElement('tr');
                            row.id = `day-${i}`;
                            const dayCell =
document.createElement('td');
                            dayCell.textContent = i;
                            row.appendChild(dayCell);
                            for (let j = 1; j <= 5; j++) {
                                const cell =
document.createElement('td');
                                const assignment =
data.find(item => {
                                    const assignmentDate = new
Date(item.date);
                                    return
assignmentDate.getDate() === i && assignmentDate.getDay() === j;
                                });
                                if (assignment) {
                                    const subject =
assignment.subject;
```

```
                                    const teacher =
assignment.teachers;
                                    cell.textContent = `$
{subject} (${teacher})`;
                                }
                                row.appendChild(cell);
                            }

monthlyScheduleBody.appendChild(row);
                        }
                    })
                    .catch(error => console.error('Error:',
error));
            }
</script>
```

Лістинг 3.12 – php скрипт для отримання даних про розклад

```
<?php
session_start();

$servername = "127.127.126.25";
$username = "root";
$password = "";
$dbname = "DB";

$conn = new mysqli($servername, $username, $password, $dbname);

if ($conn->connect_error) {
    die("Помилка з'єднання з базою даних: " . $conn-
>connect_error);
}

if (!isset($_SESSION['user_id'])) {
    echo json_encode(array("status" => "error", "message" =>
"Корисувач не увійшов у систему."));
    exit();
}

$currentUserId = $_SESSION['user_id'];

if (isset($_GET['week'])) {
    $week = $_GET['week'];
    $startDate = date('Y-m-d', strtotime($week));
    $endDate = date('Y-m-d', strtotime($startDate . ' +6
days'));
```

```
    $sql = "SELECT * FROM Schedules WHERE user_id = ? AND date
BETWEEN ? AND ?";
    $stmt = $conn->prepare($sql);
    $stmt->bind_param("iss", $currentUserId, $startDate,
$endDate);
} elseif (isset($_GET['year']) && isset($_GET['month'])) {
    $year = $_GET['year'];
    $month = $_GET['month'];
    $daysInMonth = cal_days_in_month(CAL_GREGORIAN, $month,
$year);
    $startDate = "$year-$month-01";
    $endDate = "$year-$month-$daysInMonth";

    $sql = "SELECT * FROM Schedules WHERE user_id = ? AND date
BETWEEN ? AND ?";
    $stmt = $conn->prepare($sql);
    $stmt->bind_param("iss", $currentUserId, $startDate,
$endDate);
}

$stmt->execute();
$result = $stmt->get_result();
$schedules = array();
if ($result->num_rows > 0) {
    while ($row = $result->fetch_assoc()) {
        $schedules[] = $row;
    }
}

echo json_encode($schedules);

$conn->close();
?>
```

Якщо користувач який увійшов у систему є викладачем, то є можливість додати нове заняття. Форма для додавання нового заняття має такі поля:

- Електрона адреса студента.
- Предмет.
- Дата.
- Час початку.
- Час закінчення.

При додаванні нового заняття, на електрону адресу студента, який ввів викладач, буде відправлено автоматичне сповіщення про наявність нового заняття.

Лістинг 3.13 – php скрипт додавання нового заняття

```
<?php
session_start();
header('Content-Type: application/json');

if (!isset($_SESSION['user_id']) || !
isset($_SESSION['user_email'])) {
    echo json_encode(["status" => "error", "message" =>
"Користувач не авторизований."]);
    exit();
}

$servername = "127.127.126.25";
$username = "root";
$password = "";
$dbname = "DB";
$port = 3306;

$conn = new mysqli($servername, $username, $password, $dbname,
$port);

if ($conn->connect_error) {
    echo json_encode(["status" => "error", "message" => "Помилка
з'єднання: " . $conn->connect_error]);
    exit();
}

$studentEmail = $_POST['emailInput'];
$subject = $_POST['subjectInput'];
$date = $_POST['dueDateInput'];
$startTime = $_POST['startTimeInput'];
$endTime = $_POST['endTimeInput'];

