Міністерство освіти і науки України
Харківський національний університет імені В. Н. Каразіна
Факультет комп’ютерних наук
Кафедра теоретичної та прикладної системотехніки

«Затверджую»
Зав. кафедри теоретичної та
прикладної системотехніки
_____ д.т.н., проф. С. І. Шматков
«___» _____ 2024 р

Пояснювальна записка

до кваліфікаційної роботи
бакалавра

на тему: «Комп’ютерна модель неформального навчання»

Захищено на засіданні
Атестаційної комісії № 44
протокол № __ від __.06.2024 р.
Оцінка _____ / _____
Голова Атестаційної комісії
_____ Скоб Ю. О.
(підпис) (прізвище та ініціали)

Виконав:
студент 4 курсу, групи КУ– 41
Галузь знань: 15 – Автоматизація та приладобудування
Спеціальність: 151 – «Автоматизація та комп’ютерно-інтегровані технології»
Глущенко Богдан Олександрович
(підпис)

Керівник:
канд. техн.наук, доцент
Булавін Дмитро Олексійович
_____

Рецензент:
д.т.н., доц., професор кафедри
теоретичної та прикладної інформатики
Руккас Кирило Маркович _ _

Харків – 2024

АНОТАЦІЯ

Пояснювальна записка до кваліфікаційної роботи бакалавра складається зі вступу, трьох розділів, висновків, списку використаних джерел і чотирьох додатків. Загальний обсяг роботи складає 71 сторінок, із яких 50 сторінки основної частини з 31 рисунком, 8 таблицями, 14 найменуваннями списку використаних джерел та чотирма додатками.

Метою кваліфікаційної роботи є розробка та імплементація комп'ютерної моделі неформального навчання з метою збагачення освітнього процесу та підвищення ефективності у набутті знань та навичок учасниками.

Об'єкт дослідження – неформальне навчання як процес здобуття знань та навичок поза офіційними освітніми установами.

Предмет дослідження – комп'ютерна модель, спрямована на вдосконалення неформального навчання, включаючи аналіз методів та технологій, що підтримують ефективне вивчення, взаємодію учасників та відслідковування прогресу.

Завдання, яке вирішується в кваліфікаційній роботі полягає у розробці, впровадження та оцінки ефективності комп'ютерної моделі, яка підтримує та оптимізує процеси неформального навчання. Неформальне навчання включає будь-яку форму навчання, що відбувається поза традиційними академічними установами, таких як самоосвіта, навчання на робочому місці, онлайн-курси, вебінари та інші форми навчання через Інтернет.

Область застосування – розробка веб-додатків, спрямована на дистанційне навчання студентів.

Ключові слова: комп'ютерна модель, моделювання, архітектура, Node.js, Java, HTML, CSS, React, MySQL Workbench та Server.

ABSTRACT

The explanatory note to the bachelor's thesis consists of an introduction, three chapters, conclusions, a list of references and four appendices. The total volume of the work is 71 pages, including 50 pages of the main part with 31 figures, 8 tables, 14 references and four appendices.

The purpose of the qualification work is to develop and implement a computer model of non-formal learning in order to enrich the educational process and increase the efficiency of knowledge and skills acquisition by participants.

The object of the study is non-formal learning as a process of acquiring knowledge and skills outside formal educational institutions.

The subject of the study is a computer model aimed at improving non-formal learning, including the analysis of methods and technologies that support effective learning, participant interaction, and progress tracking.

The task that is solved in the qualification work is to develop, implement and evaluate the effectiveness of a computer model that supports and optimizes non-formal learning processes. Non-formal learning includes any form of learning that takes place outside of traditional academic institutions, such as self-education, on-the-job training, online courses, webinars, and other forms of learning via the Internet.

The scope is the development of web applications aimed at distance learning for students.

Keywords: computer model, modeling, architecture, Node.js, Java, HTML, CSS, React, MySQL Workbench and Server.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ 5

ВСТУП 6

РОЗДІЛ 1. АНАЛІЗ ПРИНЦИПІВ ПОБУДОВИ НЕФОРМАЛЬНОГО НАВЧАННЯ..... 8

1.1 Аналіз систем навчання та особливостей викладання на кафедрі ТПС..... 8

1.2 Тенденції неформального навчання..... 9

1.3 Структурна модель неформальної освіти та її декопозиція..... 13

1.4 Мета, предмет, об’єкт та постановка задачі..... 21

Висновки за розділом 1 22

РОЗДІЛ 2. МОДЕЛЮВАННЯ ТА АРХІТЕКТУРА КОМП’ЮТЕРНОЇ МОДЕЛІ 23

2.1 Аналіз поняття «моделювання» 23

2.2 Ключові компоненти моделювання..... 25

2.3 Архітектура комп’ютерної моделі 26

Висновки за розділом 2 33

РОЗДІЛ 3. ДЕПЛОЙ ТА ТЕСТУВАННЯ МОДЕЛІ НЕФОРМАЛЬНОГО НАВЧАННЯ 34

3.1 Обрані мови програмування..... 34

3.2 Використання БД MySQL (Workbench, Server)..... 37

3.3 Тестування комп`ютерної моделі 40

Висновки за розділом 3..... 56

ВИСНОВКИ 57

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 59

ДОДАТКИ 60

ПЕРЕЛІК СКОРОЧЕНЬ І УМОВНИХ ПОЗНАЧЕНЬ

Кафедра ТПС – кафедра теоретичної та прикладної системотехніки;

СУБД – система управління базами даних;

ERD – (англ. Entity-Relationship Diagram) діаграма зв'язків між сутностями;

DFD – (англ. Data Flow Diagram) діаграма потоків даних;

UI – (англ. User Interface) інтерфейс користувача;

CMS – (англ. Content Management System) система управління контентом;

CTA – (англ. Call to Action) заклики до дії;

БД – бази даних;

HTML – (англ. HyperText Markup Language) мова розмітки гіпертексту;

CSS – (англ. Cascading Style Sheets) каскадні таблиці стилів;

JVM – (англ. Java Virtual Machine) віртуальна машина Java;

SQL – (англ. Structured Query Language) мова структурованих запитів.

ВСТУП

З виникненням сучасних технологій комп'ютерна модель неформального навчання стала предметом активного дослідження та зацікавленості в освітній громадськості. Завдяки поєднанню потужності комп'ютерних систем і методології неформального навчання відкриваються нові горизонти для особистісного розвитку та професійного зростання. Ця дипломна робота націлена на систематизацію знань щодо комп'ютерної моделі неформального навчання, виявлення її переваг та можливих обмежень у контексті сучасного освітнього середовища. Шляхом аналізу наукових джерел, вивчення відомих практик та проведення практичних досліджень планується розкрити потенціал комп'ютерної моделі для стимулювання самостійного навчання, розвитку критичного мислення та формування ключових компетентностей. Відповідно, метою роботи є не лише аналіз сучасного стану та тенденцій розвитку комп'ютерної моделі неформального навчання, а й розробка конкретних рекомендацій щодо її впровадження та подальшого вдосконалення з метою сприяння ефективному навчанню та особистісному зростанню.

Актуальність теми. Загальною тенденцією в сучасному світі є ріст важливості комп'ютерних технологій у сфері освіти. У цьому контексті, дослідження та розробка комп'ютерних моделей неформального навчання стає актуальною та значущою проблемою.

Завдяки поширенню інтернету та цифрових технологій, люди мають доступ до величезного обсягу інформації та навчальних ресурсів, які можна використовувати для самонавчання та професійного розвитку. Крім того, популярність онлайн-платформ навчання, таких як Coursera, Udemy, Khan Academy, демонструє попит на ці моделі.

Однією з основних переваг комп'ютерних моделей неформального навчання є їхня гнучкість та доступність. Вони дозволяють користувачам

навчатися в будь-який час і місце, враховуючи їхні потреби та розклад. Крім того, ці моделі можуть бути персоналізовані, надаючи індивідуалізований підхід до навчання, що сприяє кращому засвоєнню матеріалу.

Таким чином, дослідження та розвиток комп'ютерних моделей неформального навчання є важливим напрямком розвитку освіти. Вони допомагають забезпечити доступ до якісної освіти для всіх шарів суспільства та сприяють постійному самовдосконаленню та професійному зростанню.

РОЗДІЛ 1
АНАЛІЗ ПРИНЦИПІВ ПОБУДОВИ НЕФОРМАЛЬНОГО НАВЧАННЯ

1.1 Аналіз систем навчання та особливостей викладання на кафедрі ТПС

Кафедра теоретичної та прикладної системотехніки (ТПС) готує фахівців з системного аналізу, проектування та експлуатації складних систем. Навчання на кафедрі ґрунтується на сучасних методах та технологіях, що дозволяє студентам отримати ґрунтовні знання та навички, необхідні для роботи в цій галузі.

На кафедрі ТПС використовуються такі системи навчання:

- Лекції: лекції є основною формою викладання теоретичних дисциплін. Викладачі використовують різноманітні методи та прийоми для того, щоб зробити лекції цікавими та інформативними.
- Практичні заняття: практичні заняття проводяться для того, щоб закріпити теоретичні знання та набути практичних навичок. На практичних заняттях студенти розв'язують задачі, виконують лабораторні роботи та беруть участь у проектах.
- Семінари: семінари проводяться для того, щоб обговорити актуальні проблеми теорії та практики системотехніки. На семінарах студенти роблять доповіді, ведуть дискусії та обмінюються думками.
- Курсові та дипломні роботи: курсові та дипломні роботи дозволяють студентам застосувати теоретичні знання та практичні навички для розв'язання реальних проблем.
- Виробнича практика: виробнича практика дає можливість студентам ознайомитися з роботою підприємств та організацій у галузі системотехніки.

Викладання на кафедрі ТПС має такі особливості:

- Використання сучасних методів та технологій: викладачі кафедри використовують сучасні методи та технології навчання, такі як проблемно-орієнтоване навчання, проектно-орієнтоване навчання, використання комп'ютерних технологій тощо.
- Індивідуальний підхід до студентів: викладачі кафедри намагаються враховувати індивідуальні особливості та можливості кожного студента.
- Практична спрямованість навчання: навчання на кафедрі ТПС має практичну спрямованість. Студенти отримують знання та навички, які необхідні для роботи в реальних умовах.
- Зв'язок з практикою: кафедра ТПС має тісні зв'язки з підприємствами та організаціями у галузі системотехніки. Це дозволяє студентам проходити практику та працевлаштуватися після закінчення навчання.

Системи навчання та особливості викладання на кафедрі теоретичної та прикладної системотехніки дозволяють студентам отримати ґрунтовні знання та навички, необхідні для роботи в цій галузі. Кафедра ТПС випускає висококваліфікованих фахівців, які користуються попитом на ринку праці.

Для подальшого вдосконалення систем навчання та особливостей викладання на кафедрі ТПС рекомендується:

- Продовжувати використовувати сучасні методи та технології навчання.
- Розширювати зв'язки з підприємствами та організаціями у галузі системотехніки.
- Створити умови для самостійного навчання та наукової роботи студентів.

1.2. Тенденції неформального навчання

Нині суспільство вступило в нову фазу свого розвитку - фазу глобальної інформатизації.

Інформатизація - це процес становлення життєдіяльності суспільства, заснований на безперервному активному використанні достовірних вичерпних і своєчасних знань у всіх ключових видах людської діяльності, для чого необхідно створити систему, що складається з засобів для зберігання, збереження, обробки та передачі інформації [1]. Відповідно, інформація в даному випадку є найважливішим суспільним ресурсом, який відіграє провідну роль в освіті, економіці та інших сферах.

Важливе місце в процесі інформатизації суспільства має посідати інформатизація освіти. Впровадження сучасних інформаційних технологій у систему освіти дасть змогу підвищити функціональну ефективність системи, забезпечити її повний функціональний цикл. Ідеї створення та просування сучасних технологій навчання, що базуються на засадах загальної комп'ютеризації та інформатизації педагогічних систем, зумовлюється такими світовими тенденціями [2]:

- розвиток всесвітньої виробничої інфраструктури;
- інформатизація та автоматизація всіх галузей;
- зміна професійної структури суспільства та світоглядних поглядів людини на працю;
- інформаційна інтеграція освіти у світову систему.

Таким чином, перспективи сучасної освіти багато в чому пов'язані з удосконаленням інформаційно-комунікаційних технологій, і, своєю чергою, створенням інноваційних педагогічних методик, що ґрунтуються на їх упровадженні в одній із форм освіти - неформальній освіті.

Неформальна освіта проводиться незалежно від освітніх програм і кваліфікацій, тобто є альтернативною або такою, що доповнює формальну освіту, і має тривати протягом усього кар'єрного шляху.

Аналізуючи тенденції розвитку інформаційного суспільства, зазначимо, що система неформальної освіти може характеризуватися такими ключовими особливостями:

1. Функціонування лише за глобального використання інформаційних технологій.
2. Врахування можливостей і потреб кожного учня за рахунок індивідуалізації освіти.
3. Орієнтація на результат, що забезпечує високий рівень практичних навичок.
4. Формування тенденції безперервної освіти, тобто підвищення кваліфікації протягом усього життя.

Однією з обов'язкових і ключових особливостей неформальної освіти є спрямованість на практичну діяльність. Саме практична спрямованість неформальної освіти та можливість навчатися дистанційно більше приваблює аудиторію дорослих, ніж аудиторію абітурієнтського віку.

Навчання в такій ситуації набуває форм індивідуально-орієнтованого, гнучкого, і що найважливіше безперервного. Освітніми заходами неформальної освіти є [2]:

- короткострокові колективні заняття;
- індивідуальні заняття, що переслідують практичні цілі;
- курси підвищення кваліфікації.

Такі освітні заходи організовуються і проводяться поза формальною системою з використанням дистанційного навчання. У зв'язку з цим ефективність такої системи безпосередньо залежить від використання сучасних інформаційних технологій.

Глобальна мета повсюдного впровадження інформатизації в освіту є підвищення якості освіти та забезпечення відповідності новим вимогам постіндустріального суспільства. Така мета, безумовно, є багатофакторною і містить у собі безліч підцілей:

- перепідготовка учнів для ефективного працевлаштування в інформаційному суспільстві;
- верифікація якості освіти;
- розширення меж і ступеня доступності освіти;

- інформаційна інтеграція системи неформальної освіти у світову інфраструктуру.

Для кращого розуміння різниці між формальним та неформальним навчанням треба подивитися таблицю 1.1, де їх порівняно.

Таблиця 1.1

Порівняння формального та неформального навчання

Характеристика	Формальне навчання	Неформальне навчання
Структура	Чітко визначена програма, розклад, оцінка	Відсутність строгої структури, свобода вибору тем та темпу
Методи навчання	Лекції, практичні заняття, лабораторні роботи	Самонавчання, наставництво, дослідницька діяльність, співпраця
Оцінка успішності	Оцінки, тестування, екзамени	Офіційна оцінка може бути відсутня, основна увага - на розвиток
Середовище навчання	Фізичний клас або лабораторія	Інтернет, робоче місце, практичне застосування у реальних ситуаціях
Фінансові витрати	Зазвичай високі (плата за курси, навчальні матеріали)	Зазвичай низькі або відсутні, доступність безкоштовних ресурсів
Правила та процедури	Суворі правила та процедури	Більш вільний та гнучкий підхід, менше формальності
Гнучкість та індивідуалізація	Обмежена гнучкість у навчанні та оцінці	Висока гнучкість, можливість адаптації до потреб індивіда
Мотивація та інтереси	Залежить від оцінок та формальних цілей навчання	Залучення та стимулювання зацікавленості та особистих цілей

Неформальне навчання може бути кращим за формальне з-за своєї спроможності адаптуватися до індивідуальних потреб, сприяти самовизначенню та особистому зростанню, а також надавати практичний досвід та вміння, які можуть бути більш цінними у реальному житті та професійній кар'єрі.

1.3 Структурна модель системи неформальної освіти та її декомпозиція

Система сучасної неформальної освіти, що ґрунтується на нових інформаційно-комунікаційних технологіях, може містити в собі такі основні взаємопов'язані підсистеми: економічна, педагогічна, технологічна, організаційна, теоретико-методологічна, теоретико-методологічна (Рис. 1) [3].

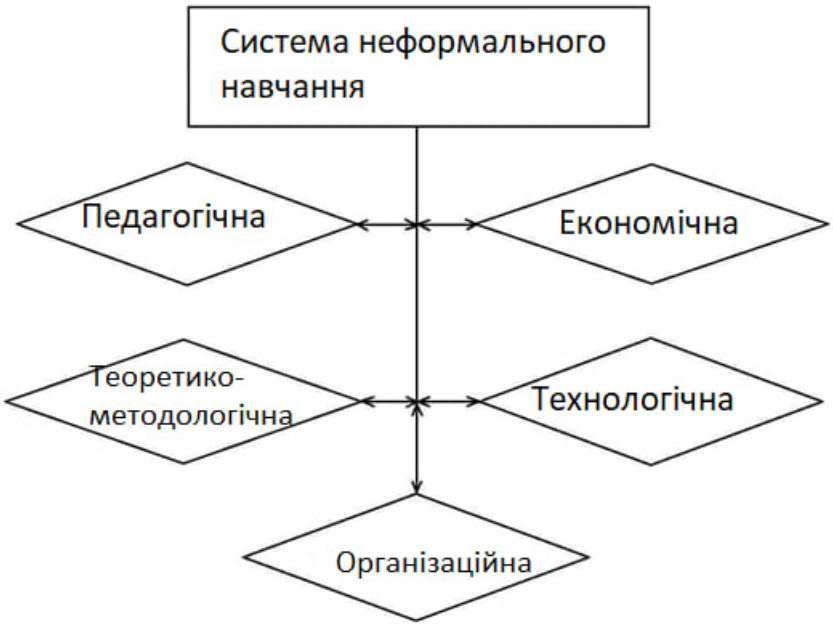


Рис. 1.1 - Структурна модель системи неформальної освіти

Розглянемо підсистеми, що входять до системи неформальної освіти, і сформулюємо основні завдання, які вирішує кожна підсистема.

Розвиток педагогічної підсистеми можна розбити на три покоління. Перше покоління характеризується дисциплінарно-орієнтованим підходом, у

якому інформаційні технології навчання розглядаються як цілісний навчальний процес, що ґрунтується на традиційному змісті, формах і методах навчання. Він підтримується класичними підручниками, задачниками та методичними посібниками. Комп'ютер у цій моделі використовується для представлення готових знань і посилення контролю за їх засвоєнням. Важливою і характерною рисою традиційної системи освіти є її навчально-дисциплінарна структура.

Таким чином, інформаційні технології першого покоління виявилися нестійкими через наявні в них протиріччя між вимогами традиційної системи навчання та невикористаними можливостями інформаційних систем.

До другого покоління належить перехідний підхід, і інформаційні технології навчання уявляються як суперечливі освітні композиції, що ґрунтуються на традиційному змісті, в яких, однак, використовується несистематизовані комбінації з класичних і модернізованих форм і методів навчання. Він підтримується традиційними підручниками, задачниками та методичними посібниками, а також сучасними комп'ютерними програмами та освітніми середовищами, здебільшого зорієнтованими на процеси всебічного дослідження моделей реального світу. Інформаційна технологія навчання другого покоління - нестійка, оскільки за своїм фундаментом вона призначена для дисциплінарно-орієнтованої системи, а за своєю надбудовою тяжіє до міждисциплінарної об'єктно-орієнтованої системи навчання. Але водночас поява таких технологій навчання свідчить про те, що відбувається природне "проростання" нових об'єктно-орієнтованих освітніх моделей.

До третього покоління належить проектно-орієнтований підхід, і технологію навчання розглядають як єдиний освітній процес, що ґрунтується на міждисциплінарному нетрадиційному змісті, формах, методах і засобах навчання. Інформаційні технології навчання третього покоління, за своїм фундаментом і надбудовою, призначені для проєктно-орієнтованої системи навчання, у процесі якої здійснюється не тільки контроль за засвоєнням знань,

а насамперед активне їхнє використання для творення в межах освітнього процесу.

