МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра комп'ютерних наук

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: **«РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ТЕСТУВАННЯ ТА КОНТРОЛЮ ЗНАНЬ УЧНІВ»**

Виконала: студентка 2 курсу, групи 8.1220
спеціальності 122 Комп'ютерні науки
(шифр і назва спеціальності)
М.В. Руденко
(ініціали та прізвище)

освітньої програми комп'ютерні науки

Керівник доцент кафедри комп'ютерних наук, доцент, к.т.н. Матвіїшина Н.В
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент доцент кафедри програмної інженерії, доцент, к.ф.-м.н., Кривохата А. Г.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя - 2021

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра комп'ютерних наук

Рівень вищої освіти магістр

Спеціальність 122 комп'ютерні науки
(шифр і назва)

Освітня програма комп'ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри
комп'ютерних наук,
к.т.н., доцент
_____ Борю С.Ю.
(підпис)
« 14 » 06 2021 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТЦІ

Руденко Марії Володимирівни
(прізвище, ім'я та по батькові)

1. Тема роботи «Розробка програмного забезпечення для тестування та контролю знань учнів»
керівник роботи Матвіїшина Надія Вікторівна
(прізвище, ім'я та по батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 09 » 06 2021 року № 850

2. Строк подання студентом роботи 19.11.2021

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Огляд інформаційних джерел та сучасний стан проблеми.
2. Аналіз предметної області.
3. Розробка програмного продукту.
4. Реалізація та апробація програмного продукту.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Презентація

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1			
2			
3			

7. Дата видачі завдання 14.06.2021

К А Л Е Н Д А Р Н И Й П Л А Н

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	28.09.2021	
2.	Збір вихідних даних.	30.09.2021	
3.	Обробка методичних та теоретичних	01.10.2021	
	джерел.		
4.	Розробка першого та другого розділу.	04.10.2021	
5.	Розробка третього та четвертого розділу.	17.10.2021	
6.	Оформлення та нормоконтроль	19.11.2021	
	кваліфікаційної роботи.		
7.	Захист кваліфікаційної роботи.	08.12.2021	

Студент _____ М.В. Руденко
(підпис) (ініціали та прізвище)

Керівник роботи _____ Н.В. Матвіїшина
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ О.Г. Спиця
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота магістра «Створення програмного забезпечення для тестування та контролю знань учнів»: 89 с., 32 Рисунок, 32 джерел, 1 додаток.

ІНФОРМАЦІЙНА СИСТЕМА, КОНТРОЛЬ ЗНАНЬ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, ПРОГРАМНИЙ МОДУЛЬ, ТЕСТУВАННЯ.

Об’єкт дослідження – організація процесу тестування школярів.

Мета роботи: розробка програмного модуля для навчання та тестування учнів.

Методи дослідження: синтез, формалізація, аналітичний, порівняльний, графічний методи, методи інтелектуального аналізу даних.

Магістерська робота присвячена вивченню питань розроблення програмного забезпечення для тестування та контролю знань учнів. Інформаційно-довідкова система складена на основі методичного плану навчального предмету і створена для покращення якості засвоєння курсу навчального предмету.

Створена можливість по перегляду повної інформації по навчальному предмету, перегляду методичних розробок та проходження тестів. При розробці даного програмного модулю використовувалася об’єктно-орієнтована мова програмування – C# та система управління базами даних MySQL.

Розроблений програмний модуль дозволяє ефективно готуватись до навчальних дисциплін та перевіряти отримані знання та навички у процесі навчання. Також розроблений програмний модуль є досить хорошим фундаментом для масштабніших проектних рішень.

SUMMARY

Master’s qualification thesis "Creation of software for testing and control of students’ knowledge": 89 pages, 32 figures, 32 references, 1 supplements.

INFORMATION SYSTEM, KNOWLEDGE CONTROL, SOFTWARE, SOFTWARE MODULE, TESTING.

The object of research is the organization of the process of testing students.

The aim of the study is development of a software module for teaching and testing students.

The methods of research are synthesis, formalization, analytical, comparative, graphic methods, methods of data mining.

The master’s thesis is devoted to the study of software development for testing and control of students’ knowledge. The master’s thesis analyzes the key principles of organization of information and reference and information retrieval systems, formulates requirements for information systems, analyzes the feasibility of automating the testing process in the learning process and considers the main ways of its organization.

There is an opportunity to view complete information on the subject, review methodological developments and take tests. The development of this software module used an object-oriented programming language – C # and MySQL database management system.

The developed software module allows you to effectively prepare for academic disciplines and test the acquired knowledge and skills in the learning process. Also, the developed software module is a good foundation for larger design solutions.

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

HTML	Hyper Text Markup Language
HTTP	Hyper Text Transfer Protocol
NET	Extensible Markup Language
URL	Uniform Resource Locator
БД	база даних
ЕОМ	електронно-обчислювальна система
ІДС	інформаційно-довідкова система
ІПМ	інформаційно-пошукова мова
ІС	інформаційна система
ПК	персональний комп’ютер
ПЗ	програмне забезпечення
СУБД	система управління базами даних

ЗМІСТ

ВСТУП

Актуальність дослідження. З кожним роком інформаційні технології все глибше проникають у сучасне суспільство. Передусім, до автоматизації процесів у багатьох галузях людської діяльності, а зокрема і діяльності навчальних закладів, спонукають величезні потоки інформації, операції з ними, наявність інноваційних технологій та стрімкий розвиток сучасного світу загалом. Автоматизація вже давно увійшла в наше життя. Вона супроводжує різні сфери професійної діяльності людини, дозволяючи прискорити, спростити і структурувати виконану роботу.

Інформація як головний фактор виробництва в формі сучасних технологій відкриває великі можливості якісного зростання на базі наступних інструментів і факторів:

- необмеженість комерційних майданчиків в Інтернеті, розвитку Інтернет-торгівлі, фінансових (фондових і валютних) бірж;
- мінімізація розміру організацій для успішної конкуренції на ринках, а також невпинного розвитку систем управління;
- активне використання інноваційних освітніх технологій і сучасних засобів в освітньому процесі;
- активізації наукових досліджень.

Постійний розвиток та невпинне вдосконалення сучасних інформаційно-комунікаційних технологій (ІКТ), їх активна реалізація значною мірою впливає на характер наукових досліджень, виробництва, освіту, соціальні взаємовідносини, культуру тощо. Це, передусім зумовлює як прямий вирішальний вплив на зміст освіти, пов’язаний з сталим рівнем науково-технічних досягнень, її удосконалення в умовах реального навчального процесу, так і опосередкований, що нерозривно пов’язаний із появою на ринку праці нових спеціальностей. Варто зазначити, що на сьогоднішній день, на тлі

глобальних змін в інформаційному суспільстві відбувається активний розвиток та інтенсивне використання ІКТ в усіх сферах життя, зокрема в освіті.

Сучасна система навчання за допомогою електронних та інформаційних технологій змінюють саму концептуальну модель освітньої активності, адже традиційне поняття навчального процесу досить важко поєднується із використанням ІКТ. Основними характерними ознаками, що відтворюють ці напрями виступають: мобільність, фокусування, доступність, універсальність, індивідуальна спрямованість викладацького складу та навчально-методичних матеріалів, результативне пристосування випускника до роботи у сучасних умовах із застосуванням новітніх інформаційних технологій.

Інформаційні технології надають можливість керувати академічними знаннями; формувати актуальні новітні системи управління якістю освіти; ефективно об’єднувати науку, освіту та інноваційні технології на основі створення освітніх кластерів. Інформаційні технології активно застосовуються для практичної реалізації навчальних програм, які полягають не тільки в інструментальних системах технологій ведення навчального процесу, але і в модернізації навчальних дисциплін та учбових планів. Саме інформаційні технології дають можливість створювати модернізовані навчально-методичні матеріали, та генерувати унікальні траєкторії навчання для школярів.

Актуальність теми обумовлена тим, що розвиток технологій автоматизації процесів, досягло такого рівня, який повинен був відбутися якісний перехід в новий стан. В даний час існує велика кількість різних підходів до вирішення даної проблеми, які базуються або на реляційному, або об’єктно-орієнтованому проектуванні.

Актуальність теми полягає в модернізації процесу проходження підготовчого курсу дисциплін для кращого засвоєння матеріалу, перспектив проходження курсу дисципліни, не маючи можливості бути в навчальному закладі з можливістю комплексного тестування з метою перевірки здобутих знань.

Сучасний стан інформаційних технологій демонструє високий зріст наочних розробок. Під наочними розробками маються на увазі створення інформаційно довідкових систем. Створення можливостей перегляду навчальної літератури, методичних розробок, перегляду матеріалів і проходження тестів по засвоєному матеріалу.

Метою кваліфікаційної роботи є розробка програмного модуля для навчання та тестування школярів.

Мета дослідження передбачає постановку і вирішення наступних **завдань**:

- комплексний аналіз інформаційно-пошукових та інформаційно-довідкових систем для оптимізації навчального процесу, їх функцій і специфіку структури;
- виявлення факторів, що впливають на побудову ефективних автоматизованих інформаційно-довідкових систем організацій;
- аналіз закономірностей і перспектив використання моделей зберігання інформації, формалізації представлення даних при побудові інформаційно-довідкових систем;
- побудова функціональної моделі майбутнього програмного модуля;
- побудувати алгоритм роботи бази даних, який пов’язуватиме всі елементи програми;
- окреслення функціональних вимог до розроблюваної програми;
- виконання проектування інформаційно-довідкової системи для комплексного ефективного тестування школярів;
- розробка програмного модуля для повноцінної роботи інформаційної системи для тестування школярів;
- оцінка роботи створеної програми.

Об’єктом дослідження виступає організація процесу тестування школярів.

Предметом дослідження є оптимізація та автоматизація роботи інформаційної системи для тестування школярів.

В даній роботі **спроектовано та розроблено** програмний модуль для тестування школярів.

Методи дослідження. У магістерській роботі використовувалися синтез, формалізація, аналітичний, порівняльний, графічний методи, а також методи інтелектуального аналізу даних.

Наукова новизна отриманих результатів. Досліджено можливості застосування інформаційних технологій для ефективної організації навчального процесу та належного тестування з метою контролю одержаних знань.

Практичне значення одержаних результатів. Розроблений програмний модуль дозволяє ефективно готуватись до навчальних дисциплін та перевіряти отримані знання та навички у процесі навчання. Також розроблений програмний модуль є досить хорошим фундаментом для масштабніших проектних рішень.

Структура роботи. Магістерська робота складається із 89 сторінок, 4 розділів, 4 схем, 32 рисунків, 32 літературних джерел та 1 додатку.

1 ОГЛЯД ІНФОРМАЦІЙНИХ ДЖЕРЕЛ ТА СУЧАСНИЙ СТАН ПРОБЛЕМИ

1.1 Роль інформаційних технологій для дистанційного навчального процесу

В умовах сьогодення електронне навчання (*e-learning*) допомагає вирішити ряд завдань, що нерозривно пов’язані із пристосуванням освітньої системи в цілому та навчальних закладів до змін, що відбулися в умовах карантину, такі як дистанційне навчання, індивідуальні навчальні плани тощо. У свою чергу автоматизована система освіти включає в себе уже наявні традиційні підходи до навчання і з застосуванням електронних технологій, але не обмежується ними.

Концепція автоматизації процесу отримання освіти – гнучкий процес, що передбачає наявність досить значного об’єму джерел, різноманітність мультимедіа, можливість легко та швидко налаштуватися під рівень знань та потреби здобувача освіти. Разом з тим змінюється роль викладача, який, у разі використання під час навчального процесу інформаційних-технологій, повинен змінювати підхід до системи контролю. Тенденції переходу інформаційних технологій окреслюють нові вимоги до вчителів: педагоги повинні мати великий багаж знань не тільки у своїй професійній сфері, а й мати широкий світогляд, вміти швидко налаштуватись до роботи у суміжних галузях та реалізовувати різноманітні технології для роботи із учнями. Такі інноваційні підходи вимагають від викладачів створювати сприятливі умови для придбання молоддю власних навичок та досвіду, а не тільки надавати готові знання [1].

Електронана освіта – це сучасне освітнє середовище; об’єднання навичок та зусиль фахівців, викладачів, вчителів, студентів та школярів для ефективного застосування всесвітніх знань та переходу до активного контенту. Провівши аналіз на основі досліджень в області автоматизації освіти, варто зазначити, що комплексне дослідження багатьох її елементів фактично не проводилося. На

сьогоднішній день уже досить популярним стає проведення навчальних занять з активним використанням мультимедійних презентацій, зроблених за допомогою усім відомих програмних пакетів *Microsoft Power Point* або *Macromedia Flash*. Однак, разом зі звичними презентаційними технологіями в освітню сферу проникають нові інформаційні технології, які дають змогу уникнути презентації у вигляді слайд-шоу. Абсолютно новим для сприйняття є те, що на інтерактивних дошках можна демонструвати навчальний матеріал, писати ключові моменти та робити письмові нотатки над зображеннями спеціальним маркером на екрані. При цьому все, що викладено та додатково зазначене на такій дошці транслюється учням, зберігається на магнітних носіях із можливістю роздруковувати матеріал або надіслати його електронною поштою [5].

У глобальній мережі Інтернет можна знайти досить багато різноманітних веб-ресурсів для автоматизованого навчання. Я вважаю доцільним зазначити найефективніші із них:

– *Khan Academy*: некомерційний освітній ресурс, що пропонує у відкритому доступі велику кількість матеріалів із різних навчальних дисциплін, тестові завдання із можливістю моментальної перевірки, а також вікторини для проведення уроків.

– *Quora*: соціальний сервіс, що широко використовується для цілого ряду цілей, і його можна сміливо віднести до одних із найкращих технологічних інструментів для педагогів. Бездоганне освітнє середовище, щоб підлітки могли навчитися шляхом обговорення освітніх тем, або, за необхідності, знайти відповіді на питання шляхом розміщення запитань у Quora. Також сервіс надає можливість пошуку питання, що потребує вирішення шляхом звернення до архіву, де необхідні відповіді об’єднані спільною тематикою.

– *Capsules*: ресурс, що надає змогу користувачеві збирати фотографії, відеозаписи, публікації в блозі та важливі документи в одному місці. Викладачі можуть використовувати його для навчання в класі та ведення онлайн проектів.

– *Google Docs*: безкоштовний хмарний застосунок, що є досить ефективним для викладачів. Педагоги можуть створювати форми зворотного

зв’язку для виконаних учнями проектів. Разом з тим, створення документів, таблиць, презентацій та їх подальший обмін між викладачем та учнями в аудиторії відбувається швидше за допомогою застосунку “Google Docs”.

– *Evernote*: найкращий сервіс для створення нотаток у режимі онлайн. Цей інструмент дозволяє користувачам бути добре організованими; робити, зберігати та синхронізувати фотографії, ідеї, нотатки, коментарі чи щось інше та отримувати безперервний доступ до них будь-де. Цей інструмент доцільно використовувати для планування уроків.

– *Socrative*: Інтернет-інструмент для викладачів. Ця система заохочує дітей до різних вправ і ігор , допомагає педагогам створювати, шукати та поширювати тести, та, безпосередньо, допомагає із проведенням тестувань. Цей інструмент, який доступний на комп’ютерах, у планшетах та на мобільних пристроях, має досить простий та зручний інтерфейс, дозволяє викладачам стежити за прогресом учнів та їх оцінками.

– *YouTube*: досить популярний відеохостинг, що містить у своїх архівах безліч важливих навчальних матеріалів, які згодом доцільно буде використовувати в аудиторії для оптимізації навчального процесу. *YouTube* також має спеціальний розділ для освіти. Завдяки деяким системним обмеженням, школярі мають змогу максимально ефективно використовувати *YouTube*, не відволікаючись на сторонні процеси.

– *Dropbox*: досить сучасний робочий простір, є одним із найкращих технологічних ресурсів для педагогів для оптимізації навчального процесу. Dropbox можна використовувати для зберігання даних, отримання доступу до них та обміну будь-якою інформацією. Також у застосунку є можливість надати для обмеженого кола користувачів (наприклад для однієї групи учнів). *Dropbox* є безкоштовним файлообмінником та досить легким у використанні.

– *SlideShare*: сервіс, що дозволяє досить легко завантажувати презентації, відеопроекти та будь-яких інші документи. *SlideShare* дозволяє користувачеві завантажити документ приватно, або зробити доступним для усіх бажаючих. [11, с. 212–213]

Цей перелік не є вичерпним, наведено , на мою думку, найважливіші застосунки для оптимізації навчального процесу.

Передумовами до розробки концепції автоматизації освітнього процесу є:

– технологічні фактори, які забезпечують нові технології та засоби для навчального процесу в сучасному інформаційно-телекомунікаційному середовищі;

– соціальні умови, включають вимоги сучасного суспільства до якості освітніх послуг та їх удосконалення;

– економічні чинники, полягають в тому, що освіта завжди робила значний внесок в розвиток макроекономіки.

Ключовою ціллю електронного навчання виступає створення належних умов для отримання максимальної ефективності у навчальному процесі. Практична реалізація смарт-навчання вимагає всебічного підходу, включаючи організаційний підхід, педагогічний та технологічний. Я вважаю доцільним буде зазначити ключові переваги запровадження до освітнього процесу інформаційних технологій:

– особистісно-орієнтований підхід у навчанні, запровадження персональних освітніх методик, що працюють та показали реальний результат;

– оперативний зворотний зв’язок з педагогами й іншими учнями;

– необмежений доступ до різноманітних джерел навчальної інформації;

– збільшення об’ємів самостійної індивідуальної та групової роботи, активізація процесу мотивування та заохочення школярів;

– запровадження новітніх освітніх технологій, що побудовані на широких можливостях навчальних автоматизованих інформаційних систем;

– реалізація змішаної форми навчального процесу;

– формування комунікативних компетентностей засобами віртуального освітнього середовища, що сприяють професійній соціалізації учнів [10, с. 14].

Разом з тим, головним недоліком інформаційних технологій можна назвати відсутність безпосередньої комунікації між школярем та викладачем, а й іноді заформалізованість протоколів спілкування в мережі. Однак,

співставивиши такі незначні недоліки та вагомі переваги, а також взявши до уваги невпинний інформаційний розвиток технологій, інформаційні-технології у навчальному процесі майбутньому ставатимуть тільки ефективнішими та дієвими.

Я вважаю виділити далі у роботі педагогічні цілі застосовування у навчальному процесі інформаційних технологій:

– активізація особистісного розвитку (естетичне виховання, логічне мислення, формування культури тощо);

– реалізація соціальної заявки (тотальна інформаційна комплексна підготовка спеціаліста у конкретній сфері);

– інтенсифікація виховного навчального процесу (максимізація якості та ефективності навчального процесу, поглиблення міжпредметних зв’язків тощо).

Визначальними напрямами використання у шкільному навчальному процесі інформаційних технологій можна визначити наступні:

– розроблення педагогічних програмних інструментів різноманітного спрямування;

– створення інтернет-платформ академічного призначення;

– опрацьовування дидактичних та методичних матеріалів;

– реалізація пошуку інформаційних відомостей різноманітних конфігурацій у локальних та глобальних мережах, активного збору такої інформації, акумуляції, зберігання, трансформації та її подальшої передачі;

– створення електронних бібліотек [8, с. 43–44].

Реформація навчального шкільного процесу, що відбувається сьогодні у нашій країні, цілеспрямована на те, щоб вивести значимість освіти у відповідність до новітнього ступеня наукового знання, максимізувати ефективність загального процесу виховної навчальної роботи та підготувати належним чином здобувачів освіти до поведінки в умовах переходу до автоматизованого суспільства. Саме тому інформаційні технології доцільно вважати невіддільним складником значимості навчального процесу, способом

оптимізації та максимізації ефективності навчання, а також допомагають у виконанні положень навчання.

