Форма № 7

ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ І. І. МЕЧНИКОВА

(повне найменування закладу вищої освіти)

Факультет математики, фізики та інформаційних технологій

(повне найменування факультету)

Кафедра механіки, автоматизації та інформаційних технологій

(повна назва кафедри)

Кваліфікаційна робота

на здобуття ступеня вищої освіти «бакалавр»

«Розробка інформаційної системи формування навичок проходження НМТ»

(тема кваліфікаційної роботи українською мовою)

«Development of an information system for the formation of skills for passing NMT»

(тема кваліфікаційної роботи англійською мовою)

Виконав: здобувач денної форми навчання
спеціальності 126 Інформаційні системи та технології
(код, назва спеціальності)
Освітня програма «Інформаційні системи та технології»
(назва)
Карайван Денис Андрійович
(прізвище, ім'я, по-батькові здобувача)

Керівник __к.т.н. Іщенко О.В._____
(науковий ступінь, вчене звання, прізвище, ініціали) (підпис)
Рецензент __д.т.н., проф. Волков В.Е.___
(науковий ступінь, вчене звання, прізвище, ініціали)

Рекомендовано до захисту:
Протокол засідання кафедри
_____
№ ___ від ____.____. 20___ р.

Завідувач(ка) кафедри
_____ Алла РАЧИНСЬКА
(підпис) (прізвище,ім'я)

Захищено на засіданні ЕК № _____
протокол № __від ____.____.20___ р.

Оцінка_____/___/_____
(за національною шкалою/шкалою ECTS/ бали)
Голова ЕК
____ Олена АРСІРІЙ
(підпис) (прізвище, ім'я)

Одеса 2025

АНОТАЦІЯ

Метою дипломної роботи є створення комплексної інформаційної системи для підтримки процесу підготовки до національного мультипредметного тесту (НМТ), яка дозволяє автоматизувати процес формування тестових завдань, їх редагування та проходження. В рамках реалізації було розроблено програмне забезпечення з графічним інтерфейсом користувача, що забезпечує інтерактивну взаємодію як для адміністратора системи, так і для користувачів-тестувальників.

Система підтримує кілька типів тестових завдань, зокрема:

1. Тести з вибором правильної відповіді, які дозволяють користувачу обрати один з наданих варіантів.
2. Завдання на відповідність, де користувачу необхідно зіставити елементи з різних списків.
3. Завдання з введенням відповіді, в яких користувач вводить текстове значення як відповідь.

Окрему увагу приділено реалізації функціоналу адміністратора. Адміністратор має змогу створювати нові тестові файли, додавати завдання до існуючих тестів, вибираючи тип питання (тест із вибором, на відповідність чи з відкритою відповіддю), а також приєднувати до завдань зображення. Додатково реалізовано модуль редагування вже створених завдань і можливість конфігурування кількості питань кожного типу в одному тесті, що забезпечує гнучкість у формуванні контрольних робіт відповідно до вимог освітнього процесу.

ABSTRACT

The objective of the thesis is to develop a comprehensive information system to support the process of preparing for the National Multisubject Test (NMT), which allows for the automation of the process of forming test tasks, their editing, and execution. As part of the implementation, software with a graphical user interface was developed, providing interactive interaction for both the system administrator and the test users.

The system supports several types of test tasks, including:

1. Multiple-choice tests that allow the user to select one of the provided options.
2. Matching tasks, where the user needs to match elements from different lists.
3. Input tasks, in which the user enters a textual value as an answer.

Particular attention is given to the administrator's functionality. The administrator is able to create new test files, add questions to existing tests by selecting the type of task (multiple-choice, matching, or open-ended), and attach images to questions. Additionally, the system includes a module for editing previously created questions and configuring the number of questions of each type in a test, offering flexibility in the formation of assessments in accordance with educational requirements.

ЗМІСТ

ВСТУП

Актуальність теми. Національний мультипредметний тест (НМТ) є важливою складовою освітнього процесу, що визначає рівень підготовки учнів до вступу у заклади вищої освіти. З огляду на постійні зміни в освітніх програмах, зростає потреба у створенні сучасних інформаційних систем, що дозволяють ефективно готуватися до тестування. Традиційні методи підготовки, такі як паперові посібники чи класичні підручники, не завжди відповідають вимогам сучасного суспільства, оскільки не забезпечують достатнього рівня інтерактивності та адаптації до індивідуальних потреб учнів.

У зв'язку з цим, створення комплексної інформаційної системи, яка дозволяє не лише формувати тестові завдання, а й інтерактивно взаємодіяти з користувачем, набуває особливої актуальності. Така система повинна забезпечувати автоматизацію процесів створення, редагування та проходження тестів, а також підтримку різних типів завдань, що дозволяє оптимізувати процес підготовки до НМТ.

Ціль та задачі дослідження. Метою дипломної роботи є розробка інформаційної системи формування навичок проходження НМТ, яка забезпечує комплексний підхід до створення та виконання тестових завдань різного типу. Система має бути зручною як для адміністраторів (викладачів), так і для користувачів (учнів), надаючи можливість налаштування тестових матеріалів та аналізу результатів.

Для досягнення поставленої мети необхідно вирішити наступні завдання:

Дослідити сучасні методи та інструменти підготовки до НМТ та проаналізувати їхні можливості.

Розробити архітектуру інформаційної системи з урахуванням вимог інтерактивності та зручності використання.

Реалізувати основні функціональні модулі системи, зокрема:

модуль адміністрування тестових файлів;

модуль створення тестових завдань з можливістю додавання зображень;

модуль виконання тестів з підтримкою декількох типів питань (з вибором відповіді, на відповідність, з введенням тексту).

Забезпечити можливість інтеграції графічних матеріалів у тестові завдання для підвищення наочності.

Провести тестування програмного продукту та проаналізувати результати для виявлення недоліків та перспектив удосконалення системи.

Об’єкт і предмет дослідження. Об’єктом дослідження є процес підготовки до національного мультипредметного тесту (НМТ) за допомогою інформаційних систем.

Предметом дослідження є програмні засоби автоматизації процесу створення та проходження тестових завдань з підтримкою різних типів питань.

1 АНАЛІЗ ПРОЦЕСУ ПІДГОТОВКИ ДО (НМТ)

1.1 Огляд літератури з питань тестування та формування навичок

У сучасній освітній системі тестування стало основним інструментом оцінювання знань, навичок і готовності до подальшого навчання. Зокрема, Національний мультипредметний тест (НМТ) як форма вступного випробування потребує від учнів не лише предметної обізнаності, а й сформованих навичок взаємодії з тестовими системами.

Згідно з численними аналітичними публікаціями, традиційні методи підготовки, зокрема робота з друкованими підручниками, посібниками та збірниками завдань, втрачають ефективність у порівнянні з інтерактивними цифровими платформами. Причиною цього є обмежений ступінь зворотного зв’язку, низька динамічність та відсутність адаптації під індивідуальні потреби учня. Як зазначає Зайцева І.О. [1], цифрові ресурси дозволяють гнучко змінювати зміст і складність завдань, що особливо важливо в умовах індивідуалізації навчання.

Сучасні дослідження свідчать, що використання інтерактивних тестових систем значно підвищує якість підготовки до стандартизованих іспитів. Наприклад, Білик Н.С. [2] підкреслює, що можливість багаторазового проходження тестів у тренувальному режимі сприяє зниженню рівня тривожності, підвищує зосередженість та формує навички швидкого прийняття рішень у ситуації обмеженого часу. Це особливо важливо в умовах, коли тести мають чітко регламентований час виконання.

Особливу увагу приділяють типології завдань. Як показує практика, поєднання завдань із вибором правильної відповіді, завдань на встановлення відповідності та запитань із відкритою відповіддю забезпечує глибше охоплення навчального матеріалу. Така комбінація дозволяє не лише перевірити пам’ять учня, але й оцінити його логіку, критичне мислення, здатність працювати з інформацією у складних умовах.

Крім того, як зазначає Сидоренко В.П. [3], візуальні елементи (схеми, діаграми, малюнки) значно підвищують якість сприйняття інформації, особливо в тестах з точних або природничих наук. Візуалізація полегшує розуміння умови, особливо у математичних та природничих дисциплінах. Саме тому більшість сучасних електронних тестових систем мають функцію додавання графічних матеріалів до завдань, що позитивно впливає на якість підготовки.

Окреме місце в дослідженнях займає гнучкість та адаптивність систем. Мова йде не лише про зміну рівня складності, а й про можливість вибору кількості питань, форматів завдань, предметних тем. Важливо, щоб система дозволяла адаптувати зміст під потреби конкретної аудиторії — наприклад, при підготовці до НМТ з математики, української мови чи історії.

Ще один важливий фактор — мотиваційна складова. Досвід використання інтерактивних платформ свідчить, що цифровий формат, можливість бачити результати в реальному часі, накопичення балів, проходження міні-тестів — усе це підвищує зацікавленість учнів, особливо серед старшокласників. Мотивація, в свою чергу, безпосередньо впливає на тривалість підготовки, послідовність навчання та загальну ефективність.

Таким чином, сучасна освітня наука та практика однозначно схиляються до того, що найефективнішою моделлю підготовки до НМТ є використання цифрових тестових платформ. Вони повинні забезпечувати не лише контроль знань, а й розвивати самостійність, здатність працювати із завданнями різної форми та складності, створювати позитивний користувацький досвід.

Розробка подібної системи потребує врахування широкого спектра чинників: психологічної адаптації учня до цифрового середовища, ергономіки інтерфейсу, гнучкості в налаштуваннях, наявності зворотного зв’язку та технічної стабільності. Саме ці аспекти є визначальними при створенні якісного програмного забезпечення для підготовки до НМТ і закладені в основу дипломного проєкту.

1.2 Визначення вимог до інформаційної системи тестування

Ефективна розробка інформаційної системи для підготовки до НМТ передбачає чітке визначення функціональних, технічних та користувацьких вимог. На етапі проєктування важливо врахувати як специфіку освітнього середовища, так і потреби різних категорій користувачів — учнів, викладачів, адміністраторів системи.

Насамперед система має забезпечити базовий функціонал, який включає:

- створення, редагування та збереження тестових завдань;
- виконання тестів користувачем з підрахунком результатів;
- зручну навігацію між завданнями;
- виведення підсумкової оцінки та часу, витраченого на проходження тесту;
- підтримку кількох типів завдань;
- відображення графічних матеріалів.

Важливою особливістю є забезпечення гнучкості у побудові тесту: адміністратор має мати змогу самостійно визначати кількість завдань кожного типу (наприклад, 10 одиничних тестів, 3 на відповідність, 5 з відкритою відповіддю) та комбінувати їх відповідно до цілей навчання або теми.

Вибір структури зберігання даних

Після аналізу можливих варіантів збереження інформації було ухвалено рішення не використовувати реляційну базу даних, а натомість зберігати дані у форматі JSON. Це рішення базується на кількох важливих аргументах:

1. Простота реалізації

Формат JSON дозволяє уникнути складних запитів, налаштування серверів та

СУБД. У межах десктопного застосунку, де всі дані зберігаються локально, JSON є оптимальним варіантом, який легко інтегрується в програму на Python.

2. Легкість у редагуванні

Завдання у форматі JSON зручно оновлювати, редагувати або копіювати вручну без спеціального програмного забезпечення. Це особливо корисно для адміністраторів, які можуть внести правки напряму у файлі, не запускаючи додаток.

3. Портативність і незалежність

JSON-файли можна легко переносити з одного комп'ютера на інший, що робить систему мобільною. Не потрібно зберігати або переносити базу даних, що зменшує ризики пошкодження або втрати інформації.

4. Відсутність надмірної складності

Для локальної системи з обмеженим числом тестів і завдань база даних є надмірною. Формат JSON повністю покриває функціональні потреби без перевантаження системи.

Функціональні вимоги до системи

На основі аналізу освітніх потреб, користувацького досвіду та типових сценаріїв використання були сформульовані такі функціональні вимоги до інформаційної системи:

- Підтримка різних типів завдань, а саме:

– з вибором однієї правильної відповіді;
– на встановлення відповідностей;
– із введенням текстової відповіді.

- Можливість додавання графіки до кожного завдання — як до питання, так і до варіантів відповідей.
- Налаштування кількості завдань кожного типу перед запуском тесту. Це дозволяє викладачам/адміністраторам швидко адаптувати тест до теми, рівня складності чи кількості часу на виконання.

- Редагування тестових файлів — можливість змінювати існуючі завдання, видаляти або додавати нові, змінювати відповіді, оновлювати зображення.
- Зрозумілий графічний інтерфейс з мінімалістичним дизайном, що забезпечує легке орієнтування в системі, навіть для користувачів без досвіду роботи з тестовими платформами.
- Виведення результатів після проходження тесту, з кількістю правильних відповідей, загальним балом та вказанням часу.
- Автоматичне збереження результатів роботи (у рамках сесії), аби у випадку випадкового закриття вікна користувач не втратив прогрес.

Нефункціональні вимоги

Окрім основного функціоналу, були сформульовані нефункціональні вимоги, які покращують загальне враження від роботи з програмою:

- Стабільність: система не повинна давати збої при роботі з великими об’ємами тестових даних чи зображень.
- Швидкість: відкриття файлів, перемикання між завданнями та збереження результатів мають відбуватись без затримок.
- Локальність: вся робота із завданнями та тестами відбувається на одному комп’ютері, без необхідності доступу до Інтернету.
- Безпека: структура даних не повинна дозволяти випадкове перезаписування чи видалення важливих елементів без підтвердження дій.

Динаміка тестування

Однією з ключових переваг розробленої системи є динамічність формування тестів. Під час запуску тесту запитання автоматично сортуються у випадковому порядку, що унеможливлює механічне запам’ятовування правильних відповідей при багаторазовому проходженні. Такий підхід відповідає сучасним підходам до тестування, де важливим є не запам’ятати розташування відповіді, а зрозуміти зміст завдання.

