

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

**Відокремлений структурний підрозділ
«Тернопільський фаховий коледж
Тернопільського національного технічного університету імені Івана Пулюя»
Відділення телекомунікацій та електронних систем
Циклова комісія комп'ютерних наук**

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи

фахового молодшого бакалавра

на тему: Розробка інформаційної системи ІТ-школи «Mriya IT»

Виконав: студент IV курсу, групи КН-423

спеціальності: 122 Комп'ютерні науки

Назар ТИПА

Керівник

Руслан СЛОБОДЯН

Рецензент

(ім'я та прізвище)

ВІДОКРЕМЛЕНИЙ СТРУКТУРНИЙ ПІДРОЗДІЛ
«ТЕРНОПІЛЬСЬКИЙ ФАХОВИЙ КОЛЕДЖ
ТЕРНОПІЛЬСЬКОГО НАЦІОНАЛЬНОГО ТЕХНІЧНОГО УНІВЕРСИТЕТУ
ІМЕНІ ІВАНА ПУЛЮЯ»

Відділення телекомунікацій та електронних систем
Циклова комісія комп'ютерних наук
Освітньо-професійний ступінь «фаховий молодший бакалавр»
Спеціальність 122 Комп'ютерні науки
Галузь знань 12 Інформаційні технології

ЗАТВЕРДЖУЮ

Голова циклової комісії
комп'ютерних наук

_____ Галина МАРЦІЯШ

« 02 » березня 2026 року

**З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ**

Типі Назару Дмитровичу

(прізвище, ім'я, по батькові)

1. Тема роботи: Розробка інформаційної системи ІТ-школи «Mriya IT»,

керівник роботи Слободян Руслан Олесійович,

(прізвище, ім'я, по батькові)

затверджені наказом вищого навчального закладу № 4/9-132 від 27.02.2026 р.

2. Строк подання студентом роботи: 19.06.2026 р.

3. Вихідні дані до роботи: технічне завдання на розробку програмного забезпечення, мова програмування JavaScript, фреймворк Express.js, бібліотеки: React, Redux; СКБД Mongo DB, стандарти: ДСТУ ISO/IEC/IEEE 29148:2025, ДСТУ ISO/IEC/IEEE 29119-1:2017, ДСТУ 3008:2015, ДСТУ 8302:2015.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити):

1 Загальний розділ

1.1 Аналітичний огляд існуючих рішень

1.2 Технічне завдання

1.2.1 Найменування та область застосування

1.2.2 Призначення розробки

1.2.3 Вимоги до функціоналу інформаційної системи

1.2.4 Вимоги до програмної документації

1.2.5 Техніко-економічні показники

1.2.6 Стадії та етапи розробки

1.2.7 Порядок тестування та прийому

2 Розробка технічного та робочого проекту

2.1 Розробка структури інформаційної системи

2.2 Створення та верстка сторінок інформаційної системи

2.3 Розробка структури бази даних інформаційної системи

2.4 Програмування інформаційної системи

2.4.1 Написання клієнтської частини

2.4.2 Написання серверної частини

2.5 Тестування інформаційної системи

3 Спеціальний розділ

3.1 Інструкція з розміщення інформаційної системи в Інтернеті

3.2 Інструкція з обслуговування та наповнення інформаційної системи

3.3 Інструкція з популяризації та підтримки інформаційної системи

4 Економічний розділ

4.1 Визначення стадій технологічного процесу та загальної тривалості

проведення розробки інформаційної системи

4.2 Визначення витрат на оплату праці та відрахувань на соціальні заходи

4.3 Розрахунок витрат на електроенергію

4.4 Розрахунок суми амортизаційних відрахувань

4.5 Обчислення накладних витрат

4.6 Складання кошторису витрат та визначення собівартості інформаційної

системи

4.7 Розрахунок ціни інформаційної системи

4.8 Визначення економічної ефективності і терміну окупності капітальних

вкладень

5 Охорона праці, техніка безпеки та екологічні вимоги

5.1 Інструкції з охорони праці, їх поділ

5.2 Системи засобів і заходів забезпечення електробезпеки

6 Висновки

Додаткові вказівки: виконання кваліфікаційної роботи із розробкою програмного продукту – інформаційної системи «Mriya IT».

5. Перелік графічного матеріалу:

1. Схема структурна клієнтської частини
2. UML-діаграма варіантів використання
3. ER-діаграма бази даних
4. Таблиця техніко-економічних показників

6. Консультанти розділів роботи

Розділ	Ім'я, прізвище та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний розділ	Любов КАЛУШКА		
Охорона праці, техніка безпеки та екологічні вимоги	Генадій ГОРЯЧЕК		

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Отримання і аналіз технічного завдання	20.03.2026	
2	Збір і узагальнення інформації	01.05.2026	
3	Написання першого розділу	15.05.2026	
4	Розробка технічного та робочого проєкту	29.05.2026	
5	Написання спеціального розділу	05.06.2026	
6	Розрахунок економічної частини	08.06.2026	
7	Написання розділу охорони праці	09.06.2026	
8	Виконання графічної частини	10.06.2026	
9	Оформлення пояснювальної записки	11.06.2026	
10	Погодження нормоконтролю	12.06.2026	
11	Попередній захист кваліфікаційної роботи	.06.2026	
12	Захист кваліфікаційної роботи	.06.2026	

7. Дата видачі завдання: 06 березня 2026 р.

Студент

(підпис)

Назар ТИПА

Керівник роботи

(підпис)

Руслан СЛОБОДЯН

ЗМІСТ

Анотація.....	7
Abstract.....	8
Вступ.....	9
1 Загальний розділ.....	10
1.1 Аналітичний огляд існуючих рішень.....	10
1.2 Технічне завдання	12
1.2.1 Найменування та область застосування	12
1.2.2 Призначення розробки.....	13
1.2.3 Вимоги до функціоналу інформаційної системи.....	13
1.2.4 Вимоги до програмної документації.....	14
1.2.5 Техніко-економічні показники	15
1.2.6 Стадії та етапи розробки	15
1.2.7 Порядок контролю та прийому.....	16
2 Розробка технічного та робочого проєкту.....	18
2.1 Розробка структури інформаційної системи.....	18
2.2 Створення та верстка сторінок інформаційної системи	20
2.3 Розробка структури бази даних інформаційної системи	23
2.4 Програмування інформаційної системи	27
2.4.1 Написання клієнтської частини.....	27
2.4.2 Написання серверної частини.....	33
2.5 Тестування інформаційної системи	37
3 Спеціальний розділ	43
3.1 Інструкція з розміщення інформаційної системи в Інтернеті	43
3.2 Інструкція з обслуговування та наповнення інформаційної системи	46
3.3 Інструкція з популяризації та підтримки інформаційної системи.....	52

					2026.КВР.122.423.22.00.00 ПЗ						
Зм.	Арк.	№ докум.	Підпис	Дата							
Розроб.		Типа Н.Д.			Розробка інформаційної системи ІТ-школи «Mriya IT» Пояснювальна записка			Літ.	Арк.	Аркушів	
Перевір.		Слободян Р.О.								5	88
Реценз.								ВСП ТФК ТНТУ КН-423 м. Тернопіль			
Н. Контр.		Приймак В.А.									
Затверд.											

4	Економічний розділ	53
4.1	Визначення стадій технологічного процесу та загальної тривалості проведення розробки інформаційної системи	53
4.2	Визначення витрат на оплату праці та відрахувань на соціальні заходи	54
4.3	Розрахунок витрат на електроенергію	56
4.4	Розрахунок суми амортизаційних відрахувань інформаційної системи.	57
4.5	Обчислення накладних витрат.....	57
4.6	Складання кошторису витрат та визначення собівартості інформаційної системи	58
4.7	Розрахунок ціни інформаційної системи.....	58
4.8	Визначення економічної ефективності і терміну окупності капітальних вкладень	59
5	Охорона праці, техніка безпеки та екологічні вимоги	61
5.1	Інструкції з охорони праці, їх поділ.....	61
5.2	Системи засобів і заходів забезпечення електробезпеки.....	65
	Висновки.....	70
	Перелік посилань	72
	Додатки	73
	Додаток А. Лістинг файлу «App.jsx».....	73
	Додаток Б. Лістинг файлу «Login.jsx».....	77
	Додаток В. Лістинг файлу «LessonDetails.jsx»	80
	Додаток Г. Лістинг файлу «server.js»	87

АНОТАЦІЯ

Тема кваліфікаційної роботи: Розробка інформаційної системи ІТ-школи «Mriya IT».

Для досягнення мети кваліфікаційної роботи виконано розробку веб-орієнтованої системи управління навчанням, призначеної для організації дистанційного навчального процесу в ІТ-школі, ведення електронного журналу успішності, адміністрування груп та завантаження навчальних медіафайлів. Створений застосунок дозволяє автоматизувати взаємодію між викладачами та студентами.

За структурою пояснювальна записка складається з п'яти розділів. Загальний обсяг записки становить 91 сторінка.

У першому загальному розділі описуються огляд існуючих освітніх рішень та аналіз технічного завдання. На основі порівняння визначаються недоліки систем аналогів та обґрунтовується необхідність розробки власного веб-ресурсу.

У другому розділі представлено процес створення програмного продукту на базі сучасного стеку. Описано архітектуру системи клієнт-сервер, структуру нереляційної бази даних, особливості проектування інтерфейсу користувача на, а також етапи функціонального тестування.

У спеціальному розділі детально висвітлено процес розгортання та інсталяції модулів у хмарному середовищі Vercel та Render. Наведено інструкцію з початкового заповнення бази даних тестовою інформацією та правила експлуатації адміністративної частини системи.

В економічному розділі наведено розрахунок вартості проєкту та оцінку термінів окупності.

У п'ятому розділі розглянуто два питання охорони праці, техніки безпеки

До складу кваліфікаційної роботи входить графічна частина, яка складається із діаграми варіантів використання, структурної схеми системи, схеми бази даних, техніко-економічних показників. Усі графічні матеріали виконані на окремих аркушах формату А1.

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		7

ABSTRACT

Thesis Topic: Development of an Information System for the “Mriya IT” IT School.

To achieve the goal of this thesis, a web-based learning management system was developed, designed to organize the distance learning process at the IT school, maintain an electronic grade book, manage groups, and upload educational media files. The created application automates interaction between instructors and students.

In terms of structure, the explanatory note consists of five chapters. The total length of the note is 91 pages.

The first general chapter provides an overview of existing educational solutions and an analysis of the technical specifications. Based on a comparison, the shortcomings of similar systems are identified, and the need to develop a proprietary web resource is justified.

The second section presents the process of creating a software product based on a modern technology stack. It describes the client-server system architecture, the structure of a non-relational database, the specifics of user interface design, and the stages of functional testing.

A special section provides a detailed overview of the process of deploying and installing modules in the Vercel and Render cloud environments. It includes instructions for initially populating the database with test data and guidelines for operating the system’s administrative interface.

The economic section presents a calculation of the project’s cost and an assessment of the payback period.

The fifth section addresses two issues related to occupational health and safety.

The thesis includes a graphical component consisting of a diagram of usage scenarios, a system architecture diagram, a database schema, and technical and economic indicators. All graphical materials are presented on separate A1-size sheets.

					2026.KBP.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		8

ВСТУП

В умовах глобальної цифровізації сучасного освітнього простору України виникає гостра потреба у впровадженні надійних та функціональних автоматизованих систем для забезпечення максимально якісної дистанційної підготовки майбутніх фахівців будь-яких технічних напрямків. Зазначені системні зміни вимагають негайного впровадження новітніх інформаційних технологій.

Через необхідність оперативної адаптації існуючих навчальних програм до динамічних вимог сучасного ринку праці розробка власних програмних рішень у галузі систем керування навчанням стає вкрай важливою та актуальною задачею сьогодні. В основу побудови загальної архітектури платформи покладено сучасний модульний принцип. Задля досягнення високої ефективності процесів передачі знань було визначено основну мету створити повноцінну систему «Mriya IT» із вбудованими функціями оцінювання та надійного збереження всіх типів освітніх ресурсів. Під час безпосередньої реалізації даного проекту розв'язано комплекс інженерних задач. Зокрема, було виконано детальне проектування структури нереляційної бази даних, реалізовано адаптивний інтерфейс користувача на базі сучасних бібліотек та побудовано стабільну серверну частину для швидкої обробки значної кількості запитів. Вказаний підхід дозволяє постійно підтримувати високу швидкість мережевої взаємодії.

У даній роботі детально обґрунтовано вибір технологічного стеку Node.js та React як найбільш оптимального середовища для розробки масштабованих веб-платформ із високим рівнем безпеки та цілісності всіх завантажених файлів.

Запропоновані у роботі програмні рішення мають вагоме практичне значення сьогодні. Практичне використання отриманих результатів охоплює роботу приватних ІТ-шкіл, організацію курсів підвищення кваліфікації та розгортання систем корпоративного навчання для забезпечення повного контролю за навчальним прогресом кожного окремого студента чи викладача.

Створена освітня платформа повністю готова до публічного мережевого доступу зараз.

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		9

1 ЗАГАЛЬНИЙ РОЗДІЛ

1.1 Аналітичний огляд існуючих рішень

Ринок програмного забезпечення пропонує багато готових рішень для організації навчального процесу онлайн. Популярні освітні платформи мають свої переваги та великі недоліки. Розглянемо детальніше такі відомі системи як Moodle та Google Classroom. Вони займають велику частку ринку та використовуються багатьма університетами по всьому світу.

Система Moodle має дуже широкий набір інструментів для викладачів та адміністраторів[1]. Програма дозволяє створювати складні тести та налаштовувати різні правила оцінювання. Інтерфейс платформи Moodle виглядає досить складно. На рисунку 1.1 представлено приклад екрану даної платформи.

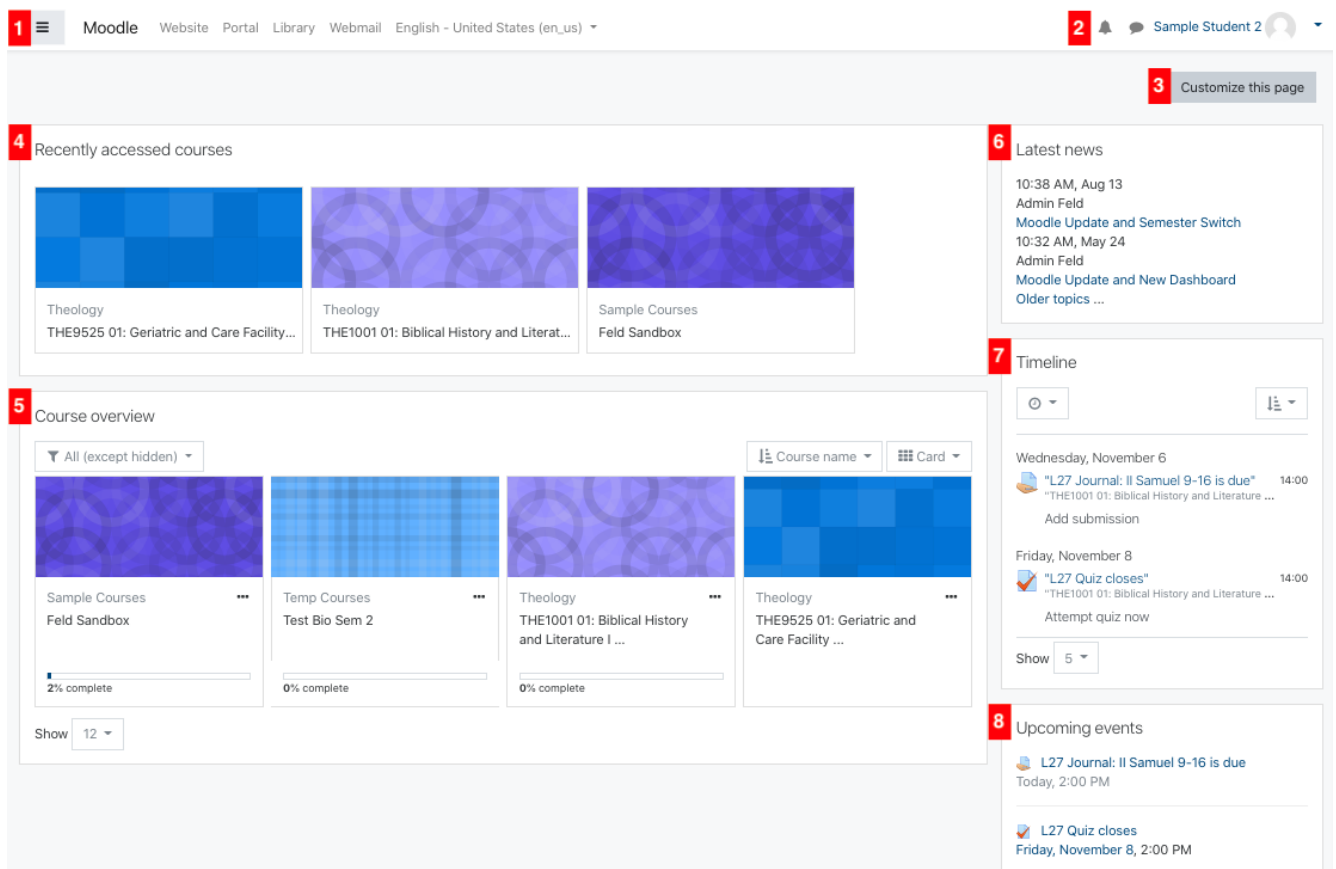


Рисунок 1.1 - Інтерфейс системи Moodle

Ця платформа працює на базі мови програмування PHP та вимагає власного сервера для встановлення. Багато готових сторінок мають перевантажений інтерфейс з десятками зайвих кнопок. Невелика школа програмування зовсім не потребує більшості цих складних функцій. Викладачі часто плутаються у великій кількості налаштувань під час створення нового курсу. Адміністрування Moodle вимагає наявності окремого технічного спеціаліста для налаштування плагінів. Монолітна архітектура системи робить процес внесення змін дуже повільним та дорогим.

Платформа Google Classroom пропонує простіший підхід до організації дистанційного навчання. Інтерфейс програми виглядає набагато чистіше та зрозуміліше для нових користувачів [2]. Зовнішній вигляд Google Classroom представлено на рисунку 1.2.

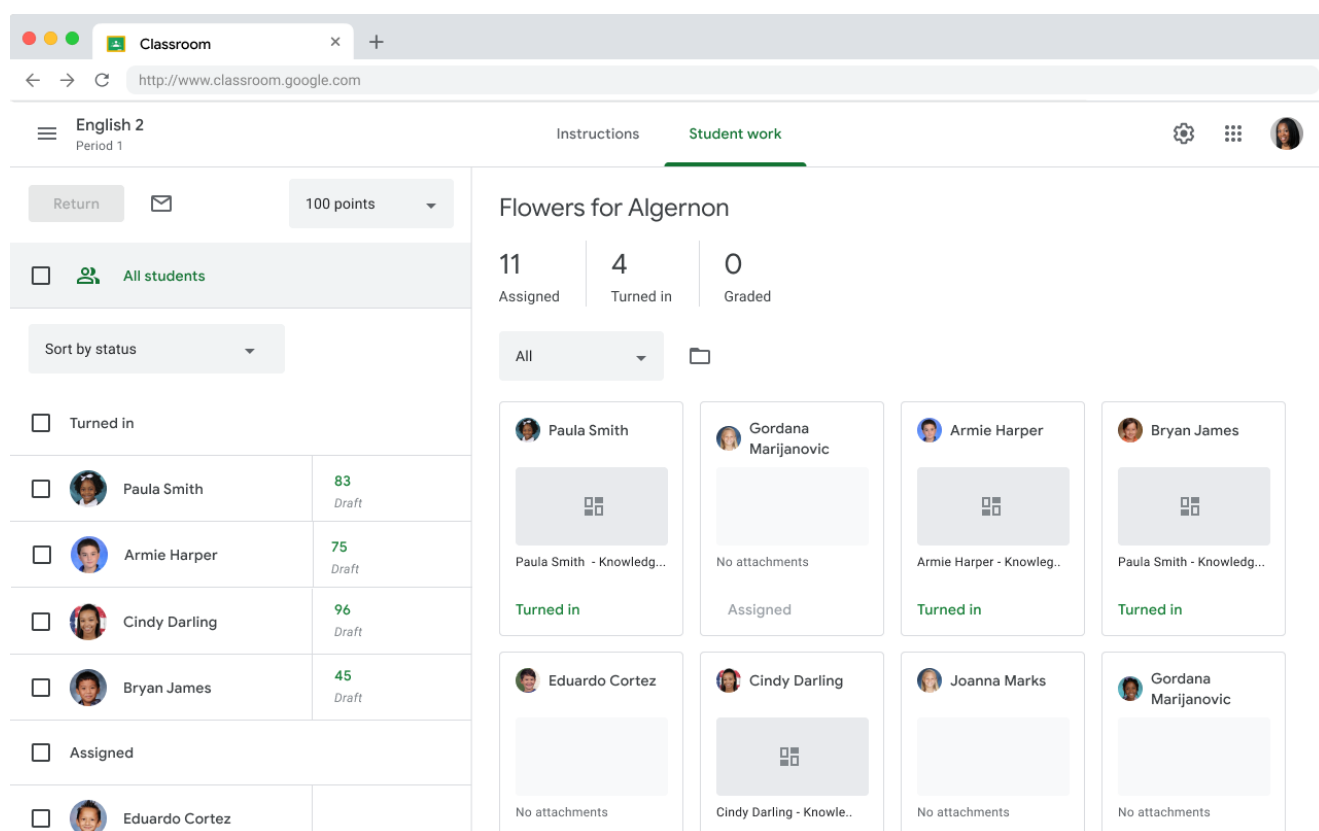


Рисунок 1.2 - Зовнішній вигляд Google Classroom

Цей хмарний сервіс добре підходить для звичайних загальноосвітніх шкіл. Проте система має суворі обмеження для специфічних курсів програмування.