На сьогоднішній день великої популярності набирають інтегровані заняття із використанням мультимедійного інструментарію. Навчальні мультимедійні презентації вважаються невіддільним елементом навчального процесу, однак таку діяльність справедливо вважають найпримітивнішим варіантом використання інформаційних технологій у шкільному навчальному процесі. Інформатизація освітнього процесу запускає розвиток інформаційних телекомунікаційних мереж, а мережа Інтернет у повній мірі забезпечує доступ до наявних об'ємів джерел інформації. Багато експертів розглядають технології Інтернет як революційний прорив, що перевершує за своєю значимістю появу персонального комп'ютера. Інструментальні засоби комп'ютерних комунікацій включають кілька форм: електронну пошту, електронну конференцію, відео конференц зв'язок, Інтернет. Ці засоби дозволяють викладачам та учням спільно використовувати інформацію, співпрацювати у вирішенні спільних проблем, публікувати свої ідеї чи коментарі, брати участь у вирішенні завдань та їх обговоренні [1].

Нинішні школярі зобов'язані самостійно вміти оволодівати необхідними навичками та знаннями. Для досягнення цієї цілі варто застосувати дидактичні особливості, які репрезентують новітні інформаційні технології, під якими розуміються ті їхні особливості, які дозволено застосовувати з дидактичною метою в виховному навчальному процесі, що виявляється у першу чергу в наступному: виучка, зберігання, класифікація, трасформація та роздрукування відомостей; виявлення даних на екрані; перспектива застосовувати модерні інформаційні технології; під'єднання до електронних баз даних; прийом та передача інформаційних відомостей; одержання даних від декількох джерел; передача інформувань заразом будь-якому числу користувачів; одночасний обмін даних із партнерами; транслювання відомостей одразу на пристрої учасникам; отримання даних від учасників конференції тощо.

В онлайн курсах переважно функціонують наступні схеми контролювання знань. Перша призначена для адаптації курсу, а наступна розроблена з метою атестації школяра. Обидва механізми дають змогу вчиняти адаптивну селекцію чергового запитання в залежності від коректності останніх результатів та перспективу розробки різних задач з одного переліку запитань [5].

Дистанційне навчання можна вважати ще однією прерогативою нинішніх інформаційних технологій, під яким вважається освітній процес, у процесі якого учні та викладач знаходяться у різних територіальних точках. Практичне застосування способів дистанційної освіти дає змогу одержувати високоякісну освіту, актуально на сьогоднішній день в умовах епідемічної ситуації, у дальніх районах, навчати осіб з різноманітними фізичними вадами тощо. Актуально зауважити, що порядок дистанційної науки не замінює, а ефективно доповняє звичний порядок освіти. Результативна реалізація дистанційної освіти допустима тільки за планомірної платформи розробки якісних мультимедіа-продуктів академічного призначення з різноманітних шкільних предметів.

1.2 Теоретичні основи розробки тестових завдань та загальний огляд педагогічного тестування

Моніторинг якості освіти, як вид наукового дослідження освітніх проблем та результатів втілення освітньої політики, здійснюється на основі певних наукових засад його організації та реалізації-правил, методів і технологій, які забезпечують можливість отримання об’єктивної та достовірної інформації, що використовуватиметься для побудови висновків та ухвалення управлінських рішень.

Останнім часом в освітньому процесі все частіше використовуються різні види тестування для контролю результатів навчання та оцінювання навчальних досягнень учнів. Такими варіантами тестування можуть бути тематичний або поточний контроль, вхідний, підсумковий або рубіжний контроль. Саме

практичне використання навчальних тестів допомагає виконанню всього функціоналу інспектування й належного оцінювання освітніх результатів, тому вони стали найбільш результативним способом педагогічних вимірювань і стали досить поширені у освітній діяльності [12].

Під терміном педагогічного тесту доцільно розуміти системну архітектоніку тестових задач, організовану за правильною закономірністю їх подання, яка покриває інформативність оцінювання ступеня і доброякісності навчальних здобутків, отриманих школярами у процесі навчання. Задачі в тестовій конфігурації зобов'язані вгамовувати конкретні стильові канони: належати до однієї предметної сфери; бути доступними та короткими; мати логічну конфігурацію повноцінної думки; розміщувати як коректні, так і хибні варіанти відповідей, еквівалентні квінтесенції задачі; виробляти ідентичні умови реалізації задач та їх належного оцінювання для усіх, хто проходить тестування. Традиційне навчальне тестування за часту запроваджують у цілях ефективного ревізування ступеня навичок та знань учнів або належного оцінювання здібності школяра відповідно застосовувати їх у різноманітних життєвих обставинах, тобто у цілях ідентифікації конструкції отриманих нею компетентностей.

Якість тестових завдань напряму залежить від великої кількості параметрів, вирішальне значення серед яких мають валідність і надійність одержаних результатів. Валідність характеризує відповідність одержаних результатів меті тестування і вимогам до обраного методу вимірювання. Це поняття визначає наскільки адекватно і якісно тест вимірює те, що означено метою тестування, тобто наскільки даний тест як метод вимірювання дає статистично вірогідні результати [14, с. 258].

Надійність тестування характеризується стабільністю (незмінністю) його результатів, тобто вона визначає наскільки статистично прийнятним є відхилення у результатах або наскільки вони збігаються при повторному проведенні того самого або аналогічного (однакового за форматом і складністю) тесту. Тому надійність тестування залежить від об'єктивності обраного методу і процедур тестування, якості тесту за добором завдань і стабільності

статистичних параметрів, що характеризують тест. Ступінь надійності тесту визначається за допомогою коефіцієнтів кореляції між результатами, одержаними однаковим методом (так зване паралельне тестування) за однакових умов. Виправданим критерієм логічного упорядкування тесту є показник складності тестових завдань. Його можна визначати на основі оцінювання передбачуваних кількості і характеру розумових дій, необхідних для успішного виконання завдання, або завдяки емпіричній оцінці складності завдань за результатами успішності їх виконання. та прогнозування успішності їх виконання як частка неправильних відповідей від їх загальної кількості [15, с. 422].

За допомогою педагогічного тестування стає можливим об’єктивніше оцінити рівень знань і вмінь учнів і структуру набутих ними компетентностей шляхом порівняння індивідуальних показників засвоєння навичок та знань для певних фрагментів навчального матеріалу.

На стартовій стадії конструювання тесту у першу чергу окреслюється головна мета тестування, у залежності від різновиду ревізування – поточний, вхідний, підсумковий або тематичний. Варто заначити, що у процесі вхідного інспектування визначається, у якій мірі школярі підготовлені до сприймання академічного матеріалу. Поточне контролювання ідентифікує процес засвоєння навичок та знань на конкретному ступені і переважно реалізує діагностуюче призначення для внесення правок у освітній процес. Тематичне інспектування визначає рубіжне оцінювання академічних здобутків школярів з конкретної частина предметної сфери знань і користується прогнозуючою направленістю в вирішальному оцінюванні навчальних результатів визначеного школяра. Головною метою підсумкового контролю варто вважати належне оцінювання освітнього результату, отриманого здобувачами освіти у ході навчання, і розкриває здобутки учнями цільових побажань до оволодіння значимістю навчального предмета.

Наступним кроком після дефініції цілей тестування, обирається різновидність тесту та виділяються підходи до його розробки. Після цього

процесу доцільно виконати аналіз основи навчального предмета в цілому, або певної теми, визначивши ті структурні компоненти навичок та знань, досвіду пізнавальної практики школярів, ціннісних постанов, які будуть оцінюватися. На базисі подібного синтезу формулюють конструкцію тесту і тактику розстановки завдань – за ступенями засвоєння, за тематичним переконанням відбиття змісту, за трудністю реалізації тощо [18, с.752–753].

Далі доцільно визначити тривалість виконування тестових завдань, його довжину (приблизна кількість тестових завдань) і створюють специфікацію тесту, тобто встановлюють перелік задач для конкретного компоненту теми або знань з урахуванням ступеня його засвоєння.

Наступним етапом після опрацьовування специфікації тесту розпочинається діяльність над розробкою самих задач. У процесі їх конструювання педагог може використати збірки тестів або самостійно розробити їх.

На стадії конструювання тесту здійснюється його експериментальна апробація у цілях ідентифікації статистичних характеристик задач. Цей етап стартує з опрацьовування методики апробаційного тестування та виучки інструктивних матеріалів. На зазначеному етапі позначається також методика оцінювання тесту та відбувається підбір шкал для інтерпретації отриманих результатів.

Далі здійснюється безпосереднє тестування, збір і статистична трансформація отриманих результатів, на базисі яких тест отримує статистично ваговиті психометричні специфіки тесту: максимальне, середнє та мінімальне значення отриманих школярами балів; коефіцієнти валідності для кожного тестового завдання окремо і тесту загалом; похибка вимірювання за заданим ступенем цінності тощо.

1.3 Огляд та аналіз існуючих програмних аналогів

На сьогоднішній день відома велика кількість інструментально-педагогічного інструментарію для розробки комп'ютерних тестів. Кожен із програмних засобів призначений для належної перевірки знань шляхом тестування, але кожен із них наділений власними характерними властивостями.

Найбільш пристосованою до нинішніх правил, формату зовнішнього незалежного оцінювання, тестування в навчальних закладах доцільно вважати програму *MyTestX*. Це програмне рішення дозволяє створювати різноманітні тестові завдання і користуватися ними на уроках. *MyTestX* – це комплекс програм (безпосереднє тестування, редактор тестових завдань та журнал результатів) для розробки та практичної реалізації тестування, опрацювання одержаних результатів, виставлення оцінок за шкалою, зазначеною в тесті. У програмі *MyTestX* можна застосовувати будь-яку кількість різновидів завдань.

Програма складається з трьох модулів:

– безпосереднє тестування (*MyTestStudent*);
– редактор тестових завдань (*MyTestEditor*);
– журнал результатів тестування (*MyTestServer*) [18].

Програмне рішення досить зручне та комфортне у використанні. Для розробки тестових завдань передбачений зручний редактор та зрозумілий інтерфейс. Програма *MyTestX* безкоштовна для використання. До кожного тестового завдання передбачена можливість прикріпити зображення або схему, звуковий файл тощо. Для будь-якого завдання дозволено окремо зазначити складність.

За присутності у навчальному закладі комп'ютерної мережі передбачена можливість організації централізованого збору та подальшого аналізу результатів проведеного тестування, застосовуючи модуль журналу тестування *MyTestServer*. За кожним тестом закріплений оптимальний час, перевищення якого істотно знижує якісні покажчики тесту.

Платформа підтримує кілька режимів: вільний, штрафний та навчальний. У навчальному розпорядку тестувальнику надходять сповіщення про його похибки, і може бути показане витлумачення до задачі. У штрафному режимі за

невірні відповіді віднімаються бали. У вільному режимі можна вирішувати тестові запитання в будь-якій послідовності.

Я вважаю, що єдиним недоліком такого програмного рішення можна вважати російський інтерфейс.

Вигляд програмного вікна платформи *MyTestX* зображено на рисунку 1.1, а саме журнал результатів тестування.

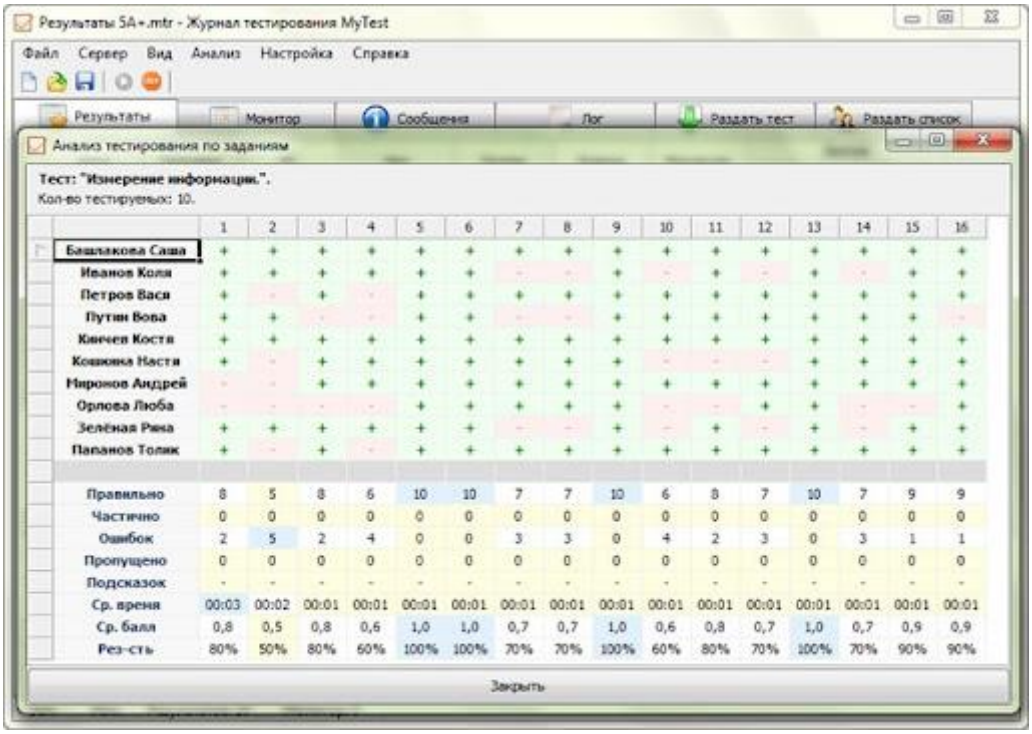

Рисунок 1.1 – Вікно програми *MyTestX*

Окрім особливих програмних інструментів для тестування та контролю знань учнів можна використовувати і традиційні програмні рішення для розробки тестових завдань, приміром, *MS Word*. На даній платформі скориставшись таким розділом як форма дозволено розробляти тестові завдання. Також, у випадку недостатньої кількості робочих місць у класі за комп'ютером, розроблені тестові завдання можна роздрукувати для письмового вирішення. Під формою розуміємо документ, який розміщує пусті поля або місця форми, розроблені для вводу даних. На платформі *Microsoft Word* є можливість реалізувати форму, розпочавши роботу із стандартного шаблону, а згодом додати компоненти керування вмістом [5].

Також за присутності у навчальному закладі мережі Інтернет є можливість розробки тестування у онлайн-форматі.

Google Форми виступає структурною частиною інструментарію *Google Drive*, що вважається одним із найлегших прийомів створити тест: компонуємо тестові завдання, обираємо варіант відповіді та отримуємо готове рішення. Сформований тест можна адресувати учням електронною поштою або розмістити на шкільній платформі. Для пришвидшення роботи можна також додати плагін *Flubaroo*, який механічно перевіряє отримані відповіді учнів і оцінює роботи адекватно заданих еталонів. *Google* Форми виступають безкоштовним рішенням, для їх використання платформі потібен тільки акаунт *Google*. Приклад тестування за допомогою ресурсу *Google* Форми зображений на рисунку 1.2.

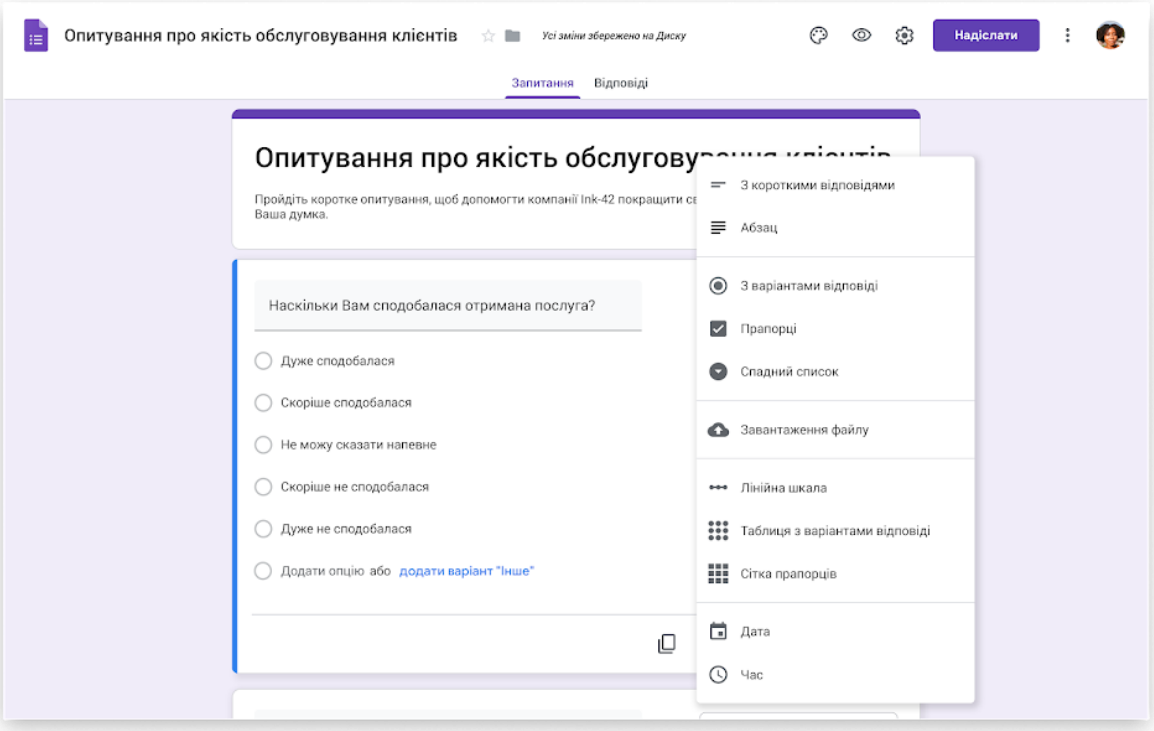

Рисунок 1.2 – приклад тестування за допомогою ресурсу *Google* Форми

LearningApps виступає онлайн-платформою для розробки інтерактивних тестів, завдань, вправ тощо. *LearningApps* – це безкоштовний сервіс для підтримки процесу викладання або самостійного навчання за допомогою

інтерактивних модулів. Користувачі можуть використовувати наявні модулі, модифікувати їх та створювати нові модулі з використанням пропонованого конструктора та шаблонів. Інтерактивні завдання скомпоновані за предметними категоріями.