$teacherEmail = $_SESSION['user_email'];
$teacherId = $_SESSION['user_id'];

$sql = "SELECT first_name, last_name FROM Users WHERE user_id
= ?";
$stmt = $conn->prepare($sql);
$stmt->bind_param("i", $teacherId);
$stmt->execute();
$result = $stmt->get_result();

if ($result->num_rows > 0) {
    $teacher = $result->fetch_assoc();
```

```
    $teacherName = $teacher['first_name'] . ' ' .
$teacher['last_name'];
} else {
    echo json_encode(["status" => "error", "message" =>
"Викладача не знайдено."]);
    exit();
}

$sql = "SELECT user_id FROM Users WHERE email = ?";
$stmt = $conn->prepare($sql);
$stmt->bind_param("s", $studentEmail);
$stmt->execute();
$result = $stmt->get_result();

if ($result->num_rows > 0) {
    $student = $result->fetch_assoc();
    $studentId = $student['user_id'];

    $sql = "INSERT INTO Schedules (user_id, teacher_id, subject,
date, start_time, end_time, teachers) VALUES
(?, ?, ?, ?, ?, ?, ?)";
    $stmt = $conn->prepare($sql);
    $stmt->bind_param("iissssss", $studentId, $teacherId,
$subject, $date, $startTime, $endTime, $teacherName);

    if ($stmt->execute()) {
        $to = $studentEmail;
        $subjectMail = 'Нове заняття від ' . $teacherName;
        $message = 'Здравствуйте! На сайті додано нове заняття
від ' . $teacherName . ".\n" .
                   'Предмет: ' . $subject . "\n" .
                   'Дата: ' . $date . "\n" .
                   'Час початку: ' . $startTime . "\n" .
                   'Час закінчення: ' . $endTime;
        $headers = 'From: ' . $teacherEmail . "\r\n" .
                   'Reply-To: ' . $teacherEmail . "\r\n" .
                   'X-Mailer: PHP/' . phpversion();

        if (mail($to, $subjectMail, $message, $headers)) {
            echo json_encode(["status" => "success", "message"
=> "Заняття успішно додано. Повідомлення надіслано студенту."]);
        } else {
            echo json_encode(["status" => "success", "message"
=> "Заняття успішно додано, але виникла помилка при надсиланні
повідомлення студенту."]);
        }
    } else {
        echo json_encode(["status" => "error", "message" =>
"Помилка при додаванні заняття: " . $stmt->error]);
    }
} else {
```

```
    echo json_encode(["status" => "error", "message" =>
"Студента з таким email не знайдено."]);
}

$stmt->close();
$conn->close();
?>
```

1.2. Розробка інтерфейсу користувача

Розробка інтерфейсу користувача розроблюваного додатку здійснювалася наступними мовами програмування: HTML5, CSS3, JavaScript. Вся розробка додатку проводилася у програмі Microsoft Visual Studio Code.

На кожній сторінці сайту наявне головне меню для швидкого переміщування між потрібними сторінками. Меню має такі пункти: розклад, завдання, навчальні матеріали, профіль, вийти. Розглянемо деякі структури компонентів сайту.

Лістинг 3.14 – код структури головного меню

```
<header class="header">
    <div class="container">
        <nav class="nav">
            <a href="#" class="logo"><img
src="../images/logo.svg" alt="" width="150" height="100"></a>
            <ul class="menu">
                <li class="menu__list"><a
href="../pages/schesudeules.html"
class="menu__link">Розклад</a></li>
                <li class="menu__list"><a
href="../pages/Assignments.html"
class="menu__link">Завдання</a></li>
                <li class="menu__list"><a
href="../pages/material.html" class="menu__link">Навчальні
матеріали</a></li>
            </ul>
            <a href="../pages/profile.html" class="profile
profile-active"><img src="../images/profile-icon.svg" alt=""
                    width="60" height="60">Профіль</a>
            <a href="../index.html" class="profile
menu__link">Вийти</a>
        </nav>
    </div>
</header>
```

Лістинг 3.15 – код структури сторінки «Профіль»