Інформаційні системи в такій моделі навчання є одним із найважливіших складових елементів, що дає змогу не тільки формувати в людині образні уявлення про навколишнє середовище, а й самій брати активну участь у їх створенні.

Структурна модель педагогічної підсистеми може мати такий вигляд (Рис. 1.2). Вона являє собою сукупність блоків, які у взаємодії між собою забезпечують процес неформальної освіти учнів.

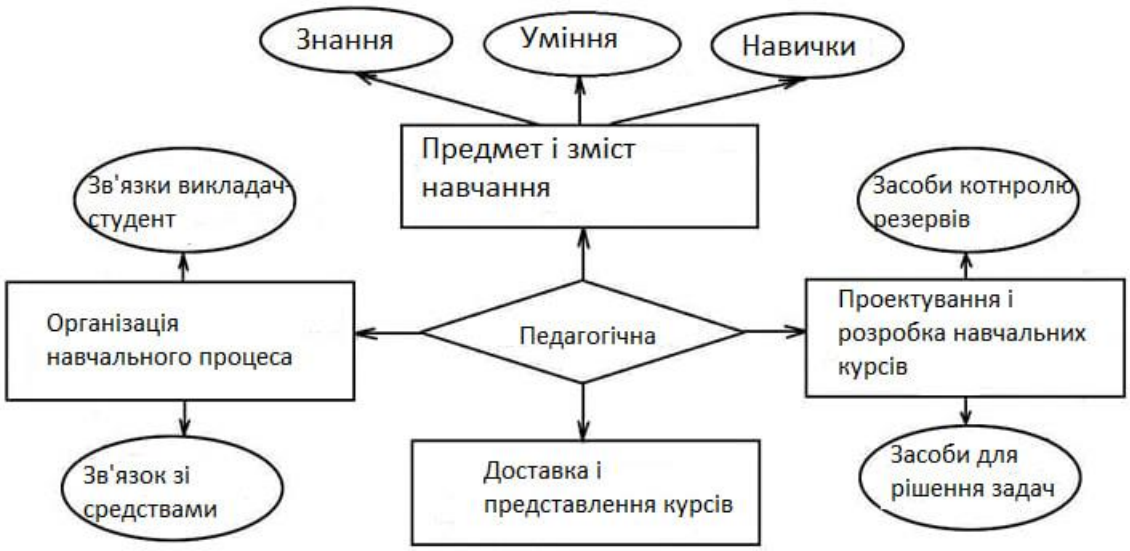


Рис. 1.2 - Педагогічна підсистема неформальної освіти

Основними елементами моделі педагогічної підсистеми неформальної освіти є блоки:

- предмет і зміст курсу - визначає сукупність знань, умінь і навичок, якими може оволодіти той, хто навчається;
- проєктування і розробка навчальних курсів - надає інструментарій для реалізації всіх видів навчальних курсів і контролю результатів їх проєктування;

- доставка та представлення курсів - використовує платформу дистанційного навчання та комп'ютерну мережу для взаємодії;
- організація навчального процесу - налагодження та підтримка зв'язків між усіма суб'єктами, залученими до процесу навчання.

На даний момент існує безліч платформ дистанційного навчання і систем управління базами даних (СУБД), які можна використовувати в такій підсистемі. Кожна з цих інформаційних технологій має свої переваги та недоліки, саме тому актуальним є вирішення завдання раціонального вибору відповідних інформаційних технологій.

Економічна підсистема в межах системи неформальної освіти забезпечує виконання таких функцій: соціально-фінансове управління, зовнішньоекономічні зв'язки, матеріально-маркетингові послуги.

Таким чином, архітектура економічної підсистеми складається з восьми блоків, які забезпечують реалізацію всіх вищеописаних функцій (рис. 1.3).

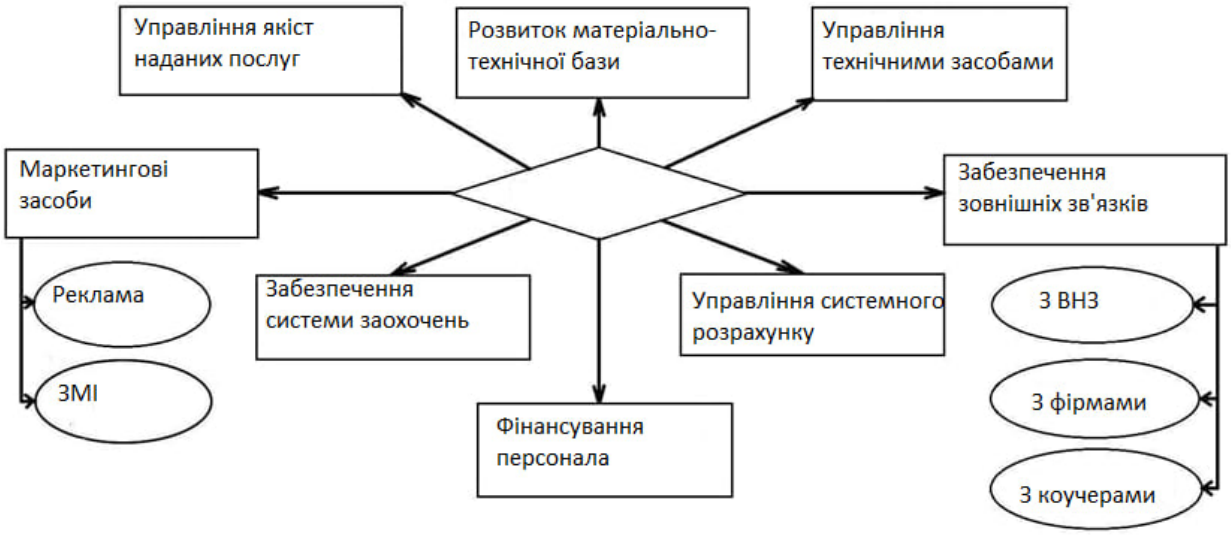


Рис. 1.3 - Економічна підсистема неформальної освіти

Функція соціально-фінансового управління представлена блоками - "Фінансування персоналу", "Забезпечення системи заохочень" і "Управління системою розрахунків". Автоматизація цих функцій може бути реалізована за

допомогою сучасних інформаційних технологій, а саме електронною білінговою системою та системою онлайн-банкінгу.

Забезпечення зовнішньоекономічної функції здійснює блок "Забезпечення зовнішніх зв'язків". За допомогою цієї функції система взаємодіє з вищими навчальними закладами, компаніями, які спеціалізуються на напрямках навчання, які представляють, а також з коучерами всього світу, що надають свої послуги.

Третя функція - матеріально-маркетингові послуги - спрямована на забезпечення постійної підтримки матеріальної бази, якості послуг і рекламного просування системи. Реалізовано цю функцію у вигляді 4 блоків: "Розвиток матеріально-технічної бази", "Управління технічними засобами", "Управління якістю надання послуг" і "Маркетингові засоби".

Ефективна робота цієї підсистеми забезпечує зростання фінансової складової всієї системи, і необхідно дуже виважено підійти до вибору інформаційних технологій, на яких вона буде реалізована.

Організаційна підсистема забезпечує взаємозв'язок усіх підсистем у єдину інформаційну мережу, використовуючи при цьому електронний документообіг, бази даних та інші комп'ютерні технології. Під час проведення дослідження модель цієї підсистеми було декомпоновано на блоки та завдання, які відображено на рис. 1.4.

Рис. 1.4 - Організаційна підсистема неформальної освіти

Організаційна підсистема пов'язана з економічною підсистемою через блок "Керування системою заохочень", з педагогічною - блоки "Керування навчальним середовищем", "Навчальний план", "Система перевірки знань", "Організація та зберігання даних". А також окремий напрямок це "Органи управління", що забезпечує автоматизований обмін і затвердження документації з надання освітніх послуг [4].

Існування системи неформальної освіти неможливе без наявності методологічної бази, що містить у своєму складі грамотно складені методики виконання різноманітних функцій.

Пропонується структурна модель теоретико-методологічної підсистеми, що ґрунтується на кількох напрямах, які реалізовано у вигляді блоків (рис. 1.5):

- "Бази знань і матеріалів" - використовується педагогічною підсистемою;
- "Способи подання даних" - взаємодія з платформами дистанційного навчання для зручності учнів;
- "Методи оцінки та перевірки знань" і "Тестування знань" - використовується в педагогічній підсистемі, за допомогою засобів технологічної підсистеми;
- "Способи накопичення знань" - методичні засади організації курсів навчання, що надалі використовуються платформою дистанційного навчання;
- "Способи та критерії відбору викладачів" - реалізація принципів призначення викладачів, які курують курси.

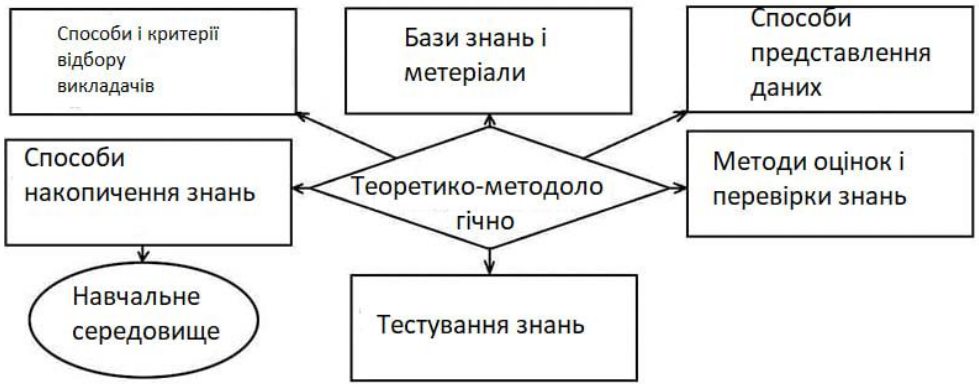


Рис. 1.5 - Теоретико-методологічна підсистема неформальної освіти

Роль технологічної підсистеми в модернізації освіти полягає в постійному використанні нових інформаційних технологій. При цьому розвиток технологічної підсистеми освіти супроводжується радикальними змінами в усіх інших підсистемах: педагогічній, організаційній, економічній - і навіть суттєво зачіпає теоретичні та методологічні засади освітньої системи. Тобто, розвиток технологічної підсистеми тягне за собою становлення принципово нової освітньої системи, що може забезпечити надання освітніх послуг мільйонам людей за скорочення питомих витрат на освіту.

Принциповою відмінністю системи неформальної освіти від формальної є її повна залежність від технологічної підсистеми. Слабо задіяна в класичній освіті, що ґрунтується на навчанні "віч-на-віч" і друкованих матеріалах, у неформальній освіті є ключовим базисом, на який спираються всі інші підсистеми. Вона організовує роботу всіх інформаційних технологій, задіяних на всіх рівнях [5].

Таким чином, розвиток і застосування інформаційних технологій дає змогу створити принципово нову освітню систему.

Структурна модель технологічної підсистеми, представлена на рис. 1.6, являє собою блоки, які забезпечують роботу всіх підсистем системи неформальної освіти [6].

Основними елементами технологічної підсистеми неформальної освіти є блоки:

- освітня платформа - забезпечення роботи платформи дистанційного навчання, що діє в педагогічній підсистемі;
- область зберігання даних - підтримка роботи СУБД, яка використовується в педагогічній і теоретико-методологічній підсистемах;
- забезпечення каналу якісної доставки - комп'ютерна мережа, з постійним контролем маршрутизації потоків;
- спосіб зв'язків (викладач-студент) - підтримка он-лайн взаємодії між усіма суб'єктами;
- система контролю знань - забезпечує взаємодію теоретико-методологічної, організаційної та педагогічної підсистем для оцінки якості проходження навчання;
- область пошуку та обміну даними і ресурсами - використання хмарних технологій для зберігання, обміну та пошуку інформації;
- система розрахунків - технічна реалізація білінгової функції, яка є ключовою в економічній підсистемі.

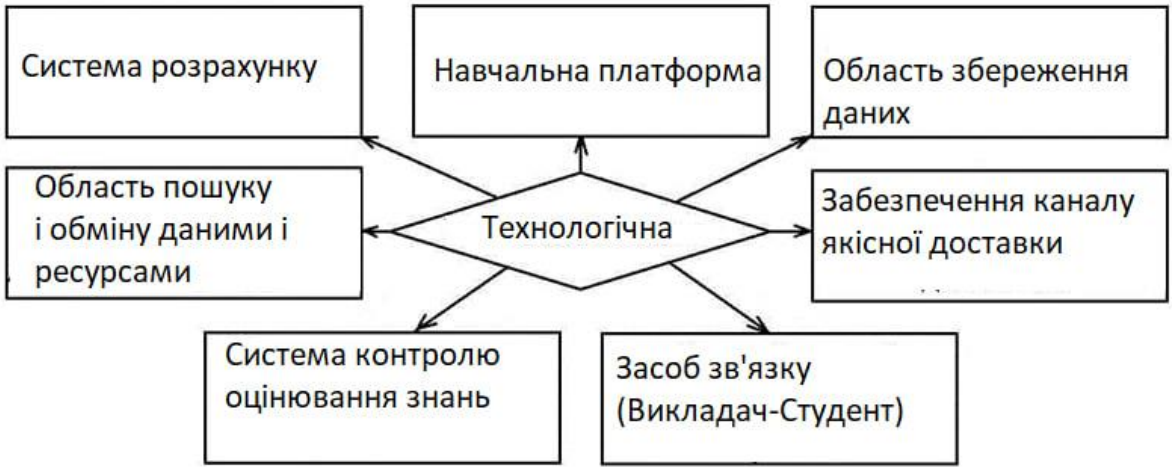


Рис.1.6 - Технологічна підсистема неформальної освіти

Використовуючи основний принцип системного аналізу, проведено декомпозицію системної моделі на часткові моделі підсистем. Це дало змогу сформулювати основні функціональні завдання кожного структурного

елемента системи. Показано, що однією з ключових підсистем є технологічна і, отже, її вдосконалення дасть змогу підвищити ефективність функціонування всієї системи в цілому.

1.4. Мета, предмет, об’єкт та постановка задачі.

Метою кваліфікаційної роботи є розробка та імплементація комп'ютерної моделі неформального навчання з метою збагачення освітнього процесу та підвищення ефективності у набутті знань та навичок учасниками.

Об’єкт дослідження – неформальне навчання як процес здобуття знань та навичок поза офіційними освітніми установами.

Предмет дослідження – комп'ютерна модель, спрямована на вдосконалення неформального навчання, включаючи аналіз методів та технологій, що підтримують ефективне вивчення, взаємодію учасників та відслідковування прогресу.

Завдання:

1. Аналіз існуючих моделей неформального навчання: порівняти програмні продукти та онлайн-платформи для неформального навчання, визначити їх переваги та недоліки.
2. Розробка власної комп’ютерної моделі: описати структуру, функціональні можливості та методи оцінки ефективності власної комп’ютерної моделі для неформального навчання.
3. Імплементація та тестування: Реалізувати власну модель та провести тестування з учасниками для визначення її ефективності.
4. Вплив моделі на навчання: Провести експерименти для оцінки впливу комп’ютерної моделі на мотивацію та засвоєння знань учасників.
5. Оптимізація та рекомендації: Використати результати досліджень для удосконалення моделей неформального навчання та розробити рекомендації щодо їх впровадження в освітній процес.

Висновки за розділом 1

Було проведено всебічний аналіз принципів побудови неформального навчання. Основну увагу зосереджено на системах навчання та особливостях викладання на кафедрі ТПС, розглянуто сучасні тенденції у сфері неформальної освіти. Було побудовано та декомпозовано структурну модель неформальної освіти, що дозволило виявити ключові елементи та взаємозв’язки в системі неформального навчання. Це дослідження створює основу для подальшого розвитку і впровадження інноваційних підходів у неформальну освіту, зокрема на рівні кафедри.

РОЗДІЛ 2
МОДЕЛЮВАННЯ ТА АРХІТЕКТУРА КОМП'ЮТЕРНОЇ МОДЕЛІ

2.1. Аналіз поняття «моделювання»

Аналітична модель є технічним представленням системи і слугує сполучною ланкою між описом системи та проектною моделлю. В аналітичному моделюванні визначаються інформація, поведінка і функціональність системи, які на етапі проектного моделювання трансформуються в архітектуру, компоненти і проекти на рівні інтерфейсів [7].

Цілі моделювання сайту:

1. Розуміння вимог: процес аналітичного моделювання допомагає зрозуміти та виокремити вимоги користувачів до програмної системи.
2. Комунікація: Аналітичне моделювання допомагає у спілкуванні між користувачами, клієнтами, розробниками, тестувальниками та іншими зацікавленими сторонами.
3. Прояснення неоднозначностей: аналітичне моделювання допомагає у вирішенні суперечок щодо вимог та проясненні неоднозначностей.
4. Виявлення вимог до даних: аналітичне моделювання допомагає визначити зв'язки даних, сутності та атрибути, необхідні системі.
5. Визначення поведінки: аналітичне моделювання допомагає визначити динамічну поведінку системи, включаючи робочі потоки, процеси та взаємодію між компонентами.
6. Визначення меж системи: аналітичне моделювання допомагає визначити межі програмної системи та її взаємодію з користувачами, іншими системами та апаратними компонентами.

Нижче можно побачити елементи моделювання сайту на рис. 2.1.

Рис. 2.1 – Елементи моделювання

Аналізуючи рис. 2.1 розберемося детальніше про кожний елемент:

- Словник даних: сховище, що містить описи всіх об'єктів даних, які використовуються або генеруються програмним забезпеченням. Це централізоване сховище, яке допомагає моделювати об'єкти даних, визначені в процесі розробки вимог до програмного забезпечення.
- Діаграма зв'язків між сутностями (ERD): описує зв'язки між об'єктами даних і використовується для моделювання даних. Атрибути кожного об'єкта в ERD можуть бути описані в термінах опису об'єкта даних, що забезпечує основу для діяльності з проєктування даних.
- Діаграма потоків даних (Data Flow Diagram, DFD): ілюструє функцію перетворення потоків даних і показує, як дані трансформуються від входу до виходу. Вона надає інформацію, яка використовується в процесі аналізу інформаційної області та слугує основою для функціонального моделювання.
- Діаграми переходів станів: показують різні режими (стани) поведінки системи та переходи між цими станами. Детально описує, як система реагує на зовнішні події, і описує дії, що виконуються в результаті певних подій.

- Специфікація процесу: зберігає опис кожної функції в DFD, включаючи входи, алгоритми перетворення та виходи. Вона також детально описує правила, обмеження продуктивності та макетні обмеження, які впливають на реалізацію процесу.
- Специфікація контролю: містить інформацію про аспекти керування програмним забезпеченням, описуючи поведінку програмного забезпечення при виникненні подій і процесів, викликаних цими подіями. Детально описує процеси, які виконуються для управління подіями.
- Опис об'єктів даних: надає вичерпну інформацію про об'єкти даних, що використовуються в програмному забезпеченні, включаючи атрибути, детально описані в ERD, охоплюючи всі об'єкти даних та їхні атрибути.

2.2. Ключові компоненти моделювання

Веб-сайт можна розділити на два ключові компоненти: інтерфейс і серверну частину [7].

Інтерфейс - це те, що бачать і з чим взаємодіють користувачі. Це все про дизайн і функціональність. Ось деякі основні компоненти інтерфейсу:

- Контент: це інформація, яку ви хочете донести до своїх відвідувачів. Він може включати текст, зображення, відео, інфографіку тощо.
- Навігація: дозволяє користувачам орієнтуватися на вашому веб-сайті. Вона включає меню, хлібні крихти та пошукові рядки.
- Дизайн: включає в себе візуальний стиль вашого веб-сайту, в тому числі макет, колірну гамму, типографіку та зображення.
- Інтерфейс користувача (UI): це кнопки, форми та інші елементи, з якими взаємодіють користувачі.

Back-end - це движок, який забезпечує роботу вашого веб-сайту. Саме тут зберігаються та обробляються всі дані. Ось деякі ключові компоненти серверної частини:

- Сервер: це комп'ютер, на якому зберігаються файли вашого веб-сайту і який робить їх доступними для користувачів.
- База даних: це місце, де зберігаються всі дані вашого веб-сайту, такі як текст, зображення та інформація про користувачів.
- Система управління контентом (CMS): це програмне забезпечення, яке дозволяє легко створювати та керувати контентом вашого веб-сайту.

