МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВІДОКРЕМЛЕНИЙ СТРУКТУРНИЙ ПІДРОЗДІЛ

«ФАХОВИЙ КОЛЕДЖ РАКЕТНО-КОСМІЧНОГО МАШИНОБУДУВАННЯ

ДНІПРОВСЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ ім. О. ГОНЧАРА»

Циклова комісія програмної інженерії

**КУРСОВИЙ ПРОЕКТ**

з навчальної дисципліни

«ОБ’ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ»

на тему: «Програма ведення обліку здачі національного мультимедійного тесту»

(вказати тему курсового проекту)

Студента IVкурсу  ПЗ-21-1  групи

галузь знань 12 «Інформаційні технології»

спеціальності 121 «Інженерія програмного забезпечення»

    Гуненко Я.М.

(прізвище та ініціали студента)

Керівниця викладачка Гапоненко Н.В.

Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_ Оцінка ECTS: \_\_\_\_\_\_\_

Члени комісії \_\_\_\_ Валентина ЛЮБОХИНЕЦЬ

(підпис) (прізвище та ініціали) .

\_\_\_\_\_\_\_\_\_ Світлана ЛАНСЬКА.

(підпис) (прізвище та ініціали) .

\_\_\_\_\_\_\_\_\_ Наталія ГАПОНЕНКО.

(підпис) (прізвище та ініціали) .

м. Дніпро - 2024 рік

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВІДОКРЕМЛЕНИЙ СТРУКТУРНИЙ ПІДРОЗДІЛ

«ФАХОВИЙ КОЛЕДЖ РАКЕТНО-КОСМІЧНОГО МАШИНОБУДУВАННЯ

ДНІПРОВСЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ ім. О. ГОНЧАРА»

Цикловакомісія програмної інженерії

|  |  |
| --- | --- |
|  | **ЗАТВЕРДЖУЮ**  Голова комісії ПІ  \_\_\_\_\_\_\_\_\_ Світлана ЛАНСЬКА  «16» вересня 2024 р. |

**ЗАВДАННЯ**

**на виконання курсового проекту**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| з дисципліни | | | Об’єктно-орієнтоване програмування | | | | | | | |
| студенту | Гуненко Ярославу Максимовичу | | | | | | | | | |
|  | (прізвище, ім'я та по батькові) | | | | | | | | | |
| Відділення | | Комп'ютерної та програмної інженерії | | | | | | | | |
| Спеціальність | | | | 121 Інженерія програмного забезпечення | | | | | | |
| Курс | IV | | | | | |  | Група (шифр) | | ПЗ-21-1 |
| 1 Тема проекту | | | | | «Програма ведення обліку здачі національного | | | | | |
| мультипредметного тесту» | | | | | | | | | | |
|  | | | | | | | | | | |
| 2 Початкові дані | | | | | | Перелік даних про предмет (назва, рік, обов`язковий/ні, | | | | |
| опис, мінімальний, максимальний бал, зразок завдань), заклад проведення ( | | | | | | | | | | |
| область, місто, тип, email, відповідальний), персональні дані особи | | | | | | | | | | |
| (прізвище, ім’я, по-батькові, дата народження, стать, номер телефону, email, | | | | | | | | | | |
| РНОКПП, паспортні дані, свідотство про освіту), дані сертифікату (особа, | | | | | | | | | | |
| PIN-код, статус, дата) результат (особа, предмет, бал, сертифікат, заклад, | | | | | | | | | | |
| дата), алгоритм ведення обліку за прізвищем, PIN-кодом, пороговим балом, | | | | | | | | | | |
| звітна форма за період з кількістю сертифікатів, звіт-сертифікат особи | | | | | | | | | | |
| Розглянуто і ухвалено на засіданні циклової комісії | | | | | | | | | програмної інженерії | |
| Протокол № 2 від 16.09.2024 р. | | | | | | | | | | |

Керівник КП \_\_\_\_\_\_\_\_\_\_\_\_\_ Наталія ГАПОНЕНКО

(підпис) (ініціали та прізвище)

Завдання до виконання

одержав студент \_\_\_\_\_\_\_\_\_\_\_\_\_ Ярослав ГУНЕНКО

(підпис) (ініціали та прізвище)

Дата видачі 16 вересня 2024 р.