Крім того, випадкове формування складу тесту — з урахуванням заздалегідь визначеної кількості завдань кожного типу — дає змогу створювати унікальні варіації тесту для кожного користувача**.** Це особливо важливо в умовах самоперевірки або проведення внутрішнього контролю знань без ризику списування.

Відповідність дизайну реальному НМТ

Інтерфейс тестування спроєктований з урахуванням максимального наближення до дизайну реального НМТ**.** Застосовано кольорову схему, характерну для офіційних тестових платформ, використано логіку навігації, яка відповідає звичному формату: відображення номера завдання, чіткий поділ між запитанням і відповідями, можливість повернення до попередніх завдань.

Цей підхід має важливе психологічне значення: учень звикає до формату, що зменшує стрес у реальних умовах іспиту. Тестування відбувається у візуальному середовищі, яке повторює структуру та логіку офіційної онлайн-платформи, зокрема: наявність таймера, підсумкового екрана з результатом, кнопок перемикання між питаннями.

Поєднання вдалого дизайну та динамічності створює реалістичне середовище, у якому учень не просто тренується, а проходить максимально наближений до справжнього формат тестування.

Таким чином, визначені вимоги до інформаційної системи сформовано на основі реальних потреб учнів та викладачів у контексті підготовки до НМТ. Усі обрані рішення спрямовані на досягнення балансу між функціональністю, простотою реалізації та зручністю користування.

1.3 Аналіз існуючих рішень та інструментів для тестування

Для ефективної розробки власної інформаційної системи підготовки до НМТ необхідно здійснити аналіз вже наявних рішень, які пропонуються як у форматі веб-платформ, так і мобільних застосунків. Такий аналіз дозволяє

визначити сильні та слабкі сторони існуючих інструментів, а також чітко сформулювати функціональні переваги майбутнього програмного продукту.

Одним із найпоширеніших рішень для онлайн-підготовки до ЗНО/НМТ є Освіта.ЮА. Ця платформа надає доступ до тестових завдань з різних предметів у веб-форматі, а також дозволяє проходити тести онлайн з моментальним підрахунком результатів. Її переваги:

- великий обсяг готових тестів, зокрема офіційних матеріалів минулих років;
- підтримка таймера, що моделює реальні умови складання іспиту;
- збереження результатів проходження;
- простий інтерфейс для кінцевого користувача.

Однак, платформа має й суттєві обмеження. По-перше, вона орієнтована виключно на роботу в браузері, що створює залежність від стабільного інтернет-з’єднання. По-друге, користувач не має можливості створювати власні тести чи редагувати наявні завдання. Таким чином, гнучкість відсутня**,** а весь функціонал зосереджено лише на проходженні фіксованих тестів.

Ще одним популярним рішенням є мобільний застосунок Просте.ЗНО**,** що позиціонується як зручний інструмент для проходження тестів у смартфоні. Його основні переваги:

- підтримка офлайн-режиму;
- адаптація до мобільних пристроїв;
- швидке завантаження та простота використання.

Однак, система також має функціональні обмеження**:**

- відсутня підтримка завдань типу "на відповідність";
- немає можливості додавати зображення до завдань;
- обмежений набір тестів, що зазвичай базується на матеріалах минулих років;
- відсутність адміністративного режиму або конструктора завдань.

Онлайн-сервіси загального призначення

Існують також платформи на кшталт Google Forms, Quizlet, Kahoot, які широко використовуються у навчальному процесі. Вони дозволяють створювати тести з різними типами запитань, проте (табл. 1.1):

- більшість із них не адаптовані під формат НМТ;
- інтерфейс може бути надто загальним або ігровим (особливо у Kahoot);
- обмежена підтримка математичних виразів або вставки графічних матеріалів в окремі елементи завдань;
- немає контролю над локальним збереженням даних або офлайн-режимом.

Платформа	Створення завдань	Режим адміністратора	Типи завдань	Графіка	Робота офлайн	Налаштування тестів
Освіта.ЮА	ні	ні	частково	частково	ні	ні
Просте.ЗНО	ні	ні	частково	ні	так	ні
Google Forms	так	частково	так	ні	частково	частково
Запропонована система	так	так	так	так	так	так

Таблиця 1.1-порівняльня характеристика аналогів.

Як видно з таблиці, жодна з існуючих платформ не забезпечує повного спектра можливостей, які є критично важливими для сучасної підготовки до НМТ. Саме тому виникає необхідність створення системи, яка об’єднує переваги наявних рішень і усуває їх обмеження.

Потреба у новій системі

Переваги запропонованої системи полягають у наступному:

- повноцінна підтримка всіх типів завдань, які зустрічаються в НМТ;
- можливість адміністративного управління тестами — створення, редагування, оновлення;

- інтеграція графічних матеріалів як до питань, так і до варіантів відповідей;
- автономна робота без інтернету;
- локальне збереження всіх даних — важлива вимога у шкільному середовищі;
- інтерфейс, наближений до формату офіційного НМТ, що знижує психологічне навантаження учня.

Таким чином, аналіз існуючих рішень підтверджує необхідність розробки нової системи, яка дозволить поєднати зручність, функціональність, гнучкість та адаптацію до реальних умов проведення тестування. Саме такі вимоги були покладені в основу програмного рішення, реалізованого в межах цієї дипломної роботи.

2 ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ ТЕСТУВАННЯ

2.1 Вибір архітектури програмної системи

Проєктування архітектури інформаційної системи є одним із ключових етапів створення ефективного та масштабованого програмного забезпечення. Саме архітектурне рішення визначає, наскільки легко буде реалізовувати, підтримувати й розширювати функціонал у майбутньому. У контексті розробки системи підготовки до НМТ важливо досягти балансу між простотою реалізації, швидкістю роботи, незалежністю від зовнішніх серверів і зручністю для користувача.

У якості архітектурної основи було обрано клієнтську однокористувацьку модель з локальним зберіганням даних. Уся логіка та обробка інформації реалізовані на стороні користувача — програма встановлюється на комп’ютер і не вимагає з’єднання з Інтернетом чи з зовнішніми серверами. Цей підхід був обраний на користь:

- простоти розгортання: система не потребує встановлення серверного ПЗ чи налаштування баз даних;
- високої автономності: можна працювати без доступу до мережі, що важливо для шкіл або домашніх користувачів;
- зручності використання: всі дані зберігаються в окремих JSON-файлах, що дозволяє швидко змінювати вміст тестів.

Основні компоненти архітектури

Архітектура системи поділена на кілька логічних блоків:

Графічний інтерфейс користувача (GUI) реалізовано за допомогою бібліотек Tkinter та CustomTkinter, що дозволяє створити інтуїтивно зрозумілий інтерфейс із сучасним візуальним стилем.

Основні елементи:

- екран вибору предмета;

- екран проходження тесту;
- панель адміністратора;
- вікна створення/редагування завдань.

Модуль адміністрування тестів відповідає за:

- створення нових тестових файлів;
- вибір типу завдання;
- введення тексту, варіантів відповідей;
- додавання графіки;
- збереження в форматі JSON.

Модуль тестування реалізує функціонал проходження тестів:

- завантаження питань з обраного файлу;
- формування динамічного списку завдань;
- облік правильних відповідей;
- виведення результату.

Модуль збереження та обробки даних відповідає за:

- зчитування і запис JSON-файлів;
- налаштування кількості питань;
- зберігання зображень (в папці images/);
- підтримку стабільної файлової структури.

Логічна структура реалізації проєкту

Щоб систематизувати процес розробки і забезпечити цілісне уявлення про всі його етапи, було створено логічну карту, яка демонструє послідовність дій — від аналізу предметної області до етапів захисту.

Схема (див. рисунок 2.1) дає змогу не лише структурувати хід проєкту, але й чітко визначити ролі та задачі на кожному з рівнів. Наприклад, спочатку здійснюється аналіз предметної області (визначення цільової аудиторії, аналіз аналогів), після чого йде проєктування (логіка, сценарії, інтерфейс), реалізація (функції, GUI), а завершальний етап включає підготовку до захисту. Такий підхід є класичним у розробці програмного забезпечення та відповідає практикам інженерії ПЗ.

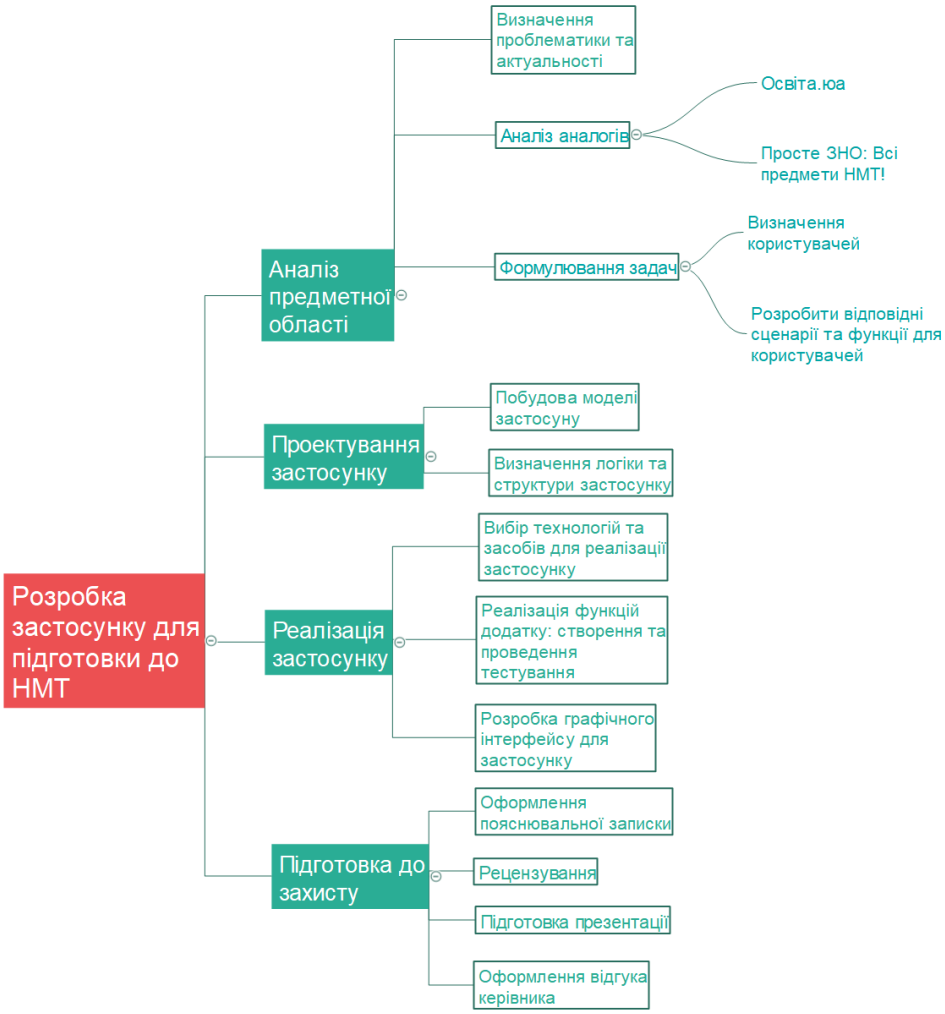

Рисунок 2.1 – Структура проекту

Обґрунтування вибору архітектури

Під час аналізу архітектурних варіантів були розглянуті інші моделі (табл. 2.2), зокрема:

Модель	Переваги	Недоліки
Клієнт-серверна (через веб)	Віддалений доступ, масштабування	Складність реалізації, потреба в хостингу, проблеми з безпекою
Хмарна модель	Автоматичне збреження, централізовані оновлення	Повна залежність від Інтернету, висока складність
Локальна файлова (обрана)	Простота, стабільність, автономність	Не підходить для одночасної роботи багатьох користувачів

Таблиця 2.2 – моделі архітектурних варіанті

У зв’язку з тим, що основними користувачами системи є викладачі та учні, які часто працюють у локальному середовищі (без мережі), було вирішено реалізувати самодостатню систему (рис. 2.3), що не потребує встановлення додаткових серверів чи інфраструктури.

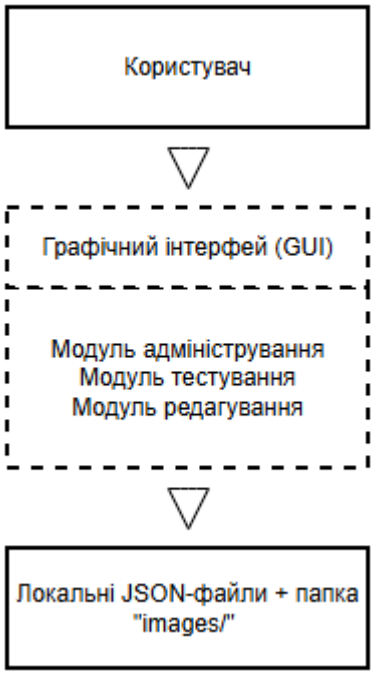

Рисунок 2.3 – архітектура системи

Гнучкість та розширюваність

Обрана архітектура дозволяє:

- додавати нові типи завдань (наприклад, "заповнення пропусків", "числові відповіді");
- розширювати формат метаданих у JSON (наприклад, рівень складності, тема);
- інтегрувати нові модулі без перебудови основної структури;
- адаптувати дизайн під інші типи тестів (не лише НМТ).

Таким чином, обрана архітектура повністю відповідає вимогам до гнучкості, стабільності та простоти використання. Вона дозволяє реалізувати необхідний функціонал без складної інфраструктури, що робить систему придатною як для індивідуального користування, так і для освітніх закладів.