Викладачі часто потребують можливості зручно перевіряти написаний код безпосередньо у браузері. Google Classroom працює як замкнута екосистема з обмеженим доступом до внутрішнього програмного інтерфейсу. Школа не може змінити логіку роботи електронного журналу під власні правила оцінювання.

Деякі інші платформи вимагають дорогої платної підписки для отримання повного доступу. Готові комерційні системи важко адаптувати під конкретні потреби невеликої школи інформаційних технологій. Виникає чітка необхідність розробки власної легкої системи на базі сучасного стеку технологій.

Власна програма дозволить гнучко змінювати розклад занять та швидко завантажувати виконані домашні завдання. Використання бібліотеки React [3] для створення інтерфейсу зробить роботу користувачів швидкою та плавною. Серверна частина на базі середовища Node.js зможе ефективно обробляти запити від багатьох студентів одночасно.

Збереження великих файлів у хмарному сховищі Cloudinary розвантажить основний сервер бази даних. Власна розробка повністю вирішує проблему залежності від сторонніх платних сервісів. Код програми належатиме школі, що дає повний контроль над даними користувачів.

1.2 Технічне завдання

1.2.1 Найменування та область застосування

Назва майбутнього програмного продукту звучить як Mriya IT. Програма створюється для використання у сфері приватної освіти. Цільовою аудиторією виступають невеликі школи програмування та різноманітні дизайнерські курси. Система допоможе зручно організувати весь навчальний процес онлайн. Студенти отримають прямий доступ до навчальних матеріалів з будь-якого комп'ютера або сучасного планшета. Викладачі зможуть проводити заняття дистанційно та контролювати успішність кожної окремої групи. Адміністрація школи отримає потужний інструмент для керування розкладом та оплатою.

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		12

1.2.2 Призначення розробки

Головна мета створення продукту полягає в об'єднанні всіх студентів та викладачів у єдиному віртуальному просторі. Кожен користувач проходить процес авторизації для входу у систему. Студенти заходять у свій особистий кабінет для перегляду розкладу занять. Вони читають конспект поточного уроку та переглядають прикріплені відеоматеріали. Після виконання практичного завдання студенти завантажують файли зі своїми звітами на сервер.

Викладачі відкривають ці завантажені файли безпосередньо через вікно браузера. Вони перевіряють виконану роботу та виставляють числові оцінки у загальний електронний журнал. Система дозволяє викладачам залишати текстові коментарі до кожної перевіреної роботи. Адміністратори самостійно керують базою користувачів через окрему панель. Вони створюють нові навчальні групи та призначають викладачів на конкретні курси. Програма максимально автоматизує рутинну роботу персоналу школи.

1.2.3 Вимоги до функціоналу інформаційної системи

Запланований функціонал повинен повністю покривати щоденні потреби навчального закладу. Перелік основних можливостей майбутньої платформи включає наступні обов'язкові пункти:

- Реєстрація та авторизація користувачів з різними правами доступу.
- Захист приватних маршрутів за допомогою надійних токенів авторизації.
- Створення нових курсів з поділом на окремі логічні модулі та уроки.
- Завантаження графічних обкладинок для курсів у хмарне сховище.
- Формування груп студентів та прив'язка створених груп до конкретного курсу.
- Ведення електронного журналу оцінок та постійний облік присутності студентів на заняттях.

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		13

- Масове швидке виставлення оцінок для всієї навчальної групи одночасно.
- Завантаження великих медіафайлів та студентських звітів через зовнішнє сховище Cloudinary.
- Вбудована підтримка формату документів для зручного читання звітів.
- Відображення текстового контенту уроків з підтримкою розмітки формату Markdown.
- Автоматичне підсвічування синтаксису програмного коду в текстах уроків.
- Система швидких внутрішніх сповіщень про появу нових оцінок або завантажені роботи.
- Перегляд загальної статистики успішності на головній сторінці інформаційної панелі.

Кожна згадана функція вимагає написання окремих програмних модулів на стороні сервера. База даних повинна надійно зберігати всю інформацію про зареєстрованих користувачів та їхні оцінки. Технологія захищених токенів допоможе безпечно ідентифікувати користувачів під час кожного нового запиту до сервера.

1.2.4 Вимоги до програмної документації

Процес розробки обов'язково супроводжується створенням необхідних документів для підтримки програми. Програміст має підготувати детальну інструкцію користувача для кожної з трьох існуючих ролей. Студенти повинні чітко розуміти процес завантаження своїх звітів після виконання домашнього завдання. Викладачі потребують покрокової інструкції щодо роботи з електронним журналом оцінок.

Адміністратор системи отримує детальне керівництво для правильного розгортання сервера на хостингу. Текст керівництва містить опис усіх прихованих змінних середовища та конфігурацію підключення до бази даних. Код серверної

					2026.KBP.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		14

частини повинен обов'язково містити зрозумілі текстові коментарі біля складних функцій. Написання чистого структурованого коду допомагає іншим програмістам легко підтримувати готовий продукт у майбутньому. Документація форматів передачі даних дозволить швидко підключити окремий мобільний додаток при такій необхідності.

1.2.5 Техніко-економічні показники

Економічна доцільність створення власної платформи підтверджується конкретними фінансовими розрахунками. Використання готових комерційних рішень вимагає щомісячної або щорічної плати за кожного активного користувача системи. Розробка власної програми вимагає лише разових фінансових інвестицій у працю програмістів.

Орієнтовний час написання вихідного коду для базової версії складає 160 людино-годин.

Регулярні витрати включають лише оплату віртуального виділеного сервера та послуг хмарної бази даних.

Використання хмарної бази даних на початковому етапі розвитку може бути абсолютно безкоштовним завдяки базовим тарифам.

Таблиця техніко-економічних показників представлена в графічній частині (2026.KBP.122.423.22.00.00 ТБ).

1.2.6 Стадії та етапи розробки

Написання вихідного коду відбувається за чітко визначеною логічною послідовністю кроків. Спочатку проєктується структура бази даних та необхідні фізичні зв'язки між різними колекціями документів. Розробник створює строгі схеми даних для користувачів, навчальних курсів, розкладу уроків та студентських звітів. Правильна архітектура даних запобігає виникненню дублювання інформації в системі.

					2026.KBP.122.423.22.00.00 ПЗ	Арк.
						15
Зм.	Арк.	№ докум.	Підпис	Дата		

Далі починається активна розробка серверної частини на базі швидкого середовища Node.js. Програміст налаштовує безпечні маршрути обміну даними за допомогою популярної бібліотеки Express [4]. Логіка роботи контролерів отримує вхідні запити від браузера та безпосередньо взаємодіє з базою даних. Окремо створюються функції для перевірки прав доступу кожного користувача до вибраного маршруту.

Після готовності та перевірки сервера розробник переходить до створення клієнтської частини програми. Інтерфейс користувача будується за допомогою компонентів відомої бібліотеки React. Програміст верстає сторінки на основі затверджених макетів візуального дизайну. Набір стилів Tailwind CSS допомагає швидко налаштовувати зовнішній вигляд елементів інтерфейсу без написання довгого коду. Інструмент Redux відповідає за надійне збереження стану програми під час роботи користувача.

На фінальному етапі створення продукту автор об'єднує обидві частини проєкту разом. Налаштовується безпечна система авторизації та передача токенів доступу при кожному запиті. Програміст перевіряє швидкість завантаження сторінок та зменшує розмір графічних файлів.

1.2.7 Порядок контролю та прийому

Процес перевірки та фінального прийому освітньої платформи Mriya IT передбачає виконання кількох послідовних кроків. Кожен етап допомагає вчасно виявити приховані програмні дефекти перед початком реального використання сайту. Порядок проведення тестування містить чітку послідовність дій.

Першим кроком потрібно розпочати роботу з повного функціонального тестування всіх можливостей розробленої системи. Необхідно перевірити правильність роботи процесів реєстрації нових користувачів, створення навчальних груп та виставлення оцінок у журнал.

Другим етапом слід перевірити сумісність платформи з різними сучасними мобільними та настільними пристроями. Сайт повинен завжди коректно та

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		16

швидко відображатися на екранах будь-якого доступного розміру. Потрібно відкрити основні сторінки платформи у різних популярних браузерах на мобільних телефонах та потужних комп'ютерах.

Далі необхідно провести детальну оцінку загальної продуктивності та стабільності роботи програми. Слід точно виміряти час завантаження сторінок з великою кількістю збережених даних. Правильне налаштування бази даних допомагає суттєво зменшити час очікування відповіді від сервера.

Четвертим кроком потрібно виконати ретельну перевірку надійності безпеки готового веб-сайту. Слід детально аналізувати правильність механізму обмеження прав доступу для різних типів облікових записів. Звичайний студент ніколи не зможе самостійно відкрити сторінку керування курсами або змінити чужу оцінку.

На п'ятому етапі необхідно зібрати відгуки від невеликої групи реальних користувачів.

Шостим кроком замовники проводять фінальний прийом готового програмного продукту. Вони ретельно перевіряють готову систему на повну відповідність початковим вимогам технічного завдання.

Для успішного захисту роботи потрібно представити членам комісії кілька обов'язкових елементів:

- повністю робоча програма з налаштованим віддаленим сервером
- вихідний програмний код проєкту на фізичному носії інформації

Спеціальна комісія з декількох осіб проводить процедуру фінального прийому платформи Мгіуа ІТ. Потрібно пройти всі етапи процедури у наступній визначеній послідовності.

- усна доповідь виконавця про успішно виконану роботу
- наочна демонстрація роботи вебсайту безпосередньо у вікні браузера
- контрольні випробування швидкості завантаження файлів студентських звітів
- відповіді розробника на поставлені запитання членів комісії. [3]

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
						17
Зм.	Арк.	№ докум.	Підпис	Дата		

2 РОЗРОБКА ТЕХНІЧНОГО ТА РОБОЧОГО ПРОЄКТУ

2.1 Розробка структури інформаційної системи

На початковому етапі проектування головною метою створення платформи «Mriya IT» було визначено забезпечення якісного дистанційного навчання для майбутніх фахівців у галузі інформаційних технологій. Завдяки впровадженню трьох рівнів доступу для студентів, викладачів та адміністраторів вдалося чітко розмежувати права користувачів на перегляд та редагування навчальних матеріалів.

Під час розробки враховано особливості кожної ролі. Під час детального аналізу технічного завдання встановлено необхідність створення гнучкого інтерфейсу, який би дозволяв легко масштабувати кількість курсів без втрати швидкості роботи.

В основу системи покладено модульний принцип, а для системного розуміння всіх процесів взаємодії між учасниками освітнього процесу та програмними модулями було розроблено відповідну модель поведінки. Взаємодію основних груп користувачів із функціональними блоками системи наочно представлено в графічній частині кваліфікаційної роботи 2026.КВР.122.423.22.00.00 ДВ.

На цій схемі можна побачити, що студенти мають доступ до базових сервісів навчання, таких як завантаження звітів та перегляд оцінок.

Викладачі наділені ширшими повноваженнями в системі. Зокрема, ними здійснюється масове виставлення балів та перевірка виконаних студентами робіт через спеціальну панель керування контентом курсів.

Найвищий рівень доступу має адміністратор ресурсу. До його обов'язків належить реєстрація нових користувачів, формування навчальних груп та глобальне налаштування параметрів усієї програмної платформи.

Щоб забезпечити логічний зв'язок між усіма розділами ресурсу, розроблено

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
						18
Зм.	Арк.	№ докум.	Підпис	Дата		

детальну карту сайту з описом структури кожної окремої сторінки та форматів даних. Загальну ієрархію всіх сторінок клієнтської частини відображено на плакаті 2026.KBP.122.423.22.00.00 СС.

На розробленій схемі можна чітко простежити поділ на публічний сегмент для гостей та закриту внутрішню панель для зареєстрованих учасників освітнього процесу. Така структура є цілком виправданою.

В межах публічної частини основну увагу зосереджено на презентації доступних програм навчання, публікації останніх новий академії та зборі відгуків від випускників курсів.

Внутрішній блок керування містить складні інструменти. Для реалізації зручної навігації обрано комбіновану модель меню, де верхня панель слугує для швидких переходів, а бокова панель забезпечує роботу в кабінеті.

Цей підхід суттєво полегшує пошук потрібного уроку. Через великий обсяг функціоналу особливу увагу приділено ергономіці робочого місця викладача, що включає можливість швидкої перевірки домашніх завдань та виставлення оцінок.

Система автоматично генерує звіти про успішність. В основу візуальної концепції проєкту покладено прагнення до мінімалізму, щоб ніякі зайві елементи дизайну не відволікали студента від засвоєння нових знань.

В ході розробки макетів головної сторінки було прийнято рішення відмовитися від складних графічних ефектів, обрано спокійну кольорову гаму з перевагою синіх та білих відтінків, які традиційно асоціюються з технологіями. А завдяки використанню шрифту Inter забезпечено високу читабельність текстових блоків на будь-яких пристроях. Всі макети було адаптовано до екранів мобільних пристроїв.

Також було детально опрацьовано інтерфейс перегляду лекційних матеріалів, оскільки саме на цій сторінці користувачі проводитимуть найбільшу кількість свого часу під час навчання. Для наочності на рисунку 2.1 представлено макет сторінки уроку.

					2026.KBP.122.423.22.00.00 ПЗ	Арк.
						19
Зм.	Арк.	№ докум.	Підпис	Дата		

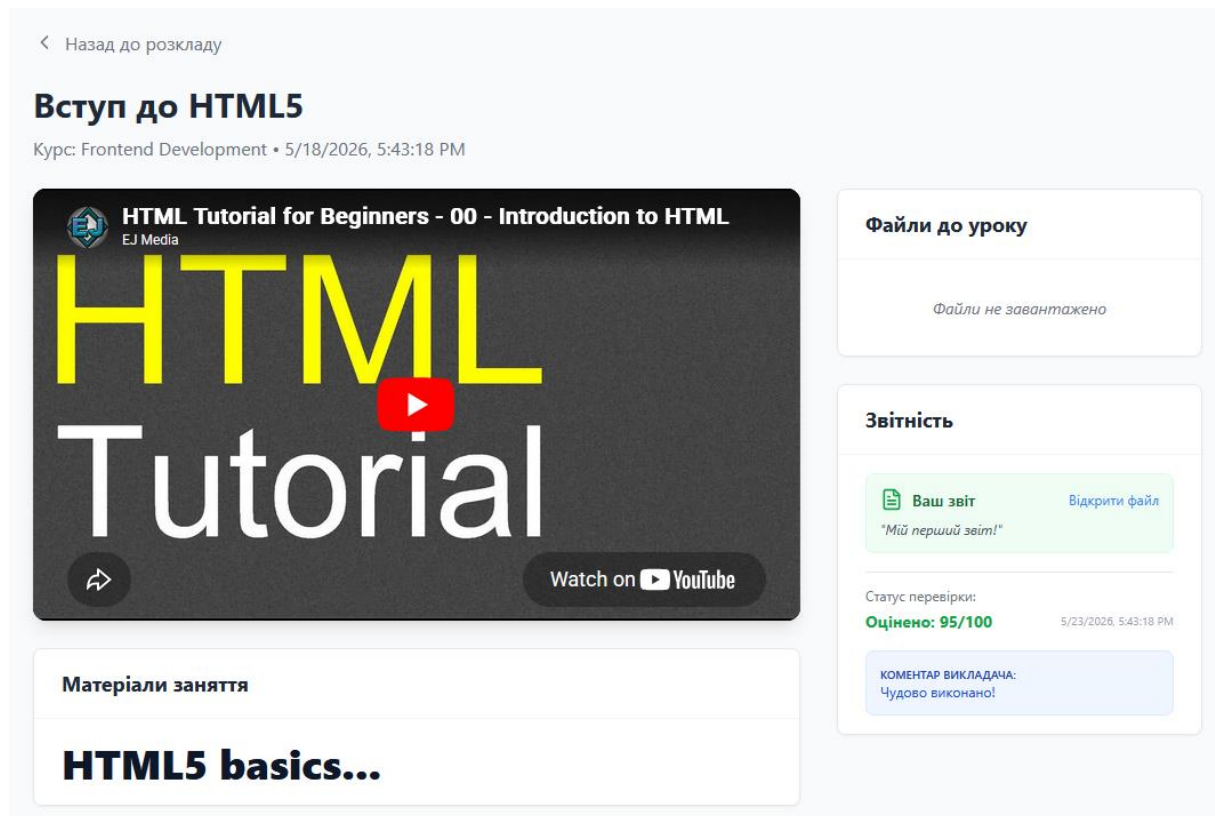


Рисунок 2.1 – Дизайн-макет сторінки перегляду навчального контенту

На макеті видно, що центральне місце займає плеєр для відеоматеріалів, а допоміжні текстові блоки та файли для завантаження розташовані у зоні периферійного зору. Це дозволяє зосередитися на головному змісті. Задля підвищення швидкості взаємодії впроваджено систему миттєвих сповіщень про нові коментарі викладача або зміну статусу перевірки звіту в особистому кабінеті студента.

У результаті виконання цього етапу робіт отримано повний перелік затверджених візуальних рішень, які стали основою для подальшої технічної верстки та програмування функцій.

2.2 Створення та верстка сторінок інформаційної системи

Під час розробки клієнтської частини навчальної платформи «Mriya IT» було обрано сучасний стек технологій на основі бібліотеки React та інструменту збірки Vite [5].

					2026.KBP.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		20

Для ініціалізації проекту було виконано команду створення додатку, що дозволило отримати оптимальну початкову структуру файлів для подальшого масштабування всієї системи.

Зокрема, початкову генерацію проекту представлено на рисунку 2.2.

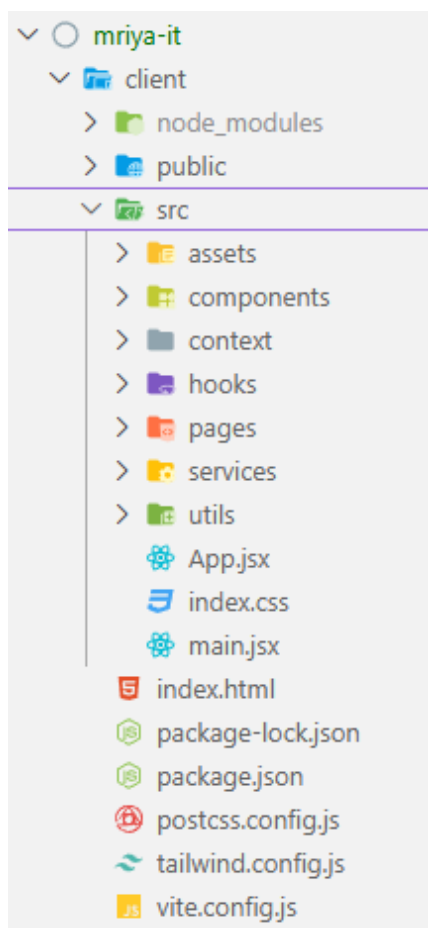


Рисунок 2.2 – Початкова структура проекту клієнтської частини «Mriya IT»

На поданій схемі можна побачити стандартний поділ на папку вихідних кодів `src`, статичні ресурси `public` та конфігураційні файли інструментів розробки.