```
<div class="container">
        <section class="profile__section">
            <form action="" class="profile__inner-form">
                <span class="span-form-profile">Прізвище <input
type="text" id="last_name"
                        class="profile___form-input"></span>
                <span class="span-form-profile">Ім’я <input
type="text" id="first_name"
                        class="profile___form-input"></span>
                <span class="span-form-profile">E-mail <input
type="email" id="email" class="profile___form-input"
                        disabled></span>
                <span class="span-form-profile">Пароль <input
type="password" class="profile___form-input"
                        id="new_password"></span>
                <span class="span-form-profile">Повторити пароль
<input type="password" class="profile___form-input"
                        id="confirm_password"></span>
                <span class="decript__error"></span>
                <button class="profile__save-btn autorization-
btn">Зберігти зміни</button>
            </form>
        </section>
    </div>
```

Лістинг 3.16 – код структури сторінки «Розклад»

```
<div class="schedule-container">
        <div class="controls">
            <div>
                <span class="week-label">Виберіть
тиждень:</span>
                <input type="week" class="week-input" id="week-
input">
                <span class="month-label hidden">Виберіть
місяць:</span>
                <input type="month" class="month-input hidden"
id="month-input">
            </div>
            <button class="view-toggle-btn" id="toggle-view-
btn">Переключити на місячний вид</button>
        </div>

        <table id="weekly-schedule">
            <thead>
                <tr>
                    <th></th>
                    <th>Понеділок</th>
```

```
                <th>Вівторок</th>
                <th>Середа</th>
                <th>Четверг</th>
                <th>П'ятниця</th>
            </tr>
        </thead>
        <tbody>
            <tr>
                <td>1 пара <br> 9.00-10.20</td>
                <td id="monday-1"></td>
                <td id="tuesday-1"></td>
                <td id="wednesday-1"></td>
                <td id="thursday-1"></td>
                <td id="friday-1"></td>
            </tr>
            <tr>
                <td>2 пара <br> 10.30-11.50</td>
                <td id="monday-2"></td>
                <td id="tuesday-2"></td>
                <td id="wednesday-2"></td>
                <td id="thursday-2"></td>
                <td id="friday-2"></td>
            </tr>
            <tr>
                <td>3 пара <br> 12.30-13.50</td>
                <td id="monday-3"></td>
                <td id="tuesday-3"></td>
                <td id="wednesday-3"></td>
                <td id="thursday-3"></td>
                <td id="friday-3"></td>
            </tr>
            <tr>
                <td>4 пара <br> 14.00-15.20</td>
                <td id="monday-4"></td>
                <td id="tuesday-4"></td>
                <td id="wednesday-4"></td>
                <td id="thursday-4"></td>
                <td id="friday-4"></td>
            </tr>
            <tr>
                <td>5 пара <br> 15.30-16.50</td>
                <td id="monday-5"></td>
                <td id="tuesday-5"></td>
                <td id="wednesday-5"></td>
                <td id="thursday-5"></td>
                <td id="friday-5"></td>
            </tr>
        </tbody>
    </table>
```