2.2 Проектування структури даних тестових завдань

Одним із основних завдань під час створення інформаційної системи є правильне моделювання та організація структури даних. Це забезпечує стабільну роботу програми, зручність редагування і гнучкість для майбутніх змін. У межах даного проєкту для зберігання тестових завдань було обрано формат JSON (JavaScript Object Notation) — просту та легку для читання текстову структуру, що широко використовується в програмуванні.

Переваги формату JSON

Формат JSON має низку переваг, які роблять його ідеальним для застосування в локальній десктопній системі:

- Людинозрозумілість: навіть без спеціальних інструментів файл можна відкрити у звичайному текстовому редакторі (наприклад, Notepad++).
- Гнучкість структури: можна легко додавати нові поля, типи даних або вкладені об'єкти.
- Просте збереження та зчитування за допомогою стандартних бібліотек Python.
- Незалежність від СУБД**:** жодних зовнішніх залежностей або налаштування серверів.

Всі тестові файли мають розширення .json та зберігаються у робочій директорії програми. Кожен файл містить список об'єктів, де кожен об'єкт відповідає одному тестовому завданню.

Загальна структура завдання

Ось приклад типового завдання з вибором відповіді:

```
single_question
│
├── type: "single_question"
├── question_text: "Скільки буде 2 + 2?"
├── answer: "Б"
├── options
│   ├── А → "3"
│   ├── Б → "4"
│   ├── В → "5"
│   └── Г → "2"
└── image_path: ""
```

Ключові поля:

- "type" — тип завдання (наприклад, "single_question", "match", "input").
- "question_text" — текст питання.
- "options" — словник варіантів відповідей (для "single_question").
- "answer" — правильна відповідь (наприклад, літера або текст).
- "image_path" — шлях до зображення, яке додається до питання (опційне поле).

Приклад завдання на відповідність:

```
match
│
├── type: "match"
├── question_text: "Установіть відповідність між числами та їх квадратами."
├── answer
│   ├── "1" → "А"
│   ├── "2" → "Б"
│   └── "3" → "В"
└── image_path: ""
```

У цьому випадку:

- "answer" є словником, де ключі — це умовні позначення (цифри), а значення — правильні відповіді (літери).
- "image_path" зберігає графіку, яка ілюструє завдання (опційно).

Приклад завдання з введенням відповіді:

```
input
│
├── type: "input"
├── question_text: "Скільки буде 5 * 6?"
├── answer: "30"
└── image_path: ""
```

Тут відповідь перевіряється за точним збігом тексту, введеного користувачем. У майбутньому цю структуру можна розширити, додавши поля для варіантів допустимих відповідей або регулярних виразів.

Логіка та переваги структури

1. Один файл — один список завдань. Це дозволяє мати окремі тести для кожної теми, класу чи предмету. Адміністратор може швидко створити новий файл або скопіювати існуючий.
2. Мінімальна надмірність. Дані не дублюються. Всі елементи мають чітку структуру без зайвих вкладень, що пришвидшує обробку і спрощує логіку коду.
3. Масштабованість. Якщо в майбутньому потрібно додати нові типи завдань (наприклад, “обрати всі правильні відповіді” чи “відповідь у вигляді формули”), це можна реалізувати, просто доповнивши поле "type" новим значенням і відповідною структурою.
4. Відповідність структурі GUI. Формат JSON логічно співпадає зі структурою графічного інтерфейсу: текст питання, зображення, варіанти відповідей, правильна відповідь — усе це безпосередньо відображається у вікнах створення та проходження тесту.

Таким чином, структура даних, побудована на форматі JSON, дозволяє реалізувати повноцінну багатофункціональну систему без потреби в складних інструментах зберігання. Вона є гнучкою, масштабованою, простою для адміністраторів та зручною для подальшої підтримки.

2.3 Проектування інтерфейсу користувача

Інтерфейс користувача (UI) є критично важливою складовою будь-якої інформаційної системи, адже саме через нього відбувається взаємодія користувача з усіма модулями програми. У контексті системи підготовки до НМТ, де користувачами є переважно учні, викладачі та адміністратори, інтерфейс має бути максимально інтуїтивним, мінімалістичним, функціональним та адаптованим до освітнього процесу.

Основні принципи дизайну

Під час проєктування інтерфейсу було дотримано наступних ключових принципів:

- Простота: мінімум непотрібних елементів, логічне групування функцій, відсутність перевантаження інформацією.
- Інтуїтивність: користувач легко розуміє, як користуватись системою, навіть без інструкції.
- Однотипність компонентів: всі кнопки, поля введення, списки та повідомлення мають однаковий стиль, що знижує когнітивне навантаження.
- Модульність: кожен функціонал винесено в окреме вікно або секцію — адміністрування, тестування, налаштування.
- Сучасність: використано бібліотеку CustomTkinter, що дозволяє створити сучасний інтерфейс з підтримкою темної/світлої теми, закруглених кутів, тіней, адаптивних кнопок тощо.

Основні екрани та логіка навігації

1. Головне вікно / Екран вибору предмета. Це перший екран, який бачить користувач.

Він містить:

- заголовок ("Оберіть дисципліну");
- кнопки вибору предмета (наразі доступна "Математика", інші — у стані розробки);
- кнопку переходу до налаштувань/адміністрування.

Кнопки реалізовані як елементи з однаковим розміром і стилем**,** з чітким відгуком на натискання. Це зручно навіть для молодших користувачів, які тільки знайомляться з комп’ютером.

2. Панель адміністратора

Адміністратор має окремий екран, що відкривається після логіну. У ньому зібрано всі ключові функції:

- вибір файлу тесту;
- створення нового файлу;
- додавання нових завдань різного типу;
- редагування існуючих;
- налаштування кількості запитань;
- запуск редактора завдань.

Інтерфейс реалізований у вигляді блоків з великими підписами, усі дії згруповані логічно. Адміністратор не переходить між вкладками — усе на одній сторінці.

3. Вікна створення завдань

Для кожного типу завдання (вибір, відповідність, введення) відкривається окреме вікно. Вони мають:

- текстове поле для питання;
- кнопку завантаження зображення;
- інтерфейс для варіантів відповіді (у вигляді списку);
- поле для введення правильної відповіді;
- кнопку збереження.

Для завдань на відповідність реалізовано два стовпці зі стрілками та списками пар. Для завдань з вибором — варіанти "А", "Б", "В", "Г", кожен з

окремим полем і можливістю прикріпити картинку. Цей рівень деталізації дозволяє створювати наочні й адаптивні тести.

4. Вікно проходження тесту (режим користувача)

Інтерфейс користувача під час проходження тесту був розроблений з урахуванням максимальної відповідності реальному вигляду НМТ**,** зокрема:

- таймер у верхньому лівому куті;
- кнопка "Завершити тест" — праворуч;
- основне питання з текстом і зображенням у центрі;
- варіанти відповідей — під питанням, з можливістю вибору кліком;
- праворуч — панель навігації по номерах завдань, із кольоровим позначенням: зелений — правильно, червоний — помилка, сірий — не відповіли.

Інтерфейс працює без прокрутки, усі елементи вміщено на один екран, що покращує фокус уваги користувача.

5. Результати тесту

Після завершення тесту відображається підсумок:

- кількість правильних відповідей;
- час виконання;
- можливість повернутись на головний екран.

У майбутньому передбачено реалізацію виводу статистики, наприклад, точність по типах завдань або збереження результатів.

Безбар’єрність і доступність

Інтерфейс розроблений з урахуванням базових принципів доступності:

- високий контраст тексту та фону;
- великі елементи керування;
- відсутність візуального перевантаження;
- підтримка керування клавіатурою.

Це робить систему зручною для дітей, людей з незначними порушеннями зору та користувачів без досвіду.

Таким чином, інтерфейс користувача системи побудований на сучасних принципах UX/UI дизайну. Він забезпечує комфортну роботу для всіх категорій користувачів, наближений до формату реального НМТ і дозволяє повністю зануритись у процес тестування без додаткових пояснень або навчання.

2.4 Сценарії використання системи

Для забезпечення зручності та ефективності користування інформаційною системою підготовки до НМТ було розроблено низку сценаріїв взаємодії, які охоплюють повний життєвий цикл роботи з тестовими матеріалами — від створення завдань до аналізу результатів. Сценарії базуються на реальних ситуаціях, які можуть виникати під час самопідготовки учнів або проведення контрольних робіт у навчальних закладах.

Основні ролі користувачів

У системі умовно виділено дві основні ролі:

1. Адміністратор (викладач, методист, керівник гуртка). Має доступ до розширеного функціоналу, пов’язаного з формуванням тестів, редагуванням завдань, налаштуванням параметрів, створенням нових тестових файлів.

2. Користувач (учень, абітурієнт). Має доступ до проходження тестів, перегляду завдань, введення відповідей, отримання результатів.

Сценарій 1: Створення нового тесту (роль — адміністратор)

Мета**:** створити новий тест для конкретної теми з визначеною кількістю запитань кожного типу.

Кроки:

1. Користувач запускає програму та натискає кнопку "Налаштування".
2. Вводить логін/пароль адміністратора.

3. Обирає "Створити новий файл" → вводить назву файлу (наприклад, algebra_test.json).
4. Вибирає тип завдання: наприклад, "Тест з вибором відповіді".
5. Вводить текст питання, додає варіанти відповідей (А–Г), прикріплює зображення (за потреби).
6. Вказує правильну відповідь.
7. Повторює ці дії для інших типів завдань.
8. Зберігає всі завдання у файл.

Результат: сформовано повноцінний тестовий файл у форматі JSON, готовий до використання учнями.

Сценарій 2: Проходження тесту (роль — учень)

Мета: пройти тест із математики, сформований адміністратором.

Кроки:

1. Користувач відкриває головне вікно, обирає предмет "Математика".
2. Система пропонує список доступних тестів → учень обирає потрібний файл.
3. Тест розпочинається з таймером та завданням №1.
4. Учень відповідає на кожне завдання (натискає варіант, вводить текст або перетягує відповідності).
5. Після проходження всіх завдань натискає "Завершити тест".
6. Система виводить результат: кількість правильних відповідей, час проходження.
7. За бажанням — повернення на головний екран.

Результат: користувач отримує миттєвий зворотний зв'язок і бачить свій рівень підготовки.

Сценарій 3: Редагування існуючого тесту (роль — адміністратор)

Мета: внести зміни у вже створений тест, наприклад, змінити формулювання питання або оновити зображення.

Кроки:

1. Вхід у режим адміністратора через екран налаштувань.

2. Вибір існуючого тестового файлу зі списку.
3. Перехід у режим редагування тестів.
4. Вибір завдання зі списку.
5. Внесення змін у текст, варіанти відповідей або зображення.
6. Збереження оновленого тесту.

Результат: тест оновлено без потреби створювати файл спочатку.

Сценарій 4: Налаштування параметрів тесту

Мета: встановити, скільки завдань кожного типу буде включено до тесту під час запуску.

Кроки:

1. Вхід до панелі адміністратора.
2. Перехід до розділу "Налаштування кількості завдань".
3. Введення кількості:
   - Тести з вибором: 15
   - На відповідність: 3
   - З відкритою відповіддю: 4
4. Збереження параметрів.

Результат: під час кожного запуску тесту система автоматично формуватиме список завдань згідно з заданими пропорціями.

Сценарій 5: Тестування на кількох пристроях (автономний режим)

Мета: використати тест на комп’ютері, де відсутній Інтернет або система адміністрування.

Кроки:

1. Адміністратор копіює створений JSON-файл та папку images/ на флешку.
2. Файл переноситься на інший комп’ютер.
3. Запускається програма (локальна версія), тест відображається без втрат у вигляді чи функціональності.

Результат: незалежна, автономна робота системи без потреби в мережевій інфраструктурі.

Гнучкість сценаріїв

Передбачена можливість швидкого переходу між режимами. Наприклад:

- Користувач може пройти тест, повернутись і переглянути питання.
- Адміністратор може одразу створити нове завдання після завершення тестування.
- Усі зміни в налаштуваннях або завданнях діють одразу, без потреби перезапуску програми.

Таким чином, система проєктувалась з урахуванням реальних сценаріїв користування в освітньому середовищі. Завдяки чітко продуманій логіці ролей та дій, її можуть ефективно використовувати як окремі учні для самостійної підготовки, так і викладачі в рамках навчального процесу.

3 РЕАЛІЗАЦІЯ ПРОГРАМНОЇ СИСТЕМИ

3.1 Вибір платформи для реалізації системи

Правильний вибір технологій для розробки програмного забезпечення є основоположним етапом, що впливає не лише на ефективність реалізації проєкту, а й на зручність його подальшого обслуговування, масштабування та використання кінцевими користувачами. Під час створення системи підготовки до НМТ було визначено кілька ключових вимог до платформи:

- підтримка графічного інтерфейсу (GUI);
- можливість працювати автономно (офлайн);
- простота встановлення та запуску без сторонніх компонентів;
- низький поріг входження для потенційного супроводу в майбутньому;
- гнучкість у роботі з файлами, зображеннями, текстовими даними.

Вибір мови програмування: Python

Мовою реалізації обрано Python — сучасну, високорівневу мову, яка має численні переваги:

- Простота синтаксису**:** дозволяє реалізувати складну логіку з мінімальними витратами часу;
- Велика екосистема**:** наявність готових бібліотек для графіки, роботи з файлами, обробки зображень тощо;
- Кросплатформеність**:** додаток працює на Windows, Linux, macOS без потреби в серйозних змінах коду;
- Широка спільнота**:** у разі виникнення помилок або потреби в розширенні — легко знайти приклади, документацію або підтримку.

Python — це також один із найпопулярніших інструментів в освіті, тому вибір цієї мови підвищує потенціал подальшого використання проєкту в навчальних цілях або його адаптації студентами.