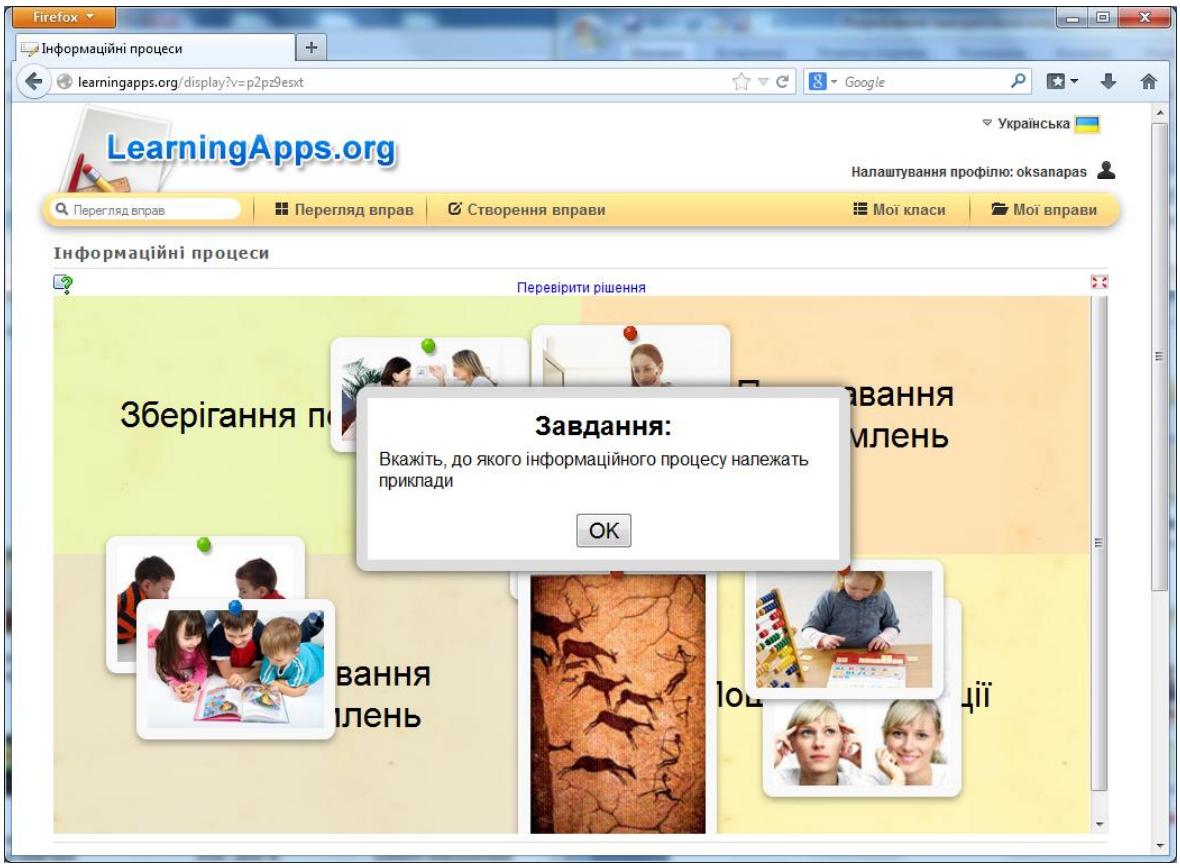

Рисунок 1.3 – Приклад форми платформи *LearningApps*

Програмне рішення *easyQuizzy* також є популярним варіантом платформи для розробки тестів. Ця програма є досить комфортною у процесі розробки завдань для контролю знань. Перевагами цієї програми можна вважати наступні характеристики: легкість користування; можливість розробки тестових завдань на один варіант правильної відповіді, на множинні відповіді та тестування на вписування відповідей; у процесі тестування *easyQuizzy* не вимагає інсталювання особливого ПЗ на пристрій, так як діє як окремий додаток самостійно від ОС. Програму *easyQuizzy варто* використовувати на уроках для жвавого ефективного опитування та для заключного ревізування знань, а також для самоосвіти [15,

с. 421]. З прикладом форми для тестування програмним забезпеченням *easyQuizzy* можна ознайомитись на рисунку 1.4.

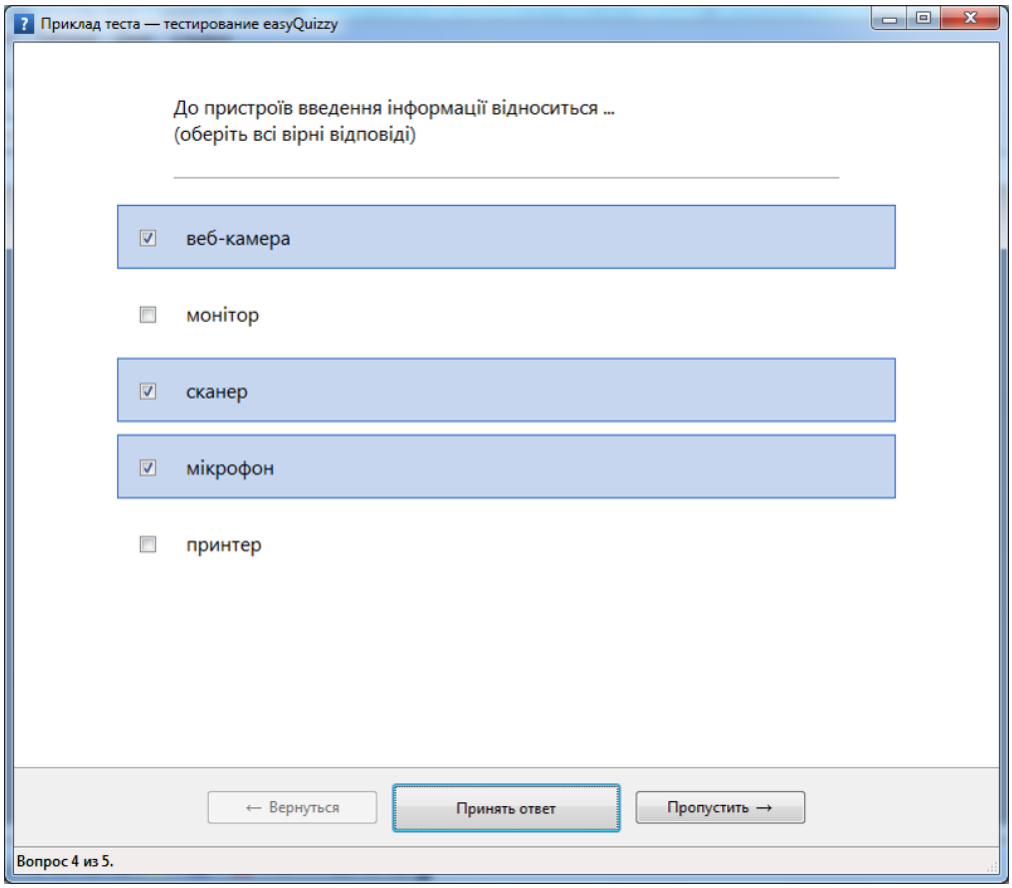

Рисунок 1.4 – Приклад форми для тестування *easyQuizzy*

1.4 Постановка задачі

У процесі активного розвитку новітніх технологій потенціал онлайн-освіти значно збільшився. Інтернет-ресурси не лише зробили навчальний процес доступним для більшого кола споживачів, але і докорінно трансформував безпосередній підхід до академічного процесу.

Колосальну роль у процесі онлайн-освіти відіграє моніторинг якості навчального процесу. Він вважається найважливішим і невід'ємним структурним компонентом нинішнього цілого освітнього процесу.

Тому було запропоновано розробити програмне забезпечення з якості оцінювання отриманих знань шляхом проведення тестування.

Основні завдання розробки системи:

– надання доступу до перегляду курсів та уроків;
– проходження уроків та тестів;
– обробка результатів тесту;
– надання результату та рекомендацій після уроку;
– доповнення системи новими курсами та уроками;
– моніторинг успішності школярів.

1.5 Висновки до розділу 1

Проведений аналіз показав, що технології інтерактивності і мультимедійності використовуються в інформаційних підсистемах освітнього і наукового напрямку, мають актуальність і практичне застосування в системній організації інформаційних ресурсів, та для навчальних програм. Також такі зазначені новітні технології активно використовуються у процесі тестування та контролю знань учнів.

При проведенні аналізу сучасних підходів до створення навчальних програм щодо застосування засобів обчислювальної техніки були виявлені проблеми, які виникають при створенні інформаційних систем; сформульовані вимоги до створення майбутнього програмного модуля, призначеного належного тестування та контролю знань учнів. Виконано функціональне моделювання програмного модуля.

Програма призначена для полегшення ефективної роботи та підготовки до навчальних предметів а також ефективного контролю засвоєного матеріалу шляхом тестування. Також в першому розділі розглянуті та проаналізовані програмні аналоги системи тестування та контролю знань учнів.

2 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

2.1 Огляд предметної області

Необхідність накопичення великих об’ємів професійно-цінних інформації і оперування ними – одна із проблем, з якою зіштовхнулися вчителі та школярі.

Активне розповсюдження інформаційних ресурсів, її подальший збір обробка та трансформація всередині сучасного інформатизованого суспільства відбувається завдяки спеціальним ресурсам. Ключова функція інформаційної системи виступає у забезпеченні актуальними інформаційними даними суспільство. Інакше кажучи інформаційна система задовольняє потребу в отриманні відомостей в межах окресленої області предмета. Кінцевим продуктом активної діяльності інформаційної системи виступають масиви даних, структуровані бази, нормативні документи [31].

Інформаційна система виступає системою, яка реалізує або в якій відбуваються інформаційні процеси такі як пошук, накопичення, збереження, трансформація та обробка інформаційних даних. Обробка даних передусім залежить від об’єму та змісту вхідних інформаційних даних, однак в процесі самої обробки інформаційні потоки не осмислюються, а лише трансформуються відповідно до алгоритмів, розроблених раніше.

Далі окреслимо активні процеси, які відбуваються в інформаційній системі:

– введення інформаційних потоків, які одержані із джерел інформації;
– обробка та трансформація інформаційних даних;
– збереження вхідних та опрацьованих інформаційних;
– виведення призначеної для користувача інформації;
– відправка / отримання інформації мережею.

Розробка інформаційної системи передбачає вирішення наступних задач:

– заповнення інформаційної системи даними специфічної вирізняльної предметної області;
– розробка інтерфейсу активного користувача (переважно графічного) для одержання потрібних інформаційних ресурсів.

До автоматизованої системи інформаційні дані поступають від джерела інформації. Такі відомості відправляють на подальше зберігання чи трансформацію у системі, а вже згодом передають активному споживачеві. Цей процес ми можемо простежити на рисунку 2.1.

Рисунок 2.1 – Процес отримання інформації

2.2 Визначення вимог до інформаційної системи

2.2.1 Вимоги до програмних характеристик

Вирішено розробити інформаційно-довідкову систему, в якій користувач матиме можливість:

– перегляду успішності своєї навчальної групи, вносити змінні, вести пошук необхідної інформації у БД;
– вести пошук необхідної навчальної літератури;

– вести пошук необхідних методичних розробок щодо запитів користувача;
– виводити відеоматеріали щодо заданих параметрів користувача; проводити тестування на якість проходження курсу дисципліни та визначення рівня засвоєння матеріалу щодо набраних балів.

Програмний модуль повинен мати простий та зручний графічний інтерфейс користувача. Для розробки інформаційно-довідкової системи необхідно вирішити низку приватних взаємопов'язаних завдань:

– вибір інструментарію та середовища розробки;
– підготовка до роботи та проектування модуля;
– розробка інформаційно-довідкової системи;
– оцінка ефективності розробленої системи.

Для реалізації даного програмного модуля були використані наступні головні бібліотеки, які необхідні для реалізації який планує функціоналу:

– *EPPlus*, бібліотека необхідна генерації звітів наявних даних в БД;
– *MySQL.Data*, яка необхідна для реалізації роботи з даними, які реалізовані в СУБД *MySQL;*
– *System.Drawing*, бібліотека, яка необхідна для роботи з графікою;
– *System.Windows.Forms*, бібліотека, яка необхідна для реалізації графічного інтерфейсу.

Також в розробці програмного модуля були використані драйвер *MySQL Connector / NET*, який виконує роль з'єднувача. Він дозволяє програмам .NET взаємодіяти з серверами *MySQL*.

Для реалізації даного програмного забезпечення були використані середовище розробки *Visual Studio* 2019, яке знаходиться у вільному доступі.

Для розробки і проектування БД, яка необхідна для маніпуляцій з даними, було розгорнуту локальний сервер за допомогою програмного забезпечення *MAMP* і з'єдналися з веб-додатком *phpMyAdmin*.

Як забезпечується взаємодія користувачів та вищеназваних системних засобів, показано на рисунку 2.2.

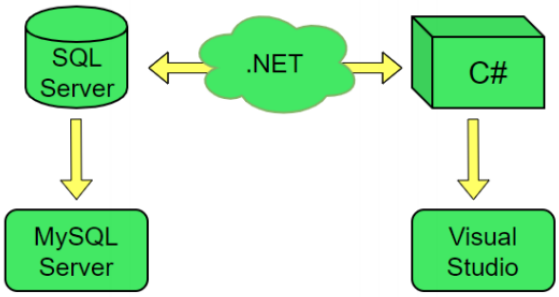

Рисунок 2.2 – Функціональний принцип роботи системи

Для виконання роботи було використано операційну систему Windows 10 корпорації Microsoft. На сьогоднішній день Windows найбільш поширена операційна система виробництва Microsoft. Основними причинами вибору саме цієї операційної системи є:

– стабільність роботи системи;
– зручний інтерфейс;
– режим мультизадачної роботи;
– популярність використання.

Також необхідно відзначити, що Windows 10 має точки відновлення, тому у разі збою в роботі системи можливе повне відновлення програм, документів та інших даних, система є багатофункціональною та при встановленні має стандартний пакет драйверів.

Програмний модуль було реалізовано на ПК з наступними характеристиками:

– операційна система: *Windows* 10 *Pro*;
– процесор: *Intel (R) Core (TM) i3-7020U CPU @ 2.30GHz 2.30 GHz*;
– оперативна пам’ять: 8,00ГБ.

Розроблений програмний модуль буде містити в собі наступні класи:

– клас *DB.cs*, для отримання інформації щодо під’єднання до сервера;
– клас *DataClass.cs*, клас для виконання маніпуляцій з даними;
– дві графічні форми для комфортного користування та виконання дій над даними, які знаходяться в БД;

– графічні ресурси, які постійно закріплені проекту, які необхідні для графічного інтерфейсу.

Виконання команд з наявними даними, які знаходяться в БД, виконується за допомогою методу *GetData* (), який реалізований в класі *DataClass.cs*. Для виконання необхідного *SQL*-запиту, необхідно внести даний запиту в подвійні лапки до параметру даного методу.

2.2.2 Вимоги до надійності та продуктивності

Надійне і, в той же час ефективне функціонування програмного модуля повинно бути забезпечено реалізацією наступних організаційних і технічних заходів:

– запобігання помилок за рахунок структурного програмування;
– безперебійне живлення технічних засобів;
– обробка виняткових ситуацій (перехоплення помилок, наприклад, розподіл на нуль) і локалізація помилок і збоїв;
– використання виключно ліцензійного програмного забезпечення, придбаним у офіційних постачальників;
– періодичне виконання діагностики щодо захисту інформаційних даних (регулярна перевірка на наявність вірусів);

Також можна сформувати такі вимоги до продуктивності роботи програмного модуля:

– час авторизації в систему не повинно перевищувати 10 секунд;
– необмежена кількість користувачів;
– для звільнення ресурсів по закінченню роботи з базою з’єднання з базою автоматично закривається;
– необмежена час зберігання інформаційних даних в базі.

2.2.3 Умови експлуатації

Для коректної та ефективної роботи розробленого програмного модуля потрібно дотримуватися встановлених правил експлуатації. Експлуатація включає в себе роботи з впровадження компонентів програмного забезпечення в експлуатацію, в тому числі конфігурацію бази даних і робочих місць користувачів, забезпечення експлуатаційної документації, проведення навчання персоналу, і безпосередньо експлуатацію, модифікацію програмного забезпечення, підготовку пропозицій до вдосконалення, розвитку та модернізації системи. Експлуатаційні показники ефективне роботи не повинні перевищувати коефіцієнт, зазначених в технічній характеристиці обладнання.

2.3 Теорія проектування баз даних

Передусім, щоб коректно та всебічно дослідити теоретичне підґрунтя проектування баз даних, слід звернутись до визначення поняття “проектування” та “база даних”. Після тлумачення необхідних нам термінів ми зможемо визначити взаємозв’язок та значущість понять. Проектування – це складний процес мисленнєвої логічної діяльності із конструювання, цілісних, завершених ідеальних моделей функціонування нових об’єктів на підставі поєднання теоретичних знань і практичного досвіду та їх практичного втілення в історичних умовах визначеного простору та часу відповідаючи інтересам та потребами конкретного соціального класу. Базуючись на дослідженнях вітчизняних фахівців, звернемося до загальноприйнятого тлумачення терміну “база даних” – це конфігуративна сукупність даних, сформованих згідно із теорією, що описує специфічні особливості цих даних та структурні зв’язки між їх складовими елементами, основною метою якої виступає зберігання, обробка та видозміна взаємопов’язаної інформації великих обсягів, містить у собі таблиці, схеми, збережені процедури та інші об’єкти [28, с. 276].

Саме тому можна стверджувати, що поняття “проектування БД” включає в себе трудомісткий процес конструювання саме такої схеми та структури бази даних, яка б відповідала загальноприйнятим вимогам активних користувачів, визначення необхідних обмежень цілісності, а також опис БД з різним рівнем деталізації, в ході якого проводиться оптимізація, уточнення та удосконалення її структури.

Можна виділити основні завдання проектування баз даних:

– забезпечення цілісності бази даних;
– зниження рівня дублювання даних;
– безпечне зберігання всієї необхідної інформації у БД;
– можливість отримання даних по всім запитам.

Взагалі, процес пректування БД є складним і вимагає концептуального підходу на основі певної методології, застосування певних методів. Методологія проектування баз даних являє собою сукупність методів, засобів, логічних зв’язків, принципів та інструментів, що застосовуються для покрокового розроблення структури бази даних. З огляду на те, що система баз даних складається із даних та програм, методологія проектування баз даних розглядається в контексті невід’ємної частини загального пректування програмних систем [21].

Підсумувавши вищесказане, можна зазначити, що методологія проектування баз даних визначає:

– безросередній процес проектування;
– методичні умови як вихідні дані для процесу проектування;
– загальноприйняті правила виконання розрахунків та принципи оцінювання альтернативних рішень на кожному кроці проектування;
– практичне відображення отриманих результатів кожної стадії проектування [5].

2.4 Моделювання предметних областей

Мова ER-моделювання – це специфічна мова ідентифікації інформаційних потреб користувача. Мова ґрунтується на теорії, адекватно до якої інформаційне забезпечення предметної області рекомендується як сукупність об’єктів, які вважаються взаємозалежними. Безпосередня процедура моделювання виявляється у виокремленні квінтесенцій ПО, установлення особливостей визначених сутностей та ідентифікація наявних між ними зв’язків. Я вважаю за доцільне розглянути визначальні специфічні властивості, номінальні відмітки й дефініції сутностей, атрибутів та зв’язків [2].

Під сутністю розуміємо уявний або реальний об’єкт зацікавлення, відомості про який піддається зберіганню або збору. Ім’я сутності дається в однині й пишеться заголовними буквами. Графічно сутність виявляється прямокутником маючи повну назву із закругленими кутами.

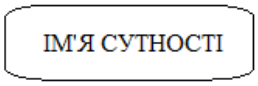

Рисунок 2.3 – Сутність

Під зв’язком розуміємо характерологічну пойменовану асоціацію, яка репрезентує інтерес двох сутностей. Зв’язок вважається подвійним в тому переконанні, що це у будь-якому випадку асоціація в точності двох сутностей або сутності із самої собою. На діаграмах зв’язки виступають лініями, що об’єднують дві сутності. Однин із видів зв’язку представлений на рисунок 2.4. Це зв’язок зі ступенем багато-до-одного.

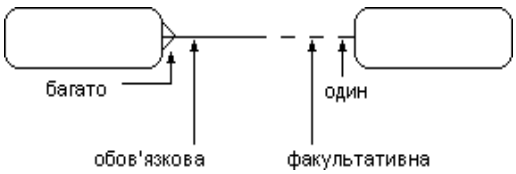

Рисунок 2.4 – Приклад зв’язку

Атрибутом вважається будь-яка риса або площина, що допомагають якісному або чисельному опису сутності, її визначенню, систематизації або відображенню її стану. Атрибутом може виступати число, текст, картинка тощо. Для заяви атрибута зазначається його ім’я в однині малими літерами. Не обов’язково вказувати атрибути на діаграмі зв’язків та сутностей, але додавання до сутності атрибуту у момент формування макета, виступає корисним. Атрибут описує лише одну сутність, до якої він занесений. Атрибут, вагомість якого має бути відома у будь-якому випадку, іменується обов’язковим, і видається символом "*" перед ім’ям [4].