```
        <table id="monthly-schedule" class="hidden">
            <thead>
                <tr>
                    <th>День</th>
                    <th>1 пара</th>
                    <th>2 пара</th>
                    <th>3 пара</th>
                    <th>4 пара</th>
                    <th>5 пара</th>
                </tr>
            </thead>
            <tbody id="monthly-schedule-body">
                <!-- Порожні осередки для місячного розкладу
будуть додані динамічно -->
            </tbody>
        </table>
    </div>
```

ВИСНОВКИ

При написані даної кваліфікаційної роботи було розроблено додаток для оптимізації навчального процесу. Робота охоплювала весь цикл розробки додатку: аналіз вимог до програмного забезпечення, розробка користувацького інтерфейс, розробка серверної частини додатку, тестування розроблюваного додатку. Результатом роботи стали наступні аспекти:

1. Проведено аналіз потреб користувачів, серед яких студенти та викладачі. Було визначено основні функціональні вимоги до програми, такі як управління розкладом, відстеження дедлайнів, автоматизоване сповіщення користувачів про нові завдання та заняття.
2. Спроектовано модульну архітектуру додатку, що дозволить додавати нові функції при потребі. У додатку використанні сучасні технології, що забезпечують ефективність та надійність використання.
3. Розроблено модулі для керування розкладом, які дозволяють створювати, видаляти та редагувати записи. Було створено систему сповіщень, що автоматично надсилає нагадування про дедлайни та нові заняття через електронну пошту.

Розроблений додаток для оптимізації навчального процесу є ефективним інструментом, що сприяє підвищенню ефективності навчального процесу.

ПЕРЕЛІК ПОСИЛАНЬ

1. Веллінг Л., Томсон Л. Розробка веб-додатків за допомогою PHP і MySQL. 5-те видання. Київ: Видавництво "БХВ-Петербург", 2019.
2. Довідник по HTML тегам [Еклектронний ресурс] – https://css.in.ua/html/tags
3. Довідник по CSS властивостям [Електронний ресурс] – https://css.in.ua/css/properties
4. Дронов С. А. (2021). HTML, JavaScript, PHP і mySQL. Джентельменський набір Web-майстра, 5 изд. Київ: Видавництво.
5. Онлайн посібник з PHP [Електронний ресурс] – https://php720.com/
6. Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5. Робин Никсон.

ДОДАТОК А – Код програми

Лістинг А.1 – код структури форми реєстрації

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

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Document</title>
    <link rel="stylesheet" href="../css/reset.css">
    <link rel="stylesheet" href="../css/style.css">
    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com"
crossorigin>
    <link href="https://fonts.googleapis.com/css2?
family=Kantumruy+Pro:ital,wght@0,100..700;1,100..700&display=swa
p"
        rel="stylesheet">
</head>

<body>
    <div class="container__inner">
        <section class="main__autorization">
            <img src="../images/registration-img.png" alt=""
class="autorization__section-img">
            <form id="registration-form"
class="autorization__section-inner" method="post">
                <input type="text" placeholder="Прізвище"
class="autorization__input" name="last_name" required>
                <input type="text" placeholder="Ім'я"
class="autorization__input" name="first_name" required>
                <input type="email" placeholder="Email"
class="autorization__input" name="email" required>
                <input type="password" placeholder="Пароль"
class="autorization__input" name="password" required>
                <div class="radio__btn-inner">
                    <input type="radio" id="student" name="role"
value="student" class="input-radio" required>
                    <label for="student" class="input-label-
radio"><span>Студент</span></label><br>
                    <input type="radio" id="teacher" name="role"
value="teacher" class="input-radio" required>
                    <label for="teacher" class="input-label-
radio"><span>Викладач</span></label><br>
                </div>
```

```
                <div class="autorization__inner-btn">
                    <a href="../index.html" class="autorization-
btn-link">Увійти</a>
                    <button type="submit" class="autorization-
btn">Реєстрація</button>
                </div>
                <span class="info_regitration" id="info"></span>
            </form>
        </section>
    </div>

    <script>
        document.getElementById('registration-
form').addEventListener('submit', function (event) {
            event.preventDefault();

            var formData = new FormData(this);

            fetch('../php/registred.php', {
                method: 'POST',
                body: formData
            })
            .then(response => response.json())
            .then(data => {
                var infoSpan = document.getElementById('info');
                if (data.status === 'success') {
                    infoSpan.textContent = data.message;
                    infoSpan.style.color = 'green';
                    infoSpan.style.fontWeight = '700';
                } else {
                    infoSpan.textContent = data.message;
                    infoSpan.style.color = 'red';
                    infoSpan.style.fontWeight = '700';
                }
            })
            .catch(error => {
                console.error('Помилка:', error);
                var infoSpan = document.getElementById('info');
                infoSpan.textContent = 'Помилка підключення';
                infoSpan.style.color = 'red';
            });
        });
    </script>
</body>

</html>
```