Вказана архітектура дозволяє легко керувати кодом. В основу візуальної частини проекту покладено використання фреймворку Tailwind CSS, який дозволяє створювати адаптивні інтерфейси через систему спеціальних утилітарних класів.

Застосовані компоненти виглядають досить привабливо. Для керування глобальним станом додатку, зокрема даними про авторизацію та сповіщення, було

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		21

впроваджено бібліотеку Redux Toolkit разом із інструментами React Redux. Це гарантує цілісність усіх даних.

Зокрема, велику увагу при розробці було приділено створенню багаторазових компонентів інтерфейсу, таких як модальні вікна, кнопки та форми вводу інформації.

Кожен елемент має чітку внутрішню логіку. Для реалізації навігації між розділами кабінету та публічними сторінками використано пакет React Router, що забезпечує швидке перемикавання контенту без оновлення сторінки.

Маршрутизація працює дуже плавно. Завдяки впровадженню контекстного API [6] вдалося реалізувати гнучку систему перевірки прав доступу, яка автоматично перенаправляє неавторизованих відвідувачів на головну сторінку для входу.

Такий підхід суттєво підвищує безпеку. В папці public розміщено головний індексний файл index.html, який слугує вхідною точкою для всього додатку через спеціальний контейнер із ідентифікатором root.

Саме структуру цього файлу подано на лістингу 2.1.

```
<!doctype html>
<html lang="uk">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Mriya IT</title>
  </head>
  <body>
    <div id="root"></div>
    <script type="module" src="/src/main.jsx"></script>
  </body>
</html>
```

Лістинг 2.1 – Структура головного файлу index.html клієнтської частини

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
						22
Зм.	Арк.	№ докум.	Підпис	Дата		

Після завантаження цього файлу браузером починається виконання основного сценарію, що відповідає за рендеринг всього дерева компонентів платформи у вказаний блок.

Уся обробка відбувається майже миттєво. Шляхом використання бібліотеки Axios було налагоджено стабільний обмін даними з серверною частиною через HTTP-запити для отримання списків курсів та результатів навчання.

Всі відповіді обробляються згідно зі стандартами. У результаті виконання цього етапу було отримано повністю працездатний клієнтський додаток, який забезпечує зручну взаємодію користувача з усіма функціями освітньої платформи.

Верстка пройшла успішну перевірку на мобільність.

2.3 Розробка структури бази даних інформаційної системи

Для забезпечення надійного зберігання та швидкої обробки великих обсягів навчальної інформації у проекті «Mriya IT» було обрано нереляційну систему керування базами даних MongoDB [7].

Завдяки документо-орієнтованій моделі вдалося досягти гнучкості у представленні складних ієрархічних структур навчальних матеріалів без необхідності виконання складних операцій об'єднання таблиць. Дані зберігаються у форматі BSON.

Під час проектування архітектури сховища основну увагу було зосереджено на визначенні оптимального складу колекцій, які б максимально точно описували предметну область освітнього процесу.

Загальну логічну схему організації даних та основні зв'язки між усіма елементами системи наочно представлено на плакаті (2026.KBP.122.423.22.00.00 БД) структури бази даних навчальної платформи «Mriya IT»

На поданій діаграмі можна побачити, що в основу архітектури покладено принцип розділення даних на кілька ключових функціональних блоків. Така декомпозиція значно спрощує розробку.

Зокрема, для обліку всіх зареєстрованих учасників освітнього процесу

					2026.KBP.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		23

розроблено колекцію User, яка містить персональні дані, зашифровані паролі та ролі доступу.

Для опису навчального контенту використовуються взаємопов'язані колекції Course та Lesson, де кожен окремий урок містить посилання на свій батьківський навчальний курс. Це забезпечує цілісність усієї інформації.

У межах колекції Group здійснюється логічне об'єднання студентів у окремі навчальні потоки із обов'язковим закріпленням за кожною групою відповідної освітньої програми. Структура дозволяє масштабувати кількість груп.

Окреме місце в ієрархії посідає колекція Report, яка слугує для фіксації результатів виконання практичних робіт та зберігання зворотного зв'язку від викладачів. Кожен документ тут містить посилання.

Для забезпечення оперативного інформування користувачів про нові події впроваджено колекцію Notification, що дозволяє відстежувати статус прочитання кожного окремого системного повідомлення. Механізм працює через унікальні ідентифікатори.

Нижче наведено детальний опис основних полів ключових колекцій бази даних проекту:

1) Колекція User (Користувачі) зберігає дані про всіх учасників системи: студентів, викладачів та адміністраторів школі містить наступні поля:

- fullName (String) – повне ім'я користувача для відображення в профілі, списках груп та сертифікатах;
- email (String) – унікальна адреса для входу в систему (логін) та отримання сповіщень;
- passwordHash (String) – зашифрований пароль для безпечного доступу до облікового запису;
- role (String) – роль (student, teacher, admin). Визначає права доступу до функцій системи;
- avatarUrl (String) – посилання на зображення профілю для персоналізації інтерфейсу;

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
						24
Зм.	Арк.	№ докум.	Підпис	Дата		

– timestamps – автоматичні поля createdAt та updatedAt для відстеження дати реєстрації;

2) Колекція Course (Курси) зберігає інформацію про навчальні програми, доступні в ІТ-школі містить наступні поля:

– title (String) – назва курсу (наприклад, "Frontend Development");

– description (String) – детальний опис курсу, що пояснює студентам, чого вони навчаться;

– teacher (ObjectId) – зв'язок із таблицею User. Вказує, який викладач відповідальний за цей курс;

– category (String) – категорія (Програмування, Дизайн тощо) для зручної фільтрації на сайті;

– imageUrl (String) – обкладинка курсу для візуального оформлення в каталозі;

– modules (Array) – список основних тем або модулів, які входять до програми курсу;

– rating (Number) – середній бал курсу, що відображає його популярність та якість;

– price (String) – вартість курсу для відображення на публічних сторінках.

3) Колекція Lesson (Заняття) зберігає кожне заняття прив'язане до певного курсу і містить навчальні матеріали школі, колекція містить наступні поля:

– course (ObjectId) – зв'язок із Course. Вказує, до якої програми належить це заняття;

– title (String) – тема конкретного уроку;

– content (String) – основний текст заняття (може містити Markdown-розмітку) ;

– videoUrl (String) – посилання на відеолекцію (наприклад, з YouTube або Cloudinary) ;

– attachments (Array) – список файлів (PDF, презентації, приклади коду),

					2026.KBP.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		25

доступних для завантаження;

- order (Number) – порядковий номер уроку для правильної черговості вивчення матеріалу;

- scheduledDate (Date) – дата проведення заняття для відображення в розкладі групи;

4) Колекція Report (Звіти/ДЗ) фіксує результати виконання практичних завдань студентами.

- student (ObjectId) – зв'язок із User. Хто саме здав роботу;
- lesson (ObjectId) – зв'язок із Lesson. До якого саме заняття цей звіт;
- fileUrl (String) – посилання на завантажений файл із виконаним завданням (архів, PDF) ;

- fileName (String) – оригінальна назва файлу для зручності викладача;
- studentComment (String) – пояснення студента до своєї роботи;
- teacherComment (String) – фідбек (відгук) викладача після перевірки роботи;

- score (Number) – оцінка за завдання (від 0 до 100 балів) ;
- graded (Boolean) – прапорець "Перевірено". Дозволяє студенту бачити, що робота оцінена;

- gradeDate (Date) – дата, коли викладач виставив оцінку;

5) Додатково використовуються допоміжні колекції:

- Group – для об'єднання студентів у потоки навчання;
- Grade – для ведення загального журналу успішності;
- News – для публікації оголошень школи;
- Testimonial – для відгуків користувачів про навчання;
- Notification – для надсилання системних сповіщень.

Зокрема, для наповнення публічної частини ресурсу було розроблено колекції News та Testimonials, які дозволяють динамічно оновлювати вміст головної сторінки академії. Це робить систему більш гнучкою.

Шляхом використання індексів за ключовими полями вдалося забезпечити

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		26

високу швидкість пошуку документів навіть за умови значного зростання кількості зареєстрованих на платформі студентів. Всі запити виконуються максимально ефективно.

У результаті завершення цього етапу робіт було отримано детальну специфікацію структури даних, що стала фундаментом для подальшої реалізації серверної логіки та інтерфейсів взаємодії. База даних готова до інтеграції.

2.4 Програмування інформаційної системи

2.4.1 Написання клієнтської частини

На початковому етапі розробки клієнтського додатку у файлі `src/main.jsx` здійснюється підключення кореневого компонента до контейнера в `index.html` для ініціалізації всього дерева інтерфейсу. Це забезпечує запуск платформи.

Для забезпечення глобального доступу до стану авторизації та навігації основний компонент обгортається у відповідні провайдери контексту, як це продемонстровано на лістингу 2.2.

```
import App from './App.jsx'

ReactDOM.createRoot(document.getElementById('root')).render(

  <React.StrictMode>
    <BrowserRouter>
      <AuthProvider>
        <App />
      </AuthProvider>
    </BrowserRouter>
  </React.StrictMode>,
)
```

Лістинг 2.2 – Підключення кореневого компонента у файлі `main.jsx`

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		27

Під час побудови архітектури було обрано компонентний підхід, де кожен елемент інтерфейсу розглядається як незалежна одиниця із власним станом та логікою відображення. Така структура полегшує масштабування.

Завдяки використанню синтаксису JSX [8] логіка керування даними поєднується із декларативним описом розмітки, що дозволяє створювати динамічні шаблони для відображення навчального контенту. Дані передаються через вхідні параметри.

Для чіткої організації програмного коду в проєкті «Mriya IT» впроваджено модульну структуру каталогів, яку представлено на рисунку 2.3.

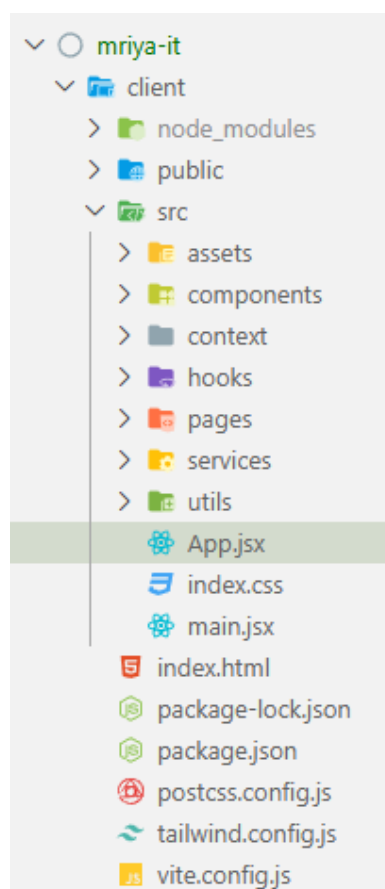


Рисунок 2.3 – Структура каталогів клієнтської частини додатка

В основу побудови додатку покладено поділ на функціональні зони. Зокрема, у папці «pages» розміщуються повноцінні сторінки ресурсу, а папка «components» містить дрібні атомарні елементи, такі як кнопки, поля вводу та картки. Це зменшує дублювання коду.

Керування маршрутами здійснюється у компоненті App.jsx, де за допомогою бібліотеки React Router визначено перелік доступних адрес та правил доступу до них. Конфігурацію маршрутизації подано на лістингу 2.3.

```
function App() {
  return (
    <div className="min-h-screen bg-gray-50">
      <LoginModal />
      <Routes>
        <Route path="/" element={<Landing />} />
        <Route path="/public-courses" element={<PublicCourses />} />
        <Route path="/public-courses/:id" element={<PublicCourseDetails />} />
      </Routes>
      <Route path="/public-news" element={<PublicNews />} />
      <Route path="/news/:id" element={<NewsDetails />} />
      <Route path="/testimonials" element={<Testimonials />} />
      <Route
        path="/dashboard"
        element={
          <ProtectedRoute>
            <Dashboard />
          </ProtectedRoute>
        }
      />
    </div>
  );
}
```

...

Лістинг 2.3 – Структура компонента App.jsx із налаштуванням маршрутів

Повний лістинг коду файлу App.jsx подано в додатку А.

Під час відвідування головної сторінки неавторизованим користувачем відображається лендінг із презентацією курсів та відгуками, що дозволяє ознайомитися із можливостями академії. Інтерфейс головної сторінки показано далі на рисунку 2.4.

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
						29
Зм.	Арк.	№ докум.	Підпис	Дата		

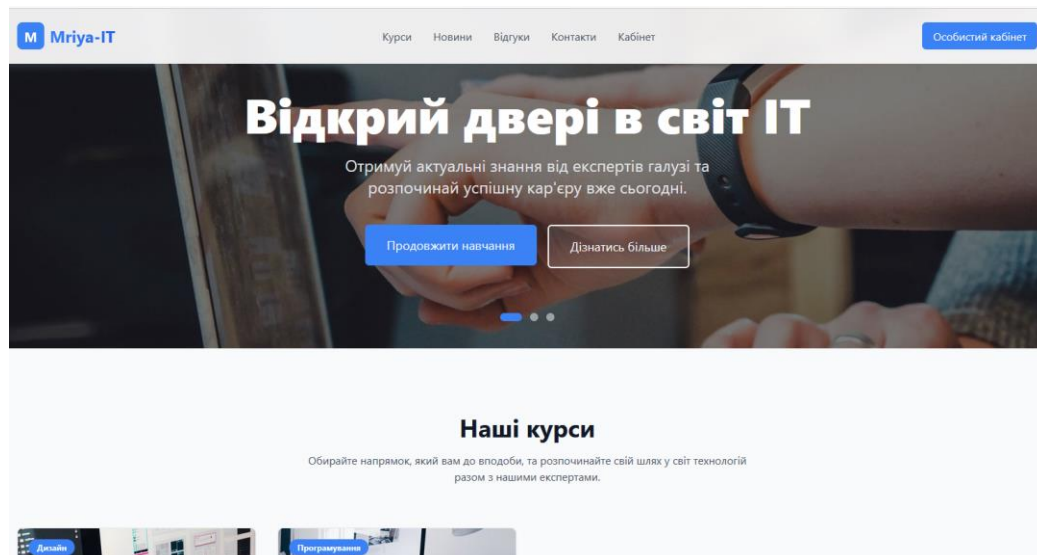


Рисунок 2.4 – Зовнішній вигляд головної сторінки (Landing)

Для отримання доступу до навчальних матеріалів відвідувачу необхідно пройти процес ідентифікації через компонент LoginModal.jsx, який викликається за допомогою контексту авторизації. Вікно входу представлено на рисунку 2.5.

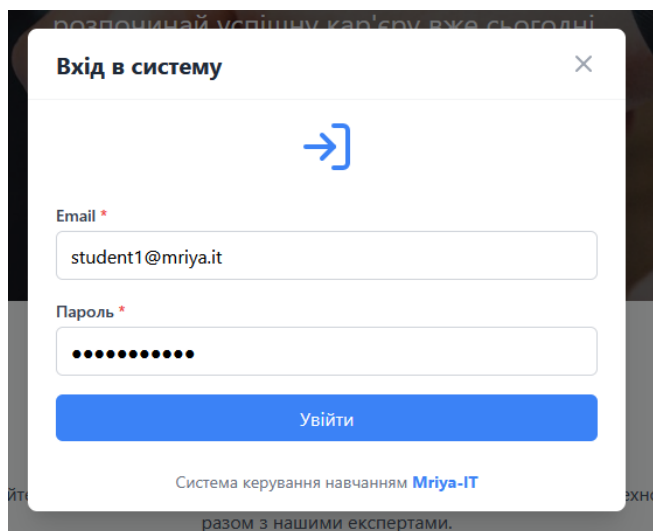


Рисунок 2.5 – Компонент LoginModal для авторизації користувачів

Лістинг коду форми авторизації подано в додатку Б.

Після успішного входу система автоматично перенаправляє студента на персональну панель керування, реалізовану в компоненті Dashboard.jsx, де відображається статистика успішності та розклад. Вигляд кабінету студента

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
						30
Зм.	Арк.	№ докум.	Підпис	Дата		

подано далі на рисунку 2.6.

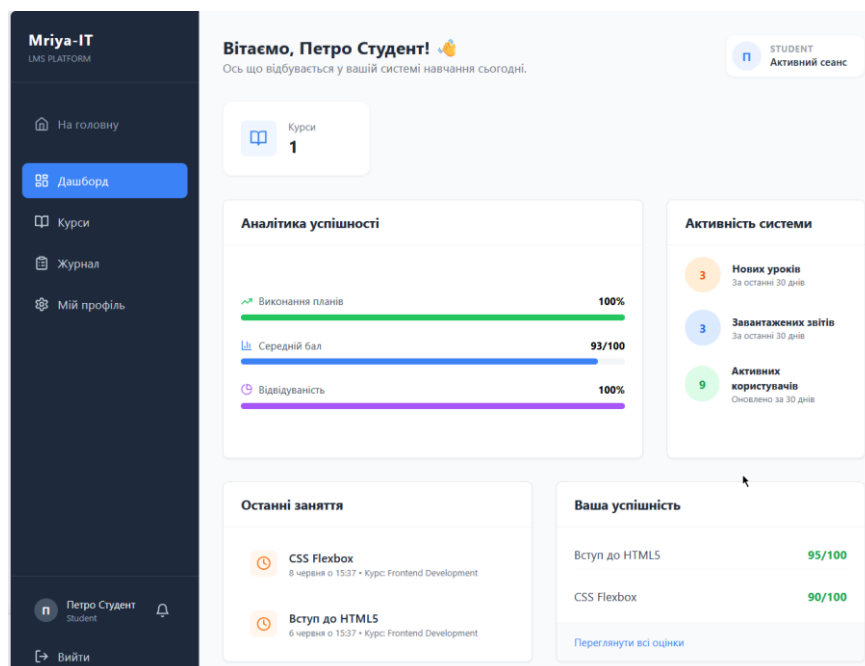


Рисунок 2.6 – Панель (Dashboard) авторизованого студента

Зокрема, для перегляду навчальних матеріалів використовується компонент LessonDetails.jsx, який інтегрує відеоплеєр, текстовий контент у форматі Markdown та модуль завантаження звітів. Сторінку уроку зображено на рисунку 2.7. Лістинг коду сторінки уроку подано в додатку В.



Рисунок 2.7 – Інтерфейс перегляду навчального заняття та здачі звіту

У разі авторизації користувача з роллю викладача стають доступними специфічні інструменти для перевірки виконаних завдань, що реалізовані у компоненті Reports.jsx. Процес перевірки показано на рисунку 2.8.

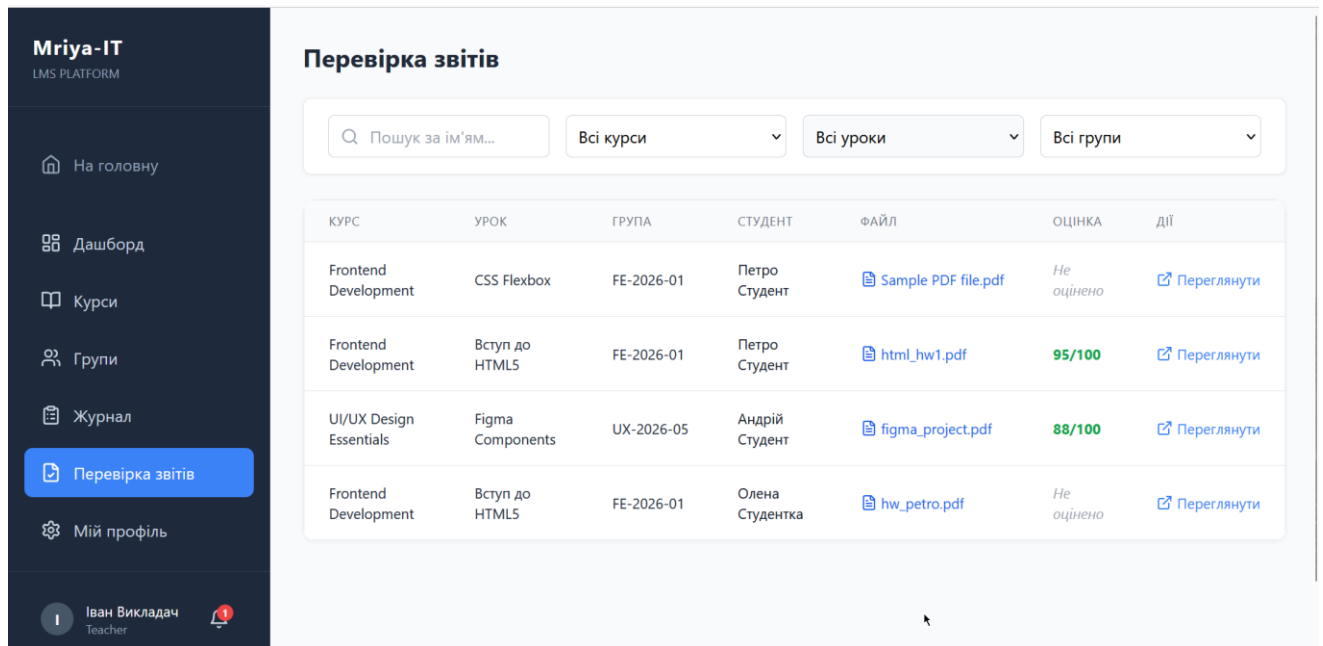


Рисунок 2.8 – Інформаційна панель викладача для перевірки звітів

Для адміністратора системи передбачено розширений функціонал керування користувачами та групами, доступний через компоненти Users.jsx та Groups.jsx відповідно. Вигляд панелі керування користувачами та групами представлено на рисунку 2.9.