Кожна сутність має ідентифікуватися за сприянням специфічної комбінації зв’язків та (або) атрибутів. Саме тому між імовірних атрибутів сутності всякчас мають бути визначені такі атрибути, які допомагають її ототожнити. Винятковий ідентифікатор визначається на *ER*-моделі символом "#" перед ім’ям усякого атрибута, що складає визначений ідентифікатор.

У процесі побудови *ER*-моделі варто використовувати вимоги, зазначені у роботі далі. У першу чергу доцільно вирізати атрибути, що ототожнюються. У конкретний часовий проміжок сутність може містити тільки одне значення атрибута. Роль протилежних атрибутів мають взаємозалежати від усього ідентифікатора. У процесі роботи доцільно виокремити такі атрибути, вагомість яких буде у взаємозв’язку тільки від одного фрагмента ідентифікатора. Атрибути мають взаємозалежати лише від виняткового ідентифікатора [22].

У мові *ER* бувають й інші поняття, однак розглянуті лише такі, які будуть використані у дипломному проекті. У процесі написання кваліфікаційної роботи було побудовано таку *ER*-модель, зображену на рисунку

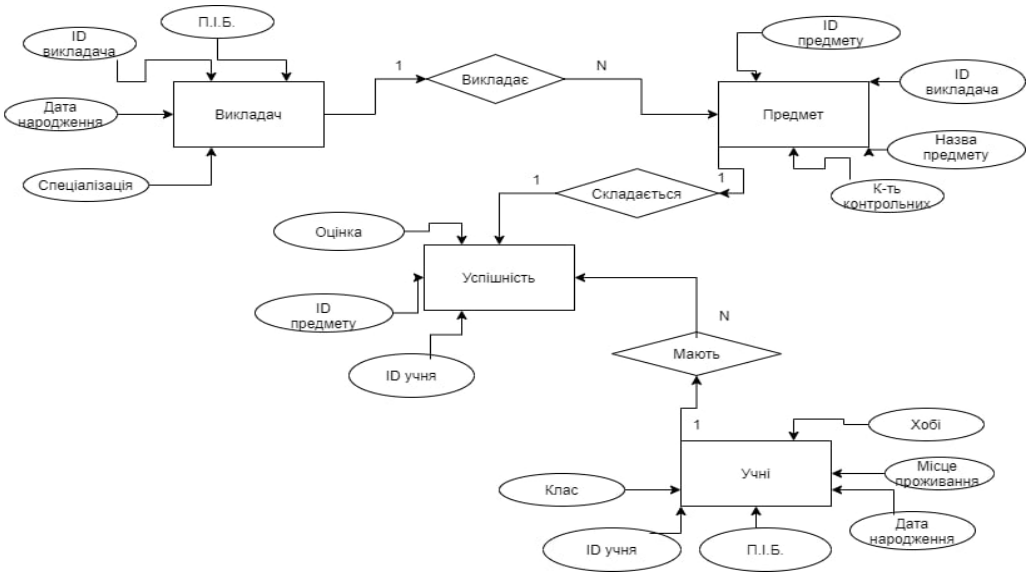

Рисунок 2.5 – ER-модель

Для комфортного користування програмним модулем приведена також блок-схема (рисунок 2.6).

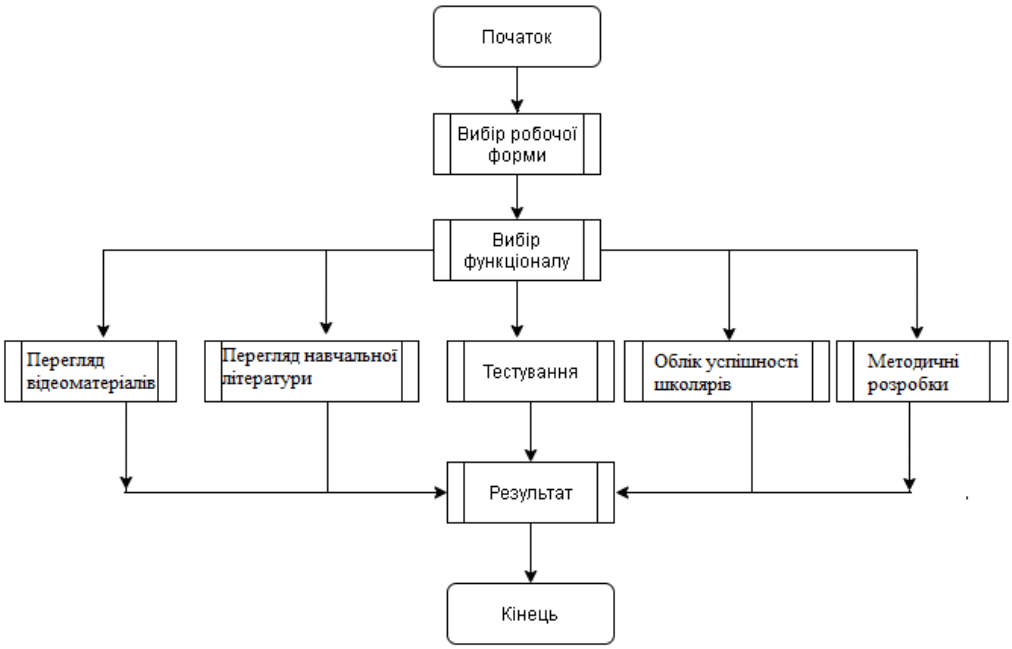

Рисунок 2.6 – Блок-схема

2.5 Етапи проектування бази даних та вибір інструментарію

Взагалі, процес проектування БД – це багатоетапний довготривалий процес, результатом якого виступає побудована активно функціонуюча інформаційна система, за допомогою якої будуть вирішуватись саме ті питання та завдання, для яких вона створювалась. Складний процес проектування БД являє собою порядок трансформації від неформального лінгвістичного опису інформаційної структури предметної області до дійсної характеристики об’єктів предметної області в термінах деякої моделі. Етап проектування БД можна вважати одним із найскладніших кроків створення БД, який не має явно вираженого початку та фінішу процесу. Проектування стартує з моменту затвердження стратегічних рішень і продовжується на етапах тестування та практичної реалізації [6].

На мою думку, можна виділити наступні етапи процесу проектування:

Визначення стратегії.

Метою запропонованого етапу виступає передусім ухвалення із замовником попереднього плану, розробленого з урахуванням наявних технічних, організаційних та економічних характеристик, що відображає як поточні, так і майбутні потреби організації формування прикладних моделей. Як показує практика, тактика створення інформаційної системи формується у результаті типізованого аналізу, на підставі якого потім будується фактична модель прикладної області. Терміни формулювання стратегії мають охоплювати незначний відрізок часу, аби планові результати та бажаний ефект проектування не втратили своєї актуальності. Результати цього кроку мають бути сформульовані конкретно та чітко, щоб замовник міг із легкістю співвіднести запропоновану виконавцем стратегію зі своїми завданнями та плановими цілями і затвердити її остаточно [17].

Основними результатами цього етапу мають бути:

– покроковий план проектування бази даних;
– опис провідних цілей, завдань, необхідних витрат , можливих ризиків та візуалізація кінцевої мети;
– узагальнена діаграма сутностей і зв’язків;

– пропозиції щодо майбутнього практичного застосування , реалізації та подолання можливих труднощів [22].

Аналіз предметної області.

Результати етапу визначення стратегії є вихідними даними для етапу аналізу, де вони проходять ретельну перевірку та деталізуються, для того щоб забезпечити предметній області адекватність моделі, гарантувати практичну реалізацію рішень і сформувати ефективну базу для наступних етапів. Цей етап проектування виступає найважчим та, беззаперечно, охоплює найбільший відрізок часу, однак, разом з тим, є найважливішим, оскільки саме на ньому генерується переважна більшість проектних рішень.

Системний аналіз предметної області має передбачає докладний словесний опис центральних об’єктів даної предметної області, специфічних ознак цих об’єктів, їх особливостей та взаємозв’язків. Аналіз предметної області включає в себе процес аналізу вихідних даних, що передбачає кодифікації всіх атрибутів та ретельного аналізу поставлених завдань, що зобов’язує використання діаграм та детального всебічного опису алгоритмів. Ретельно досліджується необхідність заходів захисту та контролю даних, їх резервному копіюванні та подальшому відновленні [11, с. 253].

Невід’ємною частиною згаданого етапу виступає оцінка наявних систем, зовнішніх та внутрішніх факторів, що прямо чи опосередковано впливають на процес, якість та фінальний результат виконаної роботи, а також терміни практичної реалізації системи. Етап аналізу повинен закінчуватися ретельним всебічним описом інформації щодо центральних об’єктів предметної області, що потрібні для вирішення поставлених задач, коротким описом алгоритмів вирішення цих питань. Невід’ємною фінальною частиною розглянутого етапу виступає опис вихідної документації, що повинна генеруватися в системі, а також аналіз вхідних матеріалів, що є безпосереднім джерелом даних, що вводяться в БД.

Основними результатами цього етапу мають бути:

– чітко поставлений та узгоджений список завдань;

– наявність діаграми сутностей і зв’язків;
– попередня оцінка розмірів системи;
– відомості про об’єм даних, очікуваний рівень продуктивності та частоту виконання поставлених завдань;
– загальна характеристика процедур, що не автоматизуються.

В проектуванні БД існують три етапи: концептуальне проектування логічне та фізичне.

Для побудови необхідної БД для створення програмного забезпечення для тестування та контролю знань учнів доцільним буде використовувати СКБД *MySQL*.

Дана система керування базами даних (СКБД) з відкритим кодом була створена як альтернатива комерційним системам. *MySQL* – одна з найпоширеніших систем керування базами даних. В першу чергу вона використовується для створення динамічних Веб-сторінок, оскільки має підтримку з боку різноманітних мов програмування [13, с. 103-104].

MySQL вважається чудовим рішенням для додатків (малих і середніх). Вихідний код сервера накопичується та поширюється на безлічі платформ. Найповніше можливості сервера виявляються в *UNIX*-системах, де є підтримка багатопоточності, що підвищує продуктивність системи в цілому.

Для некомерційного використання *MySQL* є безкоштовним. Переваги сервера *MySQL*:

– досить простий у становленні та подальшому користуванні ;
– є можливість підтримки необмеженою кількістю користувачів, що працюють одночасно із БД;
– кількість рядків у таблицях може досягати 50 млн.;
– висока швидкість виконання команд;
– наявність простої і ефективної системи безпеки.

Недоліки сервера *MySQL*:

– не реалізована підтримка трансакцій, однак є можливість використання *LOCK/UNLOCK TABLE*;

– немає належної підтримки зовнішніх ключів;

– відсутня підтримка збережених процедур та тригерів;

– відсутня підтримка представлень [16].

Однак, перелічені недоліки не є критичними при розробці інформаційних систем для робочих груп.

Для використання СКБД *MySQL* за допомогою програмного забезпечення *MAMP* було розгорнуто локальну серверну мережу. Згадане програмне забезпечення є цілком безкоштовним та має вільний код, а також є досить простим при встановленні та у використанні.

Для створення необхідної БД було прийнято рішення використовувати веб-додаток *phpMyAdmin*, який створений на мові *PHP* та має відкритий код. *phpMyAdmin* дає змогу через браузер запускати запити SQL, здійснювати адміністрування сервера *MySQL*, переглядати та змінювати вміст таблиць баз даних. Ця програма є досить популярною серед веб-розробників, оскільки надає можливість керувати базу даних *MySQL* без вводу *SQL* команд будь-якого комп'ютера, що має доступ до мережі Інтернет без встановлення додаткового програмного забезпечення.

На сьогоднішній день *phpMyAdmin* широко застосовується на практиці. Це безпосередньо пов'язано із тим, що розробники розвивають свій продукт, з огляду на всі нововведення СКБД *MySQL*. Вагомий відсоток українських провайдерів використовують цей додаток як панель керування для того, щоб надати своїм клієнтам можливість адміністрування виділених їм баз даних.

Сучасне програмне забезпечення стає щораз складнішим. Факторами, що ускладнюють програмне забезпечення виступають: розвинені можливості зручного інтерфейсу, нові нетрадиційні зони використання, керованість програми подіями тощо. На сьогоднішній день, щоб побудувати програму, не достатньо просто об'єднати в послідовність певні машинні інструкції, оператори мови високого рівня чи навіть набори процедур та модулів. Ключовим виступає проблема розробки виразної структури програми, що здатна до вільної модифікації, стійкої до змін та без наявних помилок.

2.6 Висновки до розділу 2

У другому розділі розглянуто методологію побудови інформаційної системи та досліджено доцільність її розробки. Окреслено головні вимоги до інформаційної системи, а саме вимоги до програмних характеристик, вимоги до надійності та продуктивності системи та визначено умови експлуатації програмного модуля.

Обрано необхідний інструментарій, а також аргументовано доцільність їхнього використання.

Виконано моделювання програмного середовища, а саме побудована блок-схема та ER-модель. Також визначено головні етами проектування бази даних майбутнього програмного модуля.

3 РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ

3.1 Архітектура програмної системи

Архітектура програмного продукту (*software architecture*) – специфічна структура програмного модуля, яка включає в себе першочергово програмні структурні елементи, зовні видимі характерні властивості згаданих компонентів, а також взаємовідносини між ними. Згаданий процес також стосується процесу документування архітектури програмного забезпечення.

Прикладне програмне забезпечення, призначене для вирішення завдань користувача. Виконання роботи розподілене на окремі блоки, які можуть виконуватися як послідовно, так і паралельно (в окремих потоках програмного продукту). Кожен блок виконує тільки свою специфічну роботу, наприклад, перевіряє дані, введені користувачем [10].

В архітектурі додатки можна виділити два головних компонента:

– блок графічного інтерфейсу користувача, який представляє зовнішній вигляд фінального програмного результату і виконує взаємодію з користувачем програмного продукту;
– модуль вирішенні завдання, який отримує і аналізує вхідні дані, складає і вирішує завдання, оформляє результат виконання і передає його до графічного інтерфейсу.

Графічний інтерфейс користувача дозволяє активно взаємодіяти із програмою способом, який зручний та комфортний безпосередньо для користувача . Він складається з:

– блоку взаємодії з вхідними даними;
– блоку взаємодії з вихідними даними.

Модуль вирішенні завдання виконує саму роботу програмного продукту. Створюється модуль з таких компонентів:

– блок обробки результату моделювання групує результат виконання блоку вирішенні завдання в набір масивів даних і передає в блок графічного інтерфейсу користувача;
– блок вирішенні задачі розрахунку нормованих показників і показника сумарного питомої нормованого потенціалу [12].

Оскільки програмний продукт написаний на мові, який реалізує принципи об’єктно-орієнтованого програмування, кожен з компонентів програми можна представити у вигляді об’єкта, яких є екземпляром певного класу. Тобто робота і поведінка кожного компонента архітектури описується класом. Це дозволяє створити ефективну і гнучку взаємодію об’єктів.

Для поліпшення швидкодії програмного продукту все трудомісткі операції виконуються в окремих паралельних потоках, що дозволяє мінімізувати час виконання операцій на ПК із багатоядерними процесорами.

У розробці програмного модулю була обрана клієнт-серверна архітектура. Клієнт-серверна архітектура – це такий спосіб організації структури додатків, в якій визначені для виконання системи завдання, умовним чином диференціюються між двома підкласами: клієнтом і сервером.

До сервера звертаються безліч клієнтів і часто він зберігає дані, тому він виступає “центральною” частиною. З цього випливає, що сервер може існувати без клієнтів, а от клієнти без сервера не можуть. Клієнт – програма, примірників якої зазвичай “більше”, ніж серверів [21].

Система диференціюється на окремі частини, які мають можливість виконуватися в будь-яких вузлах мережі. Кінцевий користувач, або ж безпосередньо програма активно взаємодіють із клієнтською долею системи, яка в забезпечує міжмережевий інтерфейс. Якщо виникає необхідність, то клієнтська частка системи звертається до серверної частини по мережі.

Архітектура майбутнього програмного модуля зображена на рисунку 3.1.

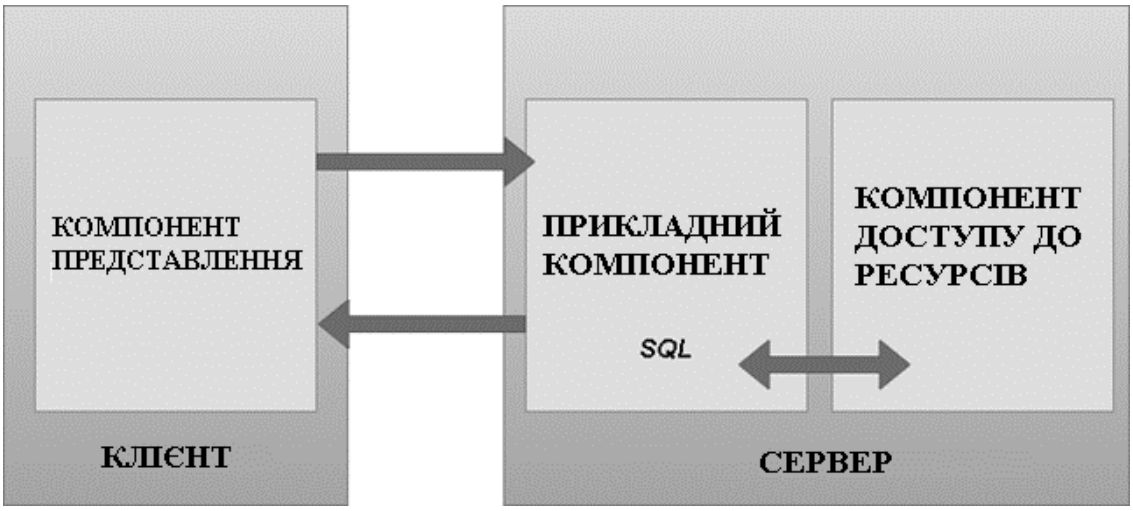

Рисунок 3.1 – Архітектура програмного модуля

Однією із переваг такої моделі взаємодії є те, що розділений вихідний код клієнтського і серверного додатків. До помітних переваг згаданої архітектури можна віднести зменшені вимоги до пристроїв користувачів, а також використовувана архітектура досить гнучка і надає змогу адміністратору локальної мережі зробити її більш захищеною [19].

Недоліками такої організації виступає той факт, що фінансові затрати на серверне обладнання значною мірою більші, аніж клієнтське. Сервер також повинен обслуговувати лише фахівець, спеціально підготовлений. Також варто відзначити, що у випадку збою сервера у локальної мережі втрачається можливість клієнта працювати у нормальних, звичних для нього умовах.

Однак, ці недоліки не є критичними, саме тому обраний такий варіант

3.2 Модель даних

Створення програмного продукту відбувається із дотриманням певних правил, що встановлюються при виборі однієї із моделей розробки. Задля реалізації поставленого завдання була обрана каскадна модель. Під моделлю зазвичай мається на увазі структура, яка визначає послідовність виконання та

взаємозв’язку процесів, дій та завдань протягом життєвого циклу. Схема каскадного процесу розробки програмного модуля зображено рисунок 3.2.

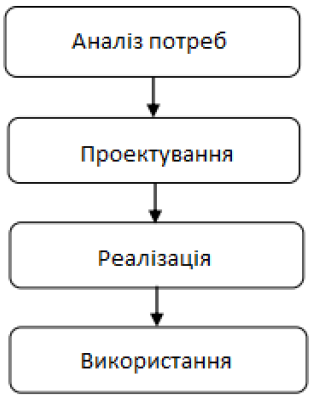

Рисунок 3.2 – Каскадна модель розробки програмного модуля

У каскадній моделі ми переходимо від однієї стадії до іншої послідовно. Спочатку повністю завершується етап «визначення вимог», у результаті виходить список вимог до програмного модуля. Після того, як вимоги повністю визначені, відбувається перехід до проектування. Після того, як проектування повністю виконано, виконується реалізація отриманого проекту. На наступній стадії процесу відбувається інтеграція окремих компонентів. Після того, як реалізація та інтеграція завершена, проводиться тестування та налагодження продукту. На цій стадії усуваються всі недоліки, що виникли на попередніх стадіях розробки. Тим самим каскадна модель передбачає, що перехід від однієї фази розробки до іншої відбувається тільки після повного та успішного завершення попередньої фази [9].