Термін виконання 18 листопада 2024 р.

ЗМІСТ

[ВСТУП 4](#_Toc182231620)

[1 ПОСТАНОВКА ЗАДАЧІ 5](#_Toc182231621)

[2 ХАРАКТЕРИСТИКА ПРОГРАМНИХ ЗАСОБІВ РЕАЛІЗАЦІЇ ПРОЕКТУ 7](#_Toc182231622)

[2.1 Опис середовища програмування 7](#_Toc182231623)

[2.2 Опис мови програмування 8](#_Toc182231624)

[2.3 Опис СКБД 10](#_Toc182231625)

[2.4 Опис основних принципів ООП 11](#_Toc182231626)

[2.5 Опис подібних програмних продуктів 14](#_Toc182231627)

[3 ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ ПРОГРАМНОГО ПРОДУКТУ 16](#_Toc182231628)

[3.1 Опис бази даних 16](#_Toc182231629)

[3.2 Проектування користувацького інтерфейсу програми 19](#_Toc182231630)

[3.3 Контроль вхідних даних програми 35](#_Toc182231631)

[4 ІНСТРУКЦІЯ З КОРИСТУВАННЯ ПРОГРАМНОЮ СИСТЕМОЮ 38](#_Toc182231632)

[Висновки 54](#_Toc182231633)

[Список Використаних ДжЕРЕЛ 55](#_Toc182231634)

[Додаток А 56](#_Toc182231635)

[Додаток Б 57](#_Toc182231636)

[Додаток В 62](#_Toc182231637)

[Додаток Г 169](#_Toc182231638)

ВСТУП

У сучасному світі точне зберігання даних та автоматизоване управління документацією потрібні скрізь. Цифрова епоха поширюється і на сферу освіти. Щороку в Україні учні, які закінчують 11-й клас, складають іспит, що відображає рівень знань, які вони здобули за цей час. Наразі, у зв'язку з воєнним станом, Зовнішнє незалежне оцінювання, скорочено ЗНО, замінено на Національний мультимедійний тест, скорочено НМТ.

Для центрів оцінювання доступно багато програмного забезпечення, але ці програми складні у вивченні та незручні у використанні. Більшість цих програм базуються на залежних від мережі інтернет WEB технологіях, що не завжди зручно. Програми повинні бути простими у використанні, швидкими, легкими в освоєнні інструментами та універсальними для центрів оцінювання будь-якого розміру. Програмне забезпечення повинно бути доступним для використання навіть недосвідченими користувачами комп'ютерів. Воно може використовуватися як однією особою (наприклад, у невеликому центрі оцінювання), так і групою осіб (наприклад, у великому центрі оцінювання з кількома посадами).

Основна мета курсового проекту - створити корисний і простий додаток для великих і малих центрів оцінювання.

1 ПОСТАНОВКА ЗАДАЧІ

Тема курсового передбачає створення програмного забезпечення для обліку та зберігання інформації про осіб, які складали НМТ, їх результатів та створення сертифікатів та включає в себе такий функціонал:

1. Ведення обліку інформації про осіб які складали НМТ:

* Додавання інформації про учасника НМТ, шляхом введення типу паспорта, номеру паспорта, ІПН, номера свідоства про освіту, статі, дати народження, електронної пошти, контактного номеру телефону та необов`язкової помітки для екстренних випадків.
* Редагування інформації про учасника НМТ.

1. Ведення обліку інформації про умови тестування затверджені МОН:

* Додавання інформації про умови проведення тестування, шляхом введення статусу обов`язковості, дати укладання умов, прохідного балу, максимального можливого балу, мінімального можливого балу.
* Редагування інформації про умови проведення тестування.

1. Ведення обліку інформації про навчальні заклади для проведення НМТ:

* Додавання інформації про умови проведення тестування, шляхом введення шифру навчального закладу, міста, області, типу, E-mail, ПІБ відповідального за проведення.
* Редагування інформації про навчальний заклад.
* Видалення інформації про навчальний заклад.