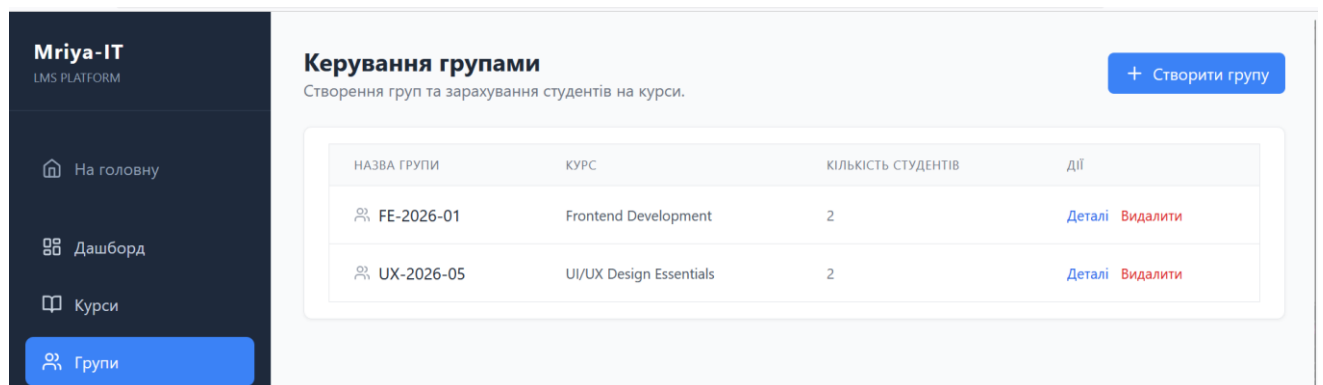


Рисунок 2.9 – Інтерфейс керування користувачами для адміністратора

Шляхом використання бібліотеки Redux Toolkit у проекті налагоджено ефективне керування глобальними станами, що дозволяє миттєво оновлювати сповіщення та дані профілю без перезавантаження сторінок. Такий підхід забезпечує високу швидкість взаємодії.

У результаті виконання робіт із написання клієнтської частини було створено інтуїтивно зрозумілий та адаптивний інтерфейс, який повністю відповідає функціональним вимогам до LMS платформи. Розробку клієнтського модуля успішно завершено.

2.4.2 Написання серверної частини

Під час створення програмного ядра навчальної платформи «Mriya IT» було обрано популярне середовище виконання Node.js разом із легким та швидким фреймворком Express.js. Це забезпечує ріст системи.

Завдяки використанню моделі обробки запитів без очікування вдалося досягти високої швидкості сервера під час одночасної роботи багатьох людей із медіаконтентом.

В основу архітектури покладено модульність. Для розгортання робочого місця було проведено налаштування проекту та встановлено повний набір програм через пакетний менеджер для керування сторонніми бібліотеками. Всі параметри вказані у package.json.

Зокрема, головний файл налаштувань серверної частини із переліком усіх підключених модулів та правилами завантаження представлено на лістингу 2.4.

Окрім базової конфігурації, особливу увагу було приділено організації маршрутизації та захисту даних користувачів. Інтеграція проміжних шарів (middleware) у структуру Express.js дозволила автоматизувати процес валідації запитів, логування помилок у реальному часі, а також впровадити сучасні протоколи шифрування для безпечного обміну інформацією. Такий підхід не лише гарантує стабільність платформи «Mriya IT» під час пікових навантажень, але й створює надійний фундамент для подальшого масштабування та легкого

					2026.KBP.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		33

ДОДАВАННЯ НОВИХ НАВЧАЛЬНИХ МОДУЛІВ.

```
{
  "name": "server",
  "version": "1.0.0",
  "description": "Backend for Mriya IT LMS Platform",
  "main": "server.js",
  "scripts": {
    "start": "node server.js",
    "dev": "nodemon server.js"
  },
  "dependencies": {
    "bcryptjs": "^2.4.3",
    "cloudinary": "^1.41.3",
    "cors": "^2.8.5",
    "dotenv": "^16.4.5",
    "express": "^4.19.2",
    "jsonwebtoken": "^9.0.2",
    "mongodb": "^7.2.0",
    "mongoose": "^8.3.4",
    "multer": "^2.1.1",
    "multer-storage-cloudinary": "^4.0.0"
  },
  "devDependencies": {
    "nodemon": "^3.1.0"
  }
}
```

Лістинг 2.4 – Конфігурація модулів у файлі package.json

У процесі розробки серверної логіки було впроваджено бібліотеку Mongoose для ефективної взаємодії із базою даних через опис схем та моделей об'єктів. Це спрощує роботу з даними.

Для забезпечення надійного захисту облікових записів використано пакет

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		34

bcryptjs, який дозволяє перетворювати паролі у складні коди перед їх збереженням у сховище. Такий підхід виключає витік даних.

Особливу увагу приділено механізму перевірки осіб, який базується на використанні веб-токенів стандарту JWT [9] для створення захищених сесій без збереження стану. Система видає унікальний ключ. Під час кожної спроби входу сервер перевіряє надані дані та створює токен, що містить код особи та її роль у системі навчання. Цей ключ передається у заголовках.

Задля розмежування прав доступу до різних частин API було розроблено спеціальні проміжні обробники, які фільтрують запити на основі перевірки коду токена. Захист працює у реальному часі.

Зокрема, впроваджено рольову модель керування, де доступ до редагування курсів або видалення користувачів дозволено лише особам із правами адміністратора чи викладача. Це запобігає несанкціонованим діям.

Для логічного групування функцій платформи було створено розгалужену систему маршрутів, де кожен напрямок взаємодії винесено у окремий програмний модуль для зручності. Структура маршрутизатора є досить прозорою. Зокрема, внутрішню організацію файлів маршрутизації та їх підключення до основного додатку для обробки вхідних запитів продемонстровано на рисунку 2.10.

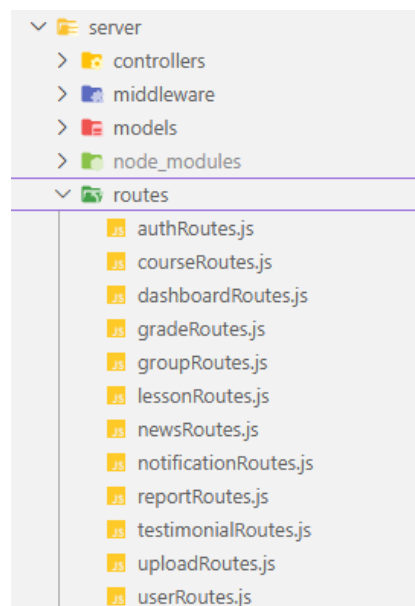


Рисунок 2.10 – Структура модулів маршрутизації серверної частини

В основу побудови API покладено поділ на такі основні блоки як вхід, керування навчальними курсами, робота з групами та система звітності.

Кожен маршрут має свій контролер.

Нижче наведено перелік ключових точок доступу серверної частини:

- /api/auth – реєстрація нових учасників, вхід до системи та отримання даних профілю поточного користувача;
- /api/courses – створення нових освітніх програм, перегляд переліку доступних напрямків та редагування описів курсів;
- /api/lessons – публікація навчальних матеріалів, додавання посилань на відеолекції та завантаження додаткових файлів;
- /api/reports – подання студентами звітів про виконання практичних робіт та виставлення оцінок з боку викладацького складу;
- /api/notifications – розсилка системних повідомлень про нові оцінки або зміни у розкладі занять через спеціальний центр сповіщень.

Лістинг коду головного файлу сервера поданий в додатку Г.

Під час обробки медіафайлів, таких як аватари користувачів чи обкладинки навчальних курсів, було реалізовано інтеграцію із хмарним сховищем Cloudinary через програмний інтерфейс.

Файли зберігаються на зовнішньому ресурсі. Завдяки використанню бібліотеки Multer сервер отримує завантажені документи, проводить їх попередню перевірку та автоматично пересилає у хмарний сервіс для оптимізації. Це зменшує навантаження на диск.

У результаті завершення робіт із програмування серверної частини було отримано стабільний програмний інтерфейс, що забезпечує повний цикл життєдіяльності навчального процесу. Всі компоненти працюють злагоджено. Створене програмне ядро дозволяє гнучко розширювати функціонал системи у майбутньому без необхідності проведення великих змін у існуючій архітектурі та логіці даних. Серверну частину повністю готово до роботи.

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
						36
Зм.	Арк.	№ докум.	Підпис	Дата		

2.5 Тестування інформаційної системи

Під час завершального етапу впровадження платформи «Mriya IT» було проведено комплексне функціональне тестування основних модулів для підтвердження повної відповідності розробленого програмного коду вимогам початкового технічного завдання. Усі заплановані заздалегідь випробування основних функцій відбувалися згідно із чітким планом.

У межах першого тестового сценарію було успішно реалізовано процедуру додавання нового студента до системи адміністратором через заповнення спеціальної модальної форми у розділі керування всіма активними користувачами. Створений запис про нову особу відразу з'явився у поточному загальному списку.

Зокрема, вікно заповнення персональних даних для ідентифікації майбутнього учасника навчального процесу наочно продемонстровано на рисунку 2.11.

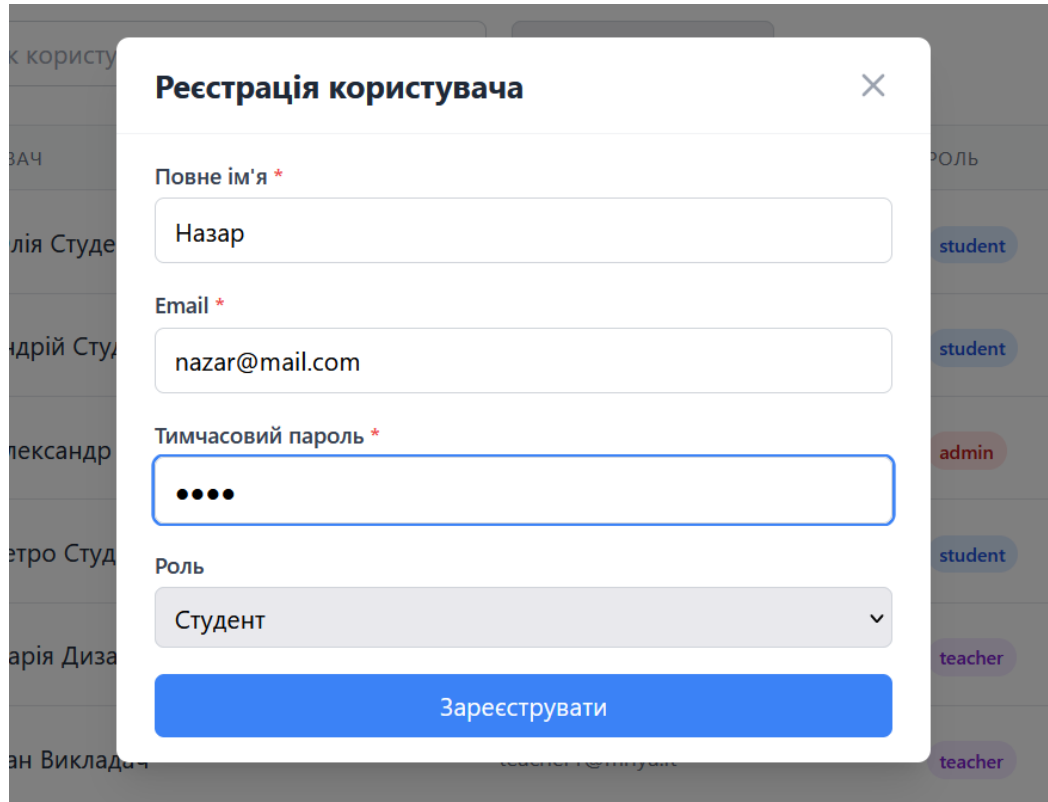


Рисунок 2.11 – Процес реєстрації нового студента адміністратором системи

Задля перевірки блоку наповнення платформи контентом було здійснено тестове формування нової освітньої програми із завантаженням опису та вибором відповідального викладача через панель керування курсами. Система опрацювала надісланий запит без жодних затримок та помилок.

Під час виконання цієї операції було використано функціонал редагування картки курсу, де вказано тривалість навчання, відповідну категорію та вартість програми для її візуальної презентації. Опис нового курсу з'явився у публічному каталозі для всіх гостей.

Результат створення нового напрямку навчання та його відображення у структурі сайту показано на рисунку 2.12.

The image shows a web application interface for creating a new course. A modal window titled "Створення нового курсу" (Create new course) is open over a blurred background of a website. The form contains the following elements:

- Назва курсу *** (Course name *): A text input field containing "Програмування веб-сайтів" (Web site programming).
- Обкладинка курсу** (Course cover): A section with a book icon, a button "Обрати файл" (Choose file), and a note "Рекомендовано: 1200x800px, до 5MB".
- Категорія** (Category): A dropdown menu with "Програмування" (Programming) selected.
- Викладач** (Instructor): A dropdown menu with "Іван Викладач" (Ivan Instructor) selected.
- Опис курсу** (Course description): A large text area with the placeholder "Опис....".
- Buttons:** "Скасувати" (Cancel) in a light blue button and "Створити курс" (Create course) in a dark blue button.

Рисунок 2.12 – Створення нового навчального курсу в адміністративній панелі

Щоб забезпечити повноцінне тестування ієрархії навчальних матеріалів, до щойно створеної освітньої програми було додано перший урок із прямою інтеграцією посилання на лекцію із зовнішнього відеоресурсу. Теоретичний матеріал став доступним для вивчення зареєстрованими студентами майже миттєво. У ході додавання заняття було окремо перевірено коректність обробки Markdown-розмітки для текстового блоку та можливість прикріплення додаткових файлів у форматі програмного коду для самостійної роботи.

Усі елементи відображаються коректно та зберігають власну початкову структуру.

Процес додавання нового заняття до обраної програми навчання представлено на рисунку 2.13.

Рисунок 2.13 – Додавання нового уроку до обраної програми навчання

З метою детального дослідження клієнтського шляху студента було проведено завантаження файлу звіту про виконання практичного завдання безпосередньо через інтерфейс особистого кабінету авторизованого учасника курсів. Файл було успішно надіслано на сервер для подальшої перевірки викладачем. Під час завантаження було додано короткий текстовий коментар до роботи та перевірено автоматичне створення сповіщення для відповідальної особи про необхідність термінового проведення оцінювання отриманих результатів.

Статус звіту в особистому кабінеті студента змінився на актуальний.

Зовнішній вигляд блоку звітності після відправки виконаної роботи продемонстровано на рисунку 2.14.

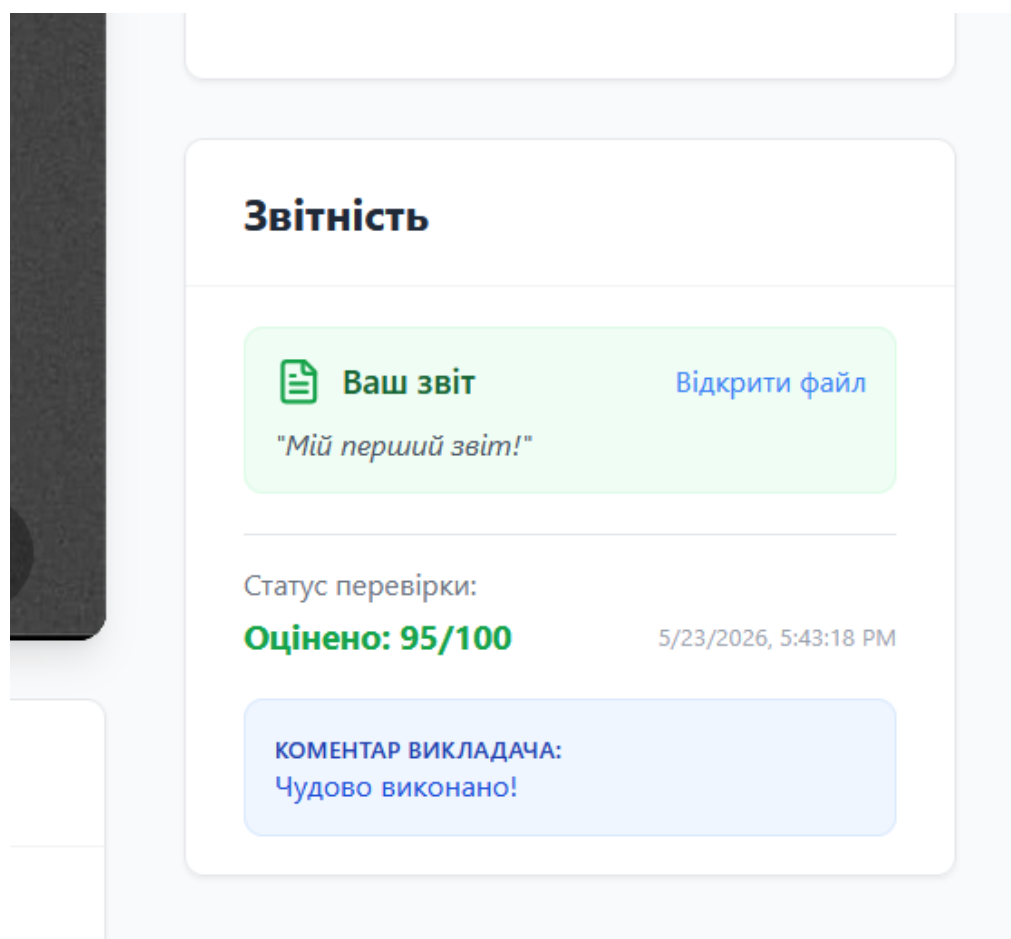


Рисунок 2.14 – Подача студентом звіту про виконання практичної роботи

Для логічного завершення тестового циклу було виконано вхід під обліковим записом викладача та проведено повне оцінювання отриманої роботи

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
						40
Зм.	Арк.	№ докум.	Підпис	Дата		

через спеціалізований модуль перевірки надісланих звітів. Виставлений бал відразу відобразився у загальному журналі успішності групи.

У процесі перевірки було вказано числовий результат за стобальною шкалою та написано розгорнутий відгук із рекомендаціями, який згодом автоматично з'явився на сторінці заняття у самого студента. Взаємодія між ролями пройшла згідно із передбаченою бізнес-логікою.

Процедуру оцінювання звіту та написання відгуку викладачем представлено на рисунку 2.15.

Перегляд звіту: Петро Студент

Sample PDF file

Коментар студента:
Мій перший звіт!

Ваш коментар (викладача):
Чудово виконано!

Виставити оцінку (0-100)
95

Зберегти оцінку

Рисунок 2.15 – Процес виставлення оцінки та написання коментаря викладачем

Окрім ручного функціонального тестування було запропоновано проведення автоматизованої перевірки серверних ендпоінтів за допомогою інструментальної платформи Postman для контролю цілісності структури всіх відповідей додатку. Цей метод є надзвичайно ефективним для пошуку прихованих помилок.

Завдяки надсиланню прямих HTTP-запитів до API вдалося встановити точність повернення даних у форматі JSON та перевірити правильність роботи кодів станів для несанкціонованих спроб доступу.