Реалізація будь-якої системи вимагає насамперед аналізу та пошуку оптимального шляху вирішення цього завдання. Після аналізу вимог та постановки завдання формується функціональна схема із зображенням взаємодії блоків програмного модуля. Функціональна схема – це схема взаємодії компонентів програмного забезпечення з описом інформаційних потоків, складу даних у потоках та зазначенням використовуваних файлів та пристроїв. Кожен блок виконує лише свою специфічну роботу, наприклад, перевіряє дані, введені користувачем.

На рисунок 3.3 зображено функціональну модель програмного продукту.

Рисунок 3.3 – Функціональна схема програмного модуля

3.3 Розробка інтерфейсу програмного модуля

Для того щоб інтерфейс програмного модуля для активного користувача був зручним, він має проектуватися на базі продуманої реалізації набору типових операцій над даними. У випадку відсутності такої бази, буде спостерігатися негативний відбиток на якості безпосередньо фінішного програмного продукту, час розробки такого продукту помітно збільшиться.

Вимоги до інтерфейсу, призначеного для користувача:

– зменшення часу виконання функцій пошуку та зчитування інформації;

– зменшення термінів вибору команди та навігації;

– збільшення загальної продуктивності користувача, яка полягає в обсязі оброблених вхідних даних за конкретний проміжок часу;

– максимізація тривалості стійкої роботи користувача.

Стильова гнучкість:

– можливість використовувати з одним і тим самим додатком різні інтерфейси, практично реалізуються із використанням таблиці стилів, разом з тим враховуючи можливість у виборі активним користувачем власних установок інтерфейсу .

Збільшення функціональності:

– передбачення можливості розвитку додатку без руйнування існуючого інтерфейсу – мається на увазі залишення у рамках уже існуючого спроектованого інтерфейсу.

Масштабованість:

– можливість налаштування та поступового розширення інтерфейсу, а також і самого додатку у випадку зростання числа активних користувачів, об’єму та змісту даних.

Адаптивність до дій користувача:

– програмний модуль має передбачати можливість введення даних і команд виключно різними варіантами та способами та багатоваріантність доступу до прикладних функцій. Також варто відзначити, що програмне забезпечення має враховувати варіант переходу та повернення від форми до форми та вміти коректно обробляти згадані ситуативні варіації.

Переносимість:

– при переході на іншу апаратну (програмну) платформу, повинен здійснюється автоматично перенесення і призначеного для користувача інтерфейсу, і кінцевого додатки.

Методи оцінки користувальницького інтерфейсу:

– для оцінки необхідного рівня зручності інтерфейсу використовуються спеціальні експертні анкети, опитувальники, формуляри, *check*-листи.

В якості методів використовують:

– спостереження за користувачами до використання програмного інтерфейсу, в процесі навчання і роботи;

– постановка і протоколювання виконання тестових завдань.

Цілі і критерії оцінки користувальницького інтерфейсу.

– ключове у програмному модулі – реалізація такого інтерфейсу, що призначений для користувача, який зможе виконати поставлене завдання ефективно та продуктивно, а також забезпечить задоволеність користувача від роботи з програмою;

– ефективність роботи полягає у забезпеченні точності, функціональної повноти і завершеності при вирішенні та практичній реалізації виробничих завдань на автоматизованому робочому місці активного користувача. Ефективність роботи відображає об’єм затрачених ресурсів у процесі вирішення поставленого завдання;

– розробка інтерфейсу, який призначений для користувача має бути спрямований на показники ефективності автоматизованої інформаційної системи, які доцільно вимірювати як кількісно, так і об’єктивно;

– продуктивність праці. Визначається середньою кількістю вирішених завдань;

– точність роботи – має на увазі кількість помилок. Показник точності враховує відсоткове відношення помилок та похибок, які зробив користувач у процесі роботи: кількісні показники помилок набору, кількість неправильних та некоректних звернень до даних, запитів тощо;

– функціональна повнота. Окреслюється тим, в якому розмірі розроблений програмний модуль користувачем, відповідає заявленим до нього вимогам; окреслює ступінь застосування первинних і трансформованих даних, переліку необхідних процедур опрацювання звітів, кількість пропущених технологічних операцій або етапів при виконанні поставленого користувачеві завдання;

– завершеність роботи. Виявляє відповідний рівень виконання виробничого завдання посереднім користувачем за окреслений термін або період, відсоткову частку необроблених заявок, кількість продукції у відсотковому співвідношенні, що знаходиться на проміжній стадії готовності, а також число користувачів, які виконали завдання у фіксований термін;

– простота освоєння. Визначається часом освоєння інтерфейсу програмного забезпечення.

Вимоги до зручності і комфортності інтерфейсу зростають зі збільшенням складності робіт і відповідальності користувача за кінцевий результат. Значний рівень ефективності та комфортності від роботи програмного продукту досягається у випадку:

– наявності прозорої навігації для користувача і цільової орієнтації в програмному модулі. Досить важливим моментом є визначення питання покроковості виконання;

– чіткості та ясності розуміння користувачем текстів, символів та значення кнопок чи іконок. У програмному модулі повинні бути використані такі символи та графічні образи, які користувач знає або повинен знати за характером його роботи або займаної посади;

– наявності допоміжних методів та способів підтримки користувача, враховуючи і процес прийняття рішення в невизначеній ситуації.

Зручний та простий інтерфейс допомагає користувачеві впоратися з втомою і напругою при роботі в умовах високої відповідальності за результат. Розробка користувальницького інтерфейсу ведеться паралельно розробці програмного продукту в цілому і в основному передує його впровадженню. Процес розробки програмного інтерфейсу умовно можна диференціювати на такі етапи:

– аналіз діяльності потенційних користувачів;
– формулювання вимог до роботи користувача;
– побудова користувальницької моделі даних.

3.4 Вибір програмного середовища та мови програмування для розробки графічного інтерфейсу

C# це об’єктно-орієнтована мова програмування з безпечною системою типізації для платформи *.NET*. Розроблена Андерсом Гейлсбергом, Скотом Вілтамутом та Пітером Гольде під егідою *Microsoft*. Синтаксис *C#* схожий із *Java* та *C++* . Мова має строгу статичну типізацію, підтримує поліморфізм, перевантаження операторів, вказівники на функції-члени класів, атрибути, події, властивості, винятки, коментарі у форматі *XML*. Рейтинг використання мов програмування зображено на Рисунок 3.4

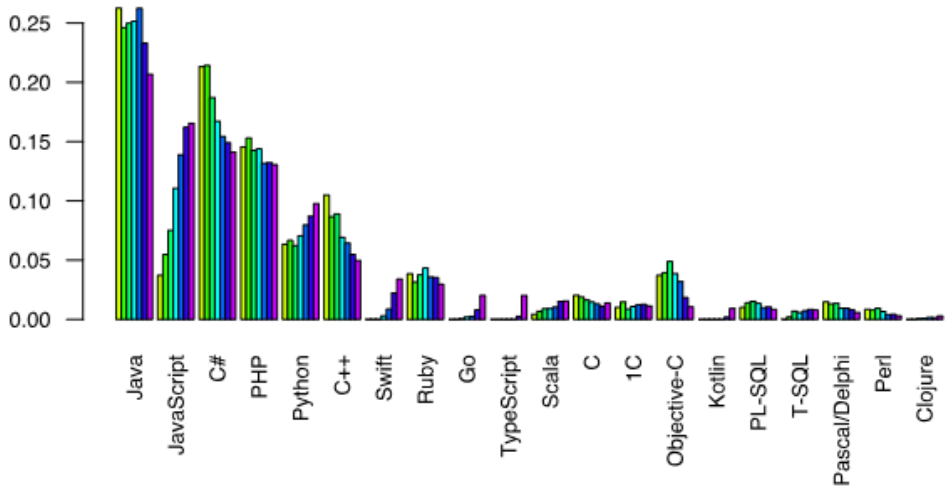

Рисунок 3.4 – Рейтинг мов програмування

Мова програмування *C#* здебільшого застосовується для формування корпоративного програмного забезпечення, фінансових проектів, наприклад для бірж та банків , зокрема хмарних сервісів мобільних додатків [16, с. 153–154].

На відміну від *Java C#* легше піддається взаємодії із кодом програм, які написані на інших програмних мовах. Варто також зазначити, що на *C#* досить часто пишуться розширення для інших мов програмування, що застосовуються в якості прошарку між бібліотекою *C#* і мовою, можливості якої планується розширити (під конкретні цілі).

C# досить часто застосовується в розробці ігор на *Unity* – одному із найпоширеніших ігрових двигунів. Отже, можна зробити висновок, що за допомогою *C#* створювалися сотні ігор, включаючи найпопулярніші. *C#* також чудово підходить під роботу системою *embedded*. *Embedded system* – спеціалізована комп'ютерна система, призначена для виконання обмеженої кількості функцій [16, с. 170].

На відміну від інших мов програмування, *C#* є високорівневою, що означає, що вона в деякій мірі має схожі риси із англійською. Мова програмування *C#* має строгу статичну типізацію, підтримує поліморфізм, перевантаження операторів, вказівники на функції-члени класів, атрибути, події, властивості, винятки, коментарі у форматі *XML*.

Microsoft Visual Studio (Рисунок 3.5) – серія продуктів компанії *Microsoft*, які включають інтегроване середовище розробки програмного забезпечення та перелік інших інструментальних засобів. Ці продукти дають можливість розробляти консольні програми, програми з графічним інтерфейсом, в тому числі з підтримкою технології *Windows Forms*, а також Веб-сайти, Веб-застосунки тощо.

Рисунок 3.5 – Програмне середовище *Visual Studio*

За допомогою платформи *.NET* для створення графічних інтерфейсів на практиці використовуються різні технології – *Window Forms, WPF*, додатки для магазину *Windows Store*. Однак найпростішою та найбільш зручною платформою досі залишається *Window Forms*.

Windows Forms використовується в *Microsoft .NET* для створення додатків, які доповнені графічним інтерфейсом. Ґрунтується він на *.NET Framework class*

library і має більш зручну та досконалу в роботі модель програмування, ніж, для прикладу, програмні інтерфейси *Win32 API* або *MFC*.

Тому, що *Windows Forms* по суті повинна включати сотні організованих класів, щоб забезпечувати всі необхідні можливості розробнику, **.***NET Framework* розбита на ряд ієрархічних розділів, що мають свої імена. *System* – виступає ключовим розділом, і призначений для опису основоположних типів даних. Приклад розробленої програми за допомогою мови програмування C# та бібліотеки *Windows Forms*, зображено на рисунку 3.6.

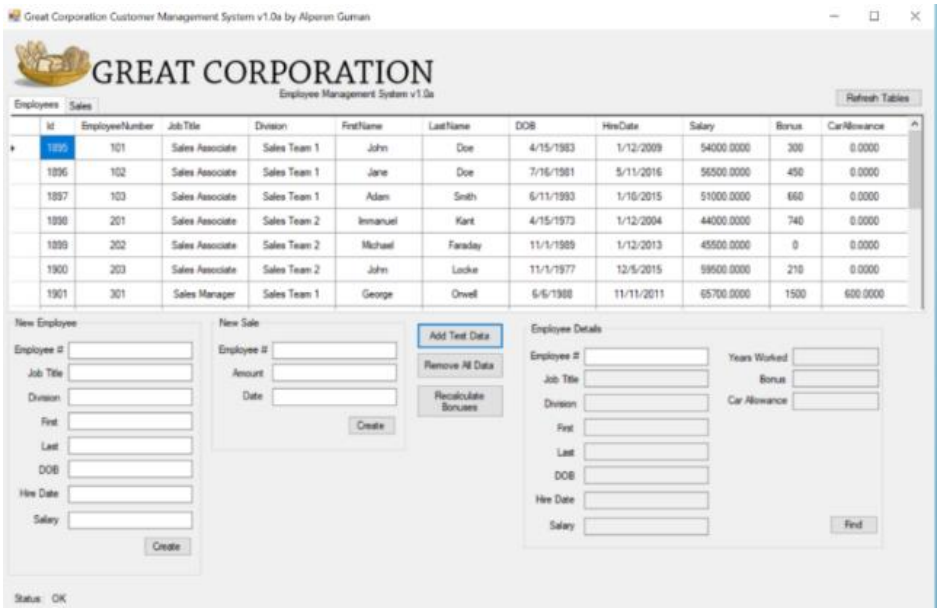

Рисунок 3.6 – Приклад розробленого програмного модулю

3.5 Посібник користувача

Для роботи користувача з розробленою програмною системою тестування та контролю знань учнів потрібні мінімальні потужності апаратного забезпечення:

– *IBM*-сумісний комп’ютер із процесором класу *Pentium-III* з тактовою частотою 200 *MHz*;

– обсяг оперативної пам’яті (*RAM*) – 128МБ;

– операційна система *Microsoft Windows* 7/8/10;

– 10МБ дискового простору;

– монітор *SVGA* (800 x 600), клавіатура.

Для встановлення системи потрібна наявність таких системних засобів:

– додаток *Microsoft Visual Studio* 2017/2018/2019;

– сервер БД *MySQL Server*;

– веб-додаток *phpMyAdmin*.

Забезпечивши переліком згаданим обладнанням та програмними засобами, користувач без будь-яких зусиль введе в експлуатацію розроблений програмний модуль, функціонал якого описаний у роботі далі.

3.6 Висновки до розділу 3

Підсумками третього розділу стало розроблення архітектури майбутнього програмного модуля, задля реалізації поставленого завдання була обрана каскадна модель та побудована схема каскадного процесу розробки програмного модуля, а також побудована функціональна модель програмного продукту.

У третьому розділі магістерської роботи було обрано мову програмування, наведено підстави її вибору, наведено її переваги та недоліки, а також обрано програмне середовище розробки програмного модуля для контролю знань учнів та тестування. Також було визначено потрібні мінімальні потужності апаратного забезпечення.

4 РЕАЛІЗАЦІЯ ТА АПРОБАЦІЯ ПРОГРАМНОГО ПРОДУКТУ

4.1 Реалізація бази даних та розробка графічного інтерфейсу

Процес розробки системи тестування та контролю знань учнів структурно можна поділити на три основні етапи:

- аналітична підготовка;
- програмування та тестування;
- документування.

Перший етап вже успішно виконано у попередніх розділах. Другий етап уособлює собою безпосередньо процес розробки, а саме:

- написання коду;
- налагодження;
- тестування;
- конфігурування.

Результатом завершення цього етапу має бути готовий. програмний модуль, готовий до використання. Під час другого етапу можливе тимчасове повернення до аналітичного етапу для вирішення архітектурних проблем, виявлених, на жаль, лише при написанні коду або тестуванні. Насамкінець залишається лише завершальний етап – документування .

Для реалізації зручного та ефективного програмного модуля необхідно визначитися із завданнями, які вирішуватиме дана система. Головні завдання будуть такі:

– безпечна авторизація в системі;
– додавання, редагування та видалення даних у відповідні таблиці бази даних;
– ведення ефективного пошуку необхідних даних користувачу залежно від запитів;
– створення можливості зручного перегляду існуючих даних у базі даних;

– реалізація можливості експорту даних у фали *excel*.

Перший етап – це проектування сховища даних, що необхідне роботи цієї системи. Для створення бази даних виконуємо вхід до сервісу *phpMyAdmin* за допомогою розгорнутого сервера, як показано на рисунку 4.1.

Рисунок 4.1 – Виконання входу до *phpMyAdmin*

За допомогою зручного графічного інтерфейсу створюємо необхідну нам базу даних під назвою *db_student* та визначаємо кодування *utf8_general_ci* для роботи з кирилицею. Надалі створена нами база даних буде в лівій частині головної сторінки. Для виконання всіх маніпуляцій із даними створюємо такі таблиці:

– *tb_discipline*
– *tb_progress*
– *tb_student*
– *tb_teacher*

Процес створення всіх таблиць досить схожий (рисунок 4.2). У кожній таблиці обов’язково створюється первинний ключ, який не може бути

негативним, і навіть автоматично додається. Для кожного поля визначається тип даних, який там зберігатиметься, довжина для типу *varchar*, а також кодування *utf8_general_ci* для роботи із кирилицею. Також для кожної таблиці та полів та їх визначається тип зберігання даних *MyISAM*, який надає велику перевагу у виконанні запитів на виведення та створення нових даних [18].

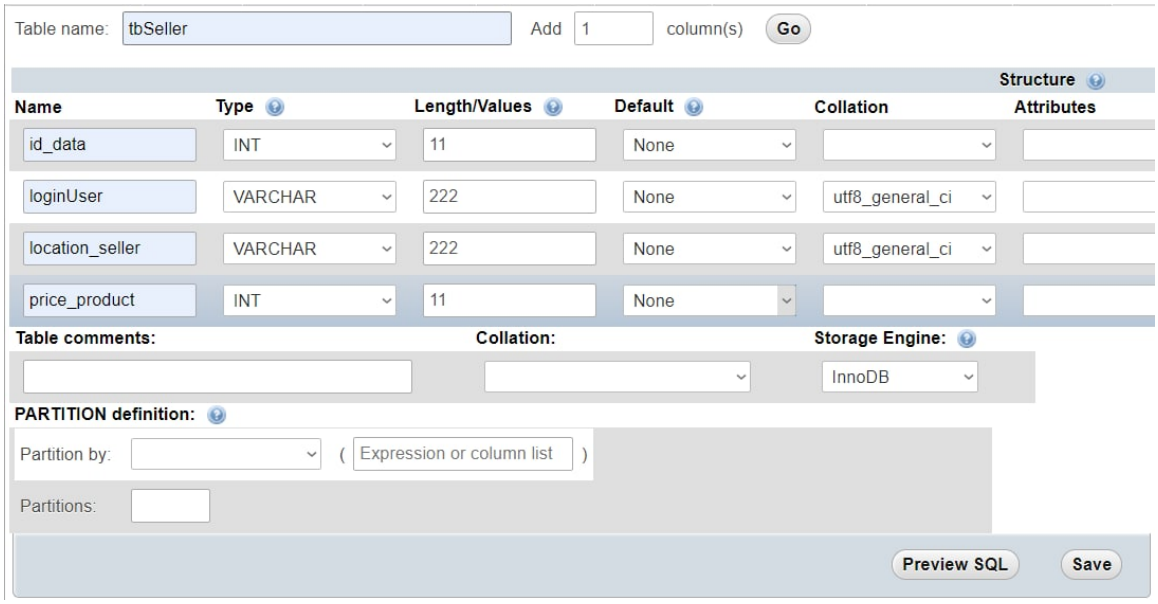

Рисунок 4.2 – Приклад створення таблиці

Для реалізації оптимального рішення та створення оптимальної структури бази даних було створено таблиці, розглянуті у роботі далі.

Таблиця Предмети (*tb_discipline*) матиме наступні атрибути: *id_discipline* – код предмету, *name_discipline* – назва предмету, *id_teacher* – код вчителя, *count_exam* – кількість контрольних робіт з предмету.

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
☐	1	id_discipline	int(11)			No	None		AUTO_INCREMENT
☐	2	name_discipline	varchar(100)	utf8_general_ci		No	None		
☐	3	id_teacher	int(12)			No	None		
☐	4	count_exam	int(11)			No	None		

Рисунок 4.3 – Таблиця Предмети

Таблиця Успішність *(tb_progress*) буде мати такі атрибути: *id_progress* номер успішності, *student* – учень, *discipline* – предмет, *score* – оцінка.