На додаток до цих основних компонентів, є багато інших елементів, які можна додати до веб-сайту, щоб покращити його функціональність і зручність для користувачів. До них відносяться:

- Заклики до дії (CTA): це підказки, які спонукають користувачів виконати певну дію, наприклад, підписатися на розсилку новин або здійснити покупку.
- Аналітика: дозволяє відстежувати, як користувачі взаємодіють з вашим сайтом, щоб ви могли приймати обґрунтовані рішення щодо дизайну та контенту.
- Безпека: це важливо для захисту вашого веб-сайту та даних користувачів від хакерів [8].

Розуміючи ключові компоненти моделі веб-сайту, ми можемо створити веб-сайт, який буде одночасно інформативним і зручним для користувачів.

2.3. Архітектура комп’ютерної моделі

Архітектура комп'ютерної моделі - це структурний та функціональний опис комп'ютерної системи або її компонентів, який визначає їх взаємозв'язки, розподіл функцій та відповідальностей, принципи взаємодії між ними та загальну організацію. Це абстрактне поняття, що описує архітектурні принципи та концепції, які лежать в основі побудови комп'ютерної системи [9].

Архітектура комп'ютерної моделі включає такі ключові аспекти:

- Характеристики апаратного забезпечення: це включає процесор, пам'ять, пристрої зберігання даних, пристрої введення/виведення та інші апаратні компоненти, що складають систему.
- Архітектурні принципи: це визначає загальну структуру системи, наприклад, як компоненти взаємодіють між собою та зовнішніми системами, які принципи додаткові до розширення системи, як використовувати абстракції для зменшення складності тощо.
- Організація пам'яті: це визначає способи організації та управління доступом до пам'яті системи, включаючи використання кешування та віртуальної пам'яті.
- Операційні системи та середовище виконання програм: це включає в себе операційну систему та інші середовища, необхідні для запуску та виконання програм на комп'ютері.
- Мережеві з'єднання: Це описує структуру та протоколи, які використовуються для з'єднання комп'ютерів у мережі та обміну даними між ними.
- Безпека: Це включає в себе заходи безпеки, які призначені для захисту системи від несанкціонованого доступу та зловмисного використання.

Архітектура комп'ютерної моделі визначає загальні принципи, які лежать в основі побудови комп'ютерної системи, і відображає взаємозв'язки між її складовими частинами. Вона служить основою для розробки та реалізації конкретних комп'ютерних систем та додатків .

На діаграмі (рис.2.2) зображено кілька елементів , таких як “Користувач”, “Адреса”, “Категорія”, “Курс”, “Розділ курсу”.

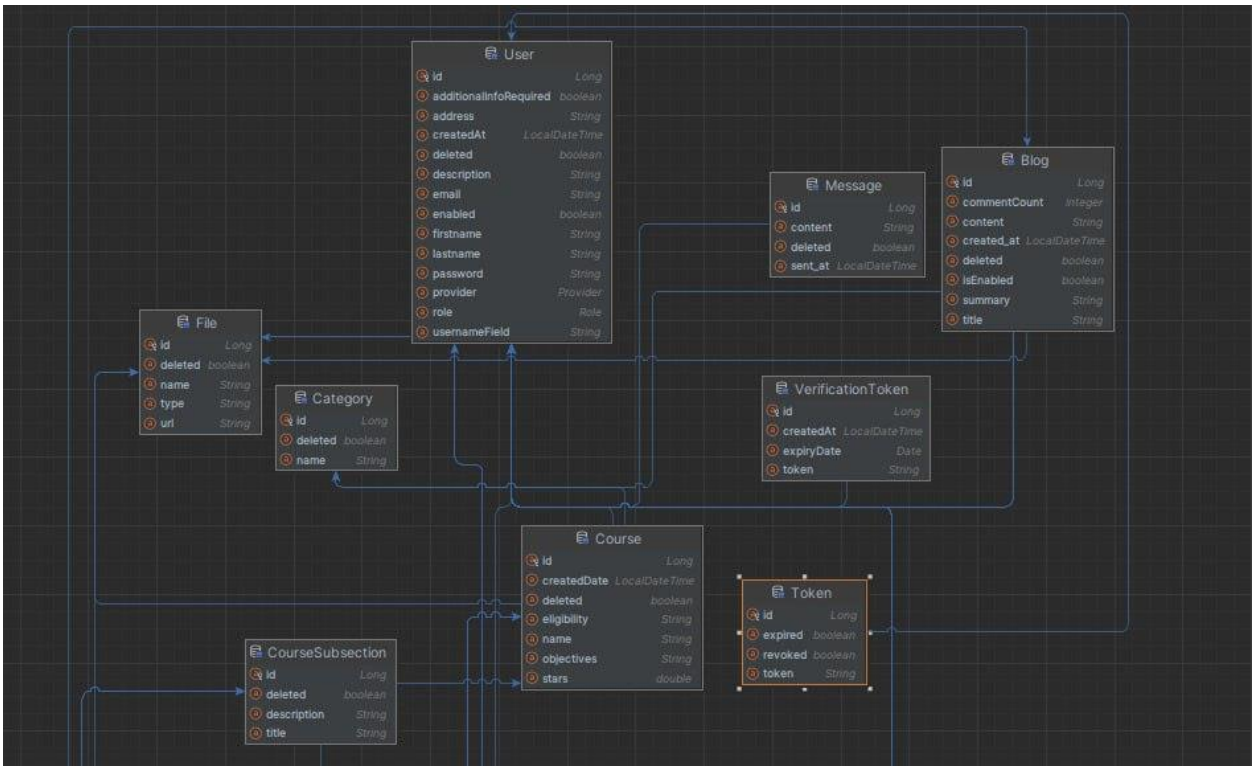


Рис. 2.2 – Діаграма класів

Елементи та їх атрибути:

1. Користувач (User):
   - id: унікальний ідентифікатор користувача;
   - username: логін користувача;
   - email: електронна пошта користувача;
   - password: пароль користувача;
   - first_name: ім’я користувача;
   - last_nam: прізвище користувача.
2. Адреса (Address):
   - id: Унікальний ідентифікатор адреси.
3. Курс (Course):
   - id: Унікальний ідентифікатор курсу;
   - title: Назва курсу;
   - description: Опис курсу;
   - start_date: Дата початку курсу;
   - end_date: Дата завершення курсу;

- category_id: Ідентифікатор категорії, до якої належить курс.

4. Розділ курсу (Course Section):
   - id: Унікальний ідентифікатор розділу курсу;
   - title: Назва розділу;
   - course_id: Ідентифікатор курсу, до якого належить розділ;
   - content: Зміст розділу.

Взаємозв’язки між елементами:

1. Користувач – Адреса:
   - Один користувач може мати одну або кілька адрес.
   - Відповідно, атрибут користувача може включати `address_id` для зв’язку з таблицею адрес.
2. Курс – Категорія:
   - Один курс належить до однієї категорії, але одна категорія може містити багато курсів.
   - Курс містить атрибут `category_id`, який зв’язується з `id` категорії.
3. Курс - Розділ курсу
   - Один курс може складатися з кількох розділів.
   - Розділ курсу містить атрибут `course_id`, який зв’язується з `id` курсу.

Для розробників така діаграма надає чітке уявлення про структуру бази даних або архітектуру системи, що дозволяє:

- Легше орієнтуватися в структурі даних та взаємозв’язках між ними.
- Зрозуміти, які дані і де зберігаються.
- Забезпечити правильну розробку і реалізацію програмного забезпечення.

Для зацікавлених сторін, наприклад, менеджерів чи клієнтів, діаграма допомагає:

- Оцінити складність системи управління даними.
- Поняти, як різні компоненти взаємодіють між собою.
- Більш ефективно планувати розробку та вдосконалення системи.

Розглянемо також ще одну схему БД на рис 2.3.

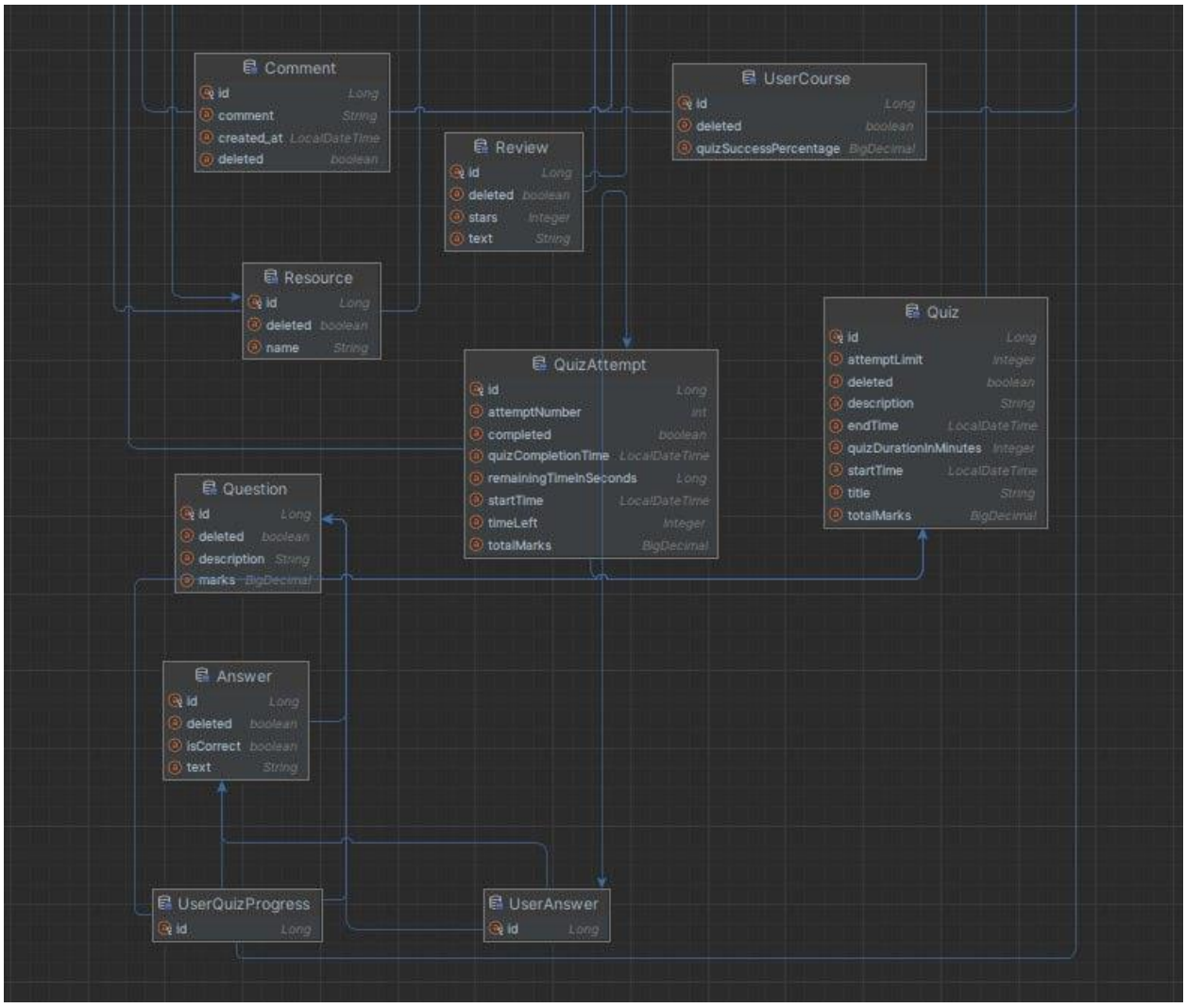


Рис. 2.3 – Схема БД

1.Comment:

- id (Long): унікальний ідентифікатор коментаря;
- comment (String): текст коментаря;
- created_at (LocalDateTime): дата і час створення коментаря;

- deleted (boolean): статус видалення коментаря.

2. Review:
   - id (Long): унікальний ідентифікатор відгуку;
   - deleted (boolean): статус видалення відгуку;
   - stars (Integer): кількість зірок (рейтинг);
   - text (String): текст відгуку.
3. Resource:
   - id (Long): унікальний ідентифікатор ресурсу;
   - deleted (boolean): статус видалення ресурсу;
   - name (String): назва ресурсу.
4. QuizAttempt:
   - id (Long): унікальний ідентифікатор спроби проходження тесту;
   - attemptNumber (Integer): номер спроби;
   - completed (boolean): статус завершення спроби;
   - quizCompletionTime (LocalDateTime): час завершення тесту;
   - remainingTimeInSeconds (Long): залишковий час у секундах;
   - startTime (LocalDateTime): час початку спроби;
   - timeLeft (Integer): залишковий час;
   - totalMarks (Integer): загальна кількість балів.
5. Question:
   - id (Long): унікальний ідентифікатор питання;
   - deleted (boolean): статус видалення питання;
   - description (String): опис питання;
   - marks (BigDecimal): кількість балів за питання.
6. Answer:
   - id (Long): унікальний ідентифікатор відповіді;
   - deleted (boolean): статус видалення відповіді;
   - isCorrect (boolean): чи є відповідь правильною;
   - text (String): текст відповіді.
7. UserQuizProgress:

- id (Long): унікальний ідентифікатор прогресу користувача в тесті.

8. UserAnswer:
   - id (Long): унікальний ідентифікатор відповіді користувача.
9. UserCourse:
   - id (Long): унікальний ідентифікатор курсу користувача;
   - deleted (boolean): статус видалення курсу користувача;
   - quizSuccessPercentage (BigDecimal): відсоток успішного проходження тесту.
10. Quiz
    - id (Long): унікальний ідентифікатор тесту;
    - attemptLimit (Integer): ліміт спроб;
    - deleted (boolean): статус видалення тесту;
    - description (String): опис тесту;
    - endTime (LocalDateTime): час завершення тесту;
    - quizDurationInMinutes (Integer): тривалість тесту в хвилинах;
    - startTime (LocalDateTime): час початку тесту;
    - title (String): назва тесту;
    - totalMarks (BigDecimal): загальна кількість балів за тест.

Взаємозв'язки між таблицями:

- Comment пов'язаний з іншими таблицями для коментування різних сутностей (курсів, відгуків, ресурсів тощо).
- Review пов'язаний з таблицями для створення відгуків про курси, ресурси або інші сутності.
- Resource може бути пов'язаний з різними сутностями, що надають додаткову інформацію або матеріали.
- QuizAttempt пов'язаний з Quiz, UserQuizProgress, UserAnswer для зберігання спроб проходження тестів користувачами.
- Question та Answer пов'язані між собою для створення запитань та відповідей до тестів.

- UserQuizProgress та UserAnswer зберігають інформацію про прогрес та відповіді користувачів у тестах.
- UserCourse пов'язаний з Quiz для відстеження успіхів користувачів у курсах.
- Quiz зберігає інформацію про тести, їхні параметри та зв'язки з іншими таблицями.

Ця структура БД дозволяє ефективно зберігати та обробляти дані про курси, тести, спроби їх проходження, а також відгуки та коментарі користувачів.

Висновки за розділом 2

У даному розділі було розглянуто основні аспекти моделювання та архітектури комп’ютерної моделі. Здійснено аналіз поняття «моделювання», де визначено його сутність та значення для досліджень і розробок. Визначено ключові компоненти моделювання, що включають в себе структурні елементи, взаємозв’язки та процеси, які забезпечують ефективне функціонування моделі. Також було розглянуто архітектуру комп’ютерної моделі, яка забезпечує організацію та взаємодію її складових, що є критично важливим для досягнення коректних і надійних результатів моделювання. В результаті проведеного аналізу сформульовано рекомендації щодо побудови ефективних комп’ютерних моделей, що враховують як теоретичні, так і практичні аспекти.

РОЗДІЛ 3
ДЕПЛОЙ ТА ТЕСТУВАННЯ МОДЕЛІ НЕФОРМАЛЬНОГО НАВЧАННЯ

3.1. Обрані мови програмування

Мною було обрано такі мови програмування як Node.js, Java, HTML, CSS, React. Нижче хочу навести короткий опис, переваги та недоліки саме цих мов.

По-перше, поговоримо про Node.js. Уявіть собі, що ви можете використовувати ту саму мову програмування як для написання коду на сервері, так і для клієнтської частини. Це саме те, що робить Node.js можливим. Завдяки своїй асинхронній, неблокуючій архітектурі, Node.js здатен обробляти тисячі одночасних запитів, не знижуючи продуктивності. Він побудований на рушії V8 від Google, тому працює неймовірно швидко [11].

Таблиця 3.1

Переваги та недоліки Node.js

Переваги	Недоліки
1.Швидкість: завдяки V8, Node.js виконує код блискавично швидко.	1. Обчислювальна важкість: Не найкращий вибір для ресурсомістких завдань.
2. Єдиний JavaScript: уся розробка, від клієнта до сервера, ведеться однією мовою.	2. Екосистема: хоча швидко розвивається, все ще відносно молода в порівнянні з іншими технологіями.
3. Багатство пакетів: завдяки npm, доступні тисячі бібліотек для будь-яких потреб.	

По-друге, мова програмування Java – це мова, яка стоїть за багатьма великими корпоративними системами. Завдяки своїй платформній незалежності, програми, написані на Java, можуть запускатися на будь-якому пристрої з JVM. Java відома своєю стабільністю та надійністю, що робить її ідеальною для розробки великих, критичних до безпеки додатків.

Таблиця 3.2

Переваги та недоліки Java

Переваги	Недоліки
1. Портативність: "Write Once, Run Anywhere" – напиши один раз, запускай де завгодно.	1. Складність: вимагає більше коду для виконання простих завдань.
2. Масштабованість: чудово підходить для великих і складних систем.	2. Продуктивність: може бути повільніше, ніж мови, що компілюються безпосередньо в машинний код.
3. Багатопоточність: легка реалізація багатопотокових програм.	

По-третє, не можна не відмітити HTML – це основа всіх веб-сторінок. Він визначає структуру веб-документів за допомогою тегів. Незалежно від того, чи створюєте ви просту статичну сторінку, чи складний веб-додаток, HTML – це ваша відправна точка.

Таблиця 3.3

Переваги та недоліки HTML

Переваги	Недоліки
1. Простота: легко вчитися і почати використовувати.	1. Обмежена функціональність: сам по собі HTML не забезпечує динамічності.
2. Широка підтримка: працює у всіх веб-браузерах.	2. Залежність від CSS та JavaScript: потребує додаткових мов для стилізації та інтерактивності.
3. Основний елемент вебу: використовується для створення структури веб-сторінок.	

Також CSS додає життя вашим веб-сторінкам, роблячи їх красивими та привабливими. З його допомогою можна змінювати кольори, шрифти, розміщення елементів і навіть створювати складні анімації.

Таблиця 3.4

Переваги та недоліки CSS

Переваги	Недоліки
1. Стилізація: легко налаштовувати зовнішній вигляд і оформлення веб-сторінок.	1. Кросбраузерність: вимагає перевірки на сумісність між різними браузерами.
2. Адаптивний дизайн: підтримує медіа-запити для різних пристроїв.	2. Складність: створення складних макетів може вимагати глибоких знань CSS.

3. Візуальні ефекти: дозволяє створювати анімації та ефекти.	

React – це бібліотека JavaScript для створення користувацьких інтерфейсів, розроблена Facebook. Вона дозволяє створювати динамічні, інтерактивні веб-додатки з високою продуктивністю, використовуючи компоненти.

Таблиця 3.5

Переваги та недоліки React

Переваги	Недоліки
1. Компонентний підхід: Легко повторно використовувати і підтримувати код.	1. Крива навчання: вимагає знань додаткових інструментів та налаштувань.
2. Висока продуктивність: віртуальний DOM забезпечує швидкий рендеринг.	2. Часті оновлення: постійні зміни в екосистемі можуть бути важкими для підтримки.
3. Екосистема: велика кількість додаткових інструментів і бібліотек.	