Вибір бібліотек для створення GUI. У Python існує кілька способів створення графічного інтерфейсу користувача (табл. 3.1):

Бібліотека	Переваги	Недоліки
Tkinter	Вбудована в Python, проста, стабільна	Обмежений візуальний стиль
PyQt / PySide	Потужний інтерфейс, сучасний дизайн	Важкий для початківців, громіздкий
Kivy	Мобільна орієнтація, сенсорний інтерфейс	Незручна для класичних десктоп-додатків
CustomTkinter	Розширення Tkinter з підтримкою тем, адаптивності, сучасного дизайну	Потребує установки, але легка

Таблиця 3.1 – варіанти створення графічного інтерфейсу

Було обрано комбінацію Tkinter + CustomTkinter, оскільки:

- Tkinter вже присутній у базовій установці Python;
- CustomTkinter забезпечує сучасний вигляд інтерфейсу без складної розмітки;
- система отримує підтримку тем (світла/темна), закруглених елементів, адаптивних форм;
- можливість легкого стилізування кнопок, інпутів, меню — без необхідності писати CSS чи HTML-код.

Це дозволяє досягти максимального балансу між функціональністю та зручністю для розробника та користувача.

Робота з графікою

Для обробки зображень використано бібліотеку Pillow (форк PIL):

- підтримка більшості форматів (PNG, JPEG, GIF);
- можливість масштабування, обрізки, перегляду;
- інтеграція з елементами Tkinter для вставки зображень у GUI;
- обробка помилок завантаження та адаптація розміру під вікно.

Це дозволяє інтегрувати зображення до кожного запитання або варіанта відповіді без ризику розтягування інтерфейсу чи спотворення.

Обробка даних: формат JSON

Для зберігання завдань і конфігурацій використано формат JSON, оскільки:

- він легкий, простий і легко зчитується/записується за допомогою стандартного модуля json;
- підтримує вкладені структури (словники, списки), що відповідає логіці завдань;
- редагується вручну, у будь-якому текстовому редакторі;
- не потребує СУБД або сторонніх бібліотек.

Таким чином, адміністратор може легко створити, змінити або скопіювати тестовий файл без спеціального програмного забезпечення.

Файлова система

Файлова структура програми спроєктована просто (рис. 3.2):

Рисунок 3.2 – файлова система програми

Це забезпечує прозорість і просту навігацію в системі навіть для початківця.

Комбінація Python + Tkinter + CustomTkinter + Pillow + JSON забезпечує:

- просту у реалізації і водночас потужну платформу;

- можливість автономної роботи без інтернету;
- швидку реакцію програми та малу кількість залежностей;
- гнучкість у зміні інтерфейсу, структури даних, розширенні функціоналу.

Це повністю відповідає потребам сучасної освітньої системи — як у школі, так і при самостійній підготовці до іспитів.

3.2 Розробка модуля адміністрування тестів

Модуль адміністрування тестів є одним із ключових елементів програмної системи, який забезпечує повний контроль над створенням, редагуванням та налаштуванням тестових завдань. Його основне призначення — надати викладачу простий у користуванні, але функціонально гнучкий інтерфейс для роботи з тестовими файлами без потреби у зміні коду чи структури файлів вручну.

Уся логіка модуля реалізована у вигляді класу AdminScreen, який наслідує CTkFrame (елемент з бібліотеки CustomTkinter). Цей клас ініціалізує повноцінне адміністративне середовище після успішної авторизації користувача з правами адміністратора.

Основна структура класу AdminScreen

При створенні екземпляра класу відбувається:

- генерація основного вікна з усіма необхідними блоками;
- завантаження доступних JSON-файлів з тестами;
- побудова інтерфейсу для додавання завдань, створення нових файлів, налаштувань кількості питань тощо.

Нижче наведено фрагмент коду, який відповідає за формування інтерфейсу головного адміністративного вікна:

```
class AdminScreen(ctk.CTkFrame):
    def __init__(self, parent):
        super().__init__(parent)
        self.root = parent
```

```
        self.root.title("Панель адміністратора")
        self.root.geometry("1000x800")
        self.root.configure(bg="#2b2b2b")
        self.configure(fg_color="#2b2b2b")
        self.pack(fill="both", expand=True)

        main_frame = ctk.CTkFrame(self, fg_color="#2e2e2e")
        main_frame.pack(pady=40, padx=40, fill="both",
expand=True)

        ctk.CTkLabel(main_frame, text="Адміністративна панель",
font=("Arial", 24, "bold"), text_color="white").pack(
            pady=20)
```

У цьому фрагменті відбувається створення контейнера main_frame, в якому розміщуються всі елементи керування. Інтерфейс виконано в темному стилі, що відповідає сучасному UI-дизайну.

Основні функції модуля адміністрування

Клас AdminScreen включає реалізацію наступних функцій:

- створення нового тестового файлу (через модальне вікно з полем для введення назви);
- вибір типу нового завдання — тест з вибором відповіді, на відповідність, або з відкритим введенням;
- відкриття відповідного вікна створення завдання;
- вбудовані налаштування кількості питань, які зберігаються у JSON-файл конфігурації;
- відкриття редактора завдань для редагування вже існуючих тестів.

Наприклад, при натисканні кнопки "Додати завдання", викликається метод on_add_question(), який відкриває нове вікно відповідно до вибраного типу завдання:

```
def on_add_question(self):
    file_name = self.file_var.get() if self.files else None
    q_type = self.type_var.get() if self.files else None
    if not file_name:
        messagebox.showwarning("Немає файлу", "Спочатку створіть
або оберіть файл тесту.")
        return

    if q_type == "Тест із вибором однієї відповіді":
```

```
        self.open_single_question_window()
    elif q_type == "Тест на співвідношення":
        self.open_match_question_window()
    elif q_type == "Питання з введенням відповіді":
        self.open_input_question_window()
```

Ця структура дозволяє гнучко масштабувати систему, додаючи нові типи завдань без зміни загальної архітектури.

Робота з файлами та збереження даних

Окрему увагу приділено взаємодії з JSON-файлами. Усі дані про завдання зберігаються у форматі, зручному для читання і редагування. При створенні нового файлу система перевіряє, чи існує вже такий файл, і якщо ні — створює новий пустий JSON.

Функція створення нового тесту:

```
def create_new_test_file(self):

    dialog = CTkInputDialog(self.root, "Новий тест", "Введіть
ім'я файлу (наприклад, tasks1.json):")
    self.root.wait_window(dialog)
    new_file = dialog.result

    if new_file:
        if not new_file.endswith(".json"):
            new_file += ".json"
        add_test_file(new_file)
        self.files.append(new_file)
        self.file_var.set(new_file)
```

Таким чином, викладач має змогу створити необмежену кількість тестів, не виходячи з вікна адміністратора, і без взаємодії з файловою системою вручну.

Клас AdminScreen реалізує повноцінну функціональність для керування вмістом тестів у системі: від створення нових питань до конфігурації структури тесту. Він є прикладом ефективної реалізації GUI на Python, побудованої з урахуванням потреб кінцевого користувача — викладача або адміністратора.

3.3 Реалізація модуля тестування

Модуль тестування є центральним елементом взаємодії між системою та кінцевим користувачем — учнем. Саме в ньому реалізовано логіку проходження тесту: починаючи з вибору потрібного предмета, через поступове проходження усіх питань, і аж до завершення сесії та перегляду результатів. Його ключовими особливостями є динамічна побудова інтерфейсу, адаптивність під різні типи завдань, наявність таймера, інтуїтивна навігація та візуальний стиль, максимально наближений до офіційного дизайну тестових платформ НМТ.

Після запуску тесту користувач спочатку обирає предмет, після чого система відкриває нове вікно вибору тестового файлу. Ці файли формуються у вигляді JSON-документів, які містять список усіх завдань певної теми або рівня складності. Система автоматично зчитує вміст відповідної директорії й формує перелік доступних для проходження тестів.

Після підтвердження вибору файл завантажується в оперативну пам’ять, завдання проходять внутрішню фільтрацію за типами, а потім із кожної категорії (вибір відповіді, відповідність, введення тексту) випадковим чином обирається певна кількість запитань. Саме цей обсяг регулюється через конфігураційний файл question_count.json, який адміністратор може змінювати в інтерфейсі налаштувань. Кінцевий список питань формується динамічно і проходить додаткове перемішування. Це важлива перевага: кожен тест запускається з унікальним порядком запитань, що робить повторне проходження менш передбачуваним і стимулює справжнє засвоєння матеріалу, а не механічне запам’ятовування позицій.

Після початку тесту на екрані з’являється основне робоче вікно. Його структура складається з центральної панелі, де відображається завдання, правої бокової панелі для навігації між питаннями, а також верхньої панелі з таймером і кнопкою завершення тесту. Кожне питання відображається у

новому прокручуваному контейнері (реалізовано через CTkScrollableFrame), що забезпечує правильне масштабування незалежно від розміру екрана.

Система підтримує три типи завдань, кожен із яких має свою структуру і візуальне представлення. Завдання з вибором правильної відповіді відображаються у вигляді текстового блоку та радіокнопок, підписаних літерами (А–Г). До кожного варіанта, а також до самого питання, може бути прикріплене зображення — наприклад, графік функції або геометрична фігура. У випадку завдань на встановлення відповідності реалізовано умовний "drag & drop": зліва розміщені числові позначення, справа — набори відповідей, які користувач може вибрати та "перетягнути" до відповідного елемента. Завдання з відкритою відповіддю мають просту структуру — текст запитання та текстове поле, куди вводиться відповідь.

Інтерфейс кожного запитання створюється динамічно в момент його відкриття. Це означає, що навіть при великій кількості запитань система не завантажує їх усі одразу, а обробляє лише поточне, економлячи ресурси й забезпечуючи плавну роботу. У разі помилки, наприклад, якщо зображення не знайдено або JSON має помилкову структуру, система виводить відповідне повідомлення без аварійного завершення програми.

Однією з важливих деталей, що наближує тест до реального НМТ, є реалізований таймер. Він розміщений у верхній частині вікна, починає рахунок одразу після старту тесту й оновлюється щосекунди. Завдяки кольоровому акценту (червоний фон та білий шрифт), таймер завжди помітний. Це дозволяє користувачу не лише контролювати свій час, а й адаптуватися до психологічного навантаження, яке присутнє на реальному іспиті.

Навігація між завданнями реалізована через набір кнопок з номерами питань, розміщених у правій панелі. Кожна кнопка має колір, що залежить від статусу запитання: сірий — ще не відповіли; зелений — правильна відповідь; червоний — помилка; темно-червоний — поточне активне питання. Це дозволяє користувачу швидко орієнтуватися, переглядати, на які

запитання він вже відповів, а також повертатися до тих, які викликали складнощі.

Після натискання кнопки «Відповісти» система зберігає введену відповідь, оновлює інтерфейс навігації та автоматично переходить до наступного невідповіденого запитання. Якщо ж усі питання вже пройдені, користувачеві виводиться підсумкова панель результатів. У ній показується кількість правильних відповідей із загальної кількості, загальний час проходження, а також пропонується кнопка повернення на головний екран. Важливо, що після завершення тесту усі відповіді блокуються, що виключає можливість їх редагування «заднім числом».

Весь цей функціонал підтримується гнучкою логікою класу TestApp, реалізованого на Python із використанням бібліотек Tkinter та CustomTkinter. Завдяки використанню об’єктно-орієнтованої моделі, кожен тип завдання, кожен елемент інтерфейсу та кожна дія користувача чітко структурована в окремих методах, що значно спрощує обслуговування та масштабування проєкту.

Таким чином, модуль тестування забезпечує не лише функціональність перевірки знань, а й створює повноцінне середовище тренування перед реальним іспитом. Його дизайн, структура та логіка максимально наближені до вимог справжнього НМТ, що робить систему корисним інструментом як для учнів, так і для викладачів.

3.4 Інтеграція графічних матеріалів у тестові завдання

В умовах сучасної освіти візуальні матеріали відіграють важливу роль у формуванні глибокого розуміння навчального контенту. Особливо це актуально в таких дисциплінах, як математика, фізика, біологія, де часто використовуються графіки, схеми, формули, малюнки. У зв’язку з цим система підготовки до НМТ реалізує повноцінну підтримку графічних

матеріалів, які можуть бути додані до будь-якого завдання — як до самого питання, так і до окремих варіантів відповіді.

Мета інтеграції зображень

- Підвищення наочності: завдання з графіками, діаграмами чи схемами легше сприймаються візуально.
- Відповідність офіційним тестам НМТ: реальні завдання часто містять ілюстрації.
- Розширення формулювань: деякі питання можна подати лише у графічному вигляді (наприклад, "знайдіть площу фігури на рисунку").
- Підтримка мультимедійного формату навчання: учні краще запам’ятовують інформацію за допомогою зорових асоціацій.

Технічна реалізація

Структура в JSON. У структурі кожного завдання передбачено поле:

“image_path”: “images/назва_файлу.jpg”

- шлях відносний до кореневої директорії проєкту;
- допускаються формати: .png, .jpg, .jpeg, .gif;
- зображення зберігаються у папці /images/.

Додавання зображення. При створенні або редагуванні завдання адміністратор може:

1. натиснути кнопку "Додати зображення";
2. обрати файл зі свого комп’ютера через файловий діалог;
3. система:
   - автоматично копіює зображення до папки images;
   - зберігає тільки назву у JSON;
   - перевіряє, чи не дублюється файл (захист від перезапису).

Відображення в інтерфейсі. При показі завдання:

- система перевіряє, чи існує зображення;
- зображення відкривається через бібліотеку Pillow;

- автоматично масштабується до ширини 700–800 пікселів (залежно від типу питання);
- інтегрується в центр екрану між текстом питання та варіантами відповіді.

Якщо зображення не знайдено, система виводить повідомлення в консоль, але продовжує роботу (це дозволяє швидко виправити помилку без критичних збоїв).