Лістинг А.2 – код структури форми авторизації

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

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Document</title>
    <link rel="stylesheet" href="css/reset.css">
    <link rel="stylesheet" href="css/style.css">
    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com"
crossorigin>
    <link href="https://fonts.googleapis.com/css2?
family=Kantumruy+Pro:ital,wght@0,100..700;1,100..700&display=swa
p"
        rel="stylesheet">
</head>

<body>
    <div class="container__inner">
        <section class="main__autorization">
            <img src="images/autorization-img.png" alt=""
class="autorization__section-img">
            <form id="login-form" class="autorization__section-
inner">
                <input type="email" placeholder="Email"
class="autorization__input" name="email" required>
                <input type="password" placeholder="Пароль"
class="autorization__input" name="password" required>
                <div class="autorization__inner-btn">
                    <button type="submit" class="autorization-
btn">Увійти</button>
                    <a href="pages/registartion.html"
class="autorization-btn-link">Реєстрація</a>
                </div>
                <span class="info_regitration" id="info"></span>
            </form>
        </section>
    </div>

    <script>
        document.getElementById('login-
form').addEventListener('submit', function(event) {
            event.preventDefault();

            var formData = new FormData(this);
```

```
            fetch('php/login.php', {
                method: 'POST',
                body: formData
            })
            .then(response => response.json())
            .then(data => {
                var infoSpan = document.getElementById('info');
                if (data.status === 'success') {
                    infoSpan.textContent = data.message;
                    infoSpan.style.color = 'green';
                    window.location.href = data.redirect;
                } else {
                    infoSpan.textContent = data.message;
                    infoSpan.style.color = 'red';
                }
            })
            .catch(error => {
                console.error('Error:', error);
                var infoSpan = document.getElementById('info');
                infoSpan.textContent = 'Ошибка подключения.';
                infoSpan.style.color = 'red';
            });
        });
    </script>
</body>

</html>
```

Лістинг А.3 – код структури сторінки «Розклад» (рівень – викладач)

```
<body>
    <div class="add_container">
        <h2>Додати заняття</h2>
        <form action="" class="add_form_schedues" id="schedule-
form">
            <label for="emailInput">Email студента:</label>
            <input class="add_form_schedues_input" type="email"
id="emailInput" name="emailInput" required><br>
            <label for="subjectInput">Предмет:</label>
            <input type="text" id="subjectInput"
name="subjectInput" required><br>
            <label for="dueDateInput">Дата:</label>
            <input type="date" id="dueDateInput"
name="dueDateInput" required><br>
            <label for="startTimeInput">Час початку:</label>
            <input type="time" id="startTimeInput"
name="startTimeInput" required><br>
            <label for="endTimeInput">Час закінчення:</label>
```

```
            <input type="time" id="endTimeInput"
name="endTimeInput" required><br>
            <button id="addSchedulesBtn"
type="submit">Додати</button>
        </form>

        <script>
            document.getElementById('schedule-
form').addEventListener('submit', function (event) {
                event.preventDefault();

                var formData = new FormData(this);

                fetch('../php/add_schedule.php', {
                    method: 'POST',
                    body: formData
                })
                    .then(response => response.json())
                    .then(data => {
                        alert(data.message);
                        if (data.status === 'success') {
                        }
                    })
                    .catch(error => {
                        console.error('Error:', error);
                        alert('Ошибка подключения.');
                    });
            });
        </script>

    </div>
    <div class="schedule-container">
        <div class="controls">
            <div>
                <span class="week-label">Виберіть
тиждень:</span>
                <input type="week" class="week-input" id="week-
input">
                <span class="month-label hidden">Виберіть
місяць:</span>
                <input type="month" class="month-input hidden"
id="month-input">
            </div>
            <button class="view-toggle-btn" id="toggle-view-
btn">Переключити на місячний вид</button>
        </div>