Бекенд частина продемонструвала високу стабільність під час інтенсивних запитів. У результаті проведених заходів було підтверджено повну готовність системи «Mriya IT» до експлуатації та відсутність критичних дефектів у основних функціональних модулях платформи.

Тестування завершено із позитивним результатом для всього проекту.

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		42

3 СПЕЦІАЛЬНИЙ РОЗДІЛ

3.1 Інструкція з розміщення інформаційної системи в Інтернеті

Для розміщення навчальної платформи Mriya IT у публічному доступі необхідно виконати покрокове розгортання її компонентів у хмарному середовищі. Через роздільну архітектуру проєкту процес розділяється на деплой серверної частини та розміщення клієнтського інтерфейсу.

Першим етапом є розгортання серверної частини інформаційної системи Node.js на платформі Render. Для цього виконайте такі послідовні дії.

- Спочатку зареєструйтеся або авторизуйтеся на сайті Render за допомогою свого облікового запису GitHub.
- У панелі керування натисніть кнопку створення нового ресурсу та оберіть пункт Web Service.
- Підключіть свій віддалений репозиторій із кодом, вибравши відповідне ім'я зі списку доступних проєктів. Візуально процес підключення репозиторію наведено на рисинку 3.1.

The screenshot shows the 'New Web Service' form in the Render dashboard. The 'Name' field is filled with 'mriya-it'. The 'Project' field is optional and has a dialog box asking to 'Create a new project to add this to?'. The 'Language' field is set to 'Node'. The 'Branch' field is set to 'main'.

Рисунок 3.1 – Процес підключення репозиторію для розгортання сервера

Після підключення репозиторію заповніть основні параметри проєкту,

вказавши назву сервісу та обравши найближчий географічний регіон. У полі команди запуску пропишіть команду запуску сервера.

Другим кроком налаштуйте змінні оточення для безпечного підключення бази даних та зовнішніх сервісів.

- Перейдіть до вкладки Environment у налаштуваннях створеного сервісу Render.
- Додайте нову змінну MONGO_URI та вкажіть рядок підключення до вашого хмарного кластера MongoDB Atlas.
- Створіть змінну JWT_SECRET та введіть секретний ключ для підпису токенів авторизації.
- Введіть параметри доступу до хмари Cloudinary для забезпечення збереження файлів, зокрема CLOUDINARY_CLOUD_NAME, CLOUDINARY_API_KEY та CLOUDINARY_API_SECRET. Наочно додавання цих параметрів зображено на рисунку 3.2.

Environment Variables

Set environment-specific config and secrets (such as API keys), then read those values from your code. [Learn more](#).

NAME_OF_VARIABLE	value	
PORT		Generate
MONGO_URI		Generate
JWT_SECRET		Generate
NODE_ENV		Generate
CLOUDINARY_CLOUD_NAME		Generate
CLOUDINARY_API_KEY		Generate
CLOUDINARY_API_SECRET		Generate

+ Add Environment Variable + Add from .env

Рисунок 3.2 – Налаштування змінних оточення (Environment Variables) на Render

Після внесення всіх змінних натисніть кнопку збереження параметрів та дочекайтеся завершення автоматичної збірки проекту.

Третім етапом є розгортання клієнтської частини на React у середовищі Vercel.

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		44

- Авторизуйтеся на сервісі Vercel через свій обліковий запис GitHub та натисніть кнопку додавання нового проєкту.
- Імпортуйте той самий репозиторій із кодом, що використовувався для серверної частини платформи.
- У налаштуваннях збірки вкажіть шлях до папки client як кореневої директорії для фронтенду.
- Оберіть інструмент збірки Vite, який автоматично підставить необхідні параметри для генерації статичних файлів. Вибір налаштувань фронтенду наведено на рисинку 3.3.

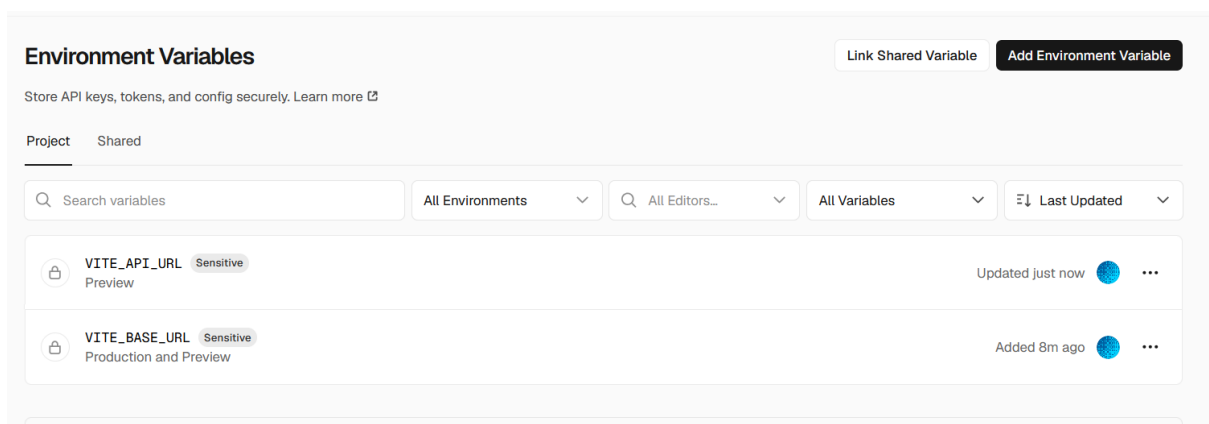


Рисунок 3.3 – Налаштування змінних оточення фронтед-частини у сервісі Vercel

Перед початком збірки обов'язково додайте адресу раніше створеного сервера Render у поле змінної оточення VITE_API_URL для забезпечення взаємодії між модулями.

Четвертим кроком запустіть процес розгортання клієнтського інтерфейсу, натиснувши кнопку Deploy. Система виконає встановлення залежностей та згенерує оптимізовані статичні файли для мережі CDN.

- Після завершення процесу збірки сервіс Vercel надасть вам автоматично згенеровану публічну адресу сайту.
- Перейдіть за цим посиланням для перевірки працездатності інтерфейсу та його взаємодії з базою даних. Вікно успішного розгортання проєкту зображено на рисунку 3.4.

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		45

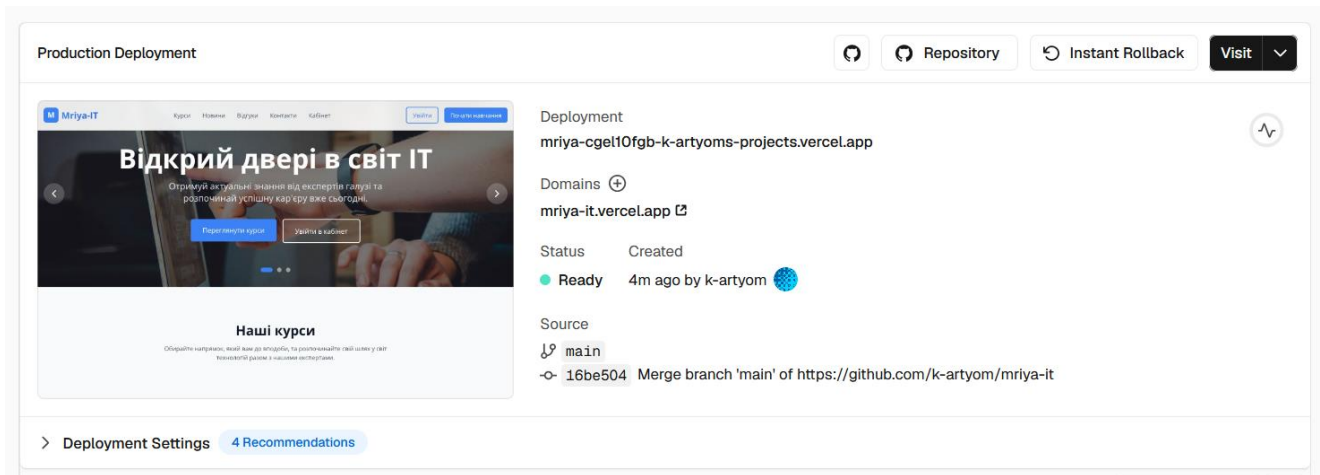


Рисунок 3.4 – Вікно успішного розгортання проекту та посилання на сайт

У результаті виконання цих інструкцій освітня платформа стає повністю доступною в Інтернеті. Кожне наступне оновлення коду в репозиторії GitHub буде автоматично запускати нову збірку на обох хмарних сервісах.

3.2 Інструкція з обслуговування та наповнення інформаційної системи

Для стабільного обслуговування та первинного наповнення навчальної платформи Mriya IT розроблено покрокову інструкцію з налаштування та адміністрування бази даних. Шляхом використання консольних інструментів та графічного інтерфейсу адміністратор може оперативно керувати контентом.

Першим кроком виконайте автоматичне заповнення бази даних початковою інформацією за допомогою вбудованих скриптів сервера.

- Відкрийте термінал у папці `server` та запустіть команду створення першого адміністратора за допомогою скрипта `seedAdmin.js`. Вона створює обліковий запис із повними правами доступу.

- Запустіть додатково команду виконання скрипта `seedData.js` для автоматичного наповнення бази даних демонстраційними курсами, уроками та відгуками студентів. Загальний результат запуску скрипта та генерації демонстративних даних наведено на рисинку 3.5.

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
						46
Зм.	Арк.	№ докум.	Підпис	Дата		

```

m Типа Назар@mriya-it\server> node .\seedData.js
MongoDB підключено для сидування...
Очищення файлів звітів у Cloudinary...
Файли в Cloudinary очищено.
База очищена.
Користувачі створені.
Курси створені.
Уроки створені.
Групи створені.
Звіти створені.
Оцінки створені.
✅ Сидування успішно завершено!

```

Рисунок 3.5 – Результат виконання скрипта створення демонстративних даних

На етапі початкової перевірки системи ці тестові дані допоможуть швидко протестувати роботу основних модулів без тривалого ручного введення. Скрипти автоматично створюють необхідні зв'язки між таблицями бази даних.

Другим кроком здійснить додавання нових користувачів із розмежуванням прав доступу через адміністративну панель керування.

- Авторизуйтеся на сайті під обліковим записом адміністратора та перейдіть до розділу користувачів.
- Натисніть кнопку додавання нового учасника та введіть його повне ім'я, електронну пошту і пароль.
- Оберіть необхідну роль у випадаючому списку, призначивши користувача студентом, викладачем або адміністратором. У графічному інтерфейсі форму реєстрації нового користувача показано на рисинку 3.6.

Рисунок 3.6 – Форма реєстрації нового користувача в адміністративній панелі

Після збереження даних новий викладач або студент зможе авторизуватися в системі та отримати доступ до особистого кабінету. Захищені токени JWT будуть генеруватися автоматично під час кожного входу.

Третім кроком виконайте створення нових освітніх програм у каталозі школи.

- Перейдіть до розділу Курси в панелі адміністратора та натисніть кнопку для створення нової програми.
- Заповніть форму, вказавши назву, детальний опис курсу, його категорію, ціну та виберіть викладача зі списку.
- Завантажте графічне зображення обкладинки курсу, яке автоматично збережеться у хмарному сховищі Cloudinary [10]. Спрощену форму створення нової програми представлено на рисунку 3.7.

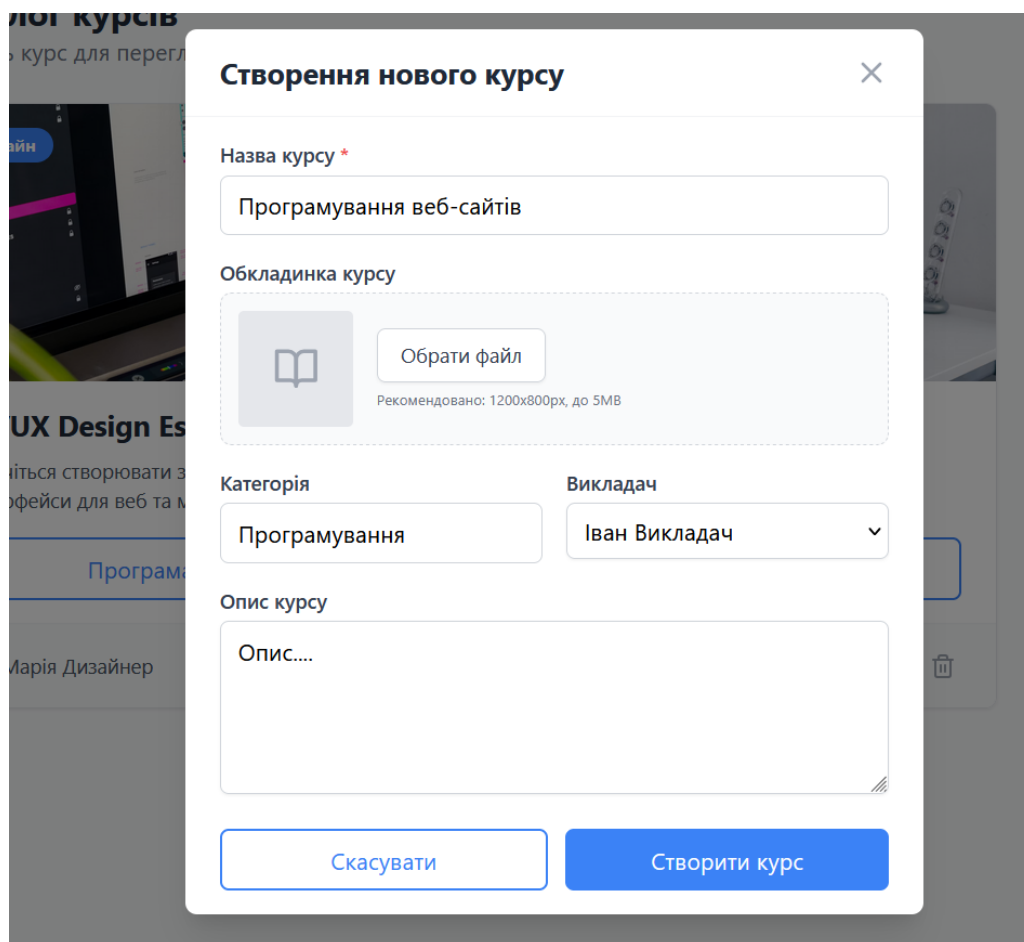


Рисунок 3.7 – Додавання нового освітнього курсу до загального каталогу

Створений курс одразу відобразиться у публічному каталозі на головній сторінці сайту для всіх неавторизованих користувачів. Відвідувачі сайту зможуть ознайомитися з його повною програмою та призначеним лектором.

Четвертим кроком організуйте процес наповнення курсу навчальними матеріалами та уроками.

- Авторизуйтеся як викладач, перейдіть до розділу Мої курси та оберіть потрібну програму для редагування.
- Натисніть кнопку додавання нового уроку та заповніть поля для заголовка та посилання на відеоматеріали.
- Введіть теоретичну частину заняття у полі текстового редактора з підтримкою розмітки Markdown для виділення заголовків та коду. Для наочності вікно створення уроку викладачем показано на рисунку 3.8.

Рисунок 3.8 – Процес додавання навчального уроку викладачем платформи

Завдяки підтримці розмітки Markdown викладачі можуть структурувати тексти уроків без залучення технічних спеціалістів. Форматування тексту та вставка блоків коду відбуваються безпосередньо у вікні редагування.

П'ятим кроком здійсніть завантаження виконаних робіт студентами для перевірки.

- Авторизуйтеся під роллю студента та відкрийте сторінку потрібного уроку у вашому розкладі.
- Ознайомтеся з матеріалами, перейдіть до блоку звіту та виберіть файл на комп'ютері для завантаження.
- Натисніть кнопку відправлення звіту для його збереження у хмарному сховищі. У робочому кабінеті форма відправлення виконаного завдання студентом показана на рисинку 3.9.

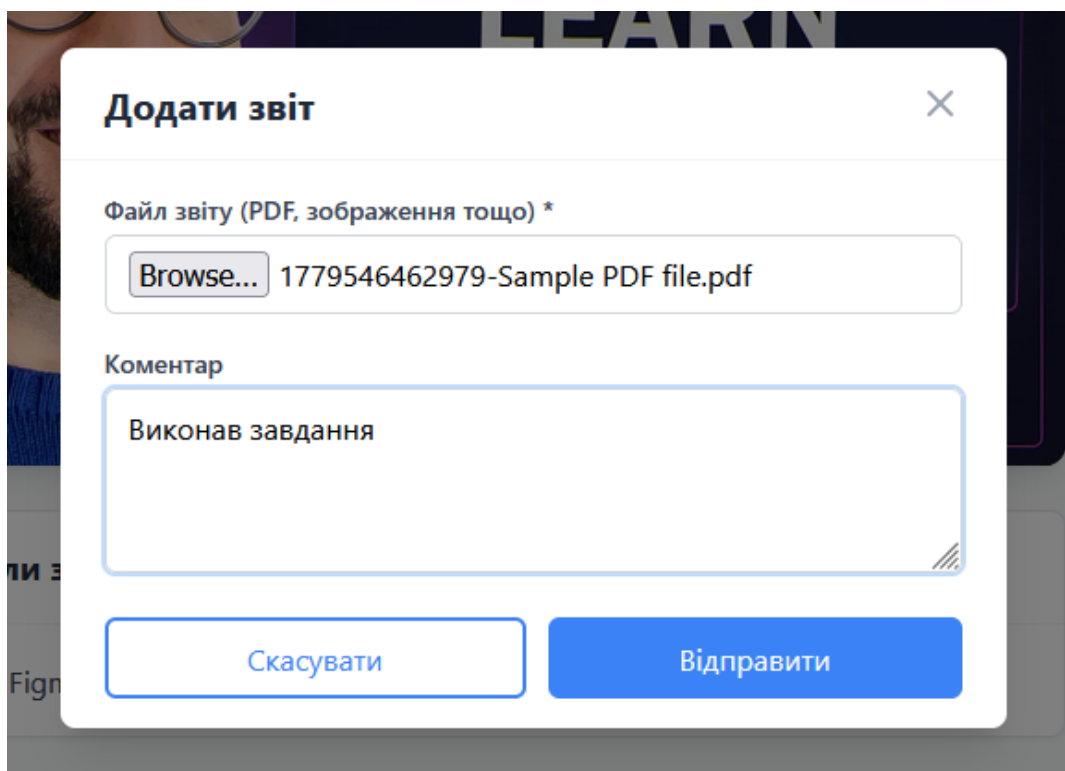


Рисунок 3.9 - Форма завантаження виконаного завдання студентом

Після успішного завантаження файлу статус домашнього завдання автоматично зміниться на очікування перевірки. Викладач отримає внутрішнє

сповіщення про надходження нової роботи.

Шостим кроком виконайте перевірку та оцінювання робіт студентів викладачем.

- Увійдіть у кабінет викладача та перейдіть до розділу перевірки домашніх завдань для перегляду нових звітів.
- Відкрийте надісланий файл студента, ознайомтеся з його вмістом та введіть оцінку за стобальною шкалою.
- Залиште текстовий коментар із рекомендаціями у спеціальному полі та натисніть кнопку збереження. У робочій області інтерфейс оцінювання робіт викладачем наведено на рисунку 3.10.

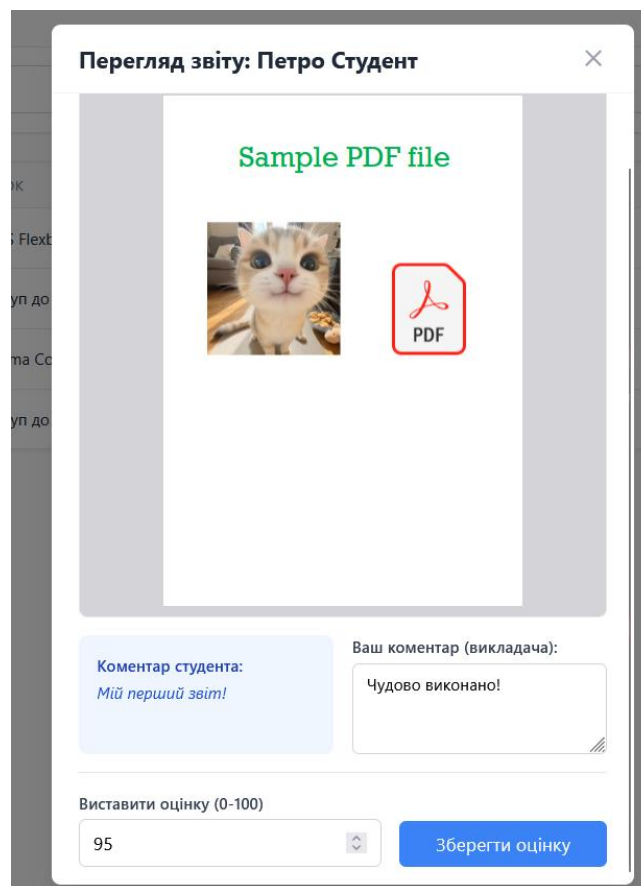


Рисунок 3.10 - Панель перевірки та оцінювання звітів викладачем

Після виставлення оцінки студент миттєво отримає сповіщення в системі про успішну перевірку його роботи викладачем. Нові бали автоматично додаються до загальної статистики успішності.