Кожна з цих мов і технологій має свої особливості та переваги, що робить їх незамінними інструментами в руках розробника, дозволяючи створювати все – від простих веб-сторінок до складних багатокористувацьких додатків.

3.2 Використання БД MySQL (Workbench, Server)

MySQL – одна з найпопулярніших систем керування базами даних з відкритим кодом, яка використовується для зберігання та управління даними

в багатьох веб-додатках та корпоративних системах. Вона забезпечує високу продуктивність, надійність і простоту використання.

MySQL Workbench – це інтегроване середовище для розробки баз даних, яке пропонує візуальні інструменти для проектування, адміністрування та розробки баз даних. Воно включає функції для моделювання даних, управління сервером та створення запитів [12].

Основні функції:

- Проектування баз даних: інструменти для створення діаграм, визначення таблиць, індексів, зв’язків та інших елементів бази даних.
- Адміністрування: засоби для управління користувачами, бекапів, відновлення даних, моніторингу продуктивності і налаштування серверу.
- Запити та розробка: вбудований SQL-редактор для написання, виконання та оптимізації запитів.

Таблиця 3.6

Переваги та недоліки MySQL Workbench

Переваги	Недоліки
1. Зручний інтерфейс: візуальні інструменти полегшують роботу з базами даних навіть для новачків.	1. Ресурсомісткість: потребує значних ресурсів системи для великих баз даних.
2. Моделювання даних: спрощує процес проектування та документування структури баз даних.	2. Залежність від GUI: деякі задачі можуть бути більш ефективно виконані за допомогою командного рядка.

3. Моніторинг та управління: надає засоби для ефективного адміністрування серверів MySQL.	

MySQL Server – це самостійний сервер, який забезпечує зберігання, управління та обробку даних. Він є основою для роботи з базами даних MySQL і надає всі необхідні функції для забезпечення високої продуктивності та надійності [13].

Основні функції:

- Зберігання даних: забезпечує зберігання даних у таблицях з підтримкою різних типів даних та індексів.
- Безпека: підтримка різних рівнів доступу, аутентифікація користувачів та шифрування даних.
- Продуктивність: оптимізовано для високої продуктивності з підтримкою масштабованості та балансування навантаження.

Таблиця 3.7

Переваги та недоліки MySQL Server

Переваги	Недоліки
1. Висока продуктивність: оптимізований для швидкої обробки запитів і ефективного управління даними.	1. Конфігурація: вимагає належного налаштування для досягнення оптимальної продуктивності.
2. Масштабованість: підтримує великі обсяги даних і високе навантаження.	2. Ресурсозатратність: потребує відповідних ресурсів для забезпечення стабільної роботи при великих навантаженнях.
3. Відкритий код: можливість використання безкоштовно з	

підтримкою широкої спільноти розробників.	

Приклади використання MySQL Workbench та Server:

1. Розробка БД:
   - Створення діаграм EER (Entity-Relationship) для проектування схеми бази даних.
   - Визначення таблиць, полів, ключів та зв’язків між таблицями.
2. Адміністрування серверу:
   - Управління користувачами та встановлення прав доступу.
   - Моніторинг стану серверу, продуктивності та ресурсів.
   - Виконання резервного копіювання та відновлення даних.
3. Розробка та оптимізація запитів:
   - Написання та виконання SQL-запитів у вбудованому редакторі.
   - Оптимізація запитів для підвищення швидкості виконання.
   - Аналіз планів виконання запитів для виявлення вузьких місць.

MySQL Workbench та Server разом утворюють потужний інструментарій для розробки, адміністрування та оптимізації баз даних, що дозволяє ефективно управляти даними та забезпечувати їхню безпеку і доступність [14].

3.3 Тестування комп`ютерної моделі

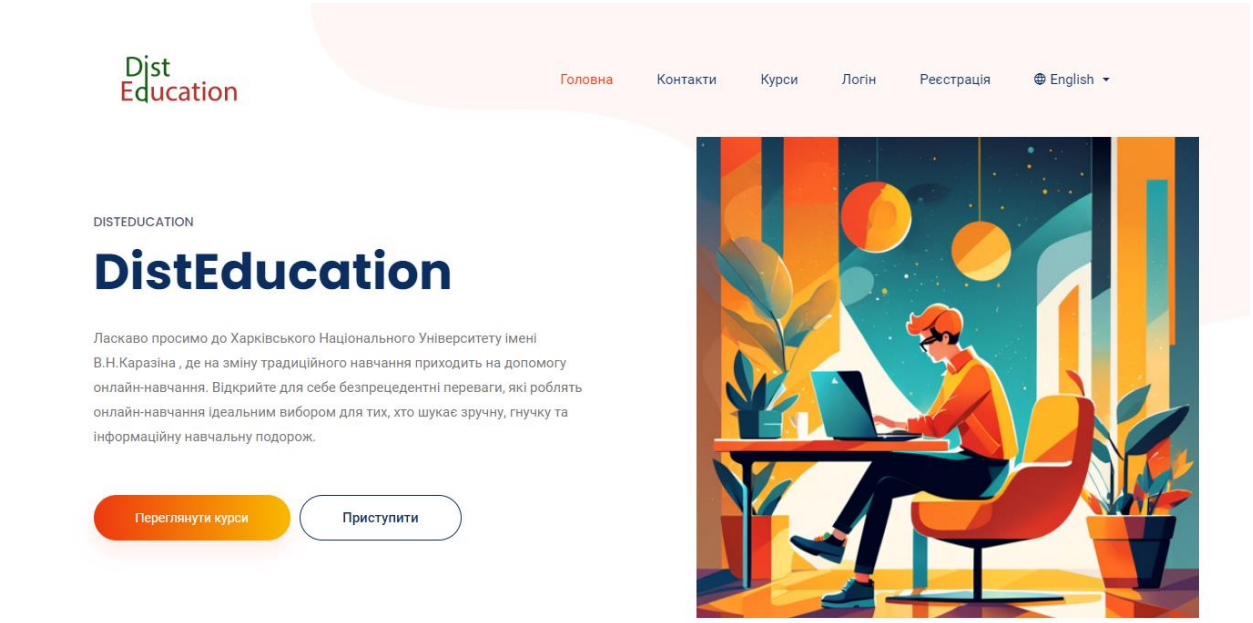


Рис 3.1 – Головна сторінка

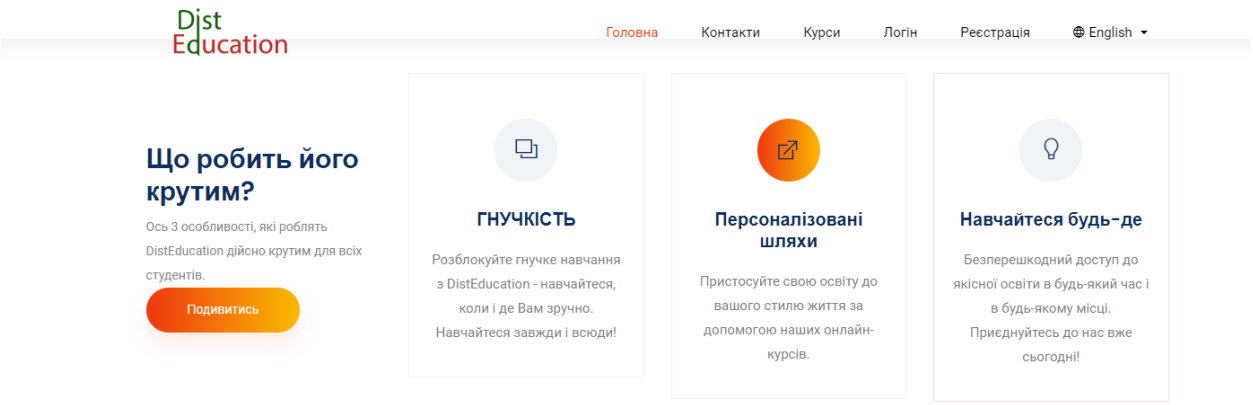


Рис 3.2 – Переваги використання сайту

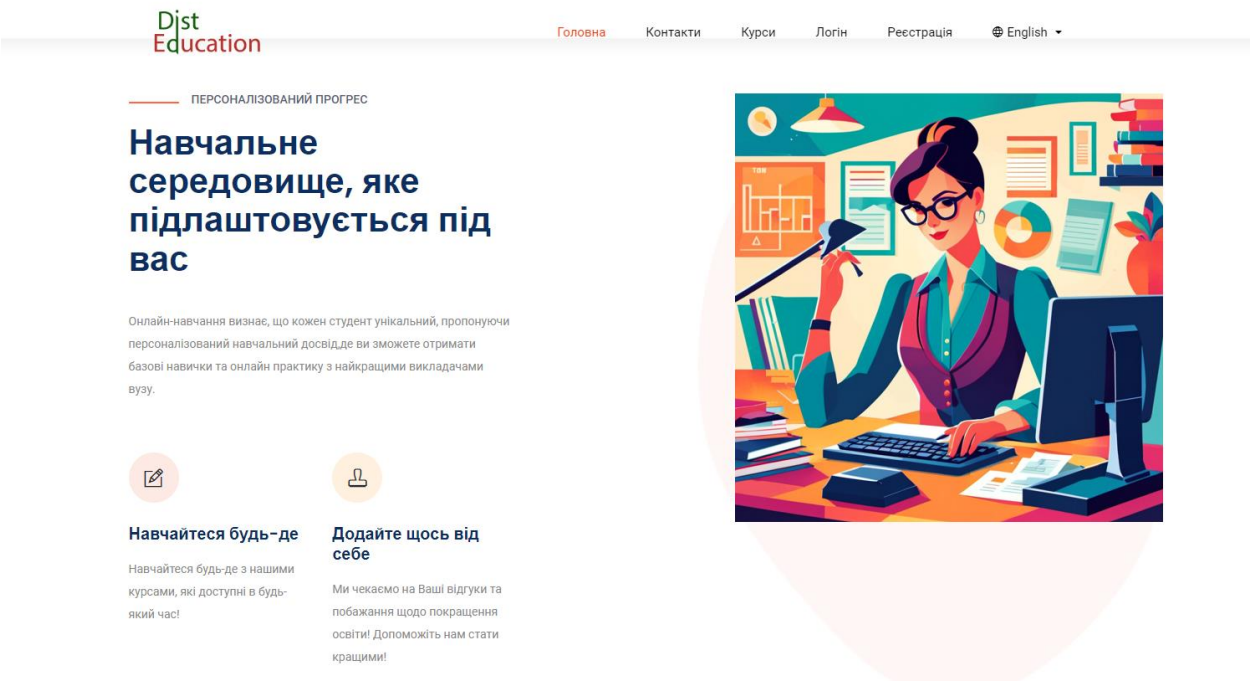


Рис 3.3 – Онлайн-навчання будь-де

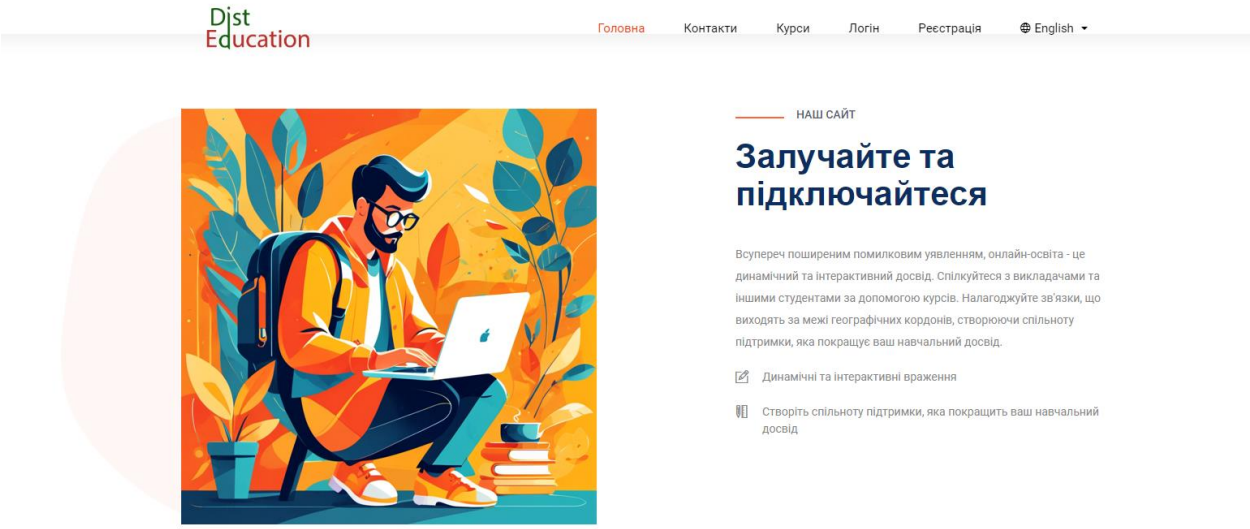


Рис 3.3 – Заклик до підключення до сайту

1. Натисніть на посилання "Забули пароль?": Клацніть на посилання, щоб перейти на сторінку скидання пароля.
2. Введіть свою адресу електронної пошти: На сторінці скидання пароля буде запропоновано ввести адресу електронної пошти, пов'язану з

вашим акаунтом. Переконайтеся, що ви вводите ту адресу, яку використовували під час реєстрації на сайті.

3. Натисніть на кнопку "Відновити".Після введення адреси електронної пошти натисніть на відповідну кнопку, щоб запросити скидання пароля (рис 3.5).

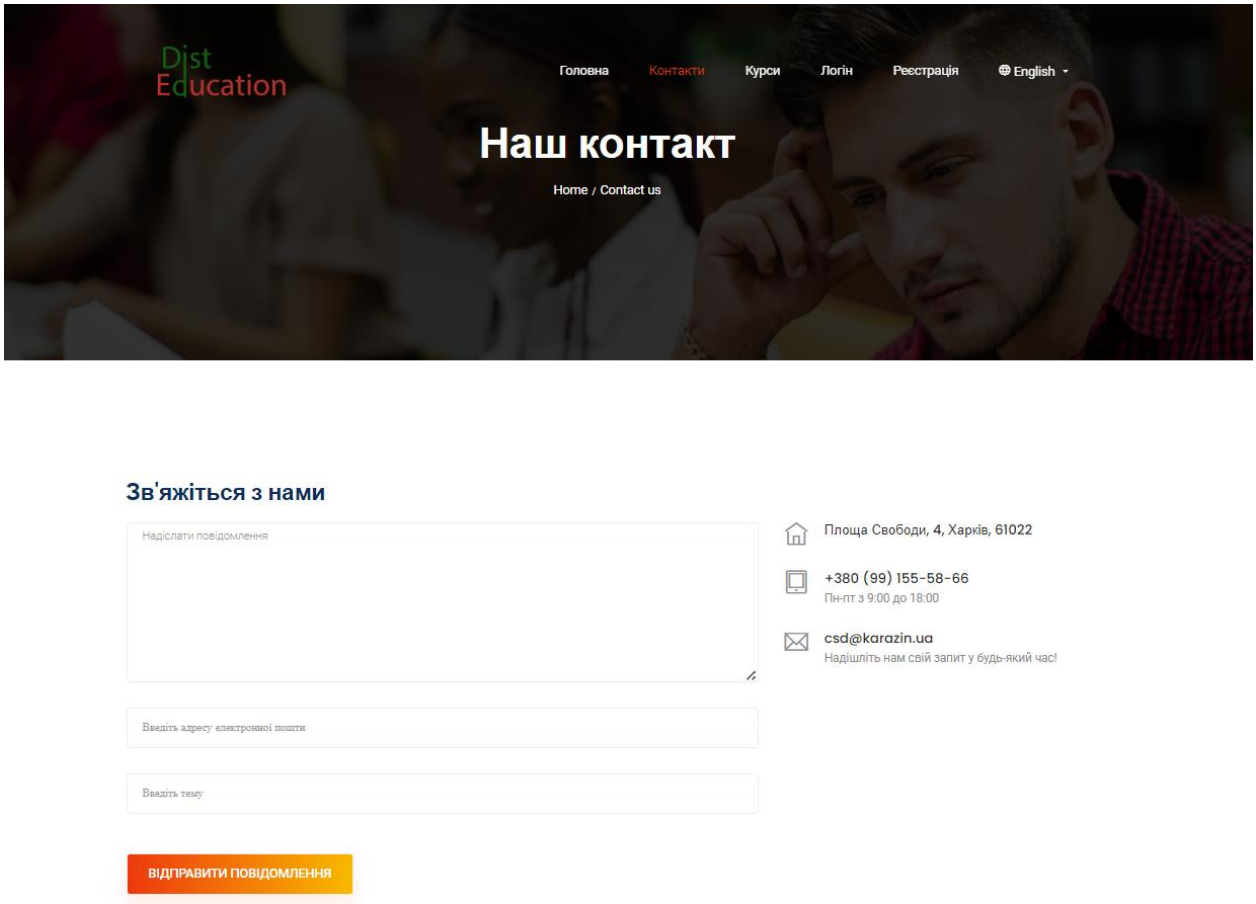


Рис 3.4 – Контактна інформація

Веб-сайт кафедри теоретичної та прикладної системотехніки (ТПС) ХНУ імені В.Н. Каразіна має простий дизайн з чіткою структурою. На сайті можна знайти контактну інформацію кафедри, опис навчальних програм, а також форму для надсилання повідомлень

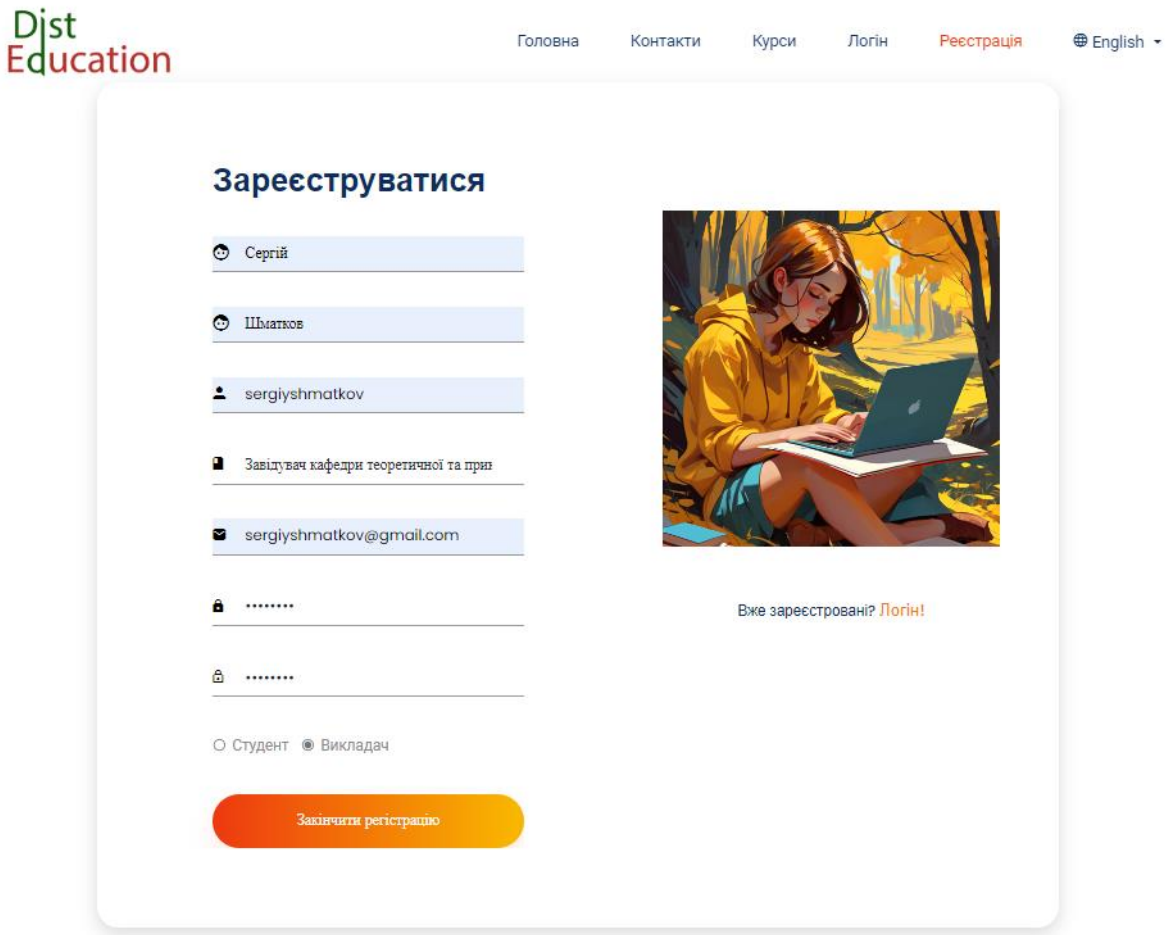


Рис 3.5 – Сторінка з формою реєстрації(Викладача)

Веб-сторінка з формою реєстрації на сайті кафедри ТПС ХНУ імені В.Н. Каразіна дозволяє користувачам створити акаунт на сайті. Користувачі можуть вибрати роль "Студент" або "Викладач" під час реєстрації.