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
☐	1	id_progress	int(11)			No	*None*		AUTO_INCREMENT
☐	2	student	int(11)			No	*None*		
☐	3	discipline	int(11)			No	*None*		
☐	4	score	int(11)			No	*None*		

Рисунок 4.4 – Таблиця Успішність

Таблиця Школярі (*tb_student*) буде мати такі атрибути: *id_student* – код школяра, *date_student* – П.І.Б. школяра, *number_class* – номер класу, *date_birth* – день народження учня, *place_birth* – місце народження та проживання, *hobby* – хобі учня.

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
☐	1	id_student	int(11)			No	*None*		AUTO_INCREMENT
☐	2	date_student	varchar(150)	utf8_general_ci		No	*None*		
☐	3	number_class	varchar(5)	utf32_general_ci		No	*None*		
☐	4	date_birth	varchar(70)	utf8_general_ci		No	*None*		
☐	5	place_birth	varchar(100)	utf8_general_ci		No	*None*		
☐	6	hobby	varchar(150)	utf8_general_ci		No	*None*		

Рисунок 4.5 – Таблиця Школярі

Таблиця Вчителі (*tb_teacher*) буде мати такі атрибути: *id_teacher* – код вчителя, *name_teacher* – П.І.Б. вчителя, *date_birth* – дата народження вчителя, *specialization* – спеціалізація (предмет).

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
☐	1	id_teacher	int(11)			No	*None*		AUTO_INCREMENT
☐	2	name_teacher	varchar(150)	utf8_general_ci		No	*None*		
☐	3	date_birth	varchar(10)	utf8_general_ci		No	*None*		
☐	4	specialization	varchar(50)	utf8_general_ci		No	*None*		

Рисунок 4.6 – Таблиця Вчителі

Для орієнтації між формами створено головне меню, яке знаходиться зверху кожної з форм. Натискаючи на назву необхідної форми, користувач потраплятиме на форму обраної форми. На кожній формі реалізована можливість виконувати маніпуляції з даними за допомогою широкого функціоналу, розташованого на панелі управління. Також на кожній формі знаходиться поля для введення даних, щоб здійснити додавання та редагування даних. Значний розмір кожної форми займає *dataGridViev* – спеціальний елемент, необхідний відображення всіх даних у даній таблиці. Знизу кожної форми знаходиться область для пошуку необхідних для користувача даних. Вигляд головного вікна зображено на рисунку 4.7.

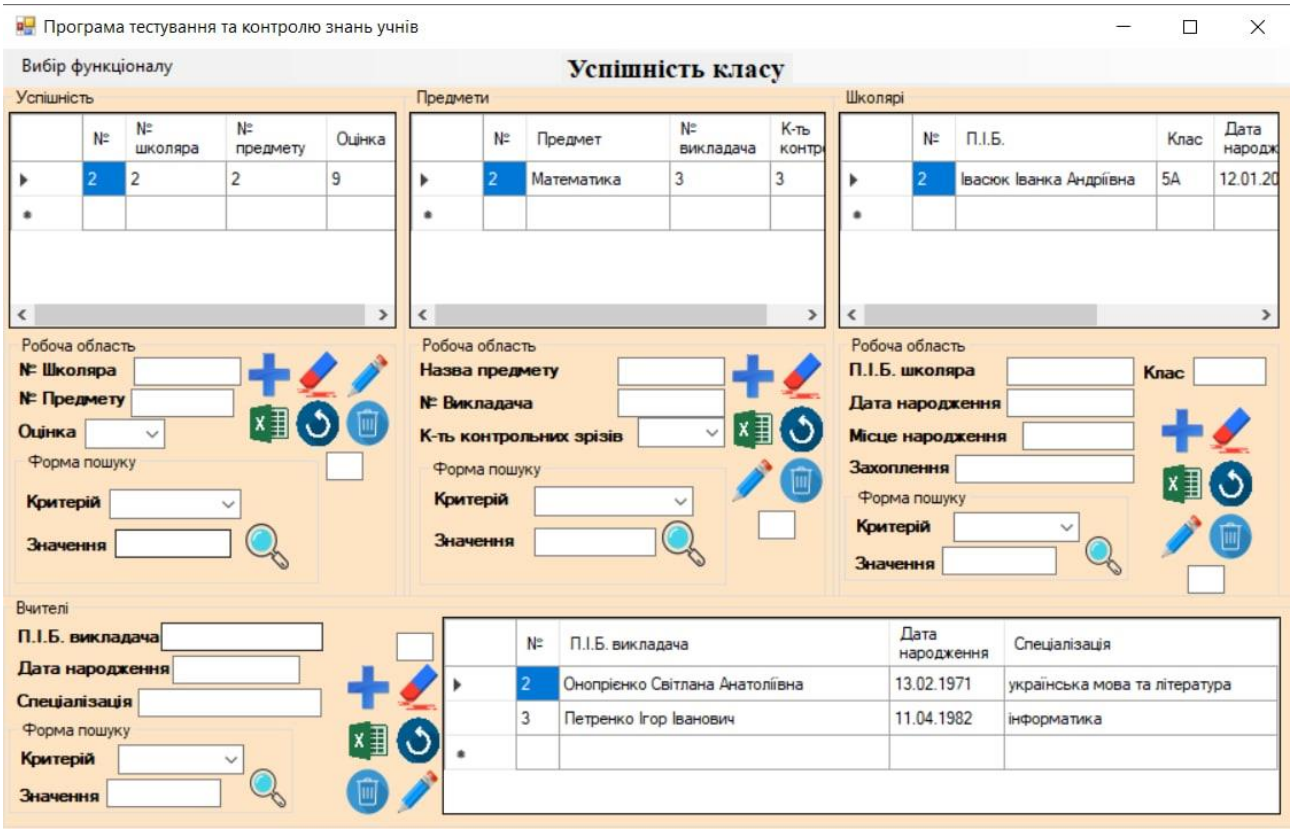

Рисунок 4.7 – Головне вікно

Для додавання нових даних користувач заповнює всі поля (при введенні не всіх даних, користувач отримає попередження та дані не будуть додані) і натискає клавішу із зображенням плюса на блакитному тлі. Щоб перевірити

наявність змін у таблиці користувачеві необхідно натиснути клавішу кругової стрілочки на блакитному фоні, і здійсниться оновлення даних із таблиці в *dataGridViev*.

Для того, щоб ввести зміни в наявних даних у таблиці, користувач заповнює всі необхідні поля і натискає клавішу із зображенням блакитного олівця. Для комфортного редагування користувачеві необхідно лише натиснути на необхідний запис у таблиці (*dataGridViev*) та необхідні записи з’являться у відповідних полях.

Для того щоб стерти помилково введені дані або значення, потрібно виділити такий запис і натиснути на клавішу гумки. Для видалення необхідного запису, користувачеві необхідно ввести номер та натиснути клавішу із зображенням кошика на червоному тлі.

У цьому програмному модулі реалізовано можливість формування файлів *excel*. Ця можливість може бути корисною для надсилання звітів іншим користувачам. Для цього адміністратор системи має активну форму натиснути на кнопку із зображення *excel* і вибрати шлях до збереження та вказати назву майбутнього файлу (рисунок 4.8).

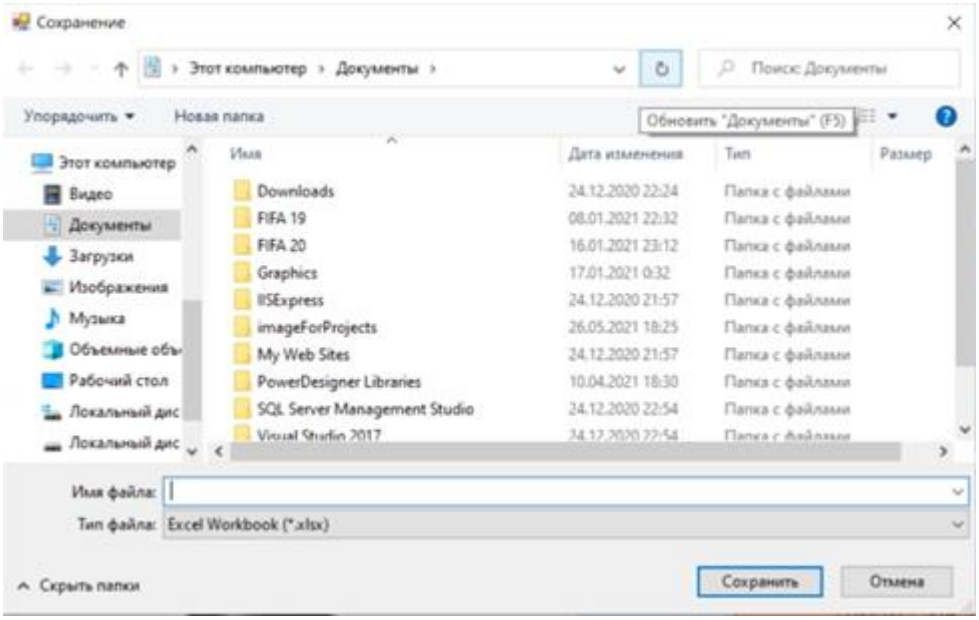

Рисунок 4.8 – Формування файлів *excel*

Для пошуку необхідних даних користувачеві необхідно в області пошуку вибрати критерії пошуку і ввести значення за яким необхідно знайти дані (рисунок 4.8-4.9).

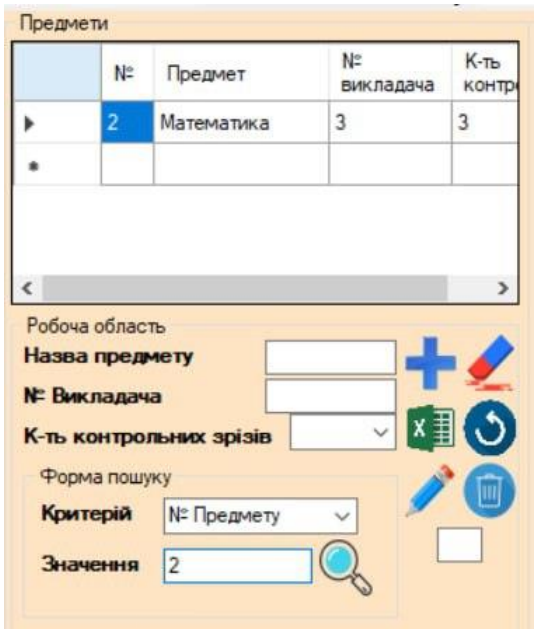

Рисунок 4.8 – Пошук за критеріями

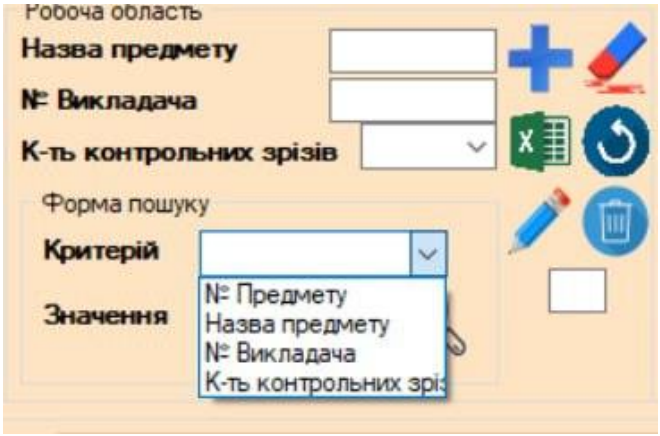

Рисунок 4.9 – Критерії пошуку

4.2 План випробувань

Для досягнення максимальної якості тестування, я вважаю за доцільне обрати найоптимальніший та найзручніший метод або спосіб, який у повній мірі забезпечить потрібний результат. Разом з тим варто обрати балансове

співвідношення між тим, що буде витрачено на тестування і якістю тестування. Це означає, що процес тестування має проходити найменш збитково, однак охоплювати якнайповніше функціональність системи [10].

Тестування програмного модулю можна ідентифікувати як:

– процес дослідження програмного забезпечення з метою одержання інформаційних даних щодо якості фінального продукту;
– процес перевірки відповідності заявлених до продукту вимог і реально реалізованої функціональності, яка відбувається, в штучно створених ситуаціях і на обмеженому наборі тестів, шляхом спостереження за його роботою.

Для того, щоб провести повне і всебічно тестування відповідності програмного продукту поставленим цілям, складений план тестування:

– перевірка системи на належну авторизацію;
– перевірка системи на можливість реєстрації нового користувача ;
– перевірка системи на перегляд навчальної літератури;
– перевірка програмного модуля на проходження тестування після засвоєння навчального матеріалу;
– тестування програмного модуля на можливість роботи у двох режимах (звичайний та преміум).

4.3 Перевірка виконання функціональних вимог

Необхідно звірити програмний продукт із заявленими вимогами. Для перевірки працездатності функцій створення та додавання нових даних виконуємо маніпуляції, зазначені в розділі раніше, вносимо необхідну інформацію та натискаємо необхідну комбінацію клавіш та отримуємо потрібний результат (рисунок 4.10).

Для того щоб внести зміни набираємо потрібний критерій, який потрібно змінити і натискаємо необхідну кнопку (рисунок 4.11)

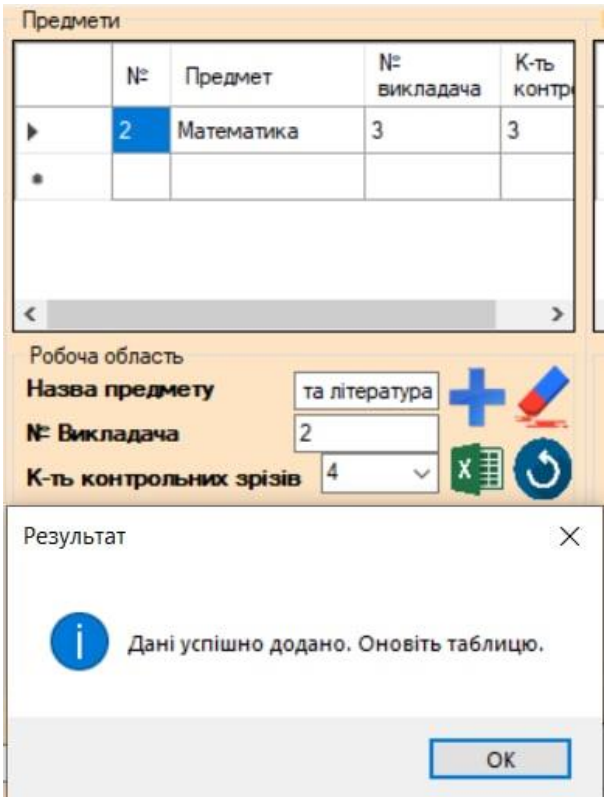

Рисунок 4.10 – Додавання нових даних

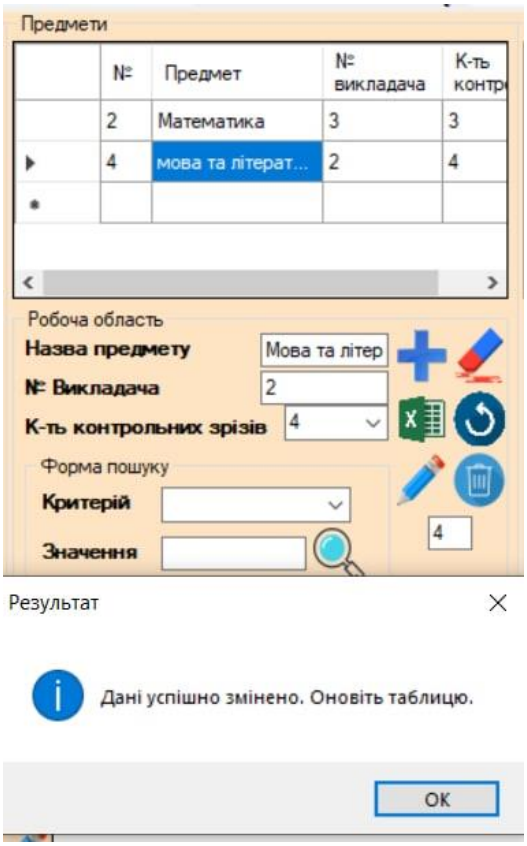

Рисунок 4.11 – Зміна даних

Видалити інформацію можна за допомогою кнопки кошика, як було згадано раніше. Перевіряємо працездатність цієї функції (рисунок 4.12)

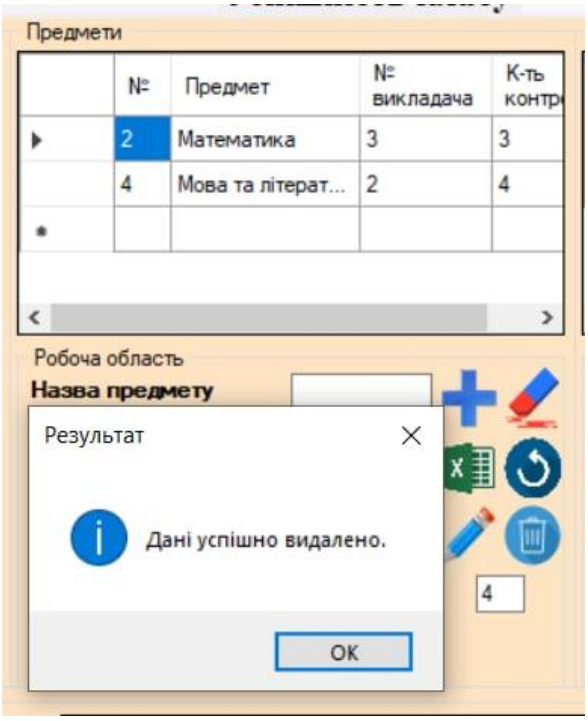

Рисунок 4.12 – Видалення даних

4.4 Демонстрація можливостей системи

Для користувача у програмному модулі передбачений широкий функціонал. Школяр може переглядати навчальну літературу, де знайомитися із необхідними літературними джерелами, підручниками та навчальними посібниками, знайомитися із методичними матеріалами, переглядами відеоуроки, проходити тестування з пройденого матеріалу для кращого засвоєння навчального матеріалу, а також передбачена можливість перегляду рейтингу своєї успішності.

Одним із наявних функціональних можливостей розробленої інформаційно-довідкової системи є пошук необхідної навчальної літератури (рисунок 4.13). Користувачеві просто потрібно вибрати авторів та назву літератури і праворуч буде виведено короткий опис цієї літератури. Якщо

користувач знайшов, що саме він хотів, він може відкрити цю навчальну літературу.