        <table id="weekly-schedule">
            <thead>
```

```
        <tr>
            <th></th>
            <th>Понеділок</th>
            <th>Вівторок</th>
            <th>Середа</th>
            <th>Четверг</th>
            <th>П'ятниця</th>
        </tr>
    </thead>
    <tbody>
        <tr>
            <td>1 пара <br> 9.00-10.20</td>
            <td id="monday-1"></td>
            <td id="tuesday-1"></td>
            <td id="wednesday-1"></td>
            <td id="thursday-1"></td>
            <td id="friday-1"></td>
        </tr>
        <tr>
            <td>2 пара <br> 10.30-11.50</td>
            <td id="monday-2"></td>
            <td id="tuesday-2"></td>
            <td id="wednesday-2"></td>
            <td id="thursday-2"></td>
            <td id="friday-2"></td>
        </tr>
        <tr>
            <td>3 пара <br> 12.30-13.50</td>
            <td id="monday-3"></td>
            <td id="tuesday-3"></td>
            <td id="wednesday-3"></td>
            <td id="thursday-3"></td>
            <td id="friday-3"></td>
        </tr>
        <tr>
            <td>4 пара <br> 14.00-15.20</td>
            <td id="monday-4"></td>
            <td id="tuesday-4"></td>
            <td id="wednesday-4"></td>
            <td id="thursday-4"></td>
            <td id="friday-4"></td>
        </tr>
        <tr>
            <td>5 пара <br> 15.30-16.50</td>
            <td id="monday-5"></td>
            <td id="tuesday-5"></td>
            <td id="wednesday-5"></td>
            <td id="thursday-5"></td>
            <td id="friday-5"></td>
```

```
                </tr>
            </tbody>
        </table>

        <table id="monthly-schedule" class="hidden">
            <thead>
                <tr>
                    <th>День</th>
                    <th>1 пара</th>
                    <th>2 пара</th>
                    <th>3 пара</th>
                    <th>4 пара</th>
                    <th>5 пара</th>
                </tr>
            </thead>
            <tbody id="monthly-schedule-body">
                <!-- Пустые ячейки для месячного расписания
будут добавлены динамически -->
            </tbody>
        </table>
    </div>

    <script>
        document.addEventListener('DOMContentLoaded', function
() {
            const weekInput = document.getElementById('week-
input');
            const monthInput = document.getElementById('month-
input');
            const weeklySchedule =
document.getElementById('weekly-schedule');
            const monthlySchedule =
document.getElementById('monthly-schedule');
            const monthlyScheduleBody =
document.getElementById('monthly-schedule-body');
            const toggleButton =
document.getElementById('toggle-view-btn');

            function formatDate(dateString) {
                const date = new Date(dateString);
                const day = String(date.getDate()).padStart(2,
'0');
                const month = String(date.getMonth() +
1).padStart(2, '0');
                const year = date.getFullYear();
                return `${day}.${month}.${year}`;
            }

            function loadWeeklySchedule(week) {
```

```
                fetch(`../php/get_schedule.php?week=${week}`)
                    .then(response => response.json())
                    .then(data => {
                        const scheduleCells =
document.querySelectorAll('#weekly-schedule td[id^="monday"],
#weekly-schedule td[id^="tuesday"], #weekly-schedule
td[id^="wednesday"], #weekly-schedule td[id^="thursday"],
#weekly-schedule td[id^="friday"]');
                        scheduleCells.forEach(cell =>
cell.textContent = '');

                        data.forEach(assignment => {
                            const date = new
Date(assignment.date);
                            const dayOfWeek = date.getDay();
                            const startTime =
assignment.start_time;
                            const subject = assignment.subject;
                            const teacher = assignment.teachers;