Коротко про форму реєстрації:

- Поля: Прізвище, ім'я, логін, пароль, підтвердження пароля, роль.
- Кнопка: Зареєструватися.

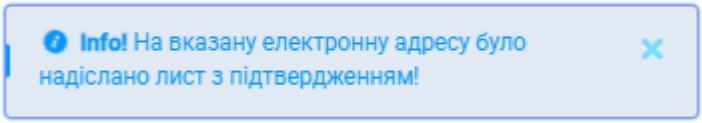


Рис 3.6 – Підтвердження реєстрації

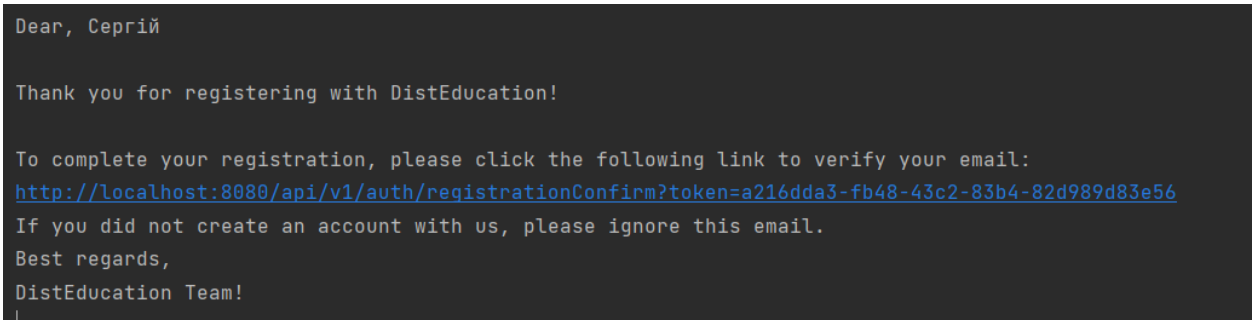


Рис 3.7 – Електроний лист для підтвердження реєстрації(Викладача)

Електронний лист дякує користувачу за реєстрацію на платформі DistEducation та містить посилання на підтвердження адреси електронної пошти.

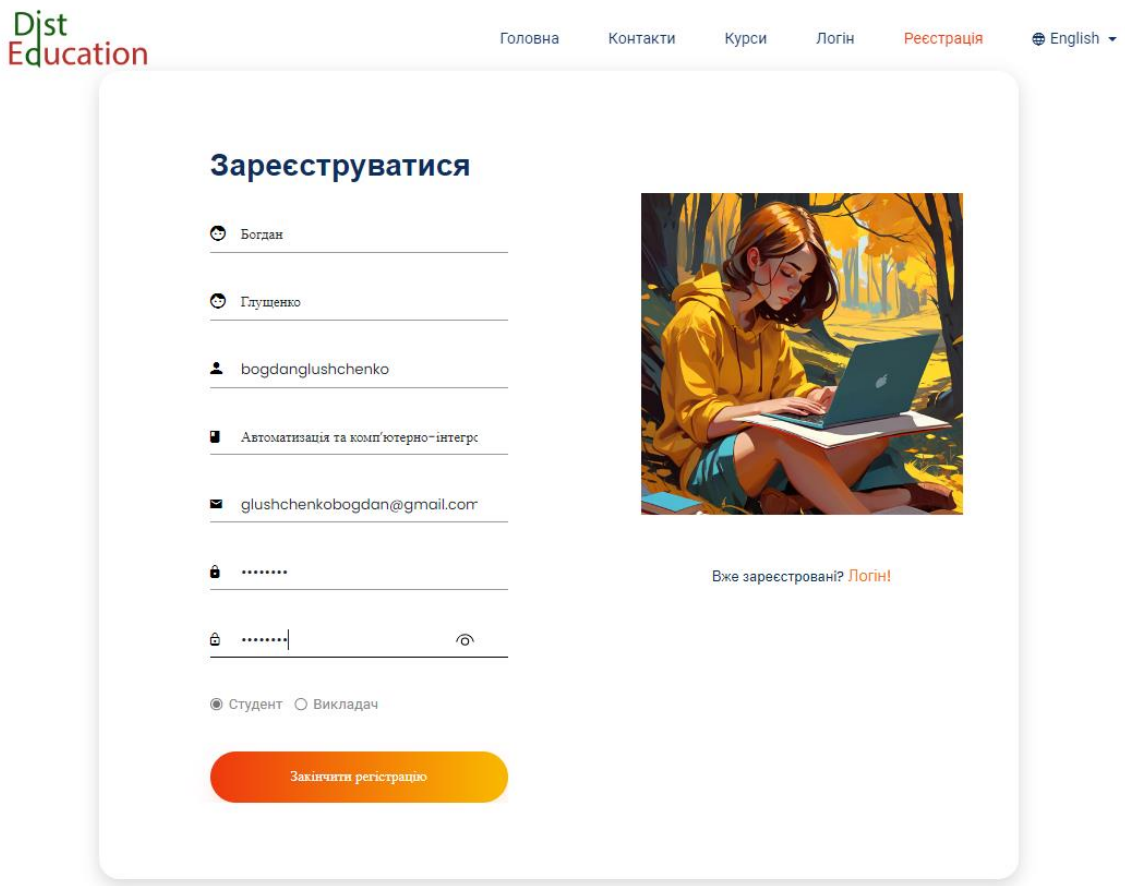


Рис 3.8 – Сторінка реєстрації(студента)

Веб-сторінка кафедри ТПС ХНУ імені В.Н. Каразіна має форму реєстрації, яка дозволяє користувачам створити акаунт на сайті. Під час реєстрації користувачі можуть обрати роль "Студент" або "Викладач".

Основні поля форми реєстрації включають: Прізвище, Ім'я, Логін, Пароль, Підтвердження пароля та Роль. Користувачі повинні ввести свої особисті дані та обрати одну з ролей.

Після введення всіх необхідних даних, користувач натискає кнопку "Зареєструватися", щоб завершити процес реєстрації на сайті.

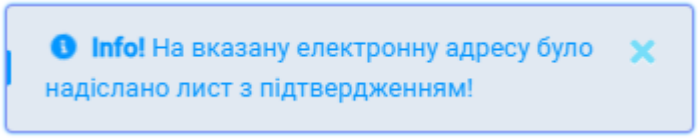


Рис 3.9 – Підтвердження реєстрації

```
Dear, Богдан

Thank you for registering with DistEducation!

To complete your registration, please click the following link to verify your email:
http://localhost:8080/api/v1/auth/registrationConfirm?token=49ffe53f-1b76-4737-b525-581ae2cab9e1
If you did not create an account with us, please ignore this email.
Best regards,
DistEducation Team!
```

Рис 3.10 – Електроний лист для підтвердження реєстрації(Студента)

Електронний лист призначений для подяки користувачеві за реєстрацію на платформі DistEducation. У листі також надається посилання для підтвердження адреси електронної пошти.

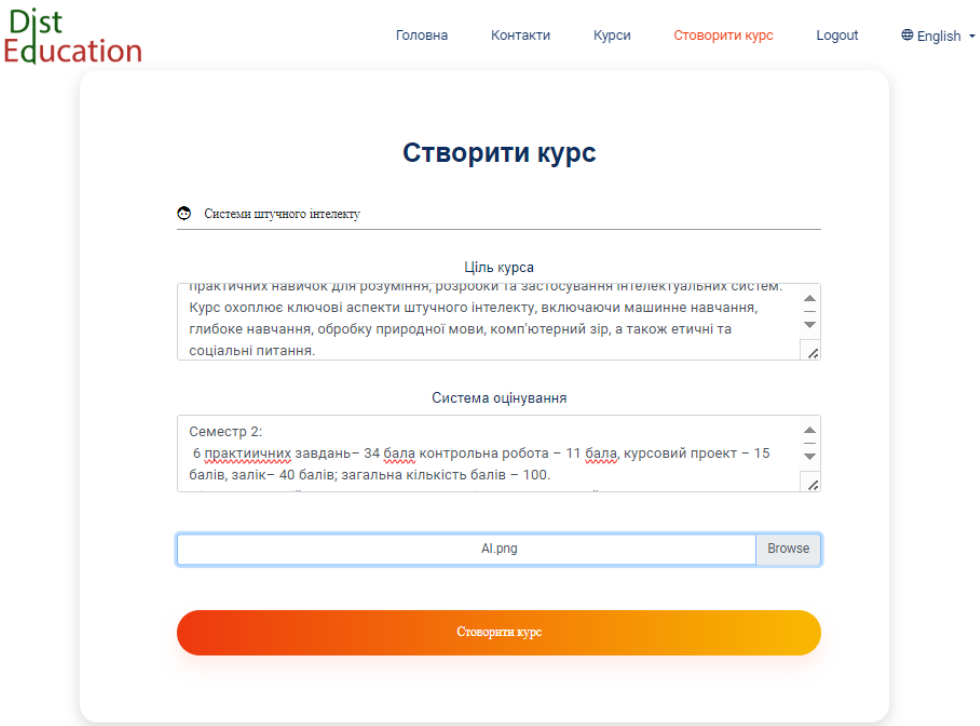
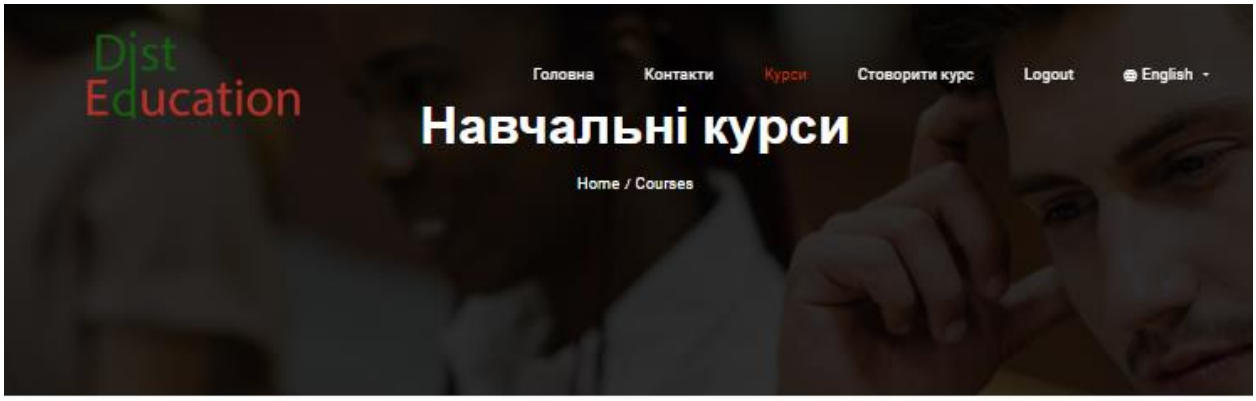


Рис 3.11 – Головна сторінка створення курсу (зі сторони викладача)

Сайт DistEducation пропонує платформу для онлайн-навчання, де викладачі можуть створювати та публікувати курси.

За допомогою простої форми ви можете:

- Надати назву, ціль курсу, систему оцінювання, додавання навчальних матеріалів.
- Завантажити зображення.
- Опублікувати ваш курс.



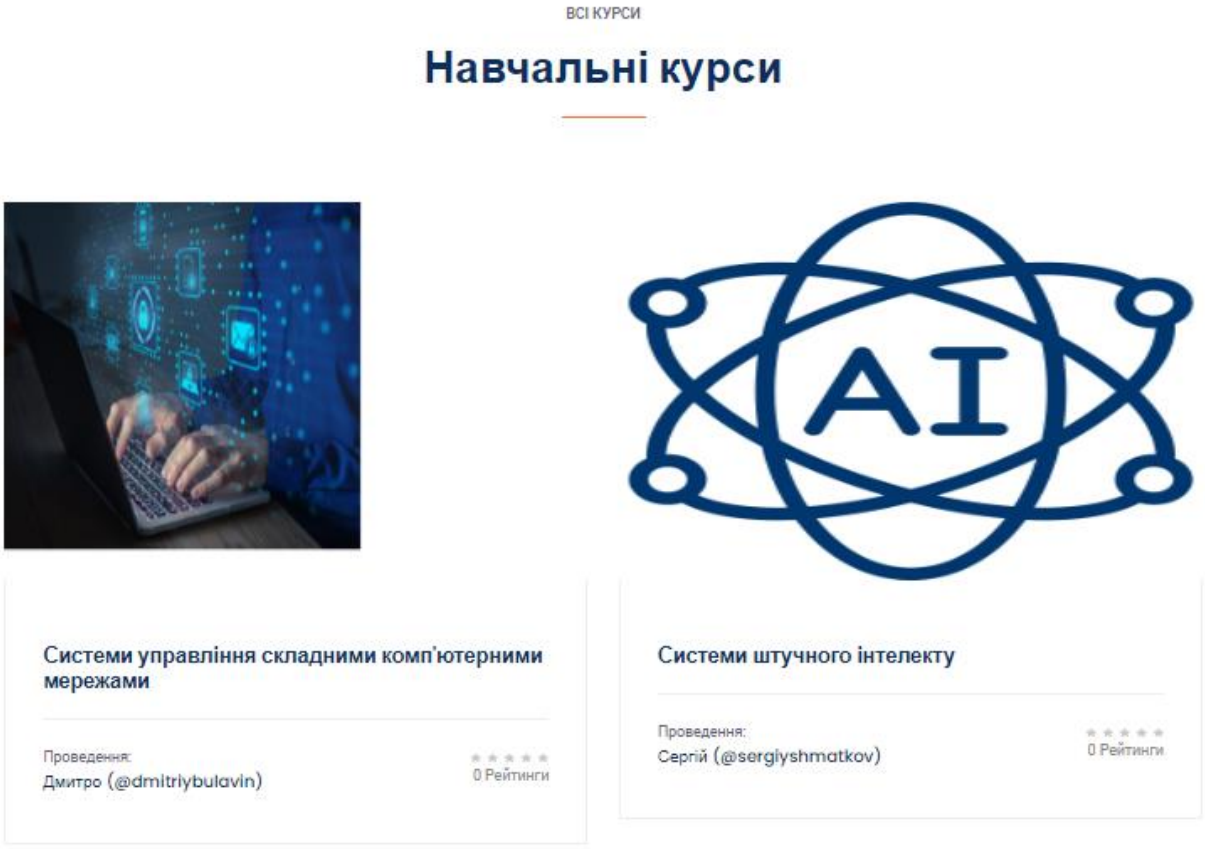


Рис 3.12 – Головна сторінка інформації про «Навчальні курси»

На цій сторінці зображено інформацію про Навчальні курси, про такий

курс як, Системи Штучного Інтелекту, де ми можемо перейти за цим курсом, та , побачити інформацію про цей курс, записи лекцій, та завдання до практичних занять.

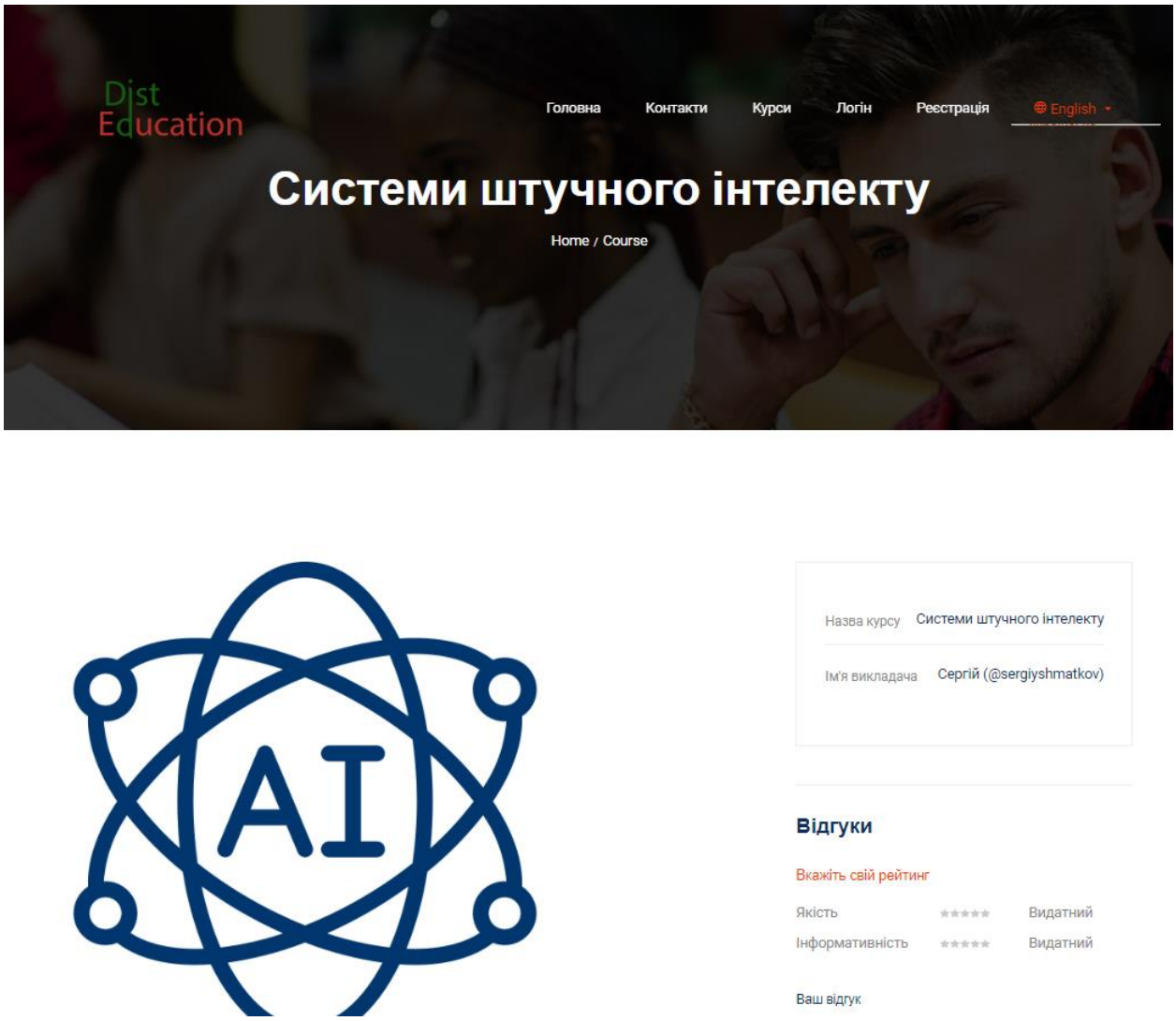


Рис 3.13 – Головна сторінка курсу «Системи Штучного Інтелекту»(зі сторони студента)

На цій сторінці, ми бачимо інформацію про реєстрацію до курсу, бачимо ім’я викладача та можемо оцінити якість та інформативність курсу.