Підтримка зображень у варіантах відповіді

Для завдань з вибором однієї правильної відповіді також передбачено окреме поле image_path для кожного варіанта:

```
"options": {
    "A": {
        "text": "(-∞; -3)",
        "image_path": ""
    },
```

Кожен варіант може мати:

- тільки текст;
- тільки зображення;
- або обидва одночасно.

У GUI:

- відображаються всі варіанти радіокнопками;
- праворуч до кожної додається картинка у форматі мініатюри (100×100 пікселів);
- користувач бачить одразу і текст, і ілюстрацію, що знижує когнітивне навантаження.

Масштабування та адаптація

Система автоматично:

- змінює розмір великих зображень (зменшує до потрібної ширини);
- зберігає пропорції;
- створює ескіз для попереднього перегляду;
- уникає «виходу» зображення за межі екрану.

Це особливо важливо для користувачів з невеликими екранами або низькою роздільною здатністю.

Практичні переваги для вчителя

Інтеграція графіки істотно полегшує роботу вчителя:

- Можливість вставляти рівняння, графіки, хімічні формули, математичні системи без необхідності програмування чи використання LaTeX;
- Завдання можуть містити зображення функцій, діаграм, таблиць, схем, які легко експортувати з будь-якого редактора (Word, GeoGebra, Paint);
- Варіанти відповідей теж можуть бути у вигляді зображень — наприклад, графіки або фрагменти рисунків;
- Створення завдання з формулою зводиться до кількох кліків: підготувати — зробити скріншот — додати — зберегти;
- Формати зображень відкриті й зрозумілі: PNG, JPEG, GIF — без специфічних вимог до конвертації;
- Один файл зображення можна багаторазово використовувати в різних завданнях і тестах.

Інтеграція графічних матеріалів в систему підготовки до НМТ значно розширює її функціональні можливості, робить завдання більш наочними та ефективними для сприйняття. Такий підхід не тільки підвищує якість підготовки, але й створює комфортне середовище, наближене до умов реального тестування.

3.5 Налаштування кількості питань у тесті

Однією з ключових вимог до гнучкої системи тестування є можливість індивідуального налаштування структури тесту залежно від конкретної мети перевірки знань: тренування, контрольна, підсумковий тест тощо. Саме тому в розробленій інформаційній системі реалізовано модуль, який дозволяє

адміністратору легко визначити кількість завдань кожного типу для майбутнього тесту.

Призначення функції

Модуль налаштувань кількості питань дає змогу:

- змінювати довжину тесту без редагування JSON-файлів;
- адаптувати тести до різного часу проходження (наприклад, 10, 20 чи 30 хвилин);
- формувати варіативні сесії підготовки, які кожного разу можуть містити різну кількість запитань певного типу;
- забезпечити рівновагу між типами завдань, залежно від складності та цілей тестування.

Це особливо зручно в контексті НМТ, де структура і розподіл типів запитань постійно оновлюється Міністерством освіти.

Інтерфейс налаштувань

Форма налаштувань відкривається в окремому модальному вікні, яке містить:

- поля для введення кількості:
  - завдань з вибором однієї правильної відповіді;
  - завдань на відповідність;
  - завдань з введенням відкритої відповіді;
- кнопку «Зберегти», яка підтверджує зміни;
- повідомлення про успішне збереження або помилку введення (якщо дані некоректні або відсутні).

Всі поля реалізовано за допомогою CTkEntry з автоматичною перевіркою типу введених даних.

Зберігання даних

Після натискання кнопки «Зберегти», значення зберігаються у файл question_count.json, структура якого має вигляд:

```
{
    "single_question": 15,
```

```
    "match": 3,
    "input": 4
}
```

Цей файл зчитується кожного разу при запуску тесту. Система формує тест на основі випадково вибраних завдань з відповідних категорій — відповідно до вказаної кількості.

Якщо у тестовому файлі немає достатньої кількості запитань певного типу — система автоматично обирає максимально доступну кількість, не викликаючи помилку.

Переваги підходу

Простота використання — адміністратор не має потреби відкривати код або редагувати JSON вручну; Гнучкість — можна формувати як короткі тести (наприклад, на 10 питань), так і повноцінні моделювання НМТ (30+ запитань); Індивідуалізація — у різних класах чи групах можна створювати тести з різною структурою; Прозорість — файл з налаштуваннями легко копіюється, змінюється, переноситься.

Модуль налаштування кількості питань є важливим інструментом, який забезпечує гнучкість та адаптивність системи до різних форматів тестування. Він дозволяє налаштовувати зміст тесту без втручання в код або структуру файлів, що значно полегшує роботу викладачів та підвищує ефективність використання системи в освітньому процесі.

4 ФУНКЦІОНАЛЬНІ МОЖЛИВОСТІ ТА ТЕСТУВАННЯ СИСТЕМ

4.1 Використання системи для створення тестових завдань

Розроблена система підготовки до Національного мультипредметного тесту (НМТ) є повноцінним інструментом як для викладача, так і для учня. Вона поєднує два основні компоненти — адміністративну частину (створення та редагування тестів) та учнівську частину (проходження тестування з урахуванням часу, дизайну та зворотного зв’язку).

Один із найважливіших функціональних елементів — це зручність створення тестових завдань. Викладачі або методисти можуть без труднощів формувати нові тести, оновлювати наявні, додавати зображення, завдання на відповідність або відкриті запитання. Усе це реалізовано без потреби в знаннях програмування: графічний інтерфейс інтуїтивно зрозумілий, а кожен крок супроводжується підказками та логічною структурою.

Після запуску застосунку відкривається головне вікно вибору дисципліни. Користувач має змогу обрати предмет (наприклад, «Математика») та перейти до подальшої взаємодії.

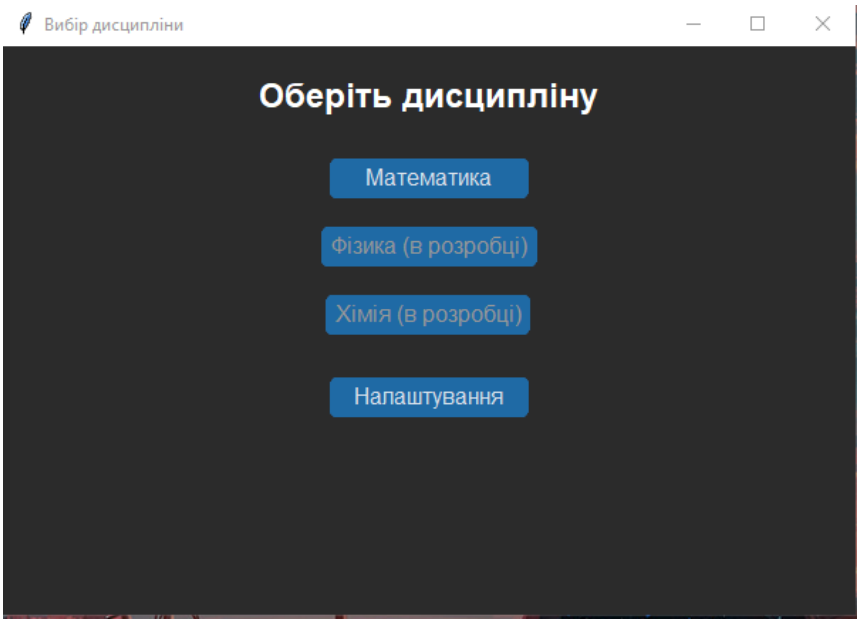

Рисунок 4.1 – головне вікно вибору предмета

Робота адміністратора

Після переходу до адміністративного інтерфейсу (через вікно авторизації), викладач отримує змогу створити новий тестовий файл або обрати існуючий із переліку. Усі тести зберігаються у форматі JSON, що дозволяє легко переносити їх між системами або редагувати вручну у разі потреби.

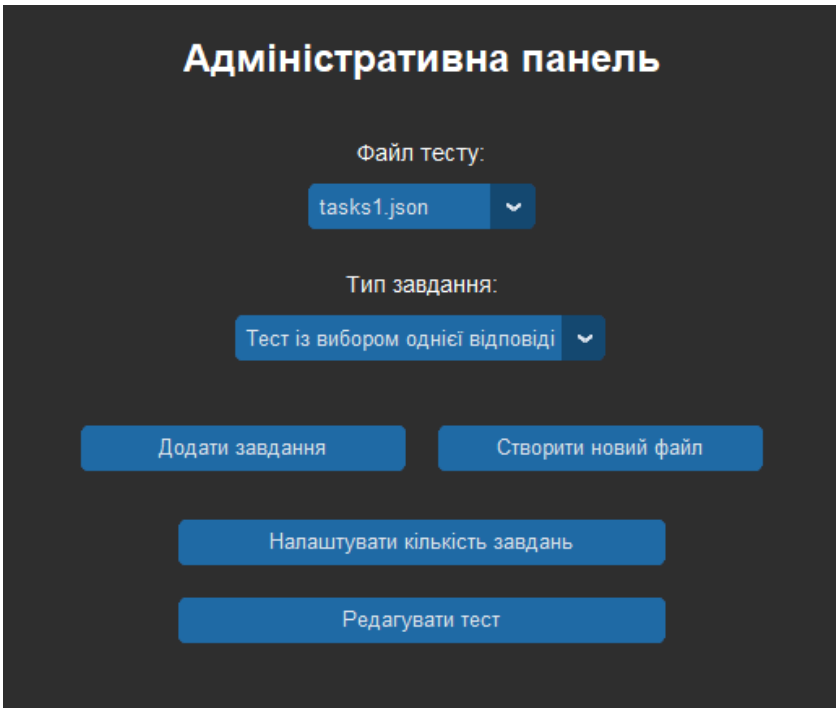

Рисунок 4.2 – панель адміністратора

Інтерфейс створення нового завдання розділений на тематичні блоки:

- поле для введення тексту запитання;
- кнопка для завантаження зображення (опційно);
- поля для варіантів відповіді або пар відповідностей;
- вибір правильного варіанту;
- кнопка збереження.

Цей підхід дозволяє педагогу швидко додати нове завдання — текстове, графічне або комбіноване — всього за кілька хвилин.

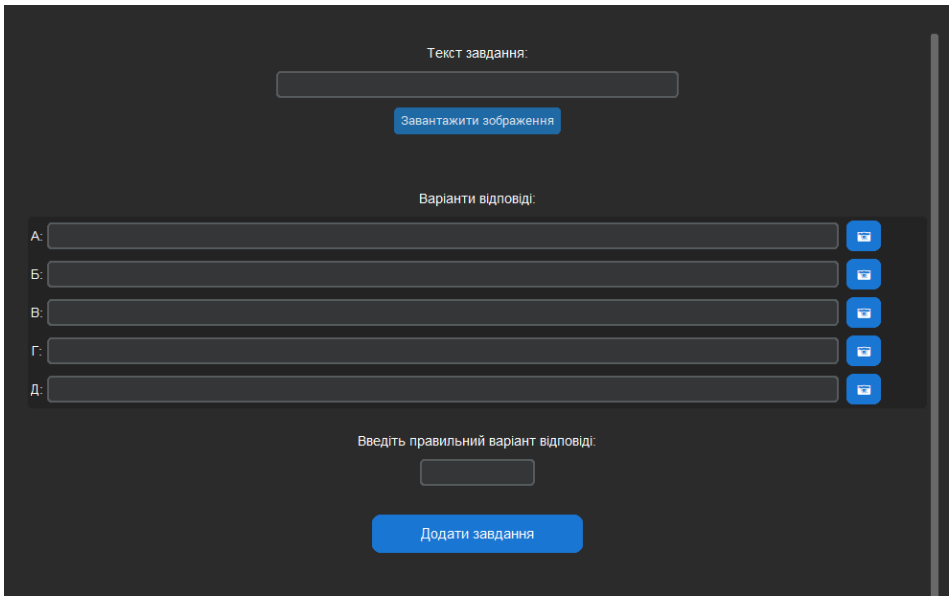

Рисунок 4.3 – вікно створення завдання з вибором відповіді

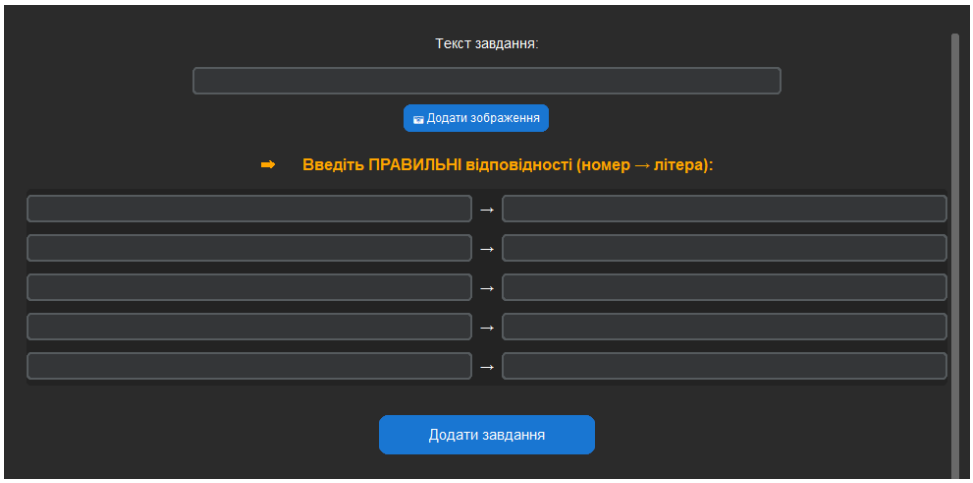

Рисунок 4.4 – створення завдання на відповідність

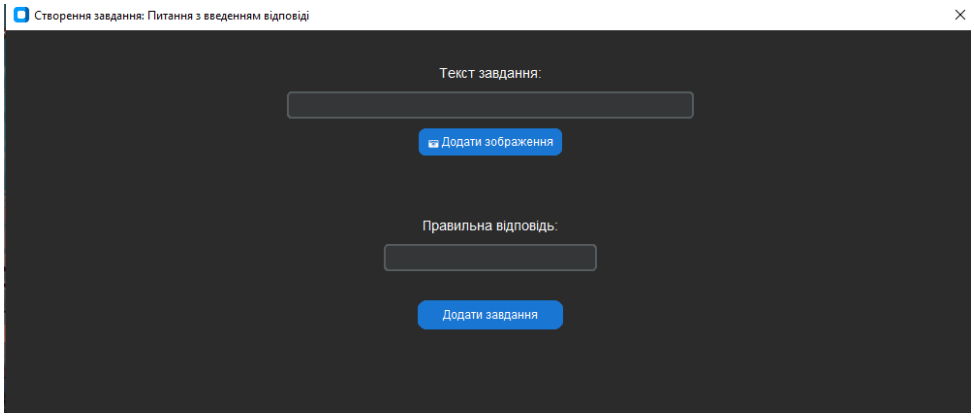

Рисунок 4.5 – створення завдання з відкритою відповіддю

Важливим аспектом є можливість додавання графічних матеріалів. Це відкриває нові можливості для створення питань із геометрії, фізики, хімії чи навіть мови. Зображення автоматично копіюються до відповідної папки, а їхній шлях додається у структуру JSON. Це повністю автоматизований процес: вчителю не потрібно нічого налаштовувати вручну.