                            let lessonNumber;
                            switch (startTime) {
                                case '09:00:00': lessonNumber =
1; break;
                                case '10:30:00': lessonNumber =
2; break;
                                case '12:30:00': lessonNumber =
3; break;
                                case '14:00:00': lessonNumber =
4; break;
                                case '15:30:00': lessonNumber =
5; break;
                                default: lessonNumber = null;
                            }

                            let cellId;
                            switch (dayOfWeek) {
                                case 1: cellId = 'monday-' +
lessonNumber; break;
                                case 2: cellId = 'tuesday-' +
lessonNumber; break;
                                case 3: cellId = 'wednesday-' +
lessonNumber; break;
                                case 4: cellId = 'thursday-' +
lessonNumber; break;
                                case 5: cellId = 'friday-' +
lessonNumber; break;
                                default: cellId = null;
                            }
```

```
                            if (cellId && lessonNumber) {
                                const cell =
document.getElementById(cellId);
                                if (cell) {
                                    cell.textContent = `$
{assignment.date}: ${subject} (${teacher})`;
                                }
                            }
                        });
                    })
                    .catch(error => console.error('Error:',
error));
            }

            function loadMonthlySchedule(year, month) {
                fetch(`../php/get_schedule.php?year=${year}
&month=${month}`)
                    .then(response => response.json())
                    .then(data => {
                        monthlyScheduleBody.innerHTML = '';

                        const daysInMonth = new Date(year,
month, 0).getDate();
                        for (let i = 1; i <= daysInMonth; i++) {
                            const row =
document.createElement('tr');
                            row.id = `day-${i}`;
                            const dayCell =
document.createElement('td');
                            dayCell.textContent = i;
                            row.appendChild(dayCell);
                            for (let j = 1; j <= 5; j++) {
                                const cell =
document.createElement('td');
                                const assignment =
data.find(item => {
                                    const assignmentDate = new
Date(item.date);
                                    return
assignmentDate.getDate() === i && assignmentDate.getDay() === j;
                                });
                                if (assignment) {
                                    const subject =
assignment.subject;
                                    const teacher =
assignment.teachers;
                                    cell.textContent = `$
{subject} (${teacher})`;
```

```
                                }
                                row.appendChild(cell);
                            }

monthlyScheduleBody.appendChild(row);
                        }
                    })
                    .catch(error => console.error('Error:',
error));
            }

            weekInput.addEventListener('change', function () {
                const week = this.value;
                loadWeeklySchedule(week);
            });

            monthInput.addEventListener('change', function () {
                const year = this.value.split('-')[0];
                const month = this.value.split('-')[1];
                loadMonthlySchedule(year, month);
            });

            toggleButton.addEventListener('click', function () {
                const weekInput = document.querySelector('.week-
input');
                const monthInput =
document.querySelector('.month-input');
                const weekLabel = document.querySelector('.week-
label');
                const monthLabel =
document.querySelector('.month-label');
                weeklySchedule.classList.toggle('hidden');
                monthlySchedule.classList.toggle('hidden');
                weekInput.classList.toggle('hidden');
                monthInput.classList.toggle('hidden');
                weekLabel.classList.toggle('hidden');
                monthLabel.classList.toggle('hidden');

                if
(monthlySchedule.classList.contains('hidden')) {
                    weekInput.value = '';
                    monthInput.value = '';
                }
            });
        });
    </script>
</body>
```

Лістинг А.4 – скрипт додавання нового завдання

```
<?php
session_start();
header('Content-Type: application/json');

if ($_SERVER['REQUEST_METHOD'] !== 'POST') {
    echo json_encode(["status" => "error", "message" => "Метод
запиту повинен бути POST"]);
    exit();
}

if (!isset($_SESSION['user_id'])) {
    echo json_encode(["status" => "error", "message" =>
"Користувач не увійшов у систему"]);
    exit();
}

$data = json_decode(file_get_contents("php://input"), true);

if (!isset($data['title']) || !isset($data['subject']) || !
isset($data['due_date']) || !isset($data['email_student'])) {
    echo json_encode(["status" => "error", "message" =>
"Недостатньо даних для додавання завдання"]);
    exit();
}