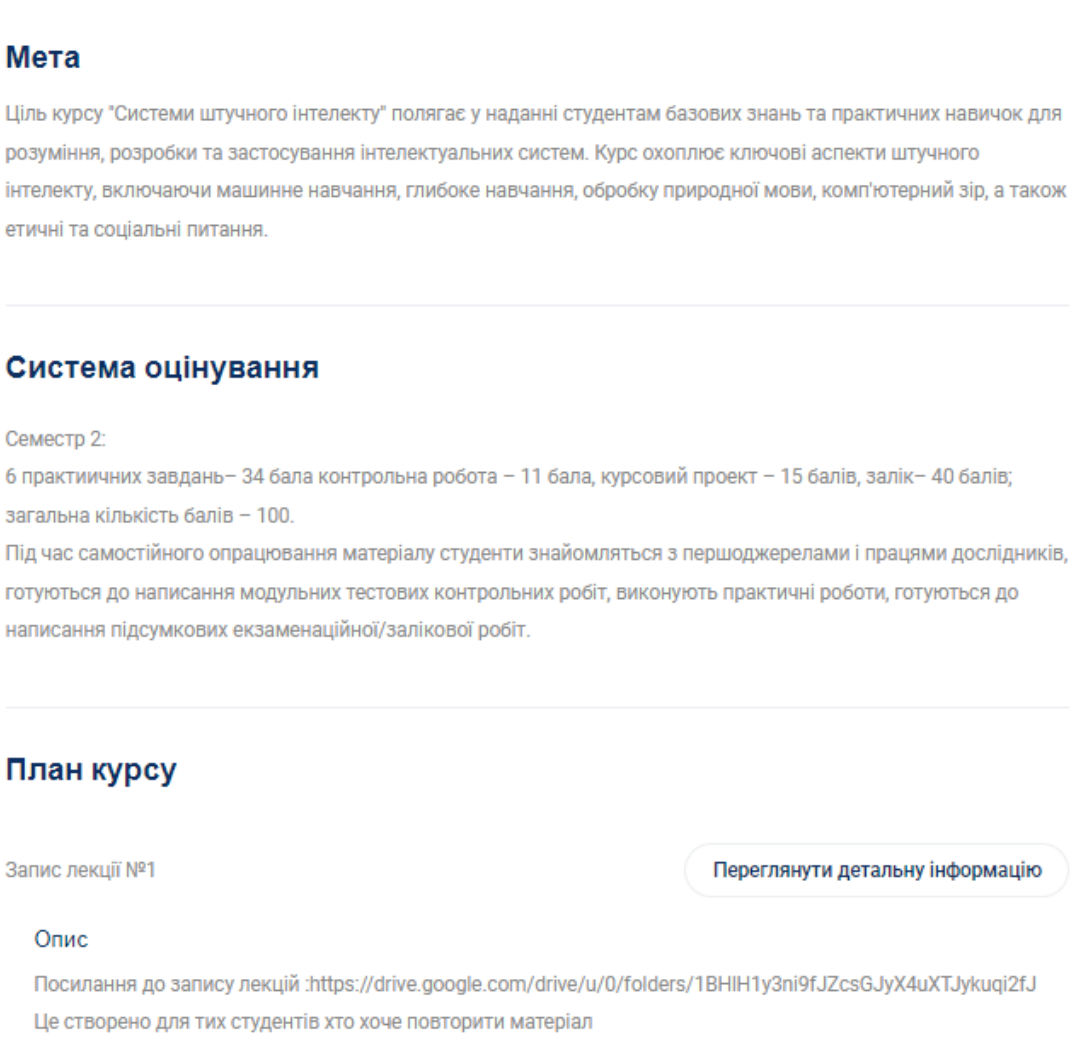


Рис 3.14 – Головна сторінка інформації про курс

На цій сторінці зображено «Мета», «Система оцінювання» та «План курсу», де в кожному присутній опис. На рис. 3.16, буде зображено скріншот про запис лекції №1.