Крім створення, система дозволяє редагувати існуючі завдання — змінити текст, оновити правильну відповідь, замінити картинку або видалити непотрібне запитання. Увесь життєвий цикл тесту — від створення до корекції — можна пройти в одному вікні, без втрати часу.

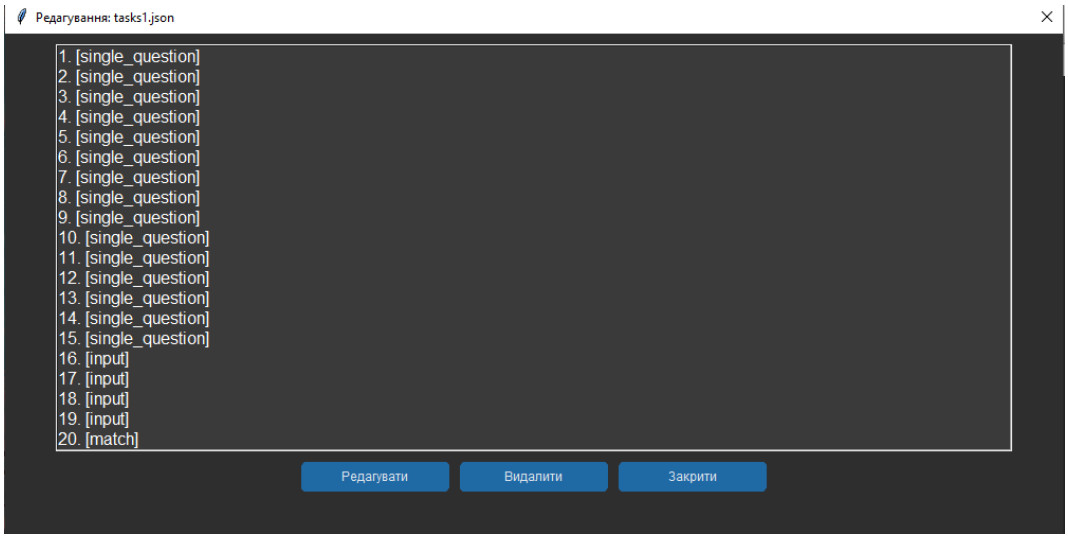

Рисунок 4.6 – редактор тестових файлів

Окремо передбачено розділ "Налаштування кількості завдань у тесті", де викладач може обрати скільки запитань кожного типу буде включено під час проходження. Наприклад, 10 завдань з вибором, 3 відповідності й 2 відкритих. Ці дані зберігаються в окремому JSON-файлі і враховуються під час формування структури тесту для учня.

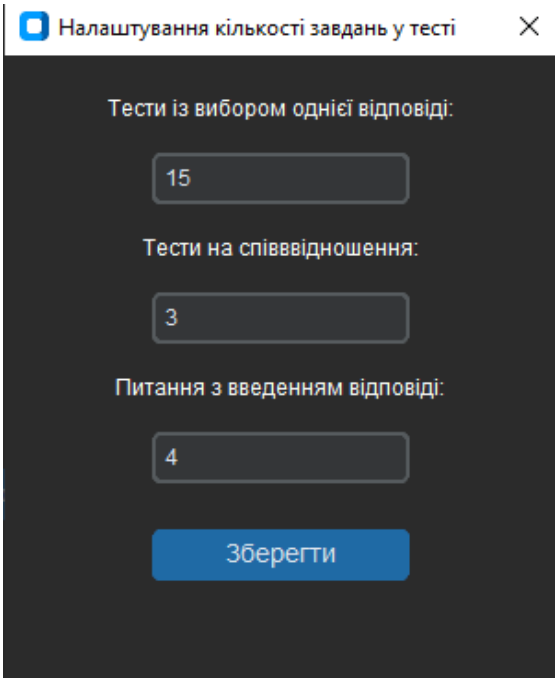

Рисунок 4.7 – вікно налаштування кількості завдань

Проходження тесту учнем

Після запуску тесту користувач бачить чистий, структурований інтерфейс. Центральна частина — це основне запитання. Справа — панель з кнопками переходу між питаннями, зверху — таймер.

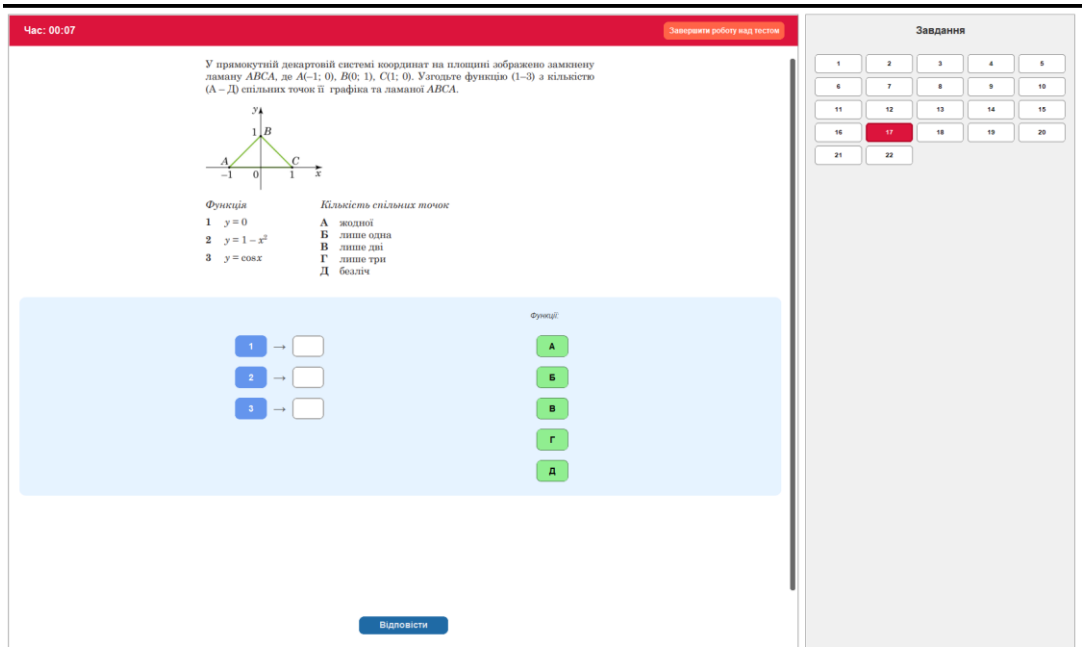

Рисунок 4.8 – інтерфейс проходження тесту

Таймер автоматично стартує при початку тесту і відображає час у хвилинах і секундах. Стиль оформлення, кольори, розташування елементів — усе максимально наближене до реального інтерфейсу офіційного НМТ.

Це дозволяє учням психологічно адаптуватися до майбутнього іспиту, уникнувши стресу через нове середовище.

Усі типи завдань відображаються з урахуванням особливостей:

- тести з варіантами відповіді мають радіокнопки;
- завдання на відповідність реалізують логіку відповідностей;
- відкриті відповіді мають текстові поля для введення.

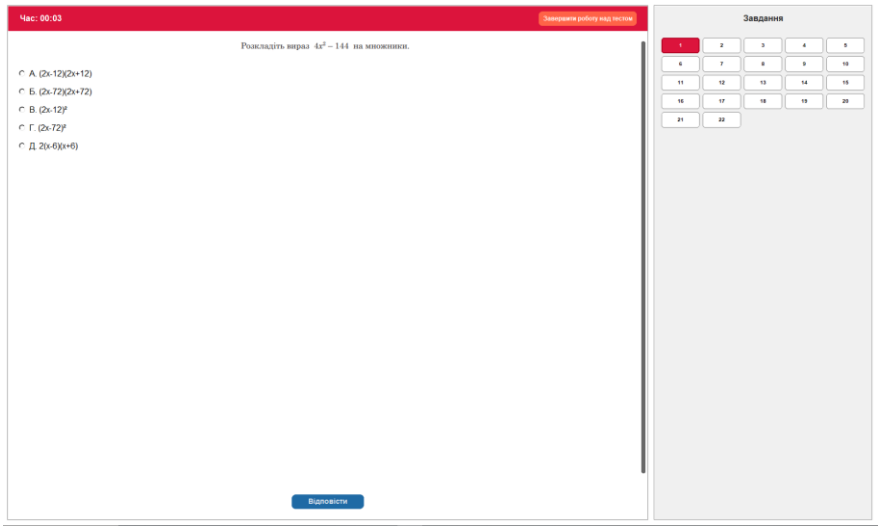

Рисунок 4.9 – тест з одним варіантом відповіді

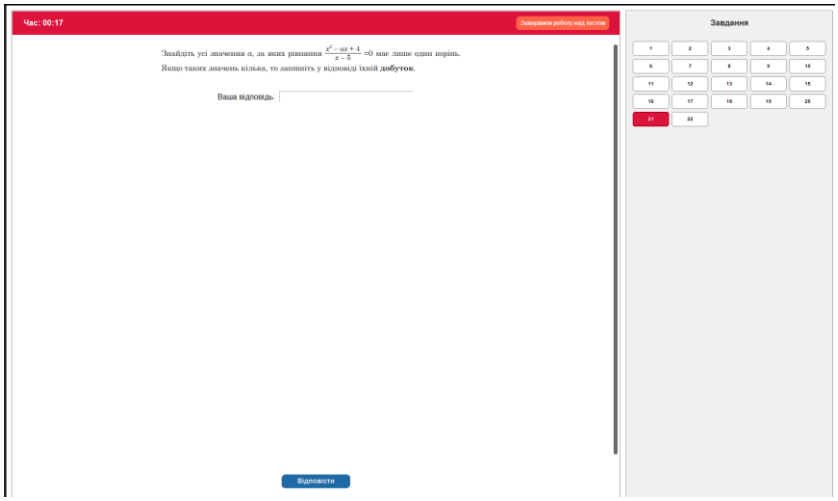

Рисунок 4.10 – питання з відкритою відповіддю

По завершенню — автоматичне підбиття підсумків: скільки правильних відповідей, скільки часу витрачено, які питання були помилковими. Усі дії користувача зберігаються протягом сесії, що дозволяє зробити аналіз після проходження.

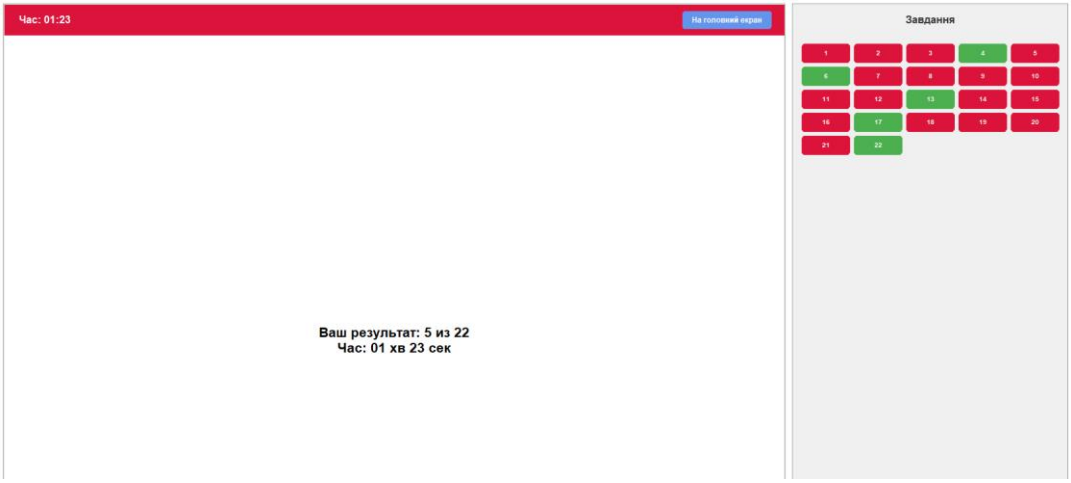

Рисунок 4.11 – фінальні результати тесту

Таким чином, програма не лише дозволяє створювати та редагувати повноцінні тести, а й надає учням можливість відчути реальну ситуацію НМТ. Гнучкий інтерфейс, підтримка графіки, динамічна генерація запитань і мінімалістичний дизайн роблять її практичним інструментом у сучасному навчальному процесі.

4.2 Тестування системи на прикладі тестових завдань

Щоб продемонструвати функціональність розробленої системи, проведемо покрокове тестування, яке охоплює весь життєвий цикл тестового завдання — від моменту його створення адміністратором до виконання учнем у режимі тесту.