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		51

3.3 Інструкція з популяризації та підтримки інформаційної системи

Для успішного масштабування навчальної платформи «Mriya IT» та залучення цільової аудиторії необхідно реалізувати комплексну стратегію інформаційного супроводу розробленого програмного продукту у мережі. Вказаний підхід дасть швидкий результат. Зокрема, під час просування системи серед вітчизняних ІТ-шкіл доцільно зосередитися на проведенні презентацій технічних можливостей для автоматизації перевірки знань та ведення журналів успішності.

Спільне партнерство є вкрай важливим кроком. Шляхом впровадження моделі безкоштовного пробного періоду для освітніх центрів вдасться наочно продемонструвати переваги модульної архітектури та зручність керування великою кількістю студентських груп. Це суттєво полегшить вибір клієнтів. Задля підвищення впізнаваності бренду академії рекомендується використовувати інструменти пошукової оптимізації та активно поширювати інформаційні матеріали про нові курси через тематичні групи у соціальних мережах.

Для залучення уваги професійної спільноти варто організовувати відкриті вебінари із залученням провідних викладачів платформи для формування позитивного іміджу та високого рівня довіри.

У процесі подальшої підтримки системи особливу увагу слід приділяти регулярному моніторингу працездатності хмарної бази даних та забезпеченню надійного резервного копіювання всієї накопиченої інформації користувачів. На практиці захист даних є пріоритетом.

Шляхом періодичного оновлення програмних залежностей на сервері вдасться мінімізувати ризики виникнення вразливостей та гарантувати швидкість обробки запитів від багатьох клієнтських додатків одночасно. Зазвичай оновлення коду відбуваються за планом.

Задля покращення функціоналу необхідно налагодити канал зворотного зв'язку із користувачами, що дозволить виправляти дрібні помилки в інтерфейсі та впроваджувати найбільш затребувані навчальні модулі.

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		52

4 ЕКОНОМІЧНИЙ РОЗДІЛ

Метою економічної частини кваліфікаційної роботи є проведення економічних розрахунків, спрямованих на визначення економічної ефективності розробки інформаційної системи ІТ-школи «Mriya IT», прийняття рішення про подальший розвиток і впровадження або ж недоцільність проведення відповідної розробки.

Об'єктом розробки є інформаційна система «Mriya IT».

Розрахунок вартості розробки виконується в декілька етапів:

- описати технологічний процес розробки із зазначенням трудомісткості кожної операції;
- визначити суму витрат на оплату праці основного і допоміжного персоналу, включаючи відрахування на соціальні заходи;
- обчислити витрати на електроенергію;
- нарахувати суму амортизаційних відрахувань;
- визначити суму накладних витрат;
- скласти кошторис та визначити собівартість робіт;
- розрахувати ціну робіт;
- визначити економічну ефективність та термін окупності.

4.1 Визначення стадій технологічного процесу та загальної тривалості проведення розробки інформаційної системи

В цьому підрозділі розглянемо основні етапи технологічного процесу для розробки інформаційної системи «Mriya IT». Для визначення загальної тривалості проведення робіт доцільно дані витрат часу по окремих операціях технологічного процесу звести у таблицю 4.1.

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		53

Таблиця 4.1 - Середній час виконання робіт по обслуговуванню та стадії (операції) технологічного процесу

№ п/п	Назва операції (стадії)	Виконавець	Середній час виконання операції, год.
1	Планування та аналіз	Керівник	10
		Розробник	5
2	Розробка технічного завдання	Керівник	5
		Розробник	10
3	Проектування інтерфейсу та баз даних	Розробник	15
4	Розробка функціоналу	Розробник	80
5	Тестування та відладка	Тестувальник	20
6	Документування	Розробник	5
7	Розгортання та підтримка	Керівник	10
Разом			160

Сумарний час виконання операцій технологічного процесу становить 160 години.

4.2 Визначення витрат на оплату праці та відрахувань на соціальні заходи

У даному підрозділі проводиться аналіз і розрахунок витрат, пов'язаних з оплатою праці та відрахуваннями на соціальні заходи, що необхідні для розробки інформаційної системи «Mriya IT».

Розмір заробітної плати залежить від складності та умов виконуваної роботи, професійно-ділових якостей працівника, результатів його праці та діяльності підприємства.

Основна заробітна плата розраховується за формулою:

$$З_{\text{осн.}} = T_c * K_r \quad (4.1)$$

де: T_c – тарифна ставка, грн. Керівник 350грн/год. Розробник 220грн/год. Тестувальник – 140грн/год; K_r – кількість відпрацьованих годин.

Отже, основна заробітна плата для:

- Керівник $З_{\text{осн.}} = 25 * 350 = 8\,750$ грн;
- Розробник $З_{\text{осн.}} = 115 * 220 = 25\,300$ грн;
- Тестувальник $З_{\text{осн.}} = 20 * 140 = 2\,800$ грн.

Сумарна основна заробітна плата становить

$$З_{\text{осн.}} = 36\,850 \text{ грн.}$$

Додаткова заробітна плата становить 10 – 15 % від суми основної заробітної плати.

$$З_{\text{дод.}} = З_{\text{осн.}} * K_{\text{допл.}} \quad (4.2)$$

де: $K_{\text{допл.}}$ – коефіцієнт додаткових виплат працівника.

Отже додаткова заробітна плата по категоріях працівників становить:

- Керівник $З_{\text{дод.2}} = 8\,750 * 0,1 = 875$ грн;
- Розробника $З_{\text{дод.2}} = 25\,300 * 0,1 = 2\,530$ грн;
- Тестувальник $З_{\text{дод.2}} = 2\,800 * 0,1 = 280$ грн.

Загальна додаткова заробітна плата становить:

$$З_{\text{дод.}} = 875 + 2\,530 + 280 = 3\,685 \text{ грн.}$$

Звідси загальні витрати на оплату праці ($B_{\text{o.n.}}$) визначаються за формулою:

$$B_{\text{o.n.}} = З_{\text{осн.}} + З_{\text{дод.}} \quad (4.3)$$

$$B_{\text{o.n.}} = 36\,850 + 3\,685 = 40\,535 \text{ грн.}$$

Єдиний соціальний внесок (ЄСВ – 22%) визначається за формулою:

$$B_{\text{ЄСВ}} = B_{\text{оп}} * 0,22 \quad (4.4)$$

$$B_{\text{ЄСВ}} = 40\,535 * 0,22 = 8\,918 \text{ грн.}$$

Проведені розрахунки витрат на оплату праці наведено у таблиці 4.2.

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
						55
Зм.	Арк.	№ докум.	Підпис	Дата		

Таблиця 4.2 – Зведені розрахунки витрат на оплату праці

№ п/п	Категорія працівників	Основна заробітна плата, грн.			Додатков а заробітна плата, грн.	ЄСВ, грн.	Всього витрати на оплату праці, грн. 6 = 3+4+5
		Тарифна ставка, грн.	К-сть годин	Фактичн о нарах. зарплати, грн.			
1	Керівник	350	25	8 750	875	2 118	11 743
2	Розробник	220	115	25 300	2 530	6 123	33 953
3	Тестувальник	140	20	2 800	280	677	3 757
Разом				36 850	3 685	8 918	49 453

Отже, загальні витрати на оплату праці становлять 49 453 грн.

4.3 Розрахунок витрат на електроенергію

Розрахуємо вартість електроенергії. Затрати на електроенергію 1-ці обладнання визначаються за формулою:

$$З_{\text{в}} = W * T * S \quad (4.5)$$

де: W – необхідна потужність, кВт; T – кількість годин роботи обладнання; S – вартість кіловат-години електроенергії (приймаємо 15, 94 грн).

В нашій системі є 1 ПК. Витрати на електроенергію для цього комп'ютера обчислимо окремо, взявши за основу, що час роботи обладнання обчислюється в залежності від виконуваних робіт згідно таблиці. 4.1 і споживані потужності наступні: комп'ютер – 0,82 кВт/год.

$$З_{\text{ек}} = 0,5 * 160 * 15,94 = 1\,275 \text{ грн.}$$

Витрати на електроенергію становлять 1 275 грн.

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		56

4.4 Розрахунок суми амортизаційних відрахувань інформаційної системи.

Характерною особливістю застосування основних фондів у процесі виробництва є їх відновлення. Для відновлення засобів праці у натуральному виразі необхідне їх відшкодування у вартісній формі, яке здійснюється шляхом амортизації. Амортизація – це процес перенесення вартості основних фондів на вартість новоствореної продукції з метою їх повного відновлення

Комп'ютери та оргтехніка належать до четвертої групи основних фондів.

Амортизація на них нараховується лише в випадку, якщо мінімально допустимі строки їх корисного використання 2 роки. Для визначення амортизаційних відрахувань застосовуємо формулу:

$$A = \frac{B_B * H_A}{100\%} * T, \quad (4.6)$$

де:

- A – амортизаційні відрахування за звітний період, грн.;
- B_B – балансова вартість групи основних фондів на початок звітного періоду, грн.;
- H_A – норма амортизації, 0,04 %.

Оскільки для написання програми та її тестування використовується один ПК, вартістю 30000,00 грн., то сума амортизаційних відрахувань становитиме:

$$A = \frac{30\,000 * 0,04}{150} * 160 = 1\,280 \text{ грн.}$$

4.5 Обчислення накладних витрат

Накладні витрати пов'язані з обслуговуванням виробництва, утриманням апарату управління підприємства (фірми) та створення необхідних умов праці. В залежності від організаційно-правової форми діяльності господарюючого суб'єкта, накладні витрати можуть становити 20–60 % від суми основної та додаткової заробітної плати працівників.

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
						57
Зм.	Арк.	№ докум.	Підпис	Дата		

$$H_B = B_{o.п.} * 0,2..0,6 \quad (4.7)$$

де: H_B – накладні витрати.

$$H_B = 40\,535 * 0,4 = 16\,214 \text{ грн.}$$

4.6 Складання кошторису витрат та визначення собівартості інформаційної системи

Для складення кошторису витрат та визначення собівартості, результати проведених вище розрахунків зведемо у таблиці 4.3.

Таблиця 4.3 - Кошторис витрат розробки інформаційної системи «Mriya IT»

№	Зміст витрат	Сума, грн.	В % до загальної суми
1.	Витрати на оплату праці	49 453	72
2.	Витрати на електроенергію	1 275	2
3.	Амортизаційні відрахування	1 280	2
4.	Накладні витрати	16 214	24
5.	Собівартість	68 222	100

Собівартість (C_B) НДР розраховуємо за формулою:

$$C_B = B_{o.п.} + B_{c.з} + 3e + A + H_B \quad (4.8)$$

Отже, собівартість дорівнює $C_B = 68\,222$ грн.

4.7 Розрахунок ціни інформаційної системи

Розрахунок ціни науково-дослідної роботи включає в себе урахування різноманітних факторів, таких як рівень рентабельності, собівартість та податкова ставка.

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		58

Ціну робіт можна визначити за формулою:

$$Ц = Св * (1 + P_{рен}) * (1 + ПДВ), \quad (4.9)$$

де: $Св$ – собівартість; $P_{рен}$ – рівень рентабельності; $ПДВ$ – ставка податку на додану вартість.

$$Ц = 68\,222 * (1 + 0,3) * (1 + 0,2) = 106\,426 \text{ грн.}$$

4.8 Визначення економічної ефективності і терміну окупності капітальних вкладень

Ефективність виробництва – це узагальнене і повне відображення кінцевих результатів використання робочої сили, засобів та предметів праці на підприємстві за певний проміжок часу.

Для визначення ефективності продукту розраховують чисту теперішню вартість (ЧТВ) і термін окупності (Ток).

$$ЧТВ = -Св + \sum_{i=1}^t \frac{\Gamma_{\Pi}}{(1+i)^t}, \quad (4.10)$$

де: $Св$ – собівартість розробки; Γ_{Π} – грошовий потік за t – ий рік; t – відповідний рік проекту; i – величина дисконтної ставки (10...15%).

$$ЧТВ = -68\,222 + \frac{38\,204}{(1+0,1)^1} + \frac{38\,204}{(1+0,1)^2} + \frac{38\,204}{(1+0,1)^3} = 26\,547 \text{ грн}$$

Якщо $ЧТВ \geq 0$, то проект може бути рекомендований до впровадження.

Термін окупності визначається за формулою:

$$T_{ок} = T_{пв} + \frac{H_{в}}{\Gamma_{пр}} \quad (4.11)$$

де: $T_{пв}$ – період до повного відшкодування витрат, років; $H_{в}$ – невідшкодовані витрати на початок року, грн.; $\Gamma_{пр}$ – грошовий потік на початок року, грн.

$$T_{ок} = 2 + \frac{2\,177}{38\,204} = 2,05 \text{ р.}$$

Всі дані внесемо в зведену таблицю 4.4.

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		59

Таблиця 4.4 – Техніко-економічні показники інформаційної системи

№ п/п	Показник	Значення
1.	Собівартість, грн.	68 222
2.	Плановий прибуток або грошовий потік, грн.	38 204
3.	Ціна, грн.	106 426
4.	Чиста теперішня вартість, грн.	26547
5.	Термін окупності, рік	2,05

Прибутковість проекту та термін окупності свідчать про його фінансову ефективність та здатність повернути капітальні вкладення протягом 2,05 року. Отже, на основі отриманих показників можна зробити висновок, що розробка інформаційної системи «Mriya IT» є доцільною з економічної точки зору.

5 ОХОРОНА ПРАЦІ, ТЕХНІКА БЕЗПЕКИ ТА ЕКОЛОГІЧНІ ВИМОГИ

5.1 Інструкції з охорони праці, їх поділ

Охорона праці є важливою складовою організації будь-якої професійної діяльності, оскільки вона спрямована на збереження життя, здоров'я та працездатності працівників. У межах підприємства або установи вимоги безпеки не можуть залишатися лише загальними положеннями, тому вони конкретизуються у внутрішніх документах. Одним з основних таких документів є інструкція з охорони праці, яка встановлює обов'язкові правила безпечного виконання робіт певного виду або роботи за певною професією [11].

Інструкція з охорони праці є нормативним актом підприємства, установи або організації. Вона визначає порядок безпечної поведінки працівника на робочому місці, у виробничих приміщеннях, на території підприємства або в іншому місці, де працівник виконує трудові обов'язки за дорученням роботодавця. Такий документ має практичне значення, тому що пояснює працівникові, які небезпечні та шкідливі фактори можуть виникати під час роботи, як їх уникати та як діяти у разі аварійної ситуації.

- Інструкція з охорони праці встановлює вимоги безпеки для конкретної професії або конкретного виду робіт;
- Інструкція з охорони праці конкретизує загальні норми законодавства з урахуванням умов конкретного підприємства;
- Інструкція з охорони праці допомагає працівникові правильно підготувати робоче місце, безпечно виконувати роботу та завершувати її без створення небезпеки;
- Інструкція з охорони праці є обов'язковою для виконання працівниками, на яких вона поширюється;

Отже, інструкція з охорони праці не є формальним документом. Вона використовується під час навчання, інструктажів, перевірки знань, організації робочого процесу та розслідування порушень вимог безпеки. Для кваліфікаційної

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		61

роботи це питання є важливим, оскільки навіть робота з комп'ютерною технікою, та мережевим обладнанням потребує визначених правил безпечної експлуатації. Поділ інструкцій з охорони праці подано в таблиці 5.1.

Таблиця 5.1 – Поділ інструкцій з охорони праці

Ознака поділу	Вид інструкції	Характеристика
За сферою дії	Типова інструкція	Типова інструкція містить загальні вимоги безпеки для певної професії або виду робіт і може використовуватися як основа для локального документа підприємства.
За сферою дії	Інструкція підприємства	Інструкція підприємства розробляється з урахуванням конкретного обладнання, приміщень, технологічних процесів і умов праці.
За призначенням	Інструкція за професією	Інструкція за професією встановлює правила безпеки для працівника певної посади або професії.
За призначенням	Інструкція за видом робіт	Інструкція за видом робіт встановлює правила безпеки для виконання конкретної роботи незалежно від назви посади працівника.
За строком застосування	Постійна інструкція	Постійна інструкція діє до її перегляду, заміни або скасування.
За строком застосування	Тимчасова інструкція	Тимчасова інструкція може застосовуватися для нових, разових або дослідних робіт до затвердження постійного документа.

Такий поділ дає змогу правильно визначити, який саме документ потрібен для конкретної ситуації. Наприклад, для офісного працівника або користувача комп'ютера може бути розроблена інструкція за професією, а для робіт з переносним електроінструментом або обслуговуванням обладнання - інструкція за видом робіт. На підприємстві важливо мати не лише загальні правила, а й документи, які відповідають реальним умовам праці .

Розроблення інструкцій здійснюється на основі нормативно-правових актів з охорони праці, технологічної документації, експлуатаційних документів на обладнання та фактичних умов виконання робіт. Перелік необхідних інструкцій зазвичай формується службою охорони праці за участю керівників структурних підрозділів і відповідальних спеціалістів. Затвердження інструкцій виконує роботодавець або уповноважена ним особа, після чого документ набирає чинності та стає обов'язковим для працівників.

- Розділ “Загальні положення” визначає сферу застосування інструкції, характеристику робочого місця, умови допуску до роботи та основні небезпечні фактори;

- Розділ “Вимоги безпеки перед початком роботи” встановлює порядок підготовки робочого місця, перевірки обладнання, засобів захисту та інструментів;

- Розділ “Вимоги безпеки під час роботи” визначає правила безпечного виконання трудових операцій і забороняє дії, які можуть створити небезпеку;

- Розділ “Вимоги безпеки після закінчення роботи” встановлює порядок вимкнення обладнання, прибирання робочого місця та повідомлення про виявлені несправності;

- Розділ “Вимоги безпеки в аварійних ситуаціях” визначає дії працівника у разі пожежі, ураження електричним струмом, пошкодження обладнання або іншої небезпечної події;

Наведена структура робить інструкцію логічною та зручною для практичного використання. Працівник отримує правила не у випадковому порядку, а відповідно до послідовності виконання роботи: підготовка, виконання,

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		63

завершення та дії у нестандартних ситуаціях. Це зменшує ризик помилок і допомагає швидше орієнтуватися у вимогах безпеки. Етапи роботи з інструкціями подано в таблиці 5.2.

Таблиця 5.2. – Етапи роботи з інструкціями

Етап роботи з інструкцією	Відповідальні особи	Практичне значення етапу
Складання переліку інструкцій	Служба охорони праці та керівники підрозділів	На цьому етапі визначається, які інструкції потрібні підприємству.
Розроблення тексту	Керівник робіт або відповідальний фахівець	На цьому етапі враховуються умови праці, обладнання, технологічні процеси та небезпечні фактори.
Узгодження	Служба охорони праці та інші відповідальні служби	На цьому етапі перевіряється відповідність інструкції вимогам охорони праці.
Затвердження	Роботодавець або уповноважена особа	На цьому етапі документ набуває чинності.
Ознайомлення працівників	Керівник робіт або відповідальна особа	На цьому етапі працівники отримують інформацію про обов'язкові правила безпечної роботи.
Перегляд і оновлення	Служба охорони праці та керівники підрозділів	На цьому етапі інструкція приводиться у відповідність до нових умов, обладнання або вимог законодавства.

Періодичний перегляд інструкцій потрібний для того, щоб вони не втрачали актуальності. Якщо змінюється обладнання, технологія, організація праці або вимоги законодавства, документ необхідно оновити. Це особливо важливо для сфер, де використовуються комп'ютери, джерела безперебійного живлення, мережеве обладнання, подовжувачі та інші електротехнічні пристрої.