1. Ведення обліку інформації про результати тестування:

* Додавання інформації про результати тестування, шляхом введення номеру паспорту учасника, навчального закладу, дати укладання умов тестування, назви навчальної дисципліни, результат тестування у балах, дата складання тестування.
* Редагування інформації про умови проведення тестування.

1. Ведення обліку інформації про навчальні дисципліни:

* Додавання інформації про навчальні дисципліни, шляхом введення назви предмету, короткого опису предмету, зразка тестування, піктограми навчальної дисципліни.
* Редагування інформації про умови навчальні дисципліни.
* Видалення інформації про умови навчальні дисципліни

1. Ведення обліку інформації про сертифікати НМТ:

* Додавання інформації про сертифікати НМТ, шляхом введення номеру паспорту учня, PIN-коду сертифікату, терміну дії сертифікату, статусу дії сертифікату, дати створення сертифікату, дати укладання умов тестування.
* Редагування інформації про сертифікати НМТ.
* Генерація сертифікатів в розширенні “\*.hltm” з можливістю встановлення шаблону для назви при першому запуску програми.

1. Вимоги до операційної системи:

* Операційна система: Windows 10 х64.
* Процесор: Intel Celeron або еквівалент.
* Графічна підсистема: DirectX 10 і вище.

Таблиця 3.1 – Виділення інформаційних об’єктів предметної області

|  |  |
| --- | --- |
| Особа | ID учня, номер паспорту, тип паспорту, ПІБ, дата народження, стать, E-mail, контактний номер телефону, Номер свідоства про освіту, примітка, ІПН |
| Навчальний заклад | Шифр навчального закладу, місто, область, тип, E-mail, ПІБ відповідального за проведення |
| Предмет | Код предмету, назва предмету, опис, зразок завдань, шлях до зображення предмету |
| Умови проходження тестування з предмету | ID умови, код предмету, максимальний бал, мінімальний бал, мінімальний бал для проходження, статус, дата укладання вимог |
| Результат тестування | Шифр результату, ID умови, ID учня, отриманий бал, статус здачі, дата проходження, шифр навчального закладу |
| Сертифікат учасника НМТ | Номер сертифікату, ID учня, PIN-код, дата створення, термін дії, статус дійсності |

2 ХАРАКТЕРИСТИКА ПРОГРАМНИХ ЗАСОБІВ РЕАЛІЗАЦІЇ ПРОЕКТУ

1. Опис середовища програмування

C++ Builder - це програмний продукт, інструмент швидкої розробки додатків (RAD), інтегроване середовище розробки (IDE) та система, що використовується програмістами для розробки програмного забезпечення мовою програмування C++. Visual C++ Builder дозволяє створювати графічні інтерфейси користувача шляхом перетягування компонентів з палітри інструментів на форми. За допомогою Builder можна створювати додатки для Windows, які використовують велику бібліотеку візуальних компонентів (VCL).C++ Builder автоматично генерує більшу частину коду програми, як тільки ви починаєте працювати над проектом. Для завершення решти логіки програми використовується текстовий редактор коду, який надає такі функції, як рефакторинг, паралельне редагування, завершення коду, збережені макроси натискання клавіш і власні комбінації клавіш. C++ Builder інтегрований в MSBuild як середовище збірки з командами збірки та компіляції, які викликають MSBuild; середовище RAD Studio надає мови програмування Delphi та C++ для розробки. Для курсових проектів корисним є C++ builder, оскільки він повністю підтримує мову програмування C++; середовище C++ builder підтримує найновіші стандарти C++, що дозволяє розробляти сучасне, ефективне програмне забезпечення. Вбудований компілятор ефективно оптимізує код і збільшує швидкість виконання програми.

Ще однією важливою особливістю цього середовища є підтримка високопродуктивних бібліотек: Visual Component Library (VCL) - це набір компонентів для побудови багатих графічних інтерфейсів користувача, VCL забезпечує швидкість і надійність. FireDAC - сучасна бібліотека доступу до даних, що підтримує широкий спектр баз даних; RTL (Runtime Library) - рутинна бібліотека часу виконання, що містить безліч корисних функцій для роботи з рядками, файлами, математичними операціями тощо.