Створення завдання з вибором відповіді

Адміністратор відкриває вікно створення завдання та обирає тип — «тест із вибором однієї правильної відповіді» (рис.4.12). У спеціальне поле вводиться текст завдання, наприклад:

«Обчисліть значення виразу: 5 – 3»

Після цього вводяться варіанти відповідей (А – Д), серед яких, наприклад:

- А: 5
- Б: 4
- В: 2 ← ☑ правильна
- Г: 3
- Д: 1

За потреби адміністратор додає зображення — наприклад, схематичну ілюстрацію чи формулу.

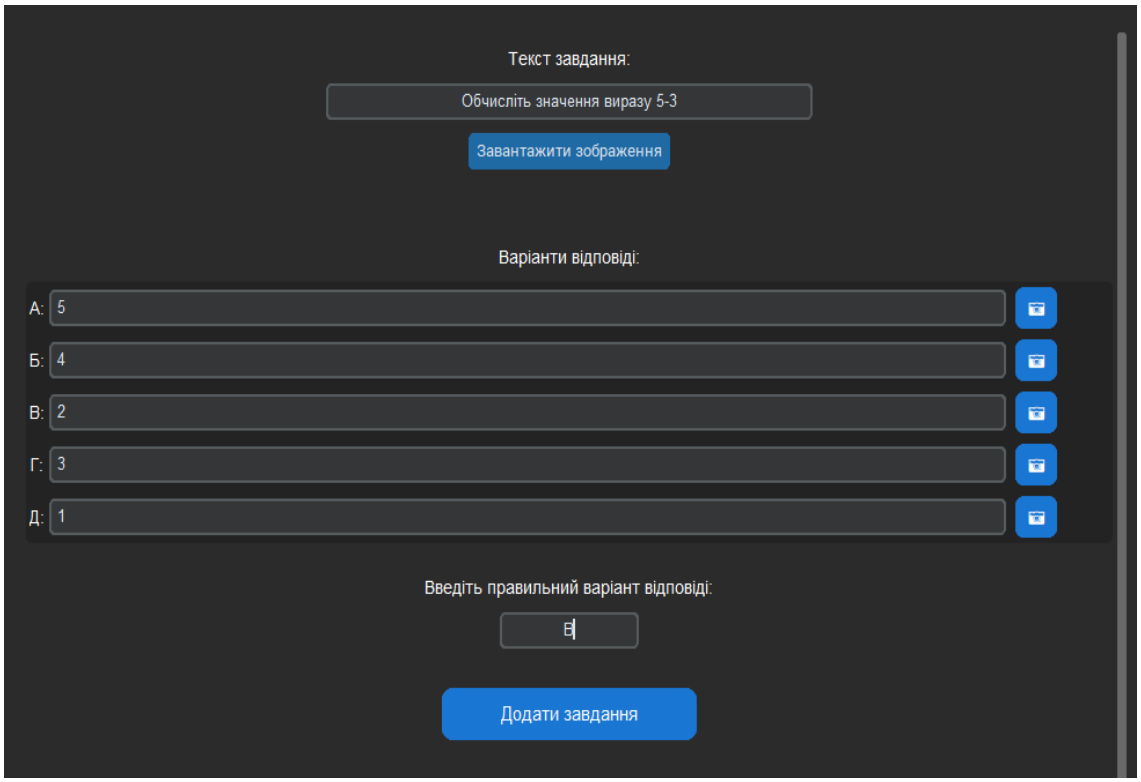

Рисунок 4.12 – вікно створення завдання з вибором відповіді

Після натискання кнопки «Додати завдання» воно зберігається у відповідному JSON-файлі. Далі система готова до використання цього завдання при проходженні тесту (рис. 4.13).

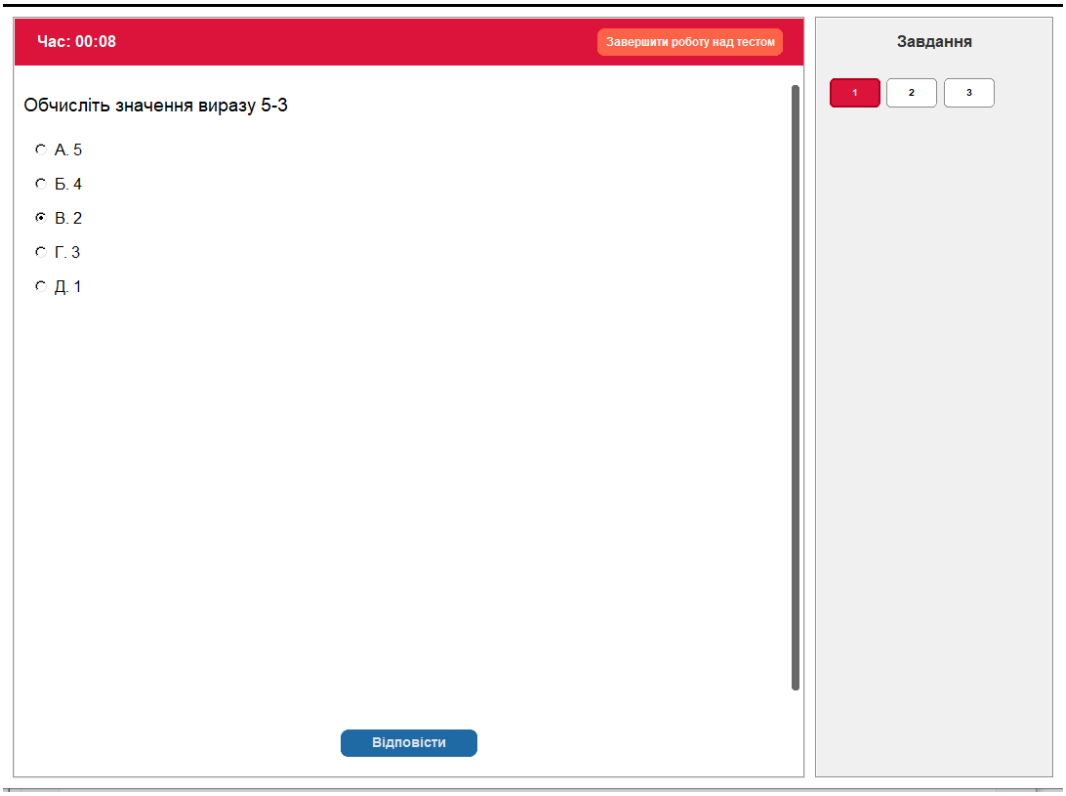

Рисунок 4.13 – створене завдання під час тестування

Створення завдання на відповідність

При створенні завдання типу «відповідність» адміністратор вводить (рис. 4.14):

- заголовок запитання або завантажує зображення (наприклад, формули, графіки або таблиці);
- у спеціальному блоці вводить правильні відповідності у форматі:

1 → А

2 → Б

3 → В

4 → Г

5 → Д

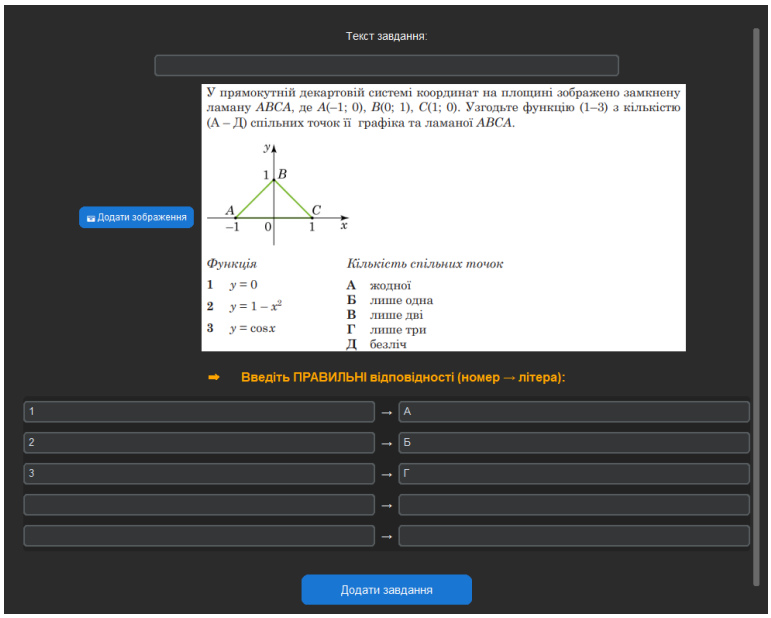

Рисунок 4.14 – вікно створення завдання на відповідність

Ці пари не відображаються учню безпосередньо (рис. 4.15). Під час проходження тесту:

- зліва автоматично виводяться номери (1–5);
- справа — літери варіантів (А–Д), які можна призначити кожному номеру;
- між ними — кнопки-«зони» для відповіді;
- користувач клікає по літері, а потім по зоні, щоб здійснити вибір.

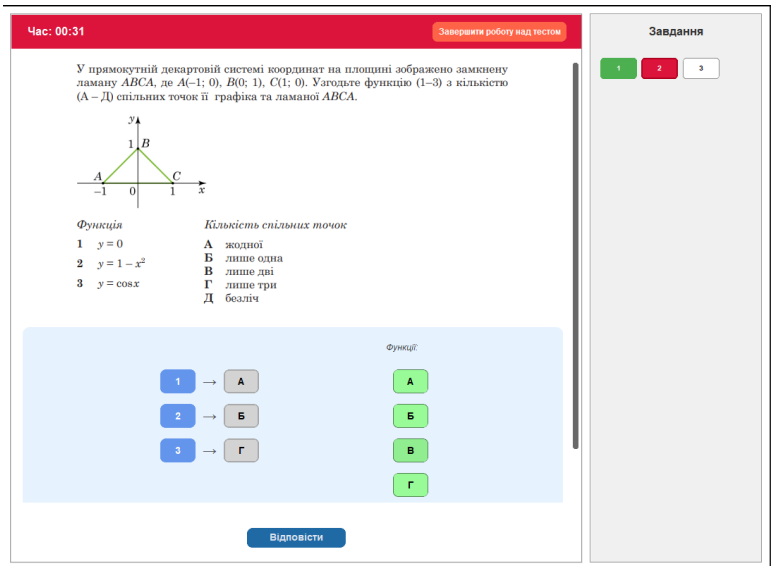

Рисунок 4.15 – створене завдання під час тестування

Важливо, що учень не вводить текст вручну. Усі дії здійснюються за допомогою інтерактивних елементів інтерфейсу, що мінімізує кількість помилок і полегшує процес.

Створення завдання з відкритою відповіддю

Останній тип — це завдання з ручним введенням відповіді (рис. 4.16, рис. 4.17), наприклад: *«Скільки градусів у сумі кутів трикутника?»*

Правильна відповідь: 180

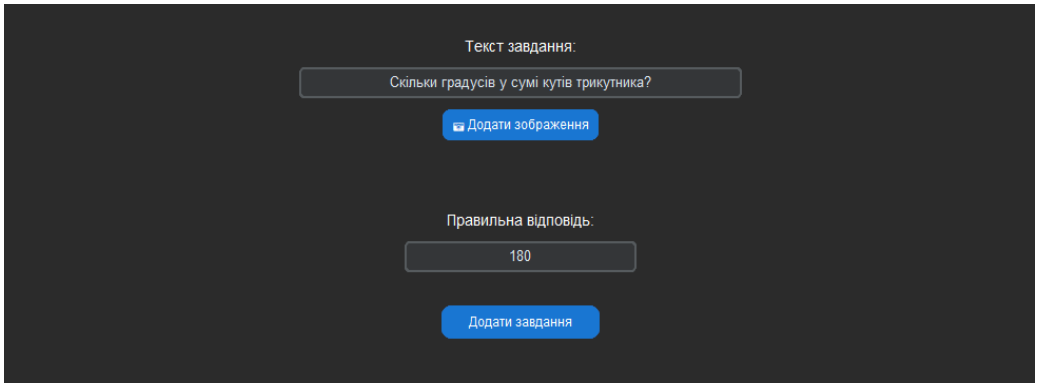

Рисунок 4.16 – вікно створення завдання відкритого типу

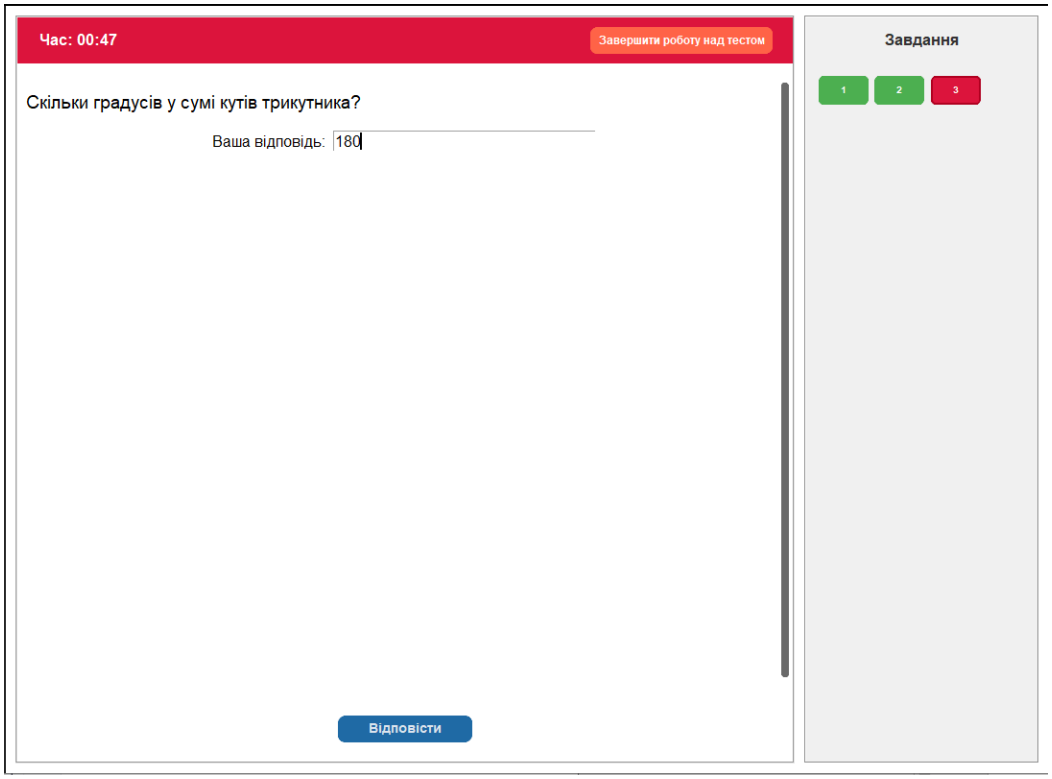

Рисунок 4.17 – створене завдання під час тестування

Система сприймає відповідь як текст, тож за потреби легко реалізується перевірка з нечутливістю до регістру, пробілів або альтернативних форм запису.