5.2 Системи засобів і заходів забезпечення електробезпеки

Електробезпека є системою організаційних і технічних заходів, які запобігають небезпечному впливу електричного струму, електричної дуги, електромагнітного поля та статичної електрики на людину. Для сучасного робочого місця це питання актуальне не лише у виробничих цехах, а й в офісах, навчальних аудиторіях, комп'ютерних класах і серверних приміщеннях. Небезпека може виникати через пошкоджену ізоляцію, несправні розетки, перевантаження електромережі, неправильне підключення обладнання або самостійний ремонт електропристроїв [12].

Основна мета електробезпеки полягає у тому, щоб не допустити контакту людини зі струмовідними частинами та зменшити наслідки можливої несправності обладнання. Для цього застосовується не один окремий засіб, а комплекс взаємопов'язаних рішень. Умовно системи засобів і заходів забезпечення електробезпеки можна поділити на технічні засоби, електрозахисні засоби та організаційно-технічні заходи.

Кожна з цих підсистем відіграє свою унікальну роль у загальній структурі захисту персоналу: технічні засоби закладаються ще на етапі проєктування та монтажу приміщень, електрозахисні - використовуються безпосередньо під час роботи, а організаційні - регламентують сам порядок дій та рівень відповідальності працівників. Спільне та послідовне впровадження цих елементів мінімізує ризики електротравматизму до найнижчого можливого рівня.

Детальна класифікація, основний зміст та практичні приклади застосування кожної з наведених систем систематизовано і подано в таблиці 5.3.

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
						65
Зм.	Арк.	№ докум.	Підпис	Дата		

Таблиця 5.3. – Етапи роботи з інструкціями

Система забезпечення електробезпеки	Основний зміст	Приклади застосування
Система технічних засобів і заходів	Ця система запобігає виникненню небезпечної напруги на корпусах обладнання та обмежує можливість контакту людини зі струмовідними частинами.	Захисне заземлення, занулення, автоматичне вимкнення живлення, ізоляція, огороження, блокування, захисне розділення мережі.
Система електрозахисних засобів	Ця система забезпечує додатковий захист працівників під час експлуатації або обслуговування електрообладнання.	Діелектричні рукавички, ізолювальні підставки, інструмент з ізольованими ручками, покажчики напруги, попереджувальні плакати.
Система організаційно-технічних заходів	Ця система встановлює порядок допуску, контролю та безпечного виконання робіт.	Призначення відповідальних осіб, навчання, інструктажі, перевірка знань, оформлення допуску, контроль справності обладнання.

Зазначені системи повинні застосовуватися разом. Технічні засоби зменшують імовірність аварійної ситуації, електрозахисні засоби захищають працівника під час виконання робіт, а організаційні заходи не допускають до небезпечних робіт осіб без відповідної підготовки. Саме поєднання цих елементів

формує реальний рівень електробезпеки.

- Захисна ізоляція відокремлює струмовідні частини від людини та запобігає випадковому дотику;
- Захисне заземлення з'єднує металеві неструмовідні частини обладнання із землею для зменшення небезпеки у разі пошкодження ізоляції;
- Автоматичне вимкнення живлення припиняє подачу електроенергії у разі аварійного режиму або витоку струму;
- Огородження і кожухи обмежують доступ до струмовідних частин обладнання;
- Блокування не дозволяє ввімкнути обладнання за умов, коли його експлуатація може бути небезпечною;
- Попереджувальні знаки та написи інформують працівників про наявність електричної небезпеки;

Технічні засоби електробезпеки мають бути передбачені ще на етапі проектування, монтажу та введення обладнання в експлуатацію. У процесі роботи вони повинні підтримуватися у справному стані. Для робочого місця з комп'ютером це означає використання справних розеток, непошкоджених кабелів, якісних подовжувачів, заземлених мережевих фільтрів і недопущення перевантаження електромережі.

Проте, окрім загальних інженерно-технічних рішень, закладених у конструкцію приміщень та обладнання, критично важливу роль відіграє безпосередній індивідуальний та колективний захист працівників під час виконання конкретних завдань або обслуговування техніки.

Цей захист забезпечується спеціальними засобами, які поділяють залежно від ступеня їхньої ізолювальної здатності, функціонального призначення та способу застосування.

Детальна класифікація електрозахисних засобів, їхнє цільове призначення та конкретні приклади використання у практичній діяльності подані в таблиці 5.4.

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		67

Таблиця 5.4. – Класифікація електрозахисних засобів

Група електрозахисних засобів	Призначення	Приклади
Основні електрозахисні засоби	Основні засоби дозволяють безпечніше виконувати окремі операції в електроустановках за умови дотримання правил і допуску підготовленого персоналу.	Ізолювальні штанги, покажчики напруги, інструмент з ізольованими ручками.
Додаткові електрозахисні засоби	Додаткові засоби посилюють захист і використовуються разом з основними засобами.	Діелектричні рукавички, діелектричне взуття, ізолювальні підставки, килимки.
Колективні засоби захисту	Колективні засоби захищають групу працівників або обмежують доступ до небезпечної зони.	Огородження, захисні екрани, плакати безпеки, сигнальні огороження.

Електрозахисні засоби повинні відповідати умовам роботи та використовуватися лише справними. Для звичайного користувача комп'ютерної техніки головним правилом є недопущення самостійного ремонту електричних частин обладнання. У разі появи запаху перегріву, іскріння, пошкодження кабелю, нагрівання вилки або нестабільної роботи пристрою необхідно припинити використання обладнання та повідомити відповідальну особу.

До робіт з електрообладнанням допускаються лише працівники, які мають відповідну підготовку, пройшли інструктаж і знають вимоги безпеки;

Роботодавець призначає відповідальних осіб за справний стан електрогосподарства та безпечну експлуатацію електроустановок;

Працівники повинні повідомляти про несправності електрообладнання, пошкодження ізоляції, розеток, вилок, подовжувачів і кабелів;

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		68

Забороняється використовувати обладнання з видимими пошкодженнями, нестійкою роботою або ознаками перегріву;

Забороняється перевантажувати подовжувачі та підключати до однієї розетки надмірну кількість потужних пристроїв;

Перевірку, ремонт і підключення електрообладнання повинен виконувати лише кваліфікований персонал;

Організаційно-технічні заходи мають особливе значення, тому що навіть якісне обладнання може стати небезпечним у разі неправильного використання. Працівник повинен дотримуватися інструкцій, не відкривати корпуси електроприладів, не торкатися оголених проводів, не працювати з електрообладнанням мокрими руками і не залишати несправні пристрої увімкненими.

Для приміщення, у якому використовується комп'ютерна техніка, важливо забезпечити справність електромережі, достатню кількість розеток, правильне розміщення кабелів, відсутність їх натягування або перетирання, а також можливість швидкого вимкнення обладнання у разі небезпеки. Усі працівники або користувачі мають знати, де розташовані вимикачі, як повідомити про несправність і до кого звернутися у разі аварійної ситуації.

Таким чином, інструкції з охорони праці та системи електробезпеки виконують спільне завдання - запобігають виробничим травмам і створюють безпечні умови праці. Інструкції визначають правила поведінки працівника, а системи електробезпеки забезпечують технічний, організаційний і додатковий захист від небезпечного впливу електричного струму. Для сучасних умов роботи, пов'язаних із використанням комп'ютерної техніки та електрообладнання, дотримання цих вимог є необхідною умовою безпечної діяльності.

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		69

ВИСНОВКИ

В процесі виконання даної кваліфікаційної роботи на тему «Розробка інформаційної системи ІТ-школи „Mriya IT“» виконано повний цикл виробничого проектування програмного продукту, починаючи від аналітичного огляду існуючих рішень та формулювання технічного завдання до програмної реалізації, тестування, публікації в мережі Інтернет і розробки супровідної документації. Поставлена мета роботи щодо створення працездатного вебзастосунку для керування освітнім процесом в ІТ-школі, ведення електронного журналу, оцінювання успішності студентів та публікації новинного контенту повністю досягнута.

Щодо відповідності завданням на кваліфікаційну роботу, розроблена система повністю задовольняє всі вимоги. Створена платформа відповідає критеріям, сформульованим у технічному завданні. У процесі роботи реалізовано публічний каталог курсів із відгуками випускників та розділом новин. Також розроблено закриту систему авторизації на основі токенів JWT з розподілом ролей на студента, викладача та адміністратора. Для викладачів створено зручний інтерфейс ведення електронного журналу успішності та перевірки звітів студентів. За допомогою хмарних сервісів Vercel та Render розроблену систему опубліковано у мережі Інтернет. Для перевірки працездатності всі функції протестовано за контрольними сценаріями.

Отримано такі основні результати та показники роботи.

- проведено аналітичний огляд популярних систем Moodle та Google Classroom, що дозволило обґрунтувати доцільність власної розробки;
- спроектовано структуру бази даних MongoDB, архітектуру клієнтської та серверної частин, а також розроблено зручні інтерфейси користувачів;
- реалізовано односторінковий вебзастосунок на React та Vite, серверний REST API на Node.js та Express із використанням Mongoose ODM;
- організовано завантаження медіафайлів та звітів студентів до хмарного сховища Cloudinary за допомогою бібліотеки Multer;

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
						70
Зм.	Арк.	№ докум.	Підпис	Дата		

- проведено успішне функціональне тестування за ролями користувачів відповідно до контрольних сценаріїв;
- виконано деплой клієнтської частини на платформу Vercel, серверної частини на хостинг Render та бази даних у хмару MongoDB Atlas;
- підготовлено інструкції з розміщення, експлуатації та наповнення контентом для адміністраторів і викладачів.

Оцінюючи самостійність виконання та новизну, роботу виконано повністю автономно. На етапі програмування весь обсяг завдань із проєктування бази даних, створення клієнтської та серверної частин виконано без сторонньої допомоги. Отримані рекомендації керівника використано для уточнення структури пояснювальної записки. У процесі виконання вперше для автора розв'язано комплексне завдання побудови предметно-орієнтованої моделі освітнього процесу в повному стеку JavaScript. Також успішно реалізовано рольову модель доступу та інтегровано хмарне сховище для оптимізації збереження файлів.

Щодо публікацій та наукової активності, підготовка статей або доповідей на конференціях у межах цього проєкту не передбачалася. Оприлюднення результатів заплановано під час офіційного захисту кваліфікаційної роботи перед комісією.

Для подальшого розвитку теми доцільно модернізувати систему Mriya IT у комерційний продукт. Перспективні напрями включають підключення власного доменного імені та платних тарифів хостингу для стабільної роботи без пауз. Доцільно також інтегрувати реальну платіжну систему для оплати курсів безпосередньо на сайті. Корисним буде впровадження розсилки сповіщень на електронну пошту. Припиняти розвиток системи не планується. Створена архітектура дозволяє легко нарощувати функціонал без зміни загальної структури платформи.

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		71

ПЕРЕЛІК ПОСИЛАНЬ

- 1) Офіційний сайт системи керування навчанням Moodle : вебсайт. URL: <https://moodle.org> (дата звернення: 15.05.2026).
- 2) Документація платформи Google Classroom : вебсайт. URL: <https://support.google.com/edu/classroom> (дата звернення: 16.05.2026).
- 3) Методичні вказівки до виконання кваліфікаційної роботи для здобувачів фахової передвищої освіти спеціальності 122 «Комп'ютерні науки» / уклад.: Марціяш Г. Я., Слободян Р. О. Тернопіль : ВСП «ТФК ТНТУ ім. І. Пулюя», 2026. 48 с.
- 4) Brown E. Web Development with Node and Express: Leveraging the JavaScript Stack. Sebastopol : O-Reilly Media, 2019. 342 p.
- 5) Офіційний інструмент збірки Vite : вебсайт. URL: <https://vite.dev> (дата звернення: 25.05.2026).
- 6) Документація середовища виконання Node.js : вебсайт. URL: <https://nodejs.org/en/docs> (дата звернення: 20.05.2026).
- 7) Bradshaw S., Brazil K., Chodorow C. MongoDB: The Definitive Guide. Sebastopol : O-Reilly Media, 2019. 514 p.
- 8) Документація бібліотеки React : вебсайт. URL: <https://react.dev> (дата звернення: 22.05.2026).
- 9) Специфікація стандарту авторизації JSON Web Tokens : вебсайт. URL: <https://jwt.io> (дата звернення: 28.05.2026).
- 10) Документація хмарного сервісу зберігання медіафайлів Cloudinary : вебсайт. URL: <https://cloudinary.com/documentation> (дата звернення: 29.05.2026)
- 11) Розробка інструкції з охорони праці на підприємстві є обов'язковою : вебсайт. URL: <https://www.pfu.gov.ua/lg/368346-rozrobka-instruktsiyi-z-ohorony-pratsi-na-pidpryyemstvi-ye-obov-yazkovoyu/> (дата звернення: 13.06.2026).
- 12) Вимоги щодо безпеки під час експлуатації електроустановок : вебсайт. URL: <https://dsp.gov.ua/wp-content/uploads/2024/08/vymohy-.pdf> (дата звернення: 13.06.2026).

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		72

ДОДАТКИ

Додаток А. Лістинг файлу «App.jsx»

```
import React, { useEffect } from 'react';
import { Routes, Route, Navigate, useLocation } from 'react-router-dom';
import { useAuth } from './context/AuthContext';
import Layout from './components/Layout';
import LoginModal from './components/LoginModal';
import Landing from './pages/Landing';
import Dashboard from './pages/Dashboard';
import Courses from './pages/Courses';
import CourseDetails from './pages/CourseDetails';
import Groups from './pages/Groups';
import GroupDetails from './pages/GroupDetails';
import LessonDetails from './pages/LessonDetails';
import Journal from './pages/Journal';
import Users from './pages/Users';
import Reports from './pages/Reports';
import Profile from './pages/Profile';
import PublicCourses from './pages/PublicCourses';
import PublicCourseDetails from './pages/PublicCourseDetails';
import PublicNews from './pages/PublicNews';
import NewsDetails from './pages/NewsDetails';
import Testimonials from './pages/Testimonials';
const ProtectedRoute = ({ children }) => {
  const { isAuthenticated, loading, openLoginModal } = useAuth();
  const location = useLocation();
  if (loading) {
    return <div className="flex h-screen items-center justify-center">Завантаження...</div>;
  }
  if (!isAuthenticated) {
    setTimeout(() => openLoginModal(), 0);
  }
}
```

					2026.KBP.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		73

```

        return <Navigate to="/" state={{ from: location }} replace />;
    }
    return <Layout>{children}</Layout>;
};
function App() {
    return (
        <div className="min-h-screen bg-gray-50">
            <LoginModal />
            <Routes>
                <Route path="/" element={<Landing />} />
                <Route path="/public-courses" element={<PublicCourses />} />
                <Route path="/public-courses/:id" element={<PublicCourseDetails />} />
            />
            <Route path="/public-news" element={<PublicNews />} />
            <Route path="/news/:id" element={<NewsDetails />} />
            <Route path="/testimonials" element={<Testimonials />} />

            <Route
                path="/dashboard"
                element={
                    <ProtectedRoute>
                        <Dashboard />
                    </ProtectedRoute>
                }
            />
            <Route
                path="/courses"
                element={
                    <ProtectedRoute>
                        <Courses />
                    </ProtectedRoute>
                }
            />
            <Route
                path="/courses/:id"

```

```

        element={
          <ProtectedRoute>
            <CourseDetails />
          </ProtectedRoute>
        }
      />
    <Route
      path="/groups"
      element={
        <ProtectedRoute>
          <Groups />
        </ProtectedRoute>
      }
    />
    <Route
      path="/groups/:id"
      element={
        <ProtectedRoute>
          <GroupDetails />
        </ProtectedRoute>
      }
    />
    <Route
      path="/lessons/:id"
      element={
        <ProtectedRoute>
          <LessonDetails />
        </ProtectedRoute>
      }
    />
    <Route
      path="/journal"
      element={
        <ProtectedRoute>

```

					2026.KBP.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		75

```

        <Journal />
    </ProtectedRoute>
  }
/>
<Route
  path="/users"
  element={
    <ProtectedRoute>
      <Users />
    </ProtectedRoute>
  }
/>
<Route
  path="/reports"
  element={
    <ProtectedRoute>
      <Reports />
    </ProtectedRoute>
  }
/>
<Route
  path="/profile"
  element={
    <ProtectedRoute>
      <Profile />
    </ProtectedRoute>
  }
/>

  { /* 404 Redirect */ }
  <Route path="*" element={<Navigate to="/" replace />} />
</Routes>
</div>
);}export default App;

```

					2026.KBP.122.423.22.00.00 ПЗ	Арк.
						76
Зм.	Арк.	№ докум.	Підпис	Дата		

Додаток Б. Лістинг файлу «Login.jsx»

```
import React, { useState } from 'react';
import { useNavigate } from 'react-router-dom';
import { useAuth } from '../context/AuthContext';
import Input from '../components/Input';
import Button from '../components/Button';
import Card from '../components/Card';
import { LogIn } from 'lucide-react';

const Login = () => {
  const [email, setEmail] = useState('');
  const [password, setPassword] = useState('');
  const [error, setError] = useState('');
  const [loading, setLoading] = useState(false);

  const { login } = useAuth();
  const navigate = useNavigate();
  const handleSubmit = async (e) => {
    e.preventDefault();
    setError('');
    setLoading(true);
    try {
      await login(email, password);
      navigate('/dashboard');
    } catch (err) {
      setError(err.response?.data?.message || 'Помилка при вході. Перевірте дані.');
```

```
    } finally {
```

```
      setLoading(false);
```

```
    }
```

```
  };
```

```
  return (
```

```
    <div className="min-h-screen flex items-center justify-center bg-gray-100 px-4">
```

					2026.KBP.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		77

```

<Card className="w-full max-w-md" title="Вхід в систему">
  <div className="flex justify-center mb-6 text-primary">
    <LogIn size={48} />
  </div>
  <form onSubmit={handleSubmit} className="space-y-4">
    <Input
      label="Email"
      type="email"
      value={email}
      onChange={(e) => setEmail(e.target.value)}
      placeholder="vash@email.com"
      required
      error={error && !email ? 'Обов'язкове поле' : null}
    />

    <Input
      label="Пароль"
      type="password"
      value={password}
      onChange={(e) => setPassword(e.target.value)}
      placeholder="....."
      required
      error={error && !password ? 'Обов'язкове поле' : null}
    />
    {error && (
      <div className="p-3 bg-red-50 border border-red-200 text-red-600 text-sm rounded-md">
        {error}
      </div>
    )}
    <Button
      type="submit"
      variant="primary"
      className="w-full mt-6"

```

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
						78
Зм.	Арк.	№ докум.	Підпис	Дата		

```

        loading={loading}
      >
        Увійти
      </Button>
    </form>
    <div className="mt-6 text-center text-gray-500 text-sm">
      Система керування навчанням <span className="font-bold text-
primary">Mriya-IT</span>
    </div>
  </Card>
</div>
);
};

export default Login;

```

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		79

Додаток В. Лістинг файлу «LessonDetails.jsx»

```
import React, { useEffect, useState } from 'react';
import { useParams } from 'react-router-dom';
import api from '../services/api';
import { useAuth } from '../context/AuthContext';
import Card from '../components/Card';
import Button from '../components/Button';
import ReportModal from '../components/ReportModal';
import ReactPlayer from 'react-player';
import Markdown from 'react-markdown';
import remarkGfm from 'remark-gfm';
import { Prism as SyntaxHighlighter } from 'react-syntax-highlighter';
import { vscDarkPlus } from 'react-syntax-highlighter/dist/esm/styles/prism';
import { Download, Video, FileText, ChevronLeft, Clock } from 'lucide-react';
import { useNavigate } from 'react-router-dom';

const LessonDetails = () => {
  const { id } = useParams();
  const navigate = useNavigate();
  const { user } = useAuth();
  const [lesson, setLesson] = useState(null);
  const [report, setReport] = useState(null);
  const [loading, setLoading] = useState(true);
  const [isReportModalOpen, setIsReportModalOpen] = useState(false);
  const fetchData = async () => {
    try {
      const lessonRes = await api.get(`/lessons/details/${id}`);
      setLesson(lessonRes.data);

      if (user?.role === 'student') {
        const reportRes = await api.get(`/reports/my/${id}`);
        setReport(reportRes.data);
      }
    }
  };
};
```