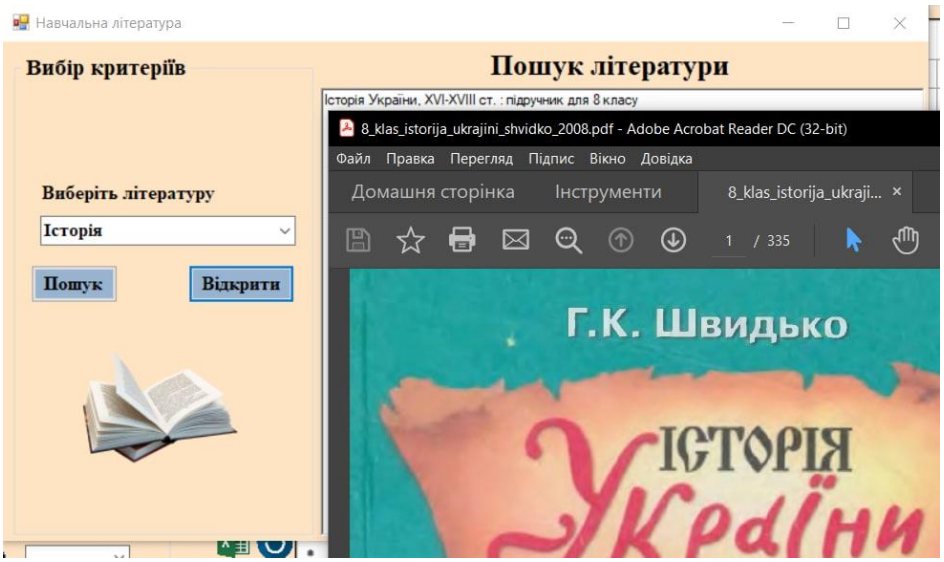

Рисунок 4.13 – Пошук навчальної літератури

Також було розроблено можливість перегляду методичних розробок відповідно до номера теми та занять, які користувач вибирає. Перевіряє короткий опис методички і відкриває її за необхідністю. Приклад виведення методичних розробок показано на рисунку 4.14.

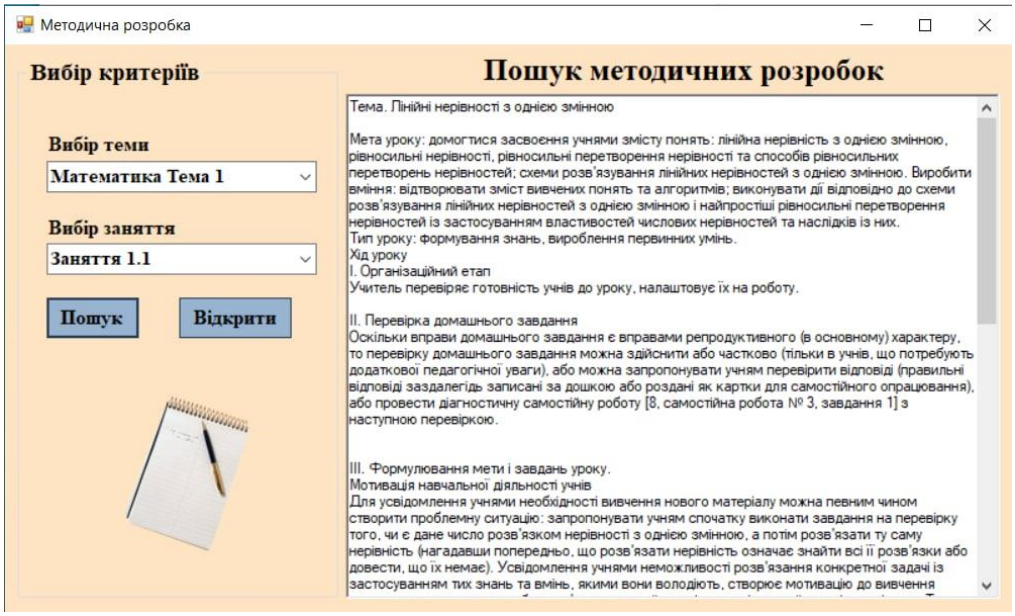

Рисунок 4.14 – Приклад пошуку методичних розробок

Для реалізації перегляду наукових відео в інформаційно-довідковій системі було додано компонент *Windows Media Player*, який відповідає за перегляд відео у розробленому програмному додатку. Щоб розпочати перегляд відео (рисунок 4.15), користувачеві необхідно вибрати необхідне відео за допомогою звичайного файлового шукача. Щоб почати перегляд відео, потрібно натиснути на клавішу "Переглянути".

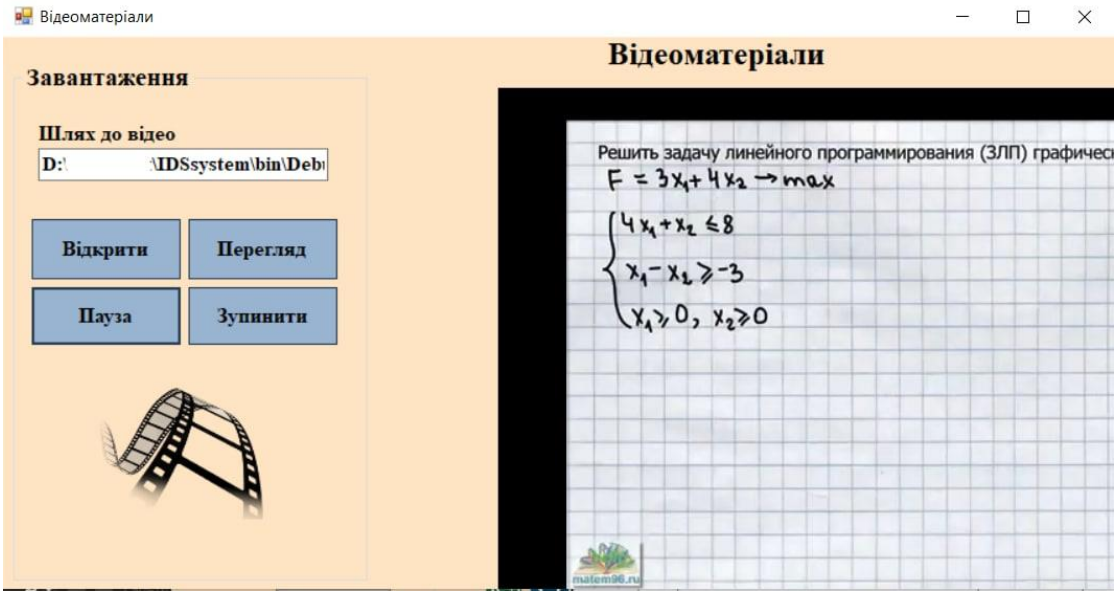

Рисунок 4.15 – Перегляд відеоматеріалів

Останнім розробленим функціоналом даної інформаційно-довідкової системи стало проходження інтерактивного тесту, щоб перевірити якість засвоєння матеріалу. Відповідно до номера запитання користувач може вибрати одну відповідь, якщо вона буде правильною, то він отримує 5 балів до своєї загальної оцінки. Спосіб перевірки якості засвоєння матеріалу показано на рисунку 4.16.

У програмному модулі розроблено тестування із 4 навчальних дисциплін: Математика, Українська мова, Географія та Історія України. Кількість отриманих балів з однієї дисципліни сумується та заноситься до таблиці успішності.

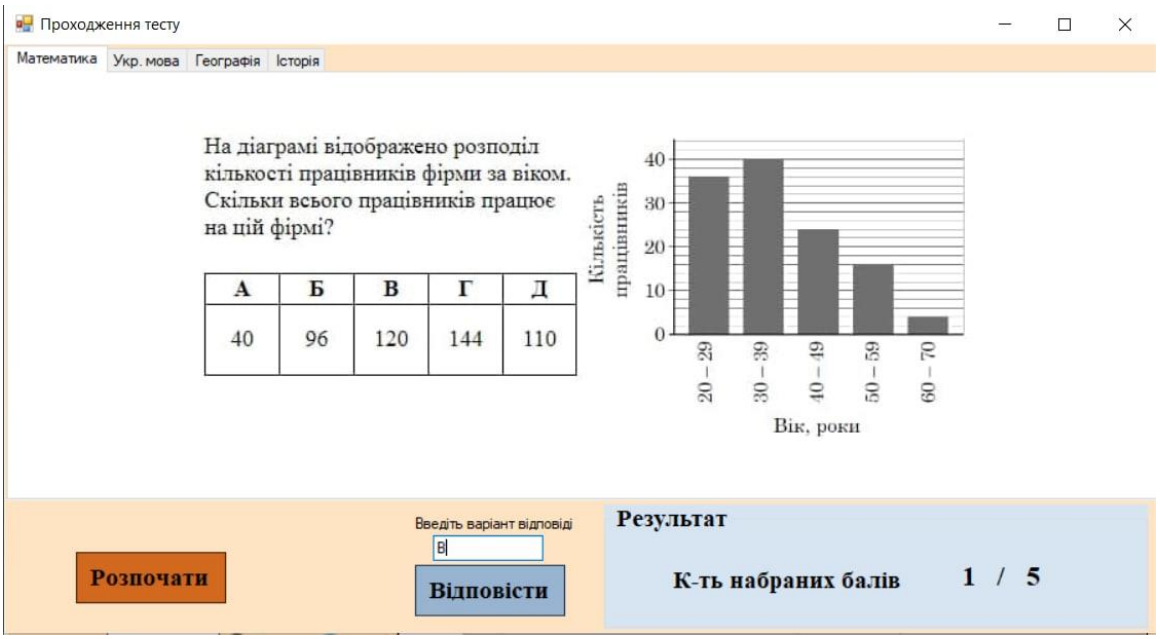

Рисунок 4.16 – Тестування

4.5 Висновки до розділу 4

Таким чином, розроблений програмний модуль для тестування та контролю знань учнів повністю відповідає висунутим вимогам. Дана інформаційна система значно полегшує навчальний процес школярам та допомагає у кращому та ефективнішому засвоєнні пройденого матеріалу.

На підставі результатів тестування, можна зробити висновок, що система повністю відповідає вимогам технічного завдання, стабільна і коректно реагує на всі дії користувача.

ВИСНОВКИ

На сучасному етапі існує необхідність подальшого впровадження інформатизації в вітчизняну освіту. Потрібно розробляти і впроваджувати більшу кількість педагогічних програмних засобів навчання з усіх навчальних предметів. На сьогоднішній день надзвичайно актуальним постає питання розробки автоматизованих інформаційних систем для організації навчального процесу та належного контролю вивченого матеріалу шляхом тестування, аргументуючи це тим, що в умовах сьогодення учні не отримують достатнього рівня знань дня належної підготовки до фахових іспитів

Дипломна робота присвячена вивченню питань програмного модуля для належної організації навчального процесу та подальшої перевірки пройденого матеріалу шляхом тестування. Інформаційна система складена на основі тематичного плану курсу дисциплін за навчальний період і створена для поліпшення якості засвоєння курсу дисципліни та належного контролю отриманих знань шляхом проходження тестування.

У роботі вирішено питання, чому саме ця тема актуальна та заслуговує бути розглянутою, також проведений аналіз актуальності інформаційних-технологій у сучасному світі, у навчальному процесі зокрема, та доцільність розробки програмного модуля для належної підготовки школяра до навчальних предметів, а також для вчителів з метою ефективного контролю опрацьованого навчального матеріалу.

Проводився аналіз технологій розробки, а саме доцільних мов програмування, аналіз навчальних методів, а також обґрунтування вибору технології розробки програмного модуля. Набуті знання допомогли створити програмний модуль для належного тестування та контролю знань учнів. Головним результатом роботи, за допомогою потужної мови програмування третього покоління – *C#* та найефективнішої СКБД *MSSQL,* виступає реалізація програмного модуля для виконання маніпуляцій з даними.

Отже, цілі і завдання моєї кваліфікаційної роботи було чітко досліджені і виконано. Розроблений програмний модуль має зрозумілий і зручний функціонал і зрозуміло кожному. Я вважаю, що дана програма зменшить час, витрачений вручну на пошук інформації, необхідної для проходження належної підготовки учня до навчальних предметів, та належного контролю пройденого матеріалу шляхом тестування.

Розроблений програмний продукт задовольняє технічним вимогам споживачів, а його впровадження на ринку є економічно доцільним.

ПЕРЕЛІК ПОСИЛАНЬ

1. Абдрахманова, Б.А. Інформаційні технології в освіті, URL: http://www.zkoipk.kz/b2/369-conf.html (дата звернення: 30.09.2021).

2. Бази даних та види архітектур інформаційних систем, URL: http://libraryno.ru/5-bazy-dannyh-i-vidy-arhitektur-ekonomicheskih-informacionnyh-sistem-2015_bd/ (дата звернення: 30.09.2021).

3. Басюк Т.М. Принципи побудови системи аналізу та просування інтернет-ресурсів: вісник Нац. ун-ту «Львівська політехніка», Львів, 2012. 12 с.

4. Бербец В. В. Поєднання форм і методів контролю навчальних досягнень учнів в процесі проектно-технологічної діяльності. Психолого-педагогічні проблеми сільської школи: науковий збірник, Київ, 2014. 20 с.

5. Бонч-Бруєвич Г.Ф., Абрамов В.О., Методика застосування Смарт-технології у навчальному процесі : навчальний посібник, Київ: КМПУ імені Б.Д. Грінченка, 2016. 102 с.

6. Бублик В.В. Об’єктно-орієнтоване програмування: підручник. К.: ІТ-книга, Київ, 2015. 624 с.

7. Булатецький В. В., Булатецька Л.В. Технології проміжного коду в корпоративних інформаційних системах: Текст лекцій нормативної навчальної дисципліни «Платформи корпоративних інформаційних систем», Луцьк: СНУ імені Лесі Українки, 2018. 48 с.

8. Вембер В. П. Роль та місце електронного підручника в навчально-методичному комплекті з навчального предмета для загальноосвітньої школи: збірник наукових праць, Київ: Інститут психології ім. Г. С. Костюка, 2019. 54 с.

9. Гришанович Т. О. Основи об’єктно-орієнтованого програмування: навч. посіб. Харків : ФОП Панов А.М., 2020. 104 с.

10. Гуржій А. М. Про проблеми наступності навчання інформаційних технологій у школі й вищому педагогічному навчальному закладі: збірник наукових праць.15 видання, Херсон, 2013. 19 с.

11. Гуревич Р. С. Інформаційно-комуникаційні технології у освіті : монографія, Вінниця, 2016. 341 с.

12. Дементієвська Н.П., Морзе Н.В. Комп'ютерні технології для розвитку учнів та вчителів, Київ: Інститут засобів навчання АПН України, 2016. 15 с.

13. Дібрівний О.А., Гребенюк В.В. Вступ до об'єктно-орієнтованого програмування C#: навч. посіб, Київ: Державний університет телекомунікацій, 2018. 190 с.

14. Кадемія М. Ю. Освіта в інформаційному суспільстві: монографія, Вінниця: НАПН України Гуржія А. М, 2016. 412 с.

15. Карташова Л. А. Електронний освітній ресурс як засіб підтримки навчання інформаційних технологій майбутніх філологів : зб. наук. праць, Київ, 2014. 427 с.

16. Коноваленко І.В. Програмування мовою C# 6.0: навчальний посібник, Тернопіль, 2016. 229с.

17. Курліщук І.І. Методика професійного навчання: Основи технології навчання. Креативні технології навчання: навчально-методичний посібник, Старобільськ, 2017. 120 с.

18. Лапінський В. В. Проектування електронних засобів навчання з урахуванням проблем управління навчальним процесом: зб. наук, праць, Київ, 2013. 866 с.

19. Моделювання систем: навч. посіб., Черкаси, 2017. 399 с.

20. Мелешко Є.В., Якименко М.С., Поліщук Л.І. Алгоритми та структури даних: навч. посіб., Кропивницький, 2019. 156 с.

21. Лебеденко М. С. Вебометричний ранг як показник ефективності електронного ресурсу підприємства: економічний вісник, Київ: КПІ, 2014. 8 с.

22. Онищенко В. В., Довженко Т. П. Спеціалізовані мови програмування: навчальний посібник, Київ, 2019. 146 с.

23. Про впровадження пілотного проекту «Learnin – SMART навчання”/ Наказ МОН №812 від 12.07.12 року. URL: http://osvita.ua (дата звернення: 20.07.2021).

24. Раскін Д. Інтерфейс: Нові напрями в проектуванні комп’ютерних систем: навч. посіб., Львів, 2014. 272 с.

25. Семеніхіна О.В. Нові парадигми у сфері освіти в умовах переходу до SMART-суспільства. URL: http://irbis-nbuv.gov.ua (дата звернення: 15.08.2021).

26. Чакраборти А. Microsoft .NET Framework. Розробка професійних проектів: навч.посіб., Київ, 2013. 896 с.

27. Тхір І.Л., Галушка В.П., Юзьків А.В. Посібник користувача ПК: навч. посіб., Тернопіль, 2017. 718 с.

28. Яцишин В. Технологія забезпечення якості процесу розробки вимог до програмного забезпечення: навч. посіб., Львів, 2018. 280 с.

29. Entity Framework [Електронний ресурс] / Документація Microsoft. — 2020, URL: https://en.wikipedia.org/wiki/Entity_Framework (дата звернення: 27.06.2021).

30. C# 7.0 in a Nutshell: The Definitive Reference: навч. посіб., Каліфорнія, 2021. 1007 с.

31. Smart-технології у вищій освіті. URL: http://library.fa.ru/exhib.asp?id=199 (дата звернення: 24.02.2021).

32. Smart-технології змінять систему освіти. URL: http://www.trainings.ru/library/education_experience/?id=14024 (дата звернення: 24.02.2021).

ДОДАТОК А

Програмний код системи тестування

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.OleDb;

namespace IDSsystem
{
 public partial class Form1 : Form
 {

 public Form1()
 {

 InitializeComponent();
 dataGridView1.DataSource = DataClass.GetData("SELECT * FROM `tb_progress`");
 dataGridView2.DataSource = DataClass.GetData("SELECT * FROM `tb_discipline`");
 dataGridView3.DataSource = DataClass.GetData("SELECT * FROM `tb_student`");
 dataGridView4.DataSource = DataClass.GetData("SELECT * FROM `tb_teacher`");



 dataGridView1.Columns[0].Width = 30;
 dataGridView1.Columns[0].HeaderText = "№";
 dataGridView1.Columns[1].Width = 70;
```

```
dataGridView1.Columns[1].HeaderText = "№ школяра";
dataGridView1.Columns[2].Width = 70;
dataGridView1.Columns[2].HeaderText = "№ предмету";
dataGridView1.Columns[3].Width = 50;
dataGridView1.Columns[3].HeaderText = "Оцінка";

dataGridView2.Columns[0].Width = 30;
dataGridView2.Columns[0].HeaderText = "№";
dataGridView2.Columns[1].Width = 100;
dataGridView2.Columns[1].HeaderText = "Предмет";
dataGridView2.Columns[2].Width = 70;
dataGridView2.Columns[2].HeaderText = "№ викладача";
dataGridView2.Columns[3].Width = 60;
dataGridView2.Columns[3].HeaderText = "К-ть контрольних";

dataGridView3.Columns[0].Width = 30;
dataGridView3.Columns[0].HeaderText = "№";
dataGridView3.Columns[1].Width = 140;
dataGridView3.Columns[1].HeaderText = "П.І.Б.";
dataGridView3.Columns[2].Width = 40;
dataGridView3.Columns[2].HeaderText = "Клас";
dataGridView3.Columns[3].Width = 80;
dataGridView3.Columns[3].HeaderText = "Дата народження";
dataGridView3.Columns[4].Width = 75;
dataGridView3.Columns[4].HeaderText = "Місце народження";
dataGridView3.Columns[5].Width = 60;
dataGridView3.Columns[5].HeaderText = "Хобі";

dataGridView4.Columns[0].Width = 30;
dataGridView4.Columns[0].HeaderText = "№";
dataGridView4.Columns[1].Width = 230;
dataGridView4.Columns[1].HeaderText = "П.І.Б. викладача";
dataGridView4.Columns[2].Width = 80;
dataGridView4.Columns[2].HeaderText = "Дата народження";
dataGridView4.Columns[3].Width = 190;
```