$title = $data['title'];
$subject = $data['subject'];
$due_date = $data['due_date'];
$email_student = $data['email_student'];
$status = 'Невиконано';

$user_id = $_SESSION['user_id'];

$servername = "127.127.126.25";
$username = "root";
$password = "";
$dbname = "DB";

$conn = new mysqli($servername, $username, $password, $dbname);

if ($conn->connect_error) {
    $error_message = "Помилка при підключенні до бази даних: " .
$conn->connect_error;
    error_log($error_message);
    echo json_encode(["status" => "error", "message" =>
$error_message]);
    exit();
```

```
}

$stmt_student = $conn->prepare("SELECT user_id FROM Users WHERE
email = ?");
$stmt_student->bind_param("s", $email_student);
$stmt_student->execute();
$result_student = $stmt_student->get_result();

if ($result_student->num_rows > 0) {
    $student_id_row = $result_student->fetch_assoc();
    $student_id = $student_id_row['user_id'];

    $sql = "INSERT INTO Assignments (title, subject, due_date,
status, user_id, teacher_id) VALUES (?, ?, ?, ?, ?, ?)";
    $stmt = $conn->prepare($sql);
    $stmt->bind_param("ssssii", $title, $subject, $due_date,
$status, $student_id, $user_id);

    if ($stmt->execute()) {
        $to = $email_student;
        $subject = "Нове завдання";
        $message = "Студенте,\n\nВаш викладач додав нове
завдання: \"{$title}\" з предмету \"{$subject}\". Просимо
ознайомитись із завданням та виконати його до зазначеного
терміну.\n\nЗ повагою,\nВаш викладач.";
        $headers = "From: marinochksgosip@gmail.com";

        if (mail($to, $subject, $message, $headers)) {
            echo json_encode(["status" => "success", "message"
=> "Завдання успішно додане. Повідомлення студенту
відправлене."]);
        } else {
            echo json_encode(["status" => "warning", "message"
=> "Завдання успішно додане, але виникла помилка при відправці
повідомленя студенту."]);
        }
    } else {
        $error_message = "Помилка при додаванні завдання: " .
$conn->error;
        error_log($error_message);
        echo json_encode(["status" => "error", "message" =>
$error_message]);
    }
} else {
    echo json_encode(["status" => "error", "message" =>
"Студента з такою електронною адресою не знайдено."]);
}

$stmt_student->close();
```

```
$stmt->close();
$conn->close();
?>
```

Лістинг А.5 – скрипт завантаження навчального матеріалу

```
<?php

if (isset($_GET['id'])) {
    $materialId = $_GET['id'];

    $servername = "127.127.126.25";
    $username = "root";
    $password = "";
    $dbname = "DB";

    $conn = new mysqli($servername, $username, $password,
$dbname);

    if ($conn->connect_error) {
        die("Помилка з'єднання: " . $conn->connect_error);
    }

    $stmt = $conn->prepare("SELECT file_path FROM Materials
WHERE material_id = ?");
    $stmt->bind_param("i", $materialId);

    $stmt->execute();

    $result = $stmt->get_result();

    if ($result->num_rows > 0) {
        $row = $result->fetch_assoc();
        $filePath = $row['file_path'];

        $stmt->close();
        $conn->close();

        if (file_exists($filePath)) {
            header('Content-Description: File Transfer');
            header('Content-Type: application/octet-stream');
            header('Content-Disposition: attachment; filename="'
. basename($filePath) . '"');
            header('Expires: 0');
            header('Cache-Control: must-revalidate');
            header('Pragma: public');
            header('Content-Length: ' . filesize($filePath));

            readfile($filePath);
```

```
            exit;
        } else {
            exit('Файл не знайдено.');
        }
    } else {
        http_response_code(404);
        exit('Матеріал не знайдено.');
    }
} else {
    http_response_code(400);
    exit('Ідентифіктор матеріалу не було передано.');
}
?>
```