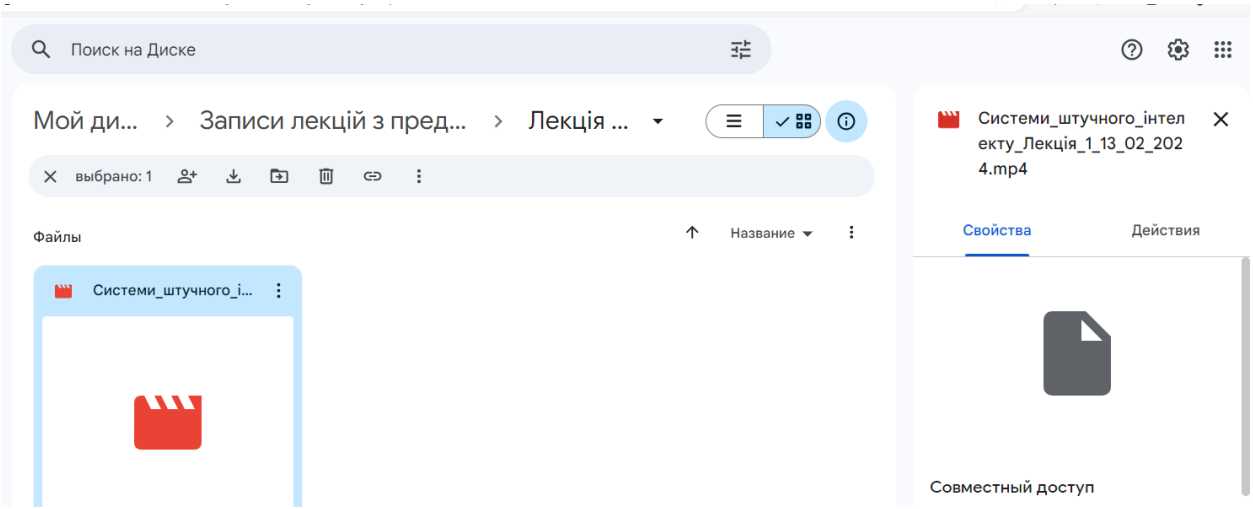


Рис 3.15 – Запис лекцій №1

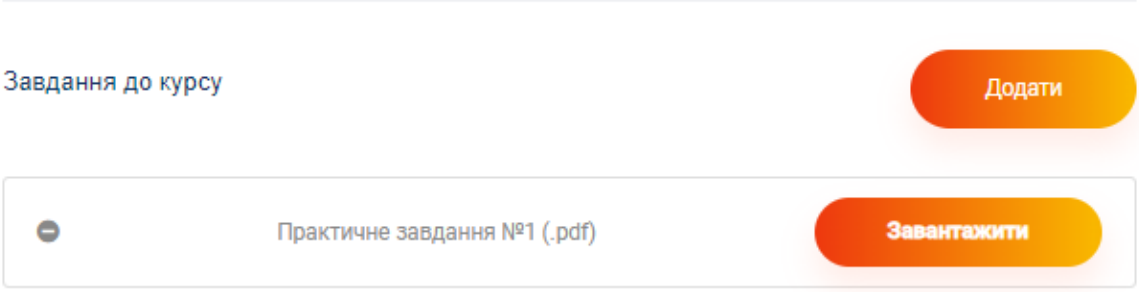


Рис 3.16 – Головна сторінка завдання до курсу

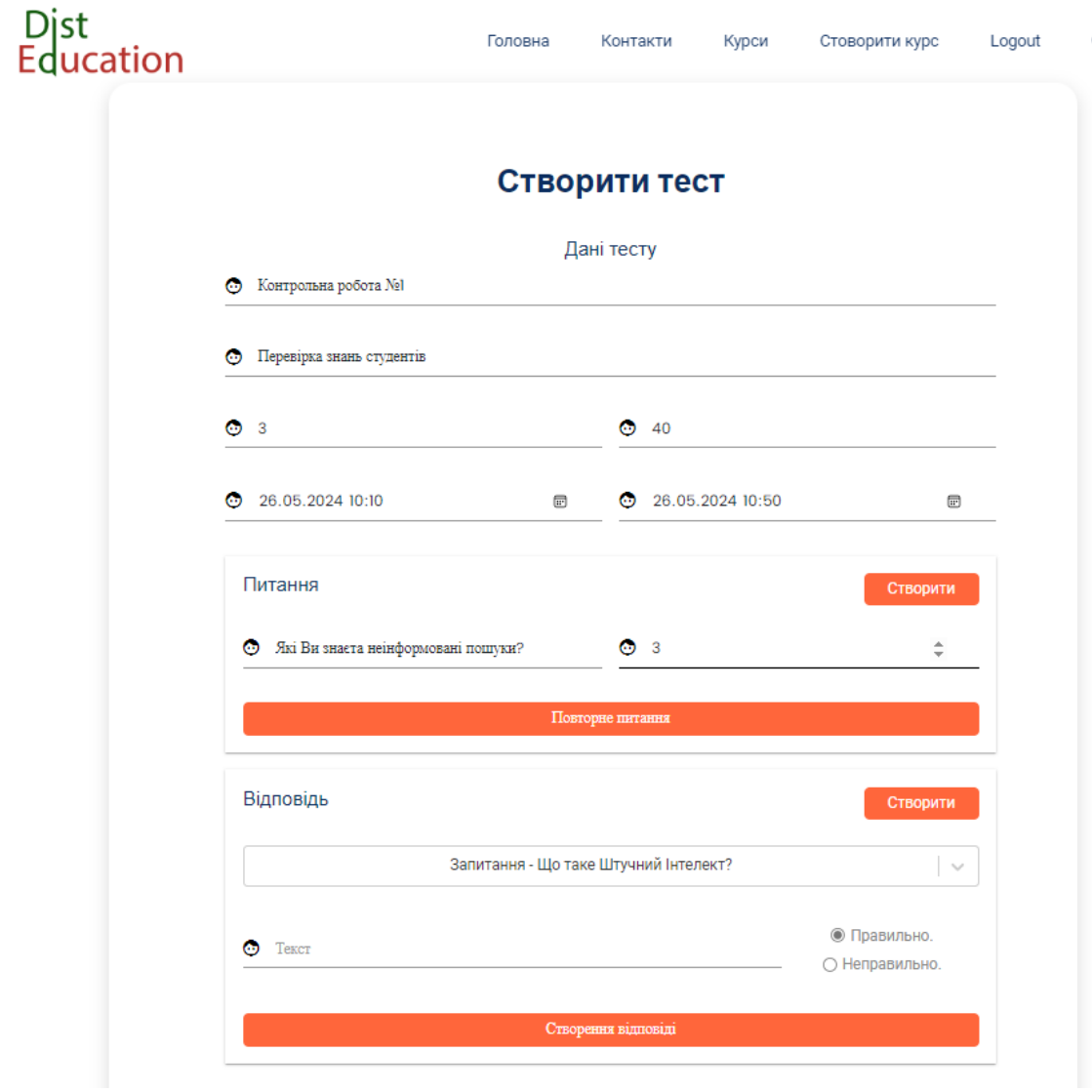


Рис 3.17 – Головна сторінка створення тесту

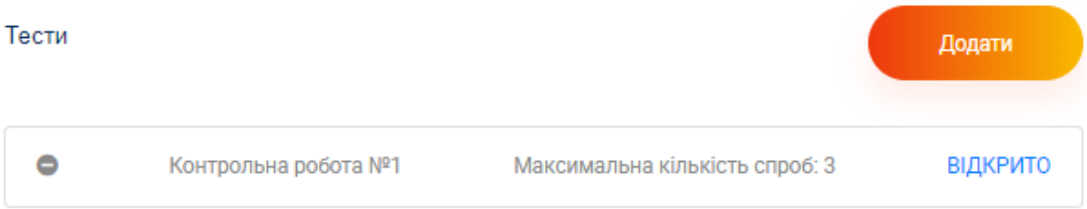


Рис 3.18 – Скріншот контрольної роботи №1

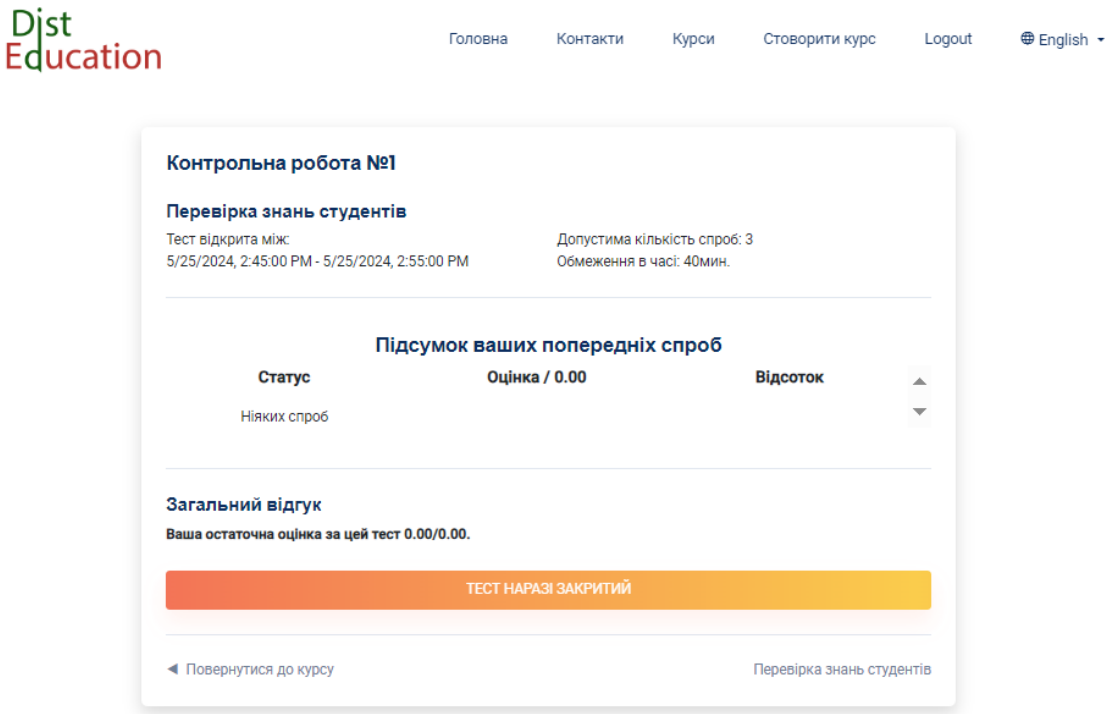


Рис. 3.19 – Тест ще недоступний

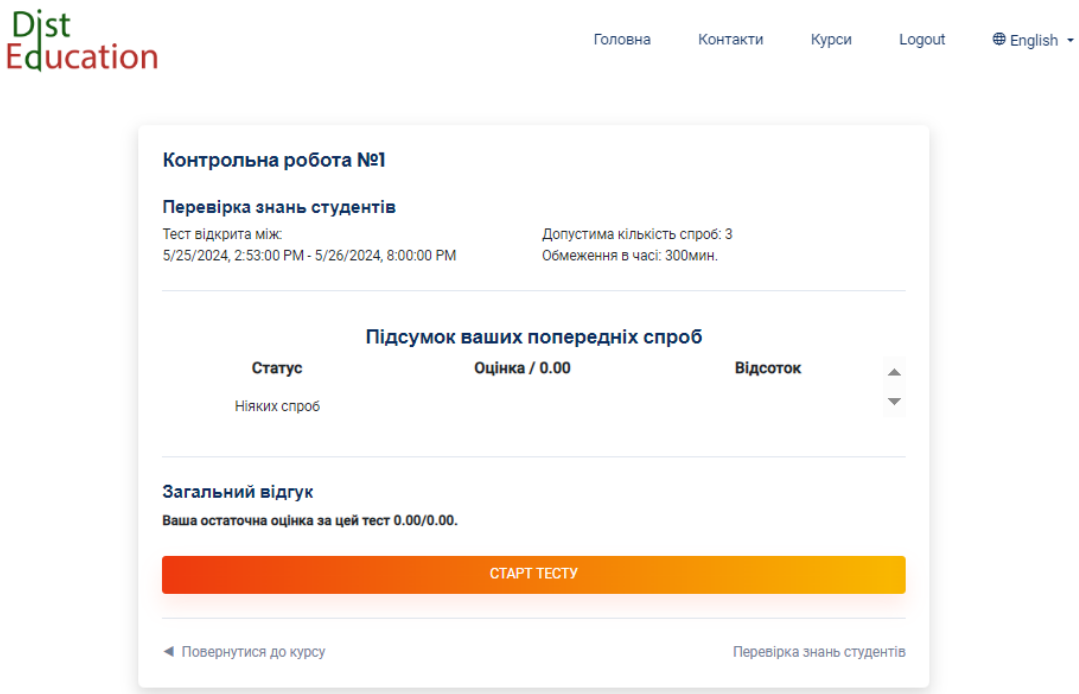


Рис. 3.20 – Тест доступний

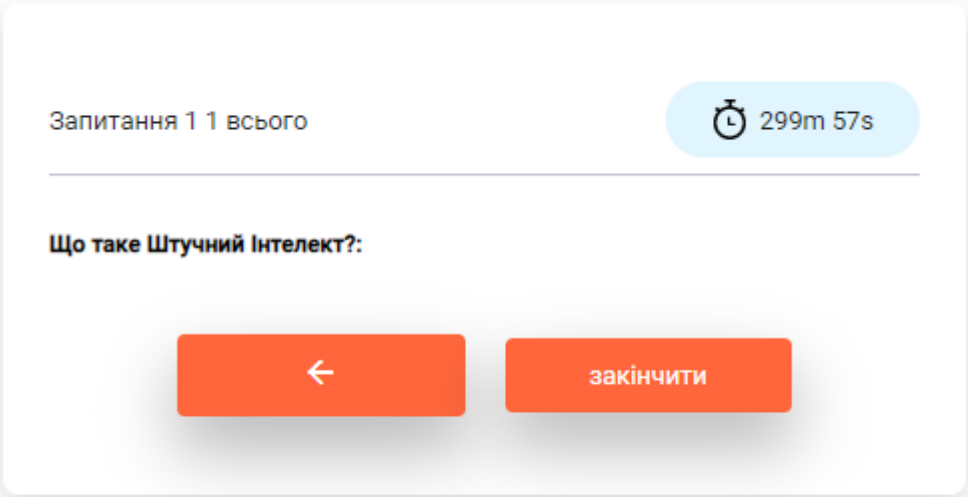


Рис. 3.21 – Питання з тесту

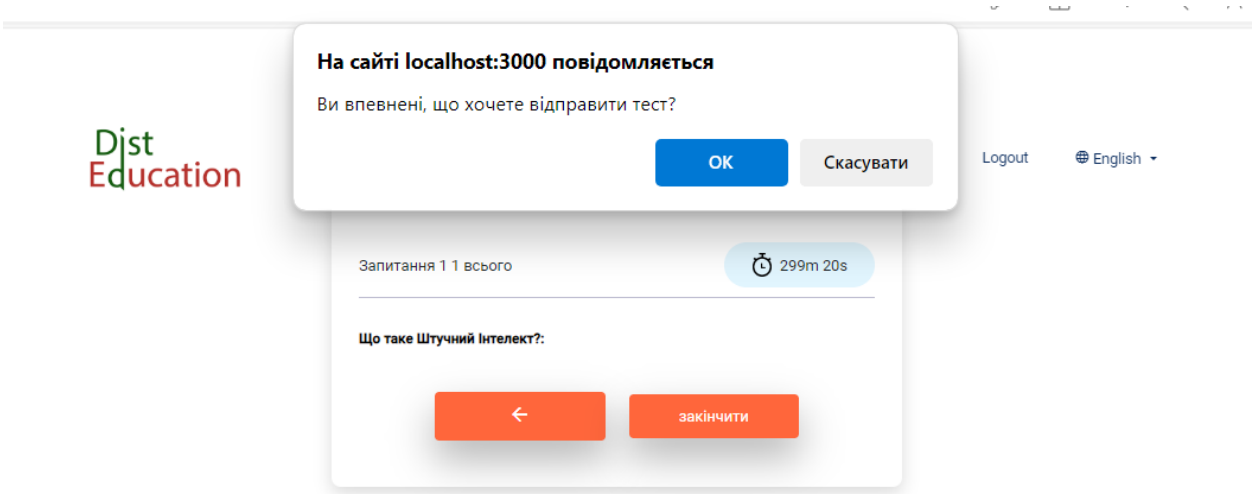


Рис. 3.22 – Відправка тесту на оцінювання

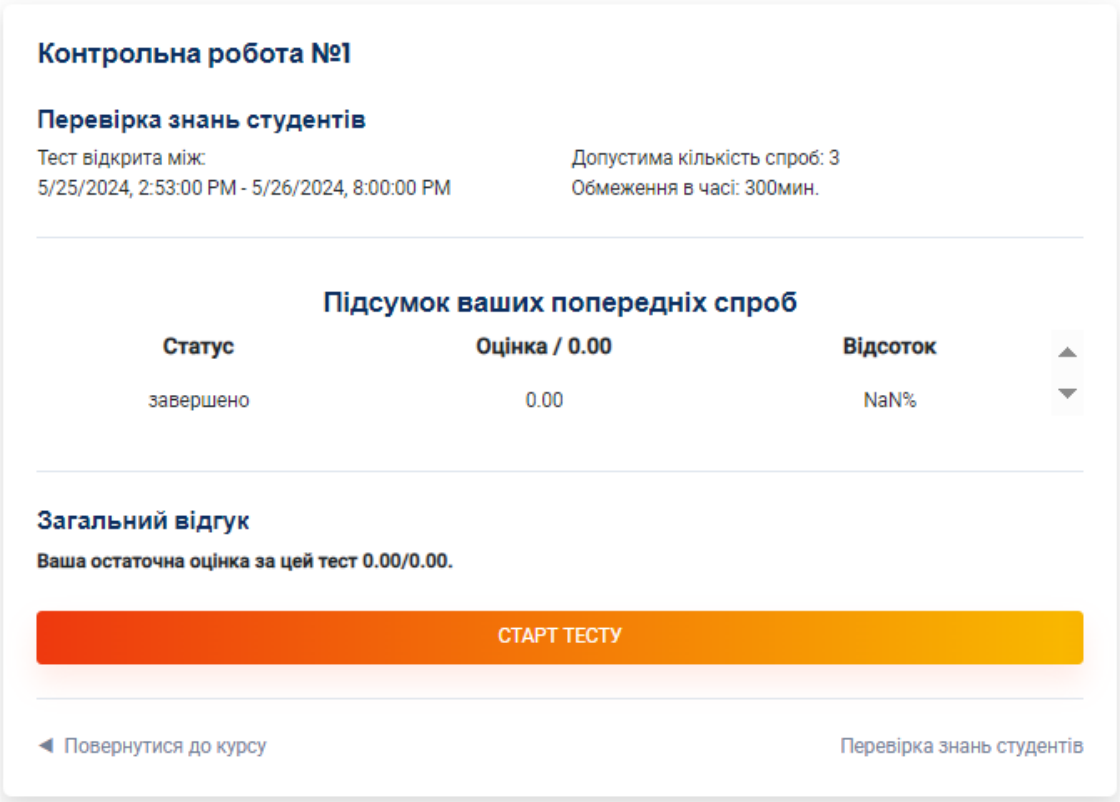


Рис. 3.23 – Результат тесту

Висновки за розділом 3

У даному розділі було детально розглянуто етапи деплою та тестування моделі неформального навчання. Зокрема, обрані мови програмування, такі як Node.js, Java, HTML, CSS та React, забезпечили ефективну реалізацію функціональності та інтерактивності системи. Використання бази даних MySQL, включаючи інструменти Workbench та Server, дозволило зберігати та управляти даними надійно та ефективно. Завершальним етапом було тестування комп’ютерної моделі, яке підтвердило коректність її роботи та готовність до використання у реальних умовах. Загалом, проведені роботи забезпечили створення стабільної та функціональної системи неформального навчання.

ВИСНОВКИ

Дослідження і розробка комп'ютерної моделі неформального навчання підтвердили важливість використання сучасних технологій для підвищення якості та доступності освітніх процесів. У ході роботи були реалізовані декілька ключових етапів, які забезпечили створення ефективної та функціональної системи навчання.

По-перше, були обрані мови програмування, такі як Node.js, Java, HTML, CSS та React. Кожна з цих мов виконувала свою важливу роль: Node.js і Java забезпечували серверну частину і бізнес-логіку, HTML та CSS відповідали за структуру та стиль веб-інтерфейсу, а React забезпечував динамічну та інтерактивну роботу користувацького інтерфейсу. Така комбінація технологій дозволила створити сучасний, швидкий і зручний інтерфейс для користувачів.

По-друге, для зберігання та управління даними була використана база даних MySQL, зокрема її інструменти Workbench та Server. MySQL забезпечила надійне зберігання даних, що є критично важливим для навчальних систем, де зберігаються результати тестів, особисті дані користувачів, навчальні матеріали тощо. Інструменти Workbench полегшили управління базою даних та її моделювання, що сприяло швидкому налаштуванню і оптимізації системи.

На завершальному етапі було проведено ретельне тестування комп’ютерної моделі. Тестування включало перевірку функціональності всіх компонентів системи, оцінку її стабільності та продуктивності. Результати тестування підтвердили, що розроблена система відповідає заявленим вимогам та готова до впровадження. Модель показала високу стабільність роботи, що є важливим для забезпечення безперебійного процесу навчання.

Загалом, розроблена комп'ютерна модель неформального навчання відкриває нові можливості для самостійного навчання. Вона забезпечує

зручний доступ до персоналізованих навчальних матеріалів, адаптується до потреб користувачів та підтримує їх у процесі навчання. Система дозволяє користувачам отримувати своєчасний зворотний зв’язок та рекомендації, що сприяє ефективнішому засвоєнню матеріалу.

Це дослідження є важливим кроком у напрямку цифровізації освіти та розвитку інноваційних методів навчання. Створена модель сприяє постійному професійному та особистісному зростанню користувачів, забезпечуючи їх сучасними інструментами для самостійного та ефективного навчання. Впровадження таких технологій у систему освіти може значно покращити якість навчання та зробити його більш доступним для широкого кола користувачів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Іваненко, О. О., & Сидоренко, П. П. (2020). Методи неформального навчання в сучасній освіті. Київ: Видавничий дім "Освіта".
2. Павленко, М. В. (2019). Технології викладання на кафедрі ТПС. Харків: Харківський національний університет.
3. Бондар, І. І. (2018). Особливості сучасних освітніх технологій. Львів: Видавництво Львівського університету.
4. Євченко, С. С. (2017). Тенденції неформального навчання в Європі. Одеса: Видавничий центр "Освіта".
5. Мельник, А. А. (2021). Структурна модель неформальної освіти. Дніпро: Видавництво Дніпровського університету.
6. Brown, J. S. (2019). Principles of Modeling in Education. Cambridge: MIT Press.
7. Smith, A. B. (2020). Key Components of Educational Modeling. New York: Routledge.
8. Johnson, P. R. (2018). Computer Model Architectures. San Francisco: Morgan Kaufmann.
9. Lee, K. C. (2017). Educational Modeling: Concepts and Applications. London: Springer.
10. Miller, T. J. (2021). Advanced Educational Modeling Techniques. Boston: Pearson.
11. Taylor, J. (2020). Programming Languages for Educational Tools. Seattle: O'Reilly Media.
12. Wang, L. (2019). MySQL Database for Educational Applications. Beijing: Tsinghua University Press.
13. Roberts, M. (2018). Testing and Deployment of Educational Models. Sydney: Wiley.
14. Kumar, S. (2021). MySQL Workbench and Server in Education. New Delhi: Tata McGraw-Hill.

ДОДАТКИ

Додаток А

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет імені В. Н. Каразіна

Факультет комп'ютерних наук
Кафедра теоретичної та прикладної системотехніки
Рівень вищої освіти (освітньо-кваліфікаційний рівень) **бакалавр**
галузь знань:**15 – Автоматизація та приладобудування**
спеціальність:**151 – Автоматизація та комп'ютерно-інтегровані технології**

ЗАТВЕРДЖУЮ
Завідувач кафедри теоретичної
та прикладної системотехніки
_____д.т.н., проф. Шматков С. І.
«21» грудня 2023 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Глущенко Богдана Олександровича

(прізвище, ім’я, по батькові студента)

1. Тема роботи «Комп’ютерна модель неформального навчання»

керівник роботи Булавін Дмитро Олексійович, канд. техн.наук, доцент

(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від “ 23 ”травня 2024 року №4101-5/895

2.Строк подання студентом роботи 31 травня 2024року

3. Перелік питань, які потрібно розробити

1. Аналіз систем навчання та особливостей викладання на кафедрі ТПС.
2. Створення архітектури моделі неформального навчання.

3. Аналіз мов програмування для розробки комп’ютерної моделі неформального навчання.
4. Розробка базових модулів моделі неформального навчання.
5. Наповнення курсу матеріалами.
6. Деплой та тестування моделі неформального навчання.

4. План роботи

№ з/п	Назви етапів роботи	Термін виконання етапів роботи
1	Аналіз научної літератури	21.12.2023 - 25.01.2024
2	Аналіз систем навчання: формального та неформального	19.12.2023 - 2.01.2024
3	Розробка структури моделі для неформального навчання	2.01.2024 - 2.02.2024
4	Вивчення мов програмування з метою створення комп'ютерної моделі неформального навчання	2.01.2024 - 2.02.2024
5	Створення моделі навчання в неформальному форматі	3.02.2024 - 30.03.2024
6	Розробка та тестування моделі неформального навчання	3.03.2024 - 30.04.2024
7	Розробка пояснювальної записки	31.03.2024 - 27.05.2024
8	Оформлення звіту за результатами переддипломної практики.	15.05.2024 – 31.05.2024
9	Представлення кваліфікаційної роботи керівнику та рецензенту	31.05.2024
10	Оформлення пояснювальної записки та підготовка презентації.	31.05.2024

5. Дата видачі завдання 21.12.2023

Студент Глущенко Б. О. ___ ___
ініціали, прізвище підпис

Керівник роботи Булавін Д. О. _____
ініціали, прізвище підпис

Додаток Б

Затверджую

_____

«___» _____ 2023 р.

Технічне завдання
на розробку програмного виробу «Комп’ютерна модель неформального навчання»

1.	Введення	1.1. Назва: Комп’ютерна модель неформального навчання 1.2. Галузь застосування: Інформаційні технології
2.	Підстава для розробки	2.1. Навчальний план за спеціальністю 151 – Автоматизація та комп'ютерно-інтегровані технології 2.2. Завдання на кваліфікаційну роботу бакалавра №4101-5/895 від «23» травня 2024 (представити як Додаток А до пояснювальної записки до кваліфікаційної роботи).
3.	Призначення розробки	3.1. Мета розробки: розробка та імплементація комп'ютерної моделі неформального навчання з метою збагачення освітнього процесу та підвищення ефективності у набутті знань та навичок учасниками. 3.2. Призначення розробки: аналіз та підтримка навчального процесу, створення інтерактивного навчального середовища, мотивування та залучення учнів, моніторинг та оцінка навчальних досягнень, адаптивність та гнучкість навчання. 3.3. Вихідні дані розробки: модель неформального навчання; вхідні дані розробки: проходження тестів та виконання завдань студентами.
4.	Технічні вимоги до програмного виробу	4.1. Вимоги до функціональних характеристик: реєстрація та управління користувачами, інтерфейс користувача, управління контентом, взаємодія та комунікація, моніторинг та оцінка, аналітика та звітність. 4.2. Вимоги до надійності: забезпечення безперебійної роботи програмного виробу при будь-яких вимогах користувача в рамках призначення виробу . 4.3.Вимоги до умов експлуатації: немає 4.4. Вимоги до складу і параметрів технічних засобів: для виконання програми повинен підходити ПК із

<table>
<tr><td></td><td></td><td colspan="2">будь-якою операційною системою сімейства Windows, Linux/Unix, Mac OS .
4.5. Вимоги до інформаційної та програмної сумісності: підтримка ОС Linux або Windows 11, підтримка мови програмування, підтримка різних платформ.
4.6. Вимоги до маркування та упаковки: вимоги до маркування та упакування не представляються.
4.7. Вимоги до транспортування і зберігання: вимоги до транспортування та зберігання не представляються.
4.8. Спеціальні вимоги: спеціальні вимоги до програмного виробу не пред'являються.</td></tr>
<tr><td>5.</td><td>Вимоги до програмної документації</td><td colspan="2">Програмною документацією до виробу «Метод аналізу інформативності змінних стану при діагностиці систем з використанням інформаційних критеріїв» вважати:
1) Справжнє Технічне завдання на розробку виробу (представити у вигляді Додатку Б до пояснювальної записки до кваліфікаційної роботи).
2) Методику розрахунку інформативності змінних стану (у вигляді глав 3.2 та 3.3 пояснювальної записки до кваліфікаційної роботи).
3) Опис виробу (представити в розділі 3 пояснювальної записки до кваліфікаційної роботи)</td></tr>
<tr><td>6.</td><td>Вимоги до техніко-економічних показників</td><td colspan="2">Програмною документацією до виробу «Комп’ютерна модель неформального навчання» вважати:
1) Справжнє Технічне завдання на розробку виробу (представити у вигляді Додатку Б до пояснювальної записки до кваліфікаційної роботи).
2) Опис програмного виробу (представити в Розділі 3 пояснювальної записки до кваліфікаційної роботи).
3) Джерела базової інформації.</td></tr>
<tr><td rowspan="4">7.</td><td rowspan="4">Стадії і етапи розробки</td><td>Дата</td><td>Назва етапу</td></tr>
<tr><td>від 21 грудня 2023
до 25 січня 2024</td><td>Аналіз научної літератури</td></tr>
<tr><td>від 19 грудня 2023
до 2 січня 2024</td><td>Аналіз систем навчання: формального та нефор-мального</td></tr>
<tr><td>від 2 січня 2024
до 2 лютого 2023</td><td>Розробка структури моделі для неформального навчання</td></tr>
</table>

		від 2 січня 2024 до 2 лютого 2024	Вивчення мов програмування з метою створення комп'ютерної моделі неформального навчання
		від 3 лютого 2024 до 30 березня 2024	Створення моделі навчання в неформальному форматі Розробка та тестування моделі неформального навчання
		від 3 березня 2024 до 30 квітня 2024	Розробка пояснювальної записки.
		від 31 березня 2024 до 27 травня 2024 від 15 травня 2024 до 31 травня 2024	Оформлення звіту за результатами переддипломної практики.
		31 травня 2024	Представлення кваліфікаційної роботи керівнику та рецензенту
		31 травня 2024	Оформлення пояснювальної записки та підготовка презентації
8.	Порядок контролю і приймання програмного продукту (моделі)	1. Перевірку ходу розробки програми виконувати раз в 3 тижні. 2. Захист розробленої моделі провести на засіданні Атестаційної комісії. 3. Пояснювальну записку подати на паперових носіях в 1 примірнику.	

Виконавець
студент групи КУ- 41
Глущенко Б. О.

Замовник
канд. техн.наук, доцент
Булавін Д. О.

Додаток В

Програма і методика випробувань програмного виробу

«Комп’ютерна модель неформального навчання»

1 Об'єкт випробувань

1. Назва програмного виробу : «Комп’ютерна модель нефрмального навчання»
2. Галузь застосування : Інформаційні технології
3. Перераховані відомості запозичуються з відповідних розділів Технічного завдання.

2. Мета випробувань

Перевірка відповідності функціональності програмної реалізації системи заявленим функціональним можливостям в технічному завданні (Додаток Б до пояснювальної записки до кваліфікаційної роботи).

3. Загальні положення

1. Підстави для проведення випробувань

Підставою для проведення випробувань є наказ про призначення атестаційної комісії.

2. Місце і тривалість випробувань

Приймальні (приймально-здавальні) випробування проводяться на базі комп'ютерного класу кафедри в період роботи атестаційної комісії.

3. Обсяг випробувань

Приймальні випробування програмного вироби проводяться в обсязі відповідному цієї програми і методики випробувань.

4. Організації, які беруть участь у випробуваннях

Приймальні випробування проводяться атестаційною комісією напередодні засідання (або в процесі засідання) за участю Замовника, Виконавця та інших осіб, присутніх на засіданні.

4. Вимоги до програми або програмного виробу

Модель повинна задовольняти наступним вимогам:

4.1. Вимоги до функціональних характеристик: реєстрація та управління користувачами, інтерфейс користувача, управління контентом, взаємодія та комунікація, моніторинг та оцінка, аналітика та звітність.

4.2. Вимоги до надійності: забезпечення безперебійної роботи програмного виробу при будь-яких вимогах користувача в рамках призначення виробу .
4.3.Вимоги до умов експлуатації: немає
4.4. Вимоги до складу і параметрів технічних засобів: для виконання програми повинен підходити ПК із будь-якою операційною системою сімейства Windows, Linux/Unix, Mac OS .
4.5. Вимоги до інформаційної та програмної сумісності: підтримка ОС Linux або Windows 11, підтримка мови програмування, підтримка різних платформ.
4.6. Вимоги до маркування та упаковки: вимоги до маркування та упакування не представляються.
4.7. Вимоги до транспортування і зберігання: вимоги до транспортування та зберігання не представляються.
4.8. Спеціальні вимоги: спеціальні вимоги до програмного виробу не пред'являються**.**

5. Вимоги до програмної документації

Документацією до виробу «Комп'ютерна модель неформального навчання» вважати:

1) Документація по мові програмування та додаткові мануали.

2) Програму і методику випробувань розробленої програми (представити як Додаток В до пояснювальної записки до кваліфікаційної роботи).

3) Опис програмного виробу (представити в Розділі 3 пояснювальної записки до кваліфікаційної роботи).

4) Джерела базової інформації.

6. Засоби і порядок випробувань

6.1 Засоби випробувань

Засоби випробувань представлено на ПК на яких встановлено наступні програмні засоби: інтерпретатор мови програмування.

6.2 Порядок проведення випробувань

Як правило, випробування проводяться в два етапи:

– ознайомчий (1-й етап);

– власне випробування програмного виробу (2-й етап).

Перелік перевірок, що проводяться на 1 етапі випробувань, включає в себе:

1) Перевірку комплектності складу програмної документації здійснюється за критерієм наявності зазначеної в ТЗ документації.

2) Перевірку якості програмної документації. Перевірку здійснювати за критерієм відповідності вимогам ГОСТ 19.301-79 ЕСПД. «Програма і методика випробувань».

Перелік перевірок, що проводяться на 2 етапі випробувань, включає в себе:

1) Перевірку відповідності технічних характеристик програми вимогам технічного завдання.

2) Перевірку ступеня виконання функціональних вимог до програми.

3) Методику проведення перевірок:

a) Запустити програмне забезпечення.

b) Порядок проведення випробувань:

– Зробити налаштування.

– Перевірити чи працює програма.

– Перевірити чи формується звіт.

4) Якщо перевірки на першому та другому етапах виконано успішно, то виріб вважається таким, що пройшов випробування.

Для проведення випробувань пропонується тест 1, тест 2 та тест 3.

Тест 1

1. Перевірка виконання програми;
2. Створення курсу;
3. Отримання результату про створений курс.