```
dataGridView4.Columns[3].HeaderText = "Спеціалізація";
}

private void успішністьГрупиToolStripMenuItem_Click(object sender, EventArgs e)
{
Form1 form = new Form1();
form.Show();
}

private void навчальнаЛітератураToolStripMenuItem_Click(object sender, EventArgs e)
{
Documentation documentation = new Documentation();
documentation.Show();
}

private void методичнаРозробкаToolStripMenuItem_Click(object sender, EventArgs e)
{
Metodical metodical = new Metodical();
metodical.Show();
}

private void відеоМатеріалиToolStripMenuItem_Click(object sender, EventArgs e)
{
Video video = new Video();
video.Show();
}

private void проходженняТестуToolStripMenuItem_Click(object sender, EventArgs e)
{
Testing testing = new Testing();
testing.Show();
}

private void pictureBox1_Click(object sender, EventArgs e)
{
```

```
 if (textBox1.Text == "" || textBox3.Text == "" || textBox5.Text == "")
 {
 MessageBox.Show("Введено пропущені дані!", "Помилка", MessageBoxButtons.OK,
MessageBoxIcon.Warning);
 return;
 }
 try
 {
 DataClass.GetData($"INSERT INTO `tb_progress`(`student`, `discipline`, `score`) VALUES
({textBox1.Text},{textBox3.Text},{textBox5.Text})");
 MessageBox.Show("Дані успішно додано. Оновіть таблицю.", "Результат",
MessageBoxButtons.OK, MessageBoxIcon.Information);
 }
 catch (Exception ex)
 {
 MessageBox.Show(ex.Message, "Помилка!", MessageBoxButtons.OK, MessageBoxIcon.Error);
 }

 textBox1.Text = "";
 textBox3.Text = "";
 textBox5.Text = "";
 DB.closedConn();
 }

 private void pictureBox2_Click(object sender, EventArgs e)
 {
 textBox1.Text = "";
 textBox3.Text = "";
 textBox5.Text = "";
 txtID.Text = "";
 textBox4.Text = "";
 comboBox1.Text = "";
 }

 private void pictureBox3_Click(object sender, EventArgs e)
```

```
{
SaveFileDialog saveFileDialog = new SaveFileDialog() { Filter = "Excel Workbook|*.xlsx" };
saveFileDialog.ShowDialog();
string path = saveFileDialog.FileName;
Microsoft.Office.Interop.Excel.Application application = new
Microsoft.Office.Interop.Excel.Application();
Microsoft.Office.Interop.Excel.Workbook workbook = application.Workbooks.Add();
Microsoft.Office.Interop.Excel.Worksheet worksheet = workbook.ActiveSheet;


for (int i = 1; i < dataGridView1.RowCount + 1; i++)
{
for (int j = 1; j < dataGridView1.ColumnCount + 1; j++)
{
worksheet.Rows[i].Columns[j] = dataGridView1.Rows[i – 1].Cells[j – 1].Value;
}
}
application.AlertBeforeOverwriting = false;
workbook.SaveAs(path);
application.Quit();
}

private void pictureBox4_Click(object sender, EventArgs e)
{
dataGridView1.DataSource = DataClass.GetData("SELECT * FROM `tb_progress`");
}

private void pictureBox5_Click(object sender, EventArgs e)
{
DataClass.GetData($"UPDATE `tb_progress` SET `student`=
{textBox1.Text},`discipline`={textBox3.Text},`score`={textBox5.Text} WHERE
`id_progress`={txtID.Text}");
MessageBox.Show("Дані успішно змінено. Оновіть таблицю", "Результат",
MessageBoxButtons.OK, MessageBoxIcon.Information);
txtID.Text = "";
```

```
textBox1.Text = "";
textBox3.Text = "";
textBox5.Text = "";
}

private void pictureBox6_Click(object sender, EventArgs e)
{
DataClass.GetData($"DELETE FROM `tb_progress` WHERE `id_progress`={txtID.Text}");
MessageBox.Show("Дані успішно видалено. Оновіть таблицю.", "Результат",
MessageBoxButtons.OK, MessageBoxIcon.Information);
txtID.Text = "";
}

private void dataGridView1_CellClick(object sender, DataGridViewCellEventArgs e)
{
txtID.Text = dataGridView1.CurrentRow.Cells[0].Value.ToString();
textBox1.Text = dataGridView1.CurrentRow.Cells[1].Value.ToString();
textBox3.Text = dataGridView1.CurrentRow.Cells[2].Value.ToString();
textBox5.Text = dataGridView1.CurrentRow.Cells[3].Value.ToString();
}

private void pictureBox7_Click(object sender, EventArgs e)
{
if (comboBox1.SelectedIndex == 0) dataGridView1.DataSource = DataClass.GetData($"SELECT
* FROM `tb_progress` WHERE `id_progress` = {textBox4.Text}");
else if (comboBox1.SelectedIndex == 1) dataGridView1.DataSource =
DataClass.GetData($"SELECT * FROM `tb_progress` WHERE `student` = {textBox4.Text}");
else if (comboBox1.SelectedIndex == 2) dataGridView1.DataSource =
DataClass.GetData($"SELECT * FROM `tb_progress` WHERE `discipline` = {textBox4.Text}");
else if (comboBox1.SelectedIndex == 3) dataGridView1.DataSource =
DataClass.GetData($"SELECT * FROM `tb_progress` WHERE `score` = {textBox4.Text}");
}

private void pictureBox13_Click(object sender, EventArgs e)
{
```

```
if (textBox11.Text == "" || textBox10.Text == "" || textBox9.Text == "")
{
MessageBox.Show("Введено пропущенные данные!", "Ошибка", MessageBoxButtons.OK,
MessageBoxIcon.Warning);
return;
}
try
{
DataClass.GetData($"INSERT INTO `tb_discipline`(`name_discipline`, `id_teacher`,
`count_exam`) VALUES (‘{textBox11.Text}’,{textBox10.Text},{textBox9.Text})");
MessageBox.Show("Дані успішно додано. Оновіть таблицю.", "Результат",
MessageBoxButtons.OK, MessageBoxIcon.Information);
}
catch (Exception ex)
{
MessageBox.Show(ex.Message, "Помилка!", MessageBoxButtons.OK, MessageBoxIcon.Error);
}

textBox11.Text = "";
textBox10.Text = "";
textBox9.Text = "";
DB.closedConn();
}

private void pictureBox12_Click(object sender, EventArgs e)
{
textBox11.Text = "";
textBox10.Text = "";
textBox9.Text = "";
textBox8.Text = "";
textBox7.Text = "";
comboBox2.Text = "";
}

private void pictureBox11_Click(object sender, EventArgs e)
```

```
{
SaveFileDialog saveFileDialog = new SaveFileDialog() { Filter = "Excel Workbook|*.xlsx" };
saveFileDialog.ShowDialog();
string path = saveFileDialog.FileName;
Microsoft.Office.Interop.Excel.Application application = new
Microsoft.Office.Interop.Excel.Application();
Microsoft.Office.Interop.Excel.Workbook workbook = application.Workbooks.Add();
Microsoft.Office.Interop.Excel.Worksheet worksheet = workbook.ActiveSheet;


for (int i = 1; i < dataGridView2.RowCount + 1; i++)
{
for (int j = 1; j < dataGridView2.ColumnCount + 1; j++)
{
worksheet.Rows[i].Columns[j] = dataGridView2.Rows[i – 1].Cells[j – 1].Value;
}
}
application.AlertBeforeOverwriting = false;
workbook.SaveAs(path);
application.Quit();
}

private void pictureBox10_Click(object sender, EventArgs e)
{
dataGridView2.DataSource = DataClass.GetData("SELECT * FROM `tb_discipline`");
}

private void pictureBox9_Click(object sender, EventArgs e)
{
DataClass.GetData($"UPDATE `tb_discipline` SET `name_discipline`=
‘{textBox11.Text}’,`id_teacher`={textBox10.Text},`count_exam`={textBox9.Text} WHERE
`id_discipline`={textBox7.Text}");
MessageBox.Show("Дані успішно змінено. Оновіть таблицю.", "Результат",
MessageBoxButtons.OK, MessageBoxIcon.Information);
textBox7.Text = "";
```

```
 textBox11.Text = "";
 textBox10.Text = "";
 textBox9.Text = "";
 }

 private void pictureBox8_Click(object sender, EventArgs e)
 {
 DataClass.GetData($"DELETE FROM `tb_discipline` WHERE
`id_discipline`={textBox7.Text}");
 MessageBox.Show("Дані успішно видалено.", "Результат", MessageBoxButtons.OK,
MessageBoxIcon.Information);
 textBox7.Text = "";
 }

 private void dataGridView2_CellClick(object sender, DataGridViewCellEventArgs e)
 {
 textBox7.Text = dataGridView2.CurrentRow.Cells[0].Value.ToString();
 textBox11.Text = dataGridView2.CurrentRow.Cells[1].Value.ToString();
 textBox10.Text = dataGridView2.CurrentRow.Cells[2].Value.ToString();
 textBox9.Text = dataGridView2.CurrentRow.Cells[3].Value.ToString();
 }

 private void pictureBox14_Click(object sender, EventArgs e)
 {
 if (comboBox2.SelectedIndex == 0) dataGridView2.DataSource = DataClass.GetData($"SELECT
* FROM `tb_discipline` WHERE `id_discipline` = {textBox8.Text}");
 else if (comboBox2.SelectedIndex == 1) dataGridView2.DataSource =
DataClass.GetData($"SELECT * FROM `tb_discipline` WHERE `name_discipline` =
‘{textBox8.Text}’");
 else if (comboBox2.SelectedIndex == 2) dataGridView2.DataSource =
DataClass.GetData($"SELECT * FROM `tb_discipline` WHERE `id_teacher` = {textBox8.Text}");
 else if (comboBox2.SelectedIndex == 3) dataGridView2.DataSource =
DataClass.GetData($"SELECT * FROM `tb_discipline` WHERE `count_exam` =
{textBox8.Text}");
 }
```

```
private void pictureBox20_Click(object sender, EventArgs e)
{
if (textBox15.Text == "" || textBox14.Text == "" || textBox13.Text == "" || textBox16.Text == "")
{
MessageBox.Show("Введено пропущені дані!", "Помилка", MessageBoxButtons.OK,
MessageBoxIcon.Warning);
return;
}
try
{
DataClass.GetData($"INSERT INTO `tb_student`(`date_student`, `number_class`, `date_birth`,
`place_birth`, `hobby`) VALUES
('{textBox15.Text}','{textBox2.Text}','{textBox14.Text}','{textBox13.Text}','{textBox16.Text}
')");
MessageBox.Show("Дані успішно оновлено. Оновіть таблицю.", "Результат",
MessageBoxButtons.OK, MessageBoxIcon.Information);
}
catch (Exception ex)
{
MessageBox.Show(ex.Message, "Помилка!", MessageBoxButtons.OK, MessageBoxIcon.Error);
}

textBox15.Text = "";
textBox14.Text = "";
textBox13.Text = "";
textBox16.Text = "";
DB.closedConn();
}

private void pictureBox19_Click(object sender, EventArgs e)
{
textBox15.Text = "";
textBox2.Text = "";
textBox14.Text = "";
```

```
textBox13.Text = "";
textBox16.Text = "";
textBox6.Text = "";
textBox12.Text = "";
comboBox3.Text = "";
}

private void pictureBox18_Click(object sender, EventArgs e)
{
SaveFileDialog saveFileDialog = new SaveFileDialog() { Filter = "Excel Workbook|*.xlsx" };
saveFileDialog.ShowDialog();
string path = saveFileDialog.FileName;
Microsoft.Office.Interop.Excel.Application application = new
Microsoft.Office.Interop.Excel.Application();
Microsoft.Office.Interop.Excel.Workbook workbook = application.Workbooks.Add();
Microsoft.Office.Interop.Excel.Worksheet worksheet = workbook.ActiveSheet;


for (int i = 1; i < dataGridView3.RowCount + 1; i++)
{
for (int j = 1; j < dataGridView3.ColumnCount + 1; j++)
{
worksheet.Rows[i].Columns[j] = dataGridView3.Rows[i – 1].Cells[j – 1].Value;
}
}
application.AlertBeforeOverwriting = false;
workbook.SaveAs(path);
application.Quit();
}

private void pictureBox17_Click(object sender, EventArgs e)
{
dataGridView3.DataSource = DataClass.GetData("SELECT * FROM `tb_student`");
}
```

```
 private void pictureBox16_Click(object sender, EventArgs e)
 {
 DataClass.GetData($"UPDATE `tb_student` SET
`date_student`='{textBox15.Text}',`number_class`='{textBox2.Text}',`date_birth`='{textBox14.T
ext}',`place_birth`='{textBox13.Text}',`hobby`='{textBox16.Text}' WHERE
`id_student`={textBox6.Text}");
 MessageBox.Show("Дані успішно змінено. Оновіть таблицю.", "Результат",
MessageBoxButtons.OK, MessageBoxIcon.Information);
 textBox6.Text = "";
 textBox15.Text = "";
 textBox14.Text = "";
 textBox2.Text = "";
 textBox13.Text = "";
 textBox16.Text = "";
 }

 private void pictureBox15_Click(object sender, EventArgs e)
 {
 DataClass.GetData($"DELETE FROM `tb_student` WHERE `id_student`={textBox6.Text}");
 MessageBox.Show("Дані успішно видалено. Оновіть таблицю.", "Результат",
MessageBoxButtons.OK, MessageBoxIcon.Information);
 textBox6.Text = "";
 }

 private void dataGridView3_CellClick(object sender, DataGridViewCellEventArgs e)
 {
 textBox6.Text = dataGridView3.CurrentRow.Cells[0].Value.ToString();
 textBox15.Text = dataGridView3.CurrentRow.Cells[1].Value.ToString();
 textBox2.Text = dataGridView3.CurrentRow.Cells[2].Value.ToString();
 textBox14.Text = dataGridView3.CurrentRow.Cells[3].Value.ToString();
 textBox13.Text = dataGridView3.CurrentRow.Cells[4].Value.ToString();
 textBox16.Text = dataGridView3.CurrentRow.Cells[5].Value.ToString();
 }

 private void pictureBox21_Click(object sender, EventArgs e)
```

```
 {
 if (comboBox3.SelectedIndex == 0) dataGridView3.DataSource = DataClass.GetData($"SELECT
* FROM `tb_student` WHERE `id_student` = {textBox12.Text}");
 else if (comboBox3.SelectedIndex == 1) dataGridView3.DataSource =
DataClass.GetData($"SELECT * FROM `tb_student` WHERE `date_student` =
‘{textBox12.Text}’");
 else if (comboBox3.SelectedIndex == 2) dataGridView3.DataSource =
DataClass.GetData($"SELECT * FROM `tb_student` WHERE `number_class` =
‘{textBox12.Text}’");
 else if (comboBox3.SelectedIndex == 3) dataGridView3.DataSource =
DataClass.GetData($"SELECT * FROM `tb_student` WHERE `date_birth` =
‘{textBox12.Text}’");
 else if (comboBox3.SelectedIndex == 4) dataGridView3.DataSource =
DataClass.GetData($"SELECT * FROM `tb_student` WHERE `place_birth` =
‘{textBox12.Text}’");
 else if (comboBox3.SelectedIndex == 5) dataGridView3.DataSource =
DataClass.GetData($"SELECT * FROM `tb_student` WHERE `hobby` = ‘{textBox12.Text}’");
 }

 private void pictureBox22_Click(object sender, EventArgs e)
 {
 if (textBox17.Text == "" && textBox20.Text == "" && textBox21.Text == "")
 {
 MessageBox.Show("Введено пропущенные данные!", "Ошибка", MessageBoxButtons.OK,
MessageBoxIcon.Warning);
 return;
 }
 try
 {
 DataClass.GetData($"INSERT INTO `tb_teacher`(`name_teacher`, `date_birth`, `specialization`)
VALUES (‘{textBox17.Text}’,’{textBox20.Text}’,’{textBox21.Text}’)");
 MessageBox.Show("Дані успішно додано. Оновіть таблицю.", "Результат",
MessageBoxButtons.OK, MessageBoxIcon.Information);
 }
 catch (Exception ex)
```

```
{
MessageBox.Show(ex.Message, "Помилка!", MessageBoxButtons.OK, MessageBoxIcon.Error);
}

textBox17.Text = "";
textBox20.Text = "";
textBox21.Text = "";

DB.closedConn();
}

private void pictureBox23_Click(object sender, EventArgs e)
{
textBox17.Text = "";
textBox18.Text = "";
textBox19.Text = "";
comboBox4.Text = "";
textBox20.Text = "";
textBox21.Text = "";
}

private void pictureBox24_Click(object sender, EventArgs e)
{
SaveFileDialog saveFileDialog = new SaveFileDialog() { Filter = "Excel Workbook|*.xlsx" };
saveFileDialog.ShowDialog();
string path = saveFileDialog.FileName;
Microsoft.Office.Interop.Excel.Application application = new
Microsoft.Office.Interop.Excel.Application();
Microsoft.Office.Interop.Excel.Workbook workbook = application.Workbooks.Add();
Microsoft.Office.Interop.Excel.Worksheet worksheet = workbook.ActiveSheet;


for (int i = 1; i < dataGridView4.RowCount + 1; i++)
{
for (int j = 1; j < dataGridView4.ColumnCount + 1; j++)
```

```
{
worksheet.Rows[i].Columns[j] = dataGridView4.Rows[i – 1].Cells[j – 1].Value;
}
}
application.AlertBeforeOverwriting = false;
workbook.SaveAs(path);
application.Quit();
}

private void pictureBox25_Click(object sender, EventArgs e)
{
dataGridView4.DataSource = DataClass.GetData("SELECT * FROM `tb_teacher`");
}

private void pictureBox26_Click(object sender, EventArgs e)
{
DataClass.GetData($"UPDATE `tb_teacher` SET `name_teacher`= ‘{textBox17.Text}’,
`date_birth` = ‘{textBox20.Text}’, `specialization` = ‘{textBox21.Text}’ WHERE
`id_teacher`={textBox18.Text}");
MessageBox.Show("Дані успішно змінено. Оновіть таблицю", "Результат",
MessageBoxButtons.OK, MessageBoxIcon.Information);
textBox17.Text = "";
textBox20.Text = "";
textBox21.Text = "";
textBox18.Text = "";
}

private void pictureBox27_Click(object sender, EventArgs e)
{
DataClass.GetData($"DELETE FROM `tb_teacher` WHERE `id_teacher`={textBox18.Text}");
MessageBox.Show("Дані успішно видалено. Оновіть таблицю.", "Результат",
MessageBoxButtons.OK, MessageBoxIcon.Information);
textBox18.Text = "";
}
```

```
 private void dataGridView4_CellClick(object sender, DataGridViewCellEventArgs e)
 {
 textBox18.Text = dataGridView4.CurrentRow.Cells[0].Value.ToString();
 textBox17.Text= dataGridView4.CurrentRow.Cells[1].Value.ToString();
 textBox20.Text = dataGridView4.CurrentRow.Cells[2].Value.ToString();
 textBox21.Text = dataGridView4.CurrentRow.Cells[3].Value.ToString();
 }

 private void pictureBox28_Click(object sender, EventArgs e)
 {
 if (comboBox4.SelectedIndex == 0) dataGridView4.DataSource = DataClass.GetData($"SELECT
* FROM `tb_teacher` WHERE `id_teacher` = {textBox19.Text}");
 else if (comboBox4.SelectedIndex == 1) dataGridView4.DataSource =
DataClass.GetData($"SELECT * FROM `tb_teacher` WHERE `name_teacher` =
‘{textBox19.Text}’");
 else if (comboBox4.SelectedIndex == 2) dataGridView4.DataSource =
DataClass.GetData($"SELECT * FROM `tb_teacher` WHERE `date_birth` =
‘{textBox19.Text}’");
 else if (comboBox4.SelectedIndex == 3) dataGridView4.DataSource =
DataClass.GetData($"SELECT * FROM `tb_teacher` WHERE `specialization` =
‘{textBox19.Text}’");
 }

 private void groupBox8_Enter(object sender, EventArgs e)
 {

 }
 }
}
```