Аналіз результатів проходження

Після завершення тесту учень натискає кнопку «Завершити тест», і система формує зведення (рис. 4.18):

- загальна кількість запитань;
- кількість правильних відповідей;
- час виконання;
- кольорову навігацію по питаннях для візуального аналізу.
-

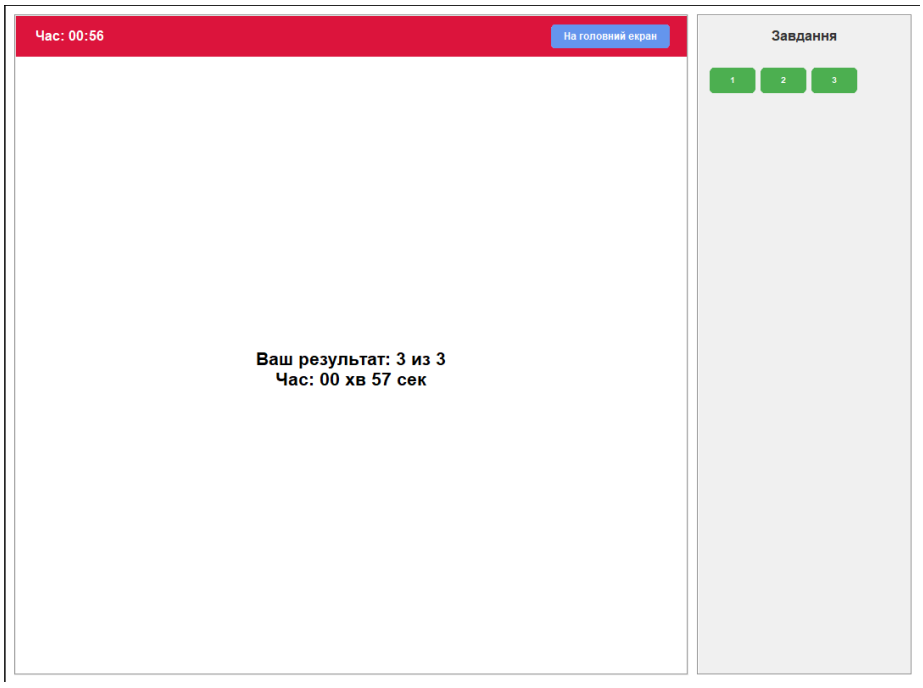

Рисунок 4.18 – вікно підсумкових результатів

Усі відповіді порівнюються з правильними, відображається статус кожного питання. Це дозволяє учню одразу побачити, де були зроблені помилки, а викладачу — оцінити прогрес і підготувати рекомендації.

Таким чином, тестування системи підтверджує її надійність, простоту використання та гнучкість. Завдання створюються швидко, без потреби в сторонньому ПЗ, а виконуються в комфортному інтерфейсі, наближеному до

офіційного середовища НМТ. Це робить програму повноцінним інструментом для щоденного використання в навчальному процесі.

4.3 Аналіз роботи системи та перспективи розвитку

Після завершення повного циклу реалізації інформаційної системи тестування було проведено детальний аналіз її роботи в умовах реального використання. Усі функціональні модулі показали стабільність, злагоджену взаємодію та відповідність поставленим цілям, що підтверджує ефективність обраної архітектури та технологічних рішень.

Комплексність реалізації

Система охоплює повний цикл роботи з тестовими матеріалами — від створення до проходження. Завдяки чіткій логіці, поділеній на адміністративний та користувацький модулі, забезпечено зручність не лише для викладачів, які формують тести, але й для учнів, які без зусиль можуть тренуватися у звичному цифровому середовищі. Це відповідає ключовим вимогам сучасної освіти, де автоматизація та адаптивність стають критично важливими.

Особливістю системи є її автономність: відсутність потреби у зовнішніх серверах чи підключенні до баз даних дозволяє запускати програму на будь-якому комп’ютері без додаткової інсталяції. Усі тести зберігаються у форматі JSON, що гарантує простоту структури, читабельність і легкість резервного копіювання.

Гнучкість адміністрування

Функціонал адміністратора побудований так, щоб забезпечити максимально просту та інтуїтивну взаємодію з системою навіть для тих користувачів, які не мають досвіду в ІТ. Створення нового завдання вимагає лише заповнення зрозумілих полів: текст питання, правильна відповідь, опційно — зображення. Система автоматично генерує структуру збереження,

записує завдання до тестового файлу та формує відповідну структуру для подальшого відображення.

Підтримка трьох типів завдань дозволяє формувати тести з різною складністю: від базового відтворення знань до логічного мислення та аналітики. Особливо важливо, що під час створення завдання на відповідність адміністратор може використовувати як текст, так і візуальні матеріали, що відкриває широкі можливості для роботи в предметах із складною термінологією або візуальною складовою.

Інтерактивність проходження тесту

Користувацький модуль реалізовано з особливою увагою до деталей. Інтерфейс максимально наближений до формату Національного мультипредметного тесту, що дозволяє не лише перевіряти знання, а й адаптувати учня до умов реального іспиту.

Важливою перевагою є динамічність інтерфейсу: для кожного типу завдання формується окремий GUI-блок, який автоматично адаптується до введених даних. Відображення зображень, масштабування, адаптація до довжини запитання — усе це реалізовано через сучасні компоненти бібліотеки CustomTkinter.

Система також оснащена таймером, панеллю навігації та функцією кольорового зворотного зв’язку (індикація правильних і неправильних відповідей), що сприяє кращому самоконтролю та підвищенню мотивації учня.

Можливості масштабування та розширення

Завдяки використанню відкритих форматів збереження та модульної архітектури система готова до подальшого розвитку. До прикладу, у майбутньому можна реалізувати:

- підтримку кабінетів учнів і викладачів з веденням персональної статистики;
- інтеграцію з хмарними сервісами для спільного доступу до тестів;
- автоматичний аналіз результатів з рекомендаціями;

- розширення типів завдань і шаблонів.

Наявна структура вже зараз дозволяє адаптувати проєкт під різні освітні потреби: шкільні тести, підготовку до ЗНО/НМТ, формуюче оцінювання або навіть внутрішню сертифікацію у коледжах і ВНЗ.

Таким чином, реалізована система демонструє вдале поєднання гнучкості, простоти використання та функціональної повноти. Її можна рекомендувати до практичного впровадження у заклади освіти, а також використовувати як платформу для створення розширених освітніх інструментів майбутнього.

ВИСНОВКИ

У межах даної дипломної роботи було розроблено повнофункціональну інформаційну систему, призначену для створення, редагування та проходження тестових завдань з орієнтацією на формат Національного мультипредметного тесту (НМТ). Враховуючи сучасні вимоги до цифрових інструментів у сфері освіти, система поєднує в собі зручність, простоту, наочність та високу адаптивність до потреб як учнів, так і викладачів.

У результаті виконано всі поставлені на початку роботи завдання. Проведено аналіз існуючих інструментів підготовки до тестування, визначено вимоги до інформаційної системи, розроблено її архітектуру та інтерфейс, реалізовано програмний продукт і проведено тестування. Система підтримує створення завдань трьох типів, дозволяє додавати зображення до запитань, зберігає дані у форматі JSON, що забезпечує гнучкість і автономність.

Особливої уваги заслуговує реалізація модуля адміністрування, який надає викладачам простий і наочний інструмент для формування тестів, без потреби у спеціальних знаннях. Інтерфейс модуля тестування наближено до реального іспитового середовища, що дозволяє учню адаптуватися до умов НМТ ще до проходження офіційного тесту.

Система є універсальною, легко масштабується та може використовуватись не лише для підготовки до НМТ, а й у рамках внутрішнього оцінювання знань у школах, коледжах та інших навчальних закладах. Результати тестування програми підтвердили її стабільну роботу, відповідність сучасним педагогічним вимогам та практичну придатність.

Отже, розроблена система може стати ефективним цифровим інструментом у сучасному освітньому середовищі, що сприятиме підвищенню якості підготовки учнів до стандартизованого тестування.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Зайцева І. О. Цифрові платформи як інструмент адаптивного навчання у старшій школі // Інформаційні технології і засоби навчання. — 2021. — Т. 83, №1. — С. 14–20.

2. Білик Н. С. Інтерактивні освітні технології у підготовці до ЗНО: ефективність та перспективи // Освітній процес: теорія і практика. — 2020. — №2. — С. 35–41.

3. Сидоренко В. П. Методика використання візуальних елементів у тестових завданнях // Педагогічні інновації: ідеї, реалії, перспективи. — 2019. — №4. — С. 22–28.

4. Державна служба якості освіти України. Рекомендації щодо підготовки до Національного мультипредметного тесту. — 2023. — https://testportal.gov.ua/

5. Ткачук В.І., Мартинюк О.А. Основи програмування мовою Python. — Вінниця: Нова Книга, 2020. — 248 с.

6. Tkinter documentation — Python Standard Library. — https://docs.python.org/3/library/tkinter.html

7. CustomTkinter — Modern GUI for Python. — https://github.com/TomSchimansky/CustomTkinter

8. JSON формат: Офіційна документація. — https://www.json.org/json-en.html

9. Радченко В.П. Програмна інженерія: навч. посіб. — Київ: КНЕУ, 2020. — 272 с.

10. Освіта.ua — онлайн-платформа для підготовки до ЗНО та НМТ. — [https://osvita.ua/test/

11. Просте.ЗНО — мобільний додаток для самопідготовки. — https://play.google.com/store/apps/details?id=com.simplezno

ДОДАТОК А

Структура JSON-файлу з тестами


```
{
    "type": "single_question",
    "question_text": "",
    "answer": "Б",
    "image_path": "image13.png",
    "options": {
        "А": {
            "text": "(-∞; -3)",
            "image_path": ""
        },
        "Б": {
            "text": "[-3; 0)",
            "image_path": ""
        },
        "В": {
            "text": "[0; 3)",
            "image_path": ""
        },
        "Г": {
            "text": "[3; 25)",
            "image_path": ""
        },
        "Д": {
            "text": "[25; +∞)",
            "image_path": ""
        }
    }
    {
    "type": "match",
    "question_text": "",
    "image_path": "image18.png",
    "answer": {
        "1": "В",
        "2": "Г",
        "3": "Б"
    },
    "options": {}
},
{
    "type": "input",
    "image_path": "images/image22.png",
    "answer": "-92,8"
},
```

ДОДАТОК Б

Реалізація основних класів

Клас: StartScreen (Початковий екран)

- __init__ — конструктор
- start_math_test — почати тест з математики
- open_settings — відкрити налаштування (вхід)

Клас: LoginScreen (Екран входу)

- __init__ — конструктор
- check_login — перевірка логіна і пароля

Клас: AdminScreen (Адміністративна панель)

- __init__ — конструктор
- open_editor — відкрити редактор
- on_add_question — обробка додавання завдання
- create_new_test_file — створити новий файл тесту
- open_question_window — відкрити вікно створення завдань *(повторюється двічі — можливо, помилка)*
- open_single_question_window — створення завдання з вибором відповіді
- select_image — вибір зображення
- select_option_image — вибір зображення для варіанту відповіді
- add_single_question — додати тест з вибором однієї відповіді
- open_match_question_window — створити завдання на співвідношення
- select_match_image — вибрати зображення для пари співвідношення
- add_match_question — додати завдання на співвідношення
- open_input_question_window — створити питання з введенням
- add_input_question — додати питання з введенням відповіді
- configure_question_counts — налаштувати кількість завдань
- save_question_counts — зберегти кількість завдань

Клас: TestApp (Тестування з таймером)

- __init__ — конструктор
- update_timer — оновлення таймера
- show_question — показати питання
- select_option — вибрати варіант (для співвідношення)
- drop_option_on_target — встановити відповідь на місце
- display_question_buttons — відобразити кнопки навігації по питаннях
- jump_to_question — перейти до питання
- check_answer — перевірити відповідь
- clear_screen — очистити екран
- show_results_confirmation — підтвердити завершення тесту
- show_results — показати результати
- quit_to_start_screen — повернутись на головний екран
- quit_test — завершити тест

ДОДАТОК В

Структура проєкту (дерево папок)

diplom/
├── build/ ← директорія для збірки
├── dist/ ← директорія з готовим .exe файлом
├── images/ ← зображення, що додаються до завдань
│
├── test_data/ ← усі тестові файли у форматі JSON
│ ├── tasks1.json
│ ├── tasks2.json
│ ├── tasks3.json
│ ├── tests_list.json ← перелік усіх доступних тестів
│
├── EditorWindow.py ← модуль редагування існуючих завдань
├── main.py ← точка входу у програму (запуск)
├── main.spec ← файл конфігурації для збірки через PyInstaller
├── question_count.json ← файл з налаштуваннями кількості завдань
├── test_manager.py ← логіка завантаження, збереження та обробки тестів