					2026.KBP.122.423.22.00.00 ПЗ	Арк.
						80
Зм.	Арк.	№ докум.	Підпис	Дата		

```

    }
  } catch (error) {
    console.error('Помилка завантаження даних', error);
  } finally {
    setLoading(false);
  }
};

useEffect(() => {
  fetchData();
}, [id, user]);

if (loading) {
  return <div className="flex justify-center p-20">Завантаження...</div>;
}

if (!lesson) {
  return <div className="text-center p-20">Заняття не знайдено</div>;
}

return (
  <div className="max-w-5xl mx-auto space-y-6">
    <button
      onClick={() => navigate(-1)}
      className="flex items-center gap-2 text-gray-500 hover:text-primary
transition-colors mb-4"
    >
      <ChevronLeft size={20} /> Назад до розкладу
    </button>
    <div className="space-y-2">
      <h1
        className="text-3xl font-bold text-gray-
800">{lesson.title}</h1>
      <p className="text-gray-500">
        Курс: {lesson.course?.title}
        {lesson.scheduledDate} && `
        • ${new
Date(lesson.scheduledDate).toLocaleString()}`
      </p>
    </div>
  </div>

```

```

<div className="grid grid-cols-1 lg:grid-cols-3 gap-8">
  <div className="lg:col-span-2 space-y-6">
    {/* Video Player Section */}
    {lesson.videoUrl && (
      <div className="bg-black rounded-xl overflow-hidden shadow-lg
aspect-video">
        <ReactPlayer
          url={lesson.videoUrl}
          width="100%"
          height="100%"
          controls
        />
      </div>
    )}
    {/* Lesson Content Section */}
    <Card title="Матеріали заняття">
      <div className="prose prose-blue max-w-none text-gray-700">
        <Markdown
          remarkPlugins={[remarkGfm]}
          components={{
            code({node, inline, className, children, ...props}) {
              const match = /language-(\w+)/.exec(className || '')
              return !inline && match ? (
                <SyntaxHighlighter
                  style={vscDarkPlus}
                  language={match[1]}
                  PreTag="div"
                  {...props}
                >
                  {String(children).replace(/\n$/, '')}
                </SyntaxHighlighter>
              ) : (
                <code className={className} {...props}>
                  {children}
                </code>
              )
            }
          }}
        />
      </div>
    </Card>
  </div>
</div>

```

```

        </code>
      )
    }
  }}
>
  {lesson.content || 'Опис відсутній.'}
</Markdown>
</div>
</Card>
</div>
<div className="space-y-6">
  {/* Attachments Section */}
  <Card title="Файли до уроку">
    {lesson.attachments && lesson.attachments.length > 0 ? (
      <div className="space-y-3">
        {lesson.attachments.map((file, idx) => {
          const fileName = typeof file === 'object' ? file.name :
(file.split('/').pop().split('-').slice(1).join('-') || `Матеріал ${idx +
1}`)});
          const fileUrl = typeof file === 'object' ? file.url :
file;

          return (
            <div key={idx} className="flex items-center justify-
between p-3 bg-gray-50 rounded-lg border border-gray-100">
              <div className="flex items-center gap-3 overflow-
hidden">
                <FileText className="text-blue-500 flex-shrink-0"
size={20} />
                <span className="text-sm font-medium truncate"
title={fileName}>{fileName}</span>
              </div>
              <a
                href={fileUrl}

```

```

        download={fileName}
        target="_blank"
        rel="noopener noreferrer"
        className="text-primary hover:text-blue-700"
      >
      <Download size={18} />
    </a>
  </div>
);
}}
</div>
) : (
  <p className="text-sm text-gray-500 italic text-center py-4">Файли не завантажено</p>
  </div>
</Card>
{/* Report Submission Section */}
<Card title="Звітність">
  {user?.role === 'student' ? (
    report ? (
      <div className="space-y-4">
        <div className="p-3 bg-green-50 rounded-lg border border-green-100">
          <div className="flex items-center justify-between mb-2">
            <div className="flex items-center gap-2">
              <FileText className="text-green-600" size={20} />
              <span className="text-sm font-semibold text-green-800">Ваш звіт</span>
            </div>
            <a
              href={report.fileUrl}
              target="_blank"
              rel="noopener noreferrer"

```

```

        className="text-primary hover:underline text-xs"
    >
        Відкрити файл
    </a>
</div>
<p        className="text-xs        text-gray-600
italic">"{report.studentComment || 'Без коментаря'}"</p>
</div>

<div className="border-t pt-3">
    <p    className="text-xs    text-gray-500    mb-1">Статус
перевірки:</p>

    {report.graded ? (
        <div className="flex items-center justify-between">
            <span    className="text-sm    font-bold    text-green-
600">Оцінено: {report.score}/100</span>
            <span className="text-[10px] text-gray-400">
                {new Date(report.gradeDate).toLocaleString()}
            </span>
        </div>
    ) : (
        <span className="text-sm font-bold text-orange-500
flex items-center gap-1">
            <Clock size={16} /> Очікує перевірку
        </span>
    )}
</div>

{report.teacherComment && (
    <div    className="p-3    bg-blue-50    rounded-lg    border
border-blue-100 mt-2">
        <p    className="text-[10px]    font-semibold    text-blue-
800 uppercase">Коментар викладача:</p>
        <p        className="text-xs        text-blue-

```

					2026.KBP.122.423.22.00.00 ПЗ	Арк.
						85
Зм.	Арк.	№ докум.	Підпис	Дата		

```

700">{report.teacherComment}</p>
        </div>
    )}
</div>
) : (
    <>
        <p className="text-sm text-gray-600 mb-4">Будь ласка,
завантажте ваш звіт після виконання практичної роботи.</p>
        <Button className="w-full" onClick={() =>
setIsReportModalOpen(true)}>Додати звіт</Button>
    </>
)
) : (
    <p className="text-sm text-gray-600 mb-4">Ви переглядаєте
цей урок як {user?.role === 'teacher' ? 'викладач' : 'адміністратор'}.</p>
    </Card>
</div>
</div>
<ReportModal
  isOpen={isReportModalOpen}
  onClose={() => setIsReportModalOpen(false)}
  lessonId={lesson._id}
  courseId={lesson.course?._id}
  onSuccess={() => {
    alert('Звіт успішно відправлено!');
    fetchData();
  }}
/>
</div>
);
};

export default LessonDetails;

```

					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		86

Додаток Г. Лістинг файлу «server.js»

```
const dns = require('node:dns');
// Примусове використання Google DNS для вирішення проблем з SRV-записами
(ECONNREFUSED)
dns.setServers(['8.8.8.8', '8.8.4.4']);

const express = require('express');
const mongoose = require('mongoose');
const cors = require('cors');
const path = require('path');
require('dotenv').config();

const app = express();

app.use(cors());
app.use(express.json());

// Відкриваємо папку uploads для доступу через URL
app.use('/uploads', express.static(path.join(__dirname, 'uploads')));

app.use('/api/auth', require('./routes/authRoutes'));
app.use('/api/users', require('./routes/userRoutes'));
app.use('/api/dashboard', require('./routes/dashboardRoutes'));
app.use('/api/courses', require('./routes/courseRoutes'));
app.use('/api/lessons', require('./routes/lessonRoutes'));
app.use('/api/grades', require('./routes/gradeRoutes'));
app.use('/api/groups', require('./routes/groupRoutes'));
app.use('/api/news', require('./routes/newsRoutes'));
app.use('/api/testimonials', require('./routes/testimonialRoutes'));
app.use('/api/upload', require('./routes/uploadRoutes'));
app.use('/api/reports', require('./routes/reportRoutes'));
app.use('/api/notifications', require('./routes/notificationRoutes'));
```

					2026.KBP.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		87

```

const connectDB = async () => {
  try {
    if (!process.env.MONGO_URI) {
      throw new Error('MONGO_URI не визначено в .env файлі');
    }

    await mongoose.connect(process.env.MONGO_URI);
    console.log('MongoDB успішно підключено');
  } catch (error) {
    console.error('Помилка підключення до MongoDB:', error.message);
    if (process.env.NODE_ENV === 'production') {
      process.exit(1);
    }
  }
};

connectDB();

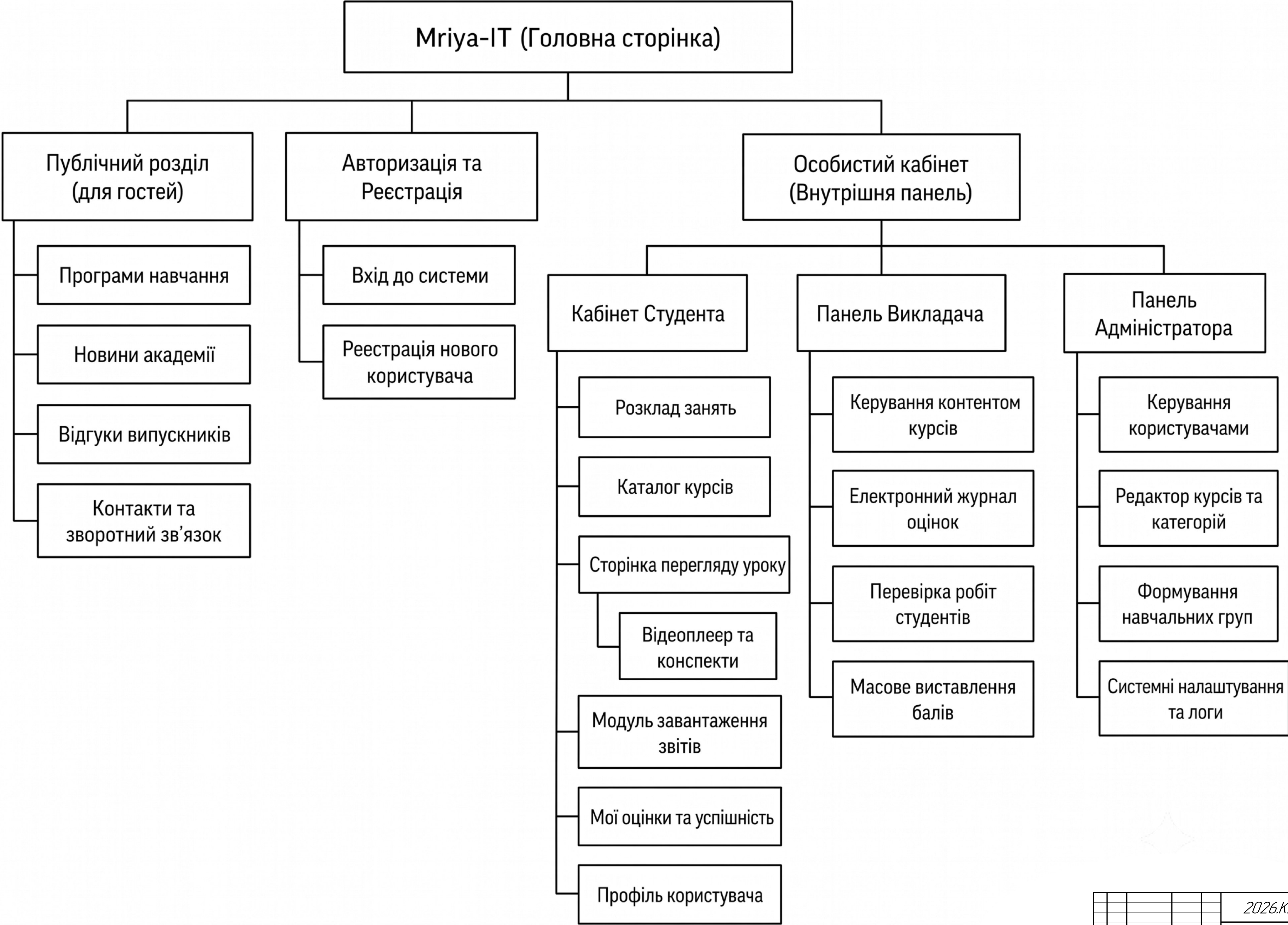
const PORT = process.env.PORT || 5000;
app.listen(PORT, () => console.log(`Сервер запущено на порту ${PORT}`));

module.exports = app;

```

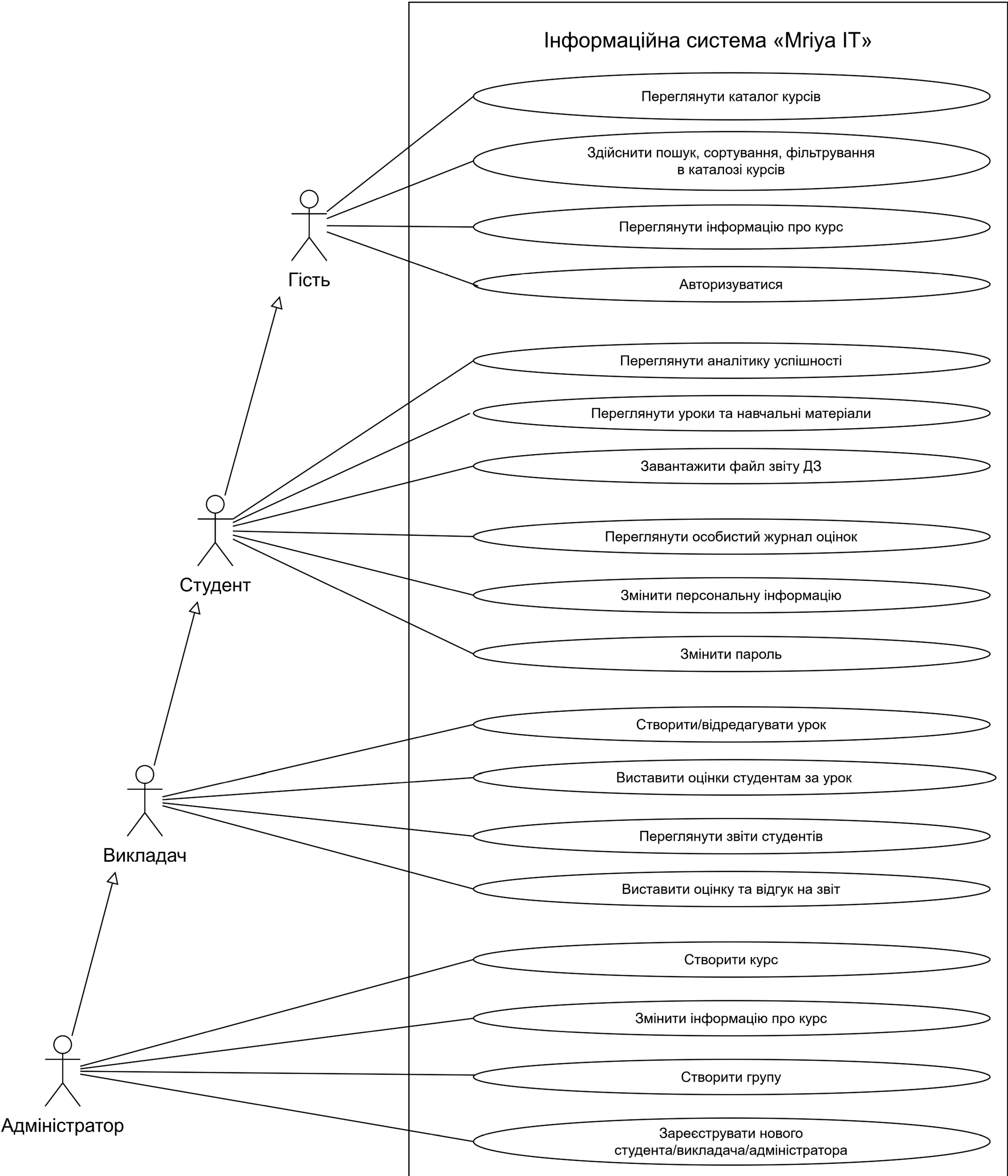
					2026.КВР.122.423.22.00.00 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		88

Схема структурна інформаційної системи

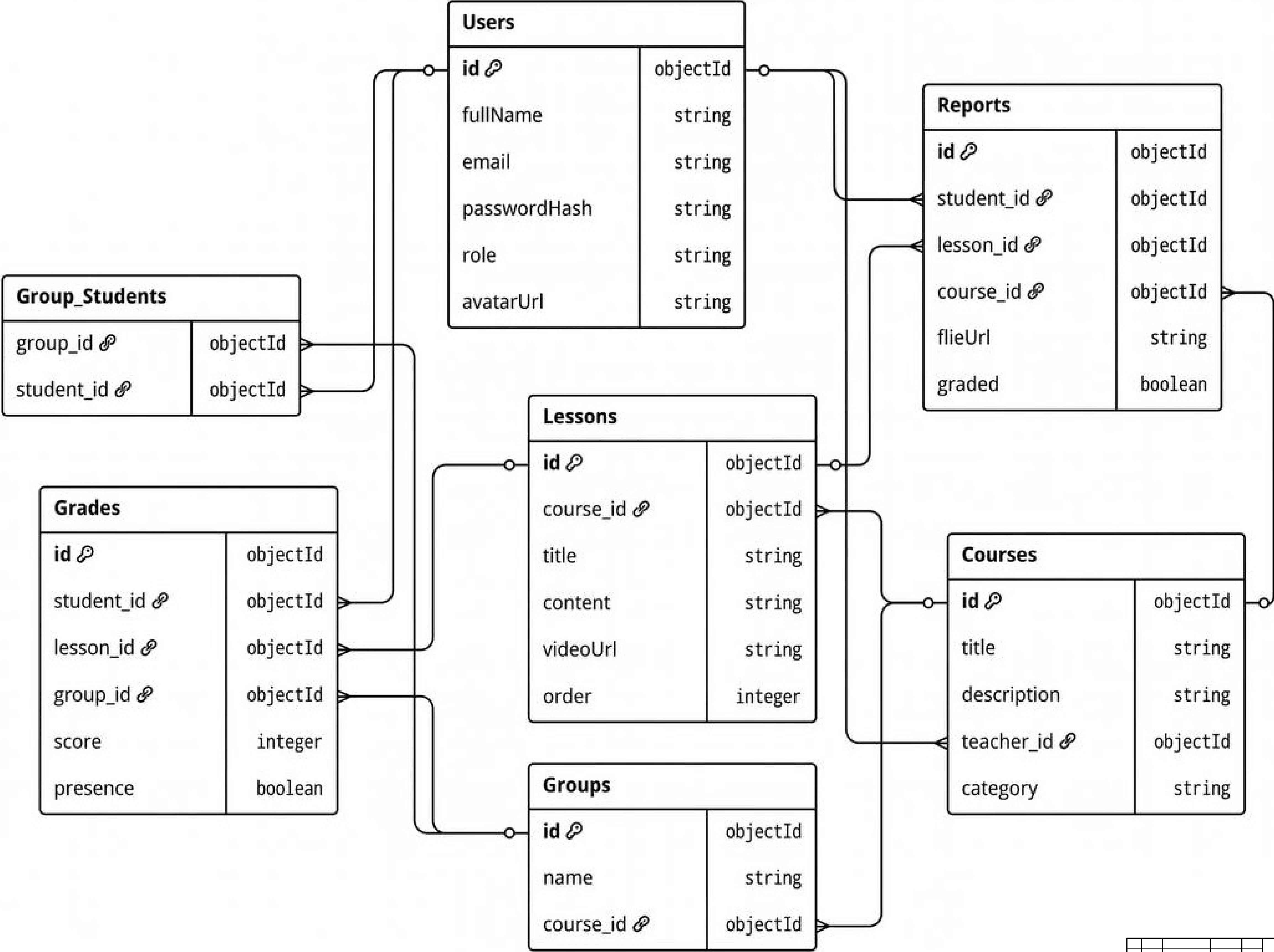


2026.KBP.122.4.23.22.00.00 CC									
Зм.	Арк.	№ докум.	Підпис.	Дата	Розробка інформаційної системи ІТ-школи «Мрія ІТ»				Лист
Розроб.	Тупа Н.Д.				Схема структурна інформаційної системи				Маса
Перевір.	Слободян Р.О.								Масштаб
Т.контр.									Архи
Рецензент									Архив
Н.контр.	Приймак В.А.								1
Затв.									ВСТ ТФК ТНТУ
					м. Тернопіль				КН-423
					Формат А1				

UML-діаграма варіантів використання



ER-діаграма бази даних



Таблиця техніко-економічних показників

№	Показник	Одиниця вимірювання	Значення
1	Мова програмування	–	JavaScript
2	Фреймворк серверної частини	–	Express.js v4.19.2
3	Фреймворк клієнтської частини	–	React.js v18.2
4	Фреймворк для стилізації інтерфейсу	–	Tailwind CSS v3.4.3
5	Обсяг програмного коду	рядок	~6 500
6	Кількість сутностей у базі даних	штук	5
7	Содівартість	грн	68 222
8	Плановий грошовий потік	грн	38 204
9	Ціна	грн	106 426
10	Чиста теперішня вартість	грн	26 547
11	Термін окупності	рік	2,05