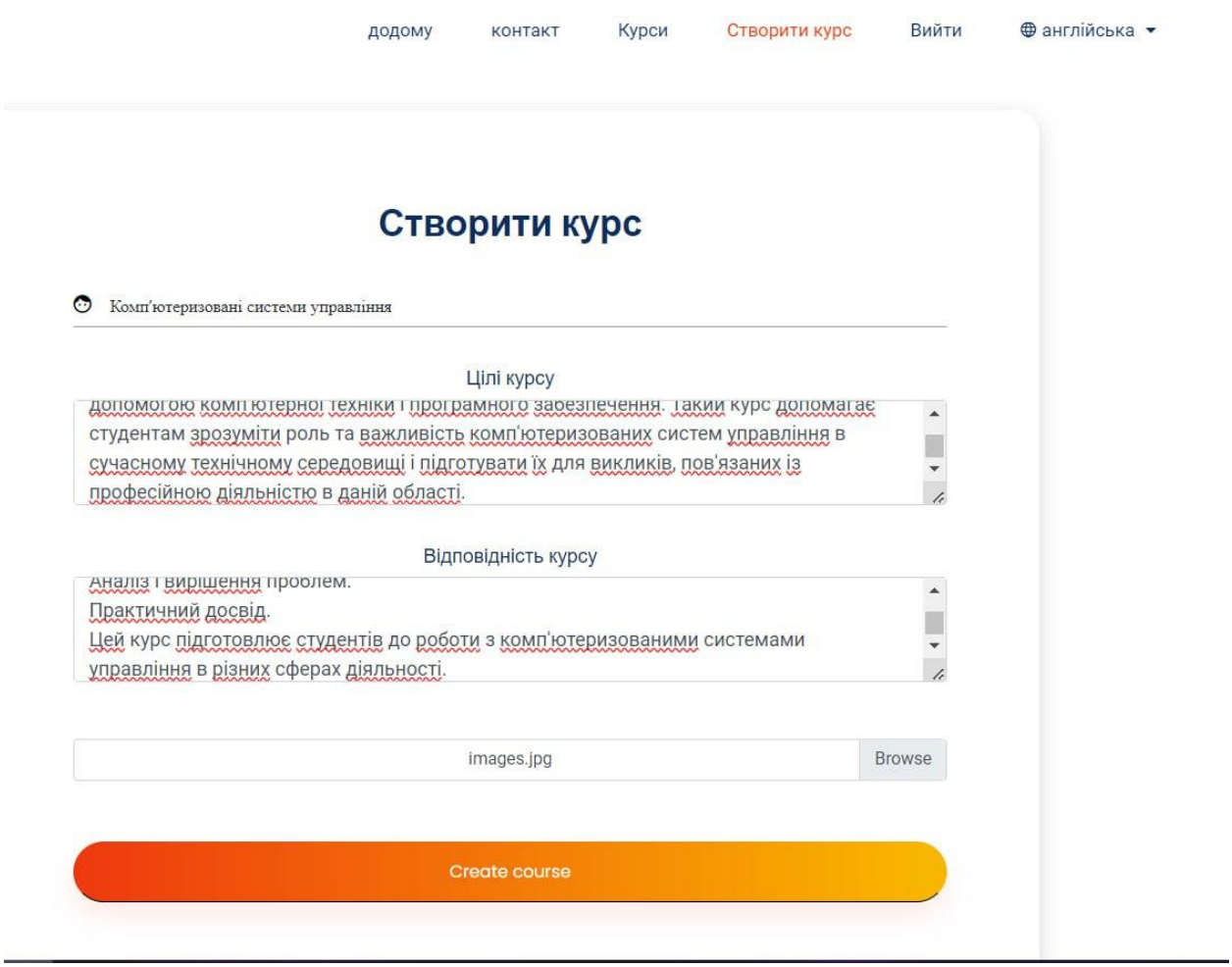


Рис. В.1 Тест 1

Тест 2

1. Перевірка виконання програми
2. Реєстрація студента;
3. Отримання результату щодо зарестрованого користувача.

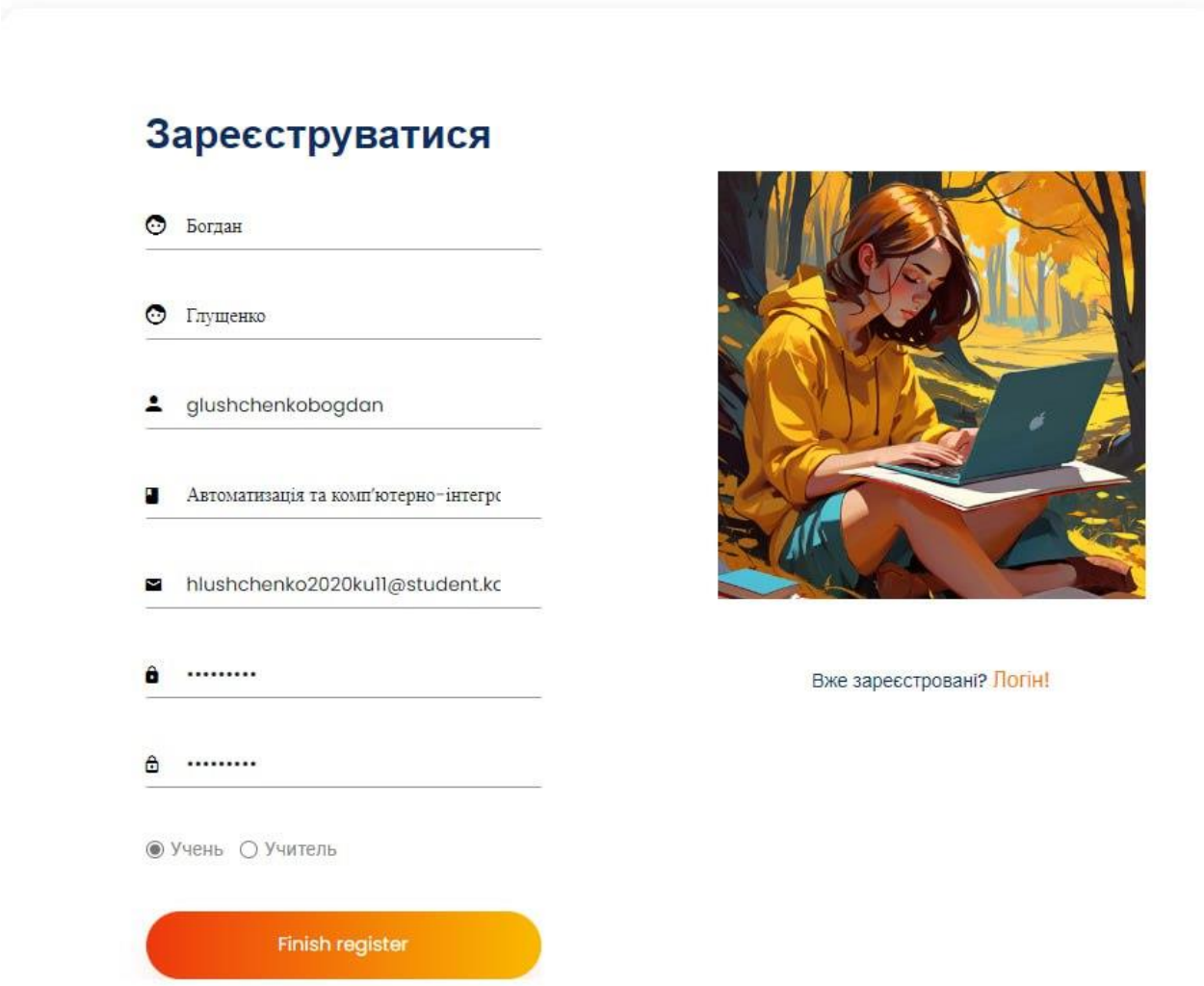


Рис. В.2 Тест 2

Тест 3

4. Перевірка виконання програми
5. Головна сторінка;
6. Отримання результату.

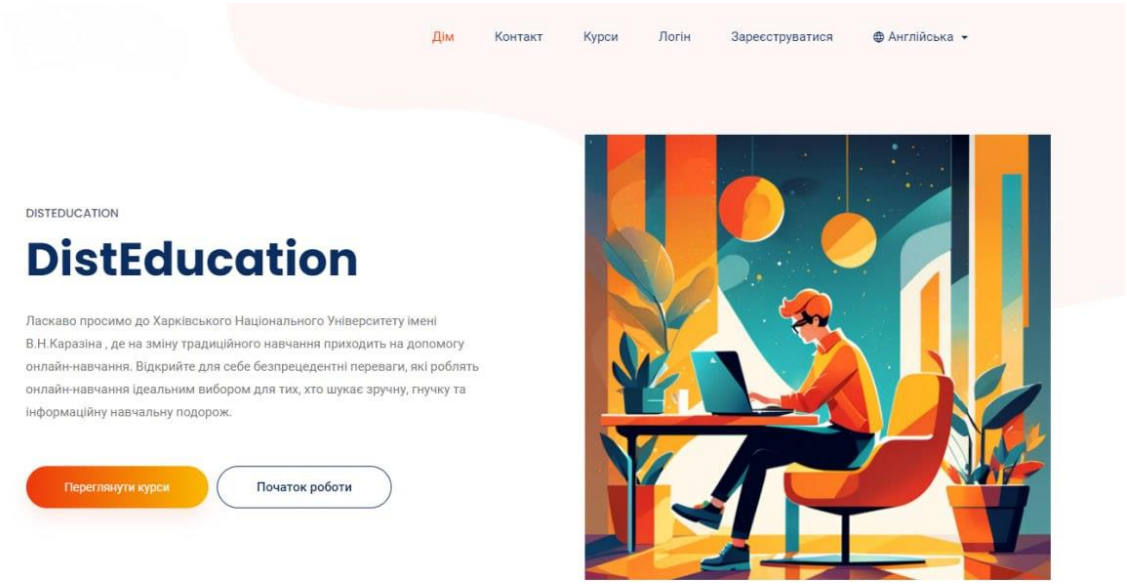


Рис. В.3 Тест 3

Тест вважається пройденим, якщо відбуваються вказані операції і їх відображення у програмному продукті.

Висновки: тест 1 успішно пройшов випробування, тест 2 успішно пройшов випробування і тест 3 успішно пройшов випробування. Випробування пройшло успішно.

Виконавець: студент групи КУ-41, Глущенко Б. О.

Додаток Г

Лістинг коду

```
import { SubmitHandler, useForm } from 'react-hook-form';
import { useTranslation } from 'react-i18next';
import { useNavigate } from 'react-router-dom';
import { apiUrlsConfig } from '../../../../config/apiUrls';
import { useAuthContext } from
'../../../../contexts/AuthContext';
import { useFetch } from '../../../../hooks/useFetch';
import useValidators from
'../../../../hooks/useValidator/useValidators';
import { PagesEnum } from '../../../../types/enums/PagesEnum';
import { RolesEnum } from '../../../../types/enums/RolesEnum';
import { IAuthResponse } from
'../../../../types/interfaces/auth/IAuthResponse';
import FormErrorWrapper from '../../../common/form-error-
wrapper/FormErrorWrapper';
import FormInput from '../../../common/form-input/FormInput';

type Inputs = {
  'First Name': string;
  'Last Name': string;
  Address: string;
  Description: string;
  role: RolesEnum;
};

// The component used to display and handle the finish register
form
export default function FinishRegisterForm() {
  const { t } = useTranslation();
  const { auth: validators } = useValidators();
  const navigate = useNavigate();
  const { loginUser } = useAuthContext();

  // Prepare fetches
  const { put, response } = useFetch<IAuthResponse>(
    apiUrlsConfig.auth.completeOAuth
  );

  // Handle form
  const { handleSubmit, register, control, reset } =
useForm<Inputs>({
    defaultValues: {
      'First Name': '',
      'Last Name': '',
      Address: '',
      Description: '',
      role: RolesEnum.USER,
    },
    mode: 'onChange',
```

```
  });

  // Handle form submit
  const onSubmit: SubmitHandler<Inputs> = async (data) => {
    const user = await put({
      firstname: data['First Name'].trim(),
      lastname: data['Last Name'].trim(),
      address: data.Address.trim(),
      description: data.Description.trim(),
      role: data['role'].trim(),
    });

    // Reset form, update auth state (login user) and navigate
to home
    if (response.ok) {
      reset();
      loginUser(user);
      navigate(PagesEnum.Home);
    }
  };

  return (
    <form
      onSubmit={handleSubmit(onSubmit)}
      className="register-form"
      id="register-form">
      <FormInput
        control={control}
        placeholder={t('finish.register.first.name')}
        name="First Name"
        type="text"
        iconClasses="zmdi zmdi-face material-icons-name"
        rules={validators.FIRST_NAME_VALIDATIONS}
      />

      <FormInput
        control={control}
        placeholder={t('finish.register.last.name')}
        name="Last Name"
        type="text"
        iconClasses="zmdi zmdi-face material-icons-name"
        rules={validators.LAST_NAME_VALIDATIONS}
      />

      <FormInput
        control={control}
        placeholder={t('finish.register.description')}
        name="Description"
        type="text"
        iconClasses="zmdi zmdi-book"
        rules={validators.DESCRIPTION_VALIDATIONS}
      />
```

```
      <FormInput
        control={control}
        placeholder={t('finish.register.address')}
        name="Address"
        type="text"
        iconClasses="zmdi zmdi-pin"
        rules={validators.ADDRESS_VALIDATIONS}
      />

      {/* No error message for role selection (there will always
be a selected role) */}
      <FormErrorWrapper message={undefined}>
        <div className="form-check-inline">
          <input
            {...register('role')}
            type="radio"
            className="form-check-input"
            value={RolesEnum.USER}
          />
          <p>{t('finish.register.role.student')}</p>
        </div>
        <div className="form-check-inline">
          <input
            {...register('role')}
            type="radio"
            className="form-check-input"
            value={RolesEnum.TEACHER}
          />
          <p>{t('finish.register.role.teacher')}</p>
        </div>
      </FormErrorWrapper>

      <div className="form-group form-button">
        <input
          type="submit"
          name="signup"
          id="signup"
          className="btn_1"
          value={t('finish.register.complete')}
        />
      </div>
    </form>
  );
}
import { RegisterOptions } from 'react-hook-form';
import { useTranslation } from 'react-i18next';

// The hook that construct validation for auth form fields
// with a message based on the user's selected language
export default function useAuthValidations() {
  const { t } = useTranslation();
```

```
// Common validations
const EMAIL_VALIDATIONS: RegisterOptions = {
  required: t('auth.email.required'),
  pattern: {
    value: /[a-z0-9]+@[a-z]+\.[a-z]{2,3}/gim,
    message: t('auth.email.invalid.pattern'),
  },
};

const PASSWORD_VALIDATIONS: RegisterOptions = {
  required: t('auth.password.required'),
  pattern: {
    value: /^(?=.*[A-Za-z])(?=.*\d)[A-Za-z\d]{8,}$/gim,
    message: t('auth.password.invalid.pattern'),
  },
};

// Register validations
const FIRST_NAME_VALIDATIONS: RegisterOptions = {
  required: t('auth.first.name.required'),
  minLength: {
    value: 2,
    message: t('auth.first.name.invalid.min-length'),
  },
};

const LAST_NAME_VALIDATIONS: RegisterOptions = {
  required: t('auth.last.name.required'),
  minLength: {
    value: 2,
    message: t('auth.last.name.invalid.min-length'),
  },
};

const USERNAME_VALIDATIONS: RegisterOptions = {
  required: t('auth.username.required'),
  pattern: {
    value: /^[a-z\d]{5,}$/gm,
    message: t('auth.username.invalid.pattern'),
  },
};

const ADDRESS_VALIDATIONS: RegisterOptions = {
  required: t('auth.address.required'),
  minLength: {
    value: 10,
    message: t('auth.address.invalid.min-length'),
  },
};

const DESCRIPTION_VALIDATIONS: RegisterOptions = {
```

```
    required: t('auth.description.required'),
    minLength: {
      value: 8,
      message: t('auth.description.invalid.min-length'),
    },
    maxLength: {
      value: 120,
      message: t('auth.description.invalid.max-length'),
    },
  };

  const REPEAT_PASSWORD_VALIDATIONS: RegisterOptions = {
    ...PASSWORD_VALIDATIONS,
    required: t('auth.repeat.password.required'),
  };

  return {
    EMAIL_VALIDATIONS,
    PASSWORD_VALIDATIONS,
    FIRST_NAME_VALIDATIONS,
    LAST_NAME_VALIDATIONS,
    USERNAME_VALIDATIONS,
    ADDRESS_VALIDATIONS,
    DESCRIPTION_VALIDATIONS,
    REPEAT_PASSWORD_VALIDATIONS,
  };
}
package com.coolSchool.coolSchool.models.entity;

import com.coolSchool.coolSchool.enums.Provider;
import com.coolSchool.coolSchool.enums.Role;
import jakarta.persistence.*;
import jakarta.validation.constraints.Email;
import jakarta.validation.constraints.NotNull;
import jakarta.validation.constraints.Size;
import lombok.AllArgsConstructor;
import lombok.Builder;
import lombok.Data;
import lombok.NoArgsConstructor;
import org.springframework.security.core.GrantedAuthority;
import
org.springframework.security.core.userdetails.UserDetails;

import java.time.LocalDateTime;
import java.util.Collection;


@Data
@Builder
@NoArgsConstructor
@AllArgsConstructor
@Entity
```

```
@Table(name = "_users")
public class User implements UserDetails {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @NotNull(message = "The name should not be null!")
    private String firstname;

    @NotNull(message = "The name should not be null!")
    private String lastname;

    @Email(message = "Email should be a well-formatted email!")
    @NotNull(message = "The email should not be null!")
    @Column(unique = true)
    private String email;

    private String password;

    @NotNull(message = "The address should not be null!")
    private String address;

    @NotNull(message = "The username should not be null!")
    @Column(name = "username")
    private String usernameField;

    @NotNull(message = "The description shot not be null!")
    @Size(min = 1, max = 120, message = "The description must be
between 60 and 120 symbols!")
    private String description;

    @NotNull
    @Enumerated(EnumType.STRING)
    private Role role;

    @ManyToOne
    @JoinColumn(name = "file_id")
    private File profilePic;

    @NotNull
    @Enumerated(EnumType.STRING)
    private Provider provider;

    @Column(name = "is_additional_info_required", nullable =
false)
    private boolean additionalInfoRequired;

    @Column(name = "is_deleted", nullable = false)
    private boolean deleted;
    @Column(name = "enabled")
    private boolean enabled;
```

```
    private LocalDateTime createdAt;

    @Override
    public Collection<? extends GrantedAuthority>
getAuthorities() {
        return role.getAuthorities();
    }

    @Override
    public String getPassword() {
        return password;
    }

    @Override
    public String getUsername() {
        return email;
    }

    @Override
    public boolean isAccountNonExpired() {
        return true;
    }

    @Override
    public boolean isAccountNonLocked() {
        return true;
    }

    @Override
    public boolean isCredentialsNonExpired() {
        return true;
    }

    public boolean isEnabled() {
        return enabled;
    }

    public void setEnabled(boolean enabled) {
        this.enabled = enabled;
    }

    @PrePersist
    void prePersist() {
        if (this.provider == null) {
            this.provider = Provider.LOCAL;
        }
    }
}
package com.coolSchool.coolSchool.services.impl.security;

import com.coolSchool.coolSchool.models.entity.User;
import com.coolSchool.coolSchool.services.UserService;
```

```
import
com.coolSchool.coolSchool.services.impl.security.events.OnRegist
rationCompleteEvent;
import org.jetbrains.annotations.NotNull;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.ApplicationListener;
import org.springframework.context.MessageSource;
import org.springframework.mail.SimpleMailMessage;
import org.springframework.mail.javamail.JavaMailSender;
import org.springframework.stereotype.Component;

import java.util.UUID;

/**
 * Component responsible for handling registration confirmation
emails.
 */
@Component
public class RegistrationListener implements
ApplicationListener<OnRegistrationCompleteEvent> {

    @Autowired
    private UserService service;

    @Autowired
    private MessageSource messages;

    @Autowired
    private JavaMailSender mailSender;

    @Override
    public void onApplicationEvent(@NotNull
OnRegistrationCompleteEvent event) {
        this.confirmRegistration(event);
    }

    private void confirmRegistration(OnRegistrationCompleteEvent
event) {
        User user = event.getUser();
        String token = UUID.randomUUID().toString();
        service.createVerificationToken(user, token);

        String recipientAddress = user.getEmail();
        String subject = "Cool School Registration
Confirmation";
        String confirmationUrl = event.getAppUrl() +
"auth/registrationConfirm?token=" + token;

        // Construct the email message
        String message = "Dear, " + user.getFirstname() + "\n\n"
                + "Thank you for registering with
DistEducation!\n\n"
```

```
                + "To complete your registration, please click
the following link to verify your email:\n"
                + confirmationUrl + "\n"
                + "If you did not create an account with us,
please ignore this email.\n"
                + "Best regards,\n"
                + "DistEducation Team!";

        SimpleMailMessage email = new SimpleMailMessage();
        email.setTo(recipientAddress);
        email.setSubject(subject);
        email.setText(message);

        System.out.println(message);

//        mailSender.send(email);
    }
}
```