C++ Builder дозволяє створювати додатки для Windows, macOS, Android та iOS з єдиної кодової бази; додатки, створені за допомогою C++ Builder, використовують нативні компоненти платформи, що забезпечує високу продуктивність та інтеграцію з операційною системою.

1. Опис мови програмування

Для виконання курсового проєкту було обрано мову програмування C++ з кількох вагомих причин. По-перше, C++ – це універсальна мова загального призначення, яка дозволяє створювати високопродуктивні, ефективні та надійні програмні системи. Її потужність полягає у можливості низькорівневого доступу до пам'яті та апаратних ресурсів, що робить її ідеальним вибором для розробки критичних до продуктивності додатків.

Однією з ключових особливостей C++ є підтримка об'єктно-орієнтованого програмування. Ця парадигма дозволяє моделювати реальний світ за допомогою класів та об'єктів. Класи визначають структуру даних та поведінку об'єктів, а об'єкти є екземплярами цих класів. Такий підхід сприяє модульності, повторюваності коду та легкості підтримки великих програмних систем.

Мова програмування C++ створена для підтримки широкого спектру програмних парадигм, включаючи процедурне, об'єктно-орієнтоване та шаблонне програмування. Застосування C++ дозволяє розробникам писати код, який є не тільки функціональним, але і високопродуктивним, завдяки доступу до низькорівневих ресурсів і можливості контролю над пам'яттю. Така гнучкість робить C++ одним із найпопулярніших виборів для розробки продуктивних і надійних додатків.[2]

C++ пропонує унікальну можливість поєднання низькорівневих операцій з високим рівнем абстракції, що є однією з найсильніших сторін мови. Завдяки таким інструментам, як шаблони та стандартна бібліотека шаблонів (STL), C++ забезпечує програмістів можливістю писати загальний та повторно використовуваний код. Це значно підвищує ефективність розробки і дає змогу створювати рішення, які легко адаптуються до різних потреб. [3]

C++ надає можливість розробникам використовувати потужну комбінацію об'єктно-орієнтованого і процедурного програмування, що сприяє створенню комплексних і гнучких програмних систем. Завдяки підтримці таких концепцій, як поліморфізм, успадкування і інкапсуляція, мова дозволяє програмістам структурувати код більш ефективно. Це, в свою чергу, сприяє легкості масштабування і підтримки програмного забезпечення. [1]

Шаблони (templates) – це ще одна потужна особливість C++, яка дозволяє створювати універсальні алгоритми та структури даних. За допомогою шаблонів можна писати код, який працює з різними типами даних, не дублюючи його. Це значно зменшує обсяг коду та підвищує його надійність.

Стандарт The C Language став основою для багатьох сучасних мов програмування, зокрема для C++. У цьому стандарті викладені базові принципи програмування на мові C, такі як контроль пам'яті, робота з вказівниками та операції на низькому рівні. Ці основні елементи мови C збереглися в C++ і є важливою частиною ефективної роботи з системними ресурсами. [7]

Стандарт ISO/IEC 14882:2011 для мови C++ встановлює чіткі вимоги і правила щодо структури, функцій та можливостей мови, що сприяє її сумісності з сучасними програмними системами. Дотримання стандарту забезпечує розробникам можливість створювати програми, які відповідають високим вимогам до якості та продуктивності. Крім того, стандарт містить рекомендації щодо оптимізації та забезпечення безпеки, що підвищує надійність і стабільність розроблених програм.[4]

Стандарт The C Language став основою для багатьох сучасних мов програмування, зокрема для C++. У цьому стандарті викладені базові принципи програмування на мові C, такі як контроль пам'яті, робота з вказівниками та операції на низькому рівні. Ці основні елементи мови C збереглися в C++ і є важливою частиною ефективної роботи з системними ресурсами. [6]

STL – це багата колекція контейнерів, алгоритмів та ітераторів, яка є невід'ємною частиною сучасного C++. STL забезпечує ефективні та зручні засоби для роботи з даними, що значно спрощує розробку складних програмних систем.

Стандартна бібліотека шаблонів (STL) є невід'ємною частиною мови C++ і забезпечує розробників набором інструментів для ефективної роботи з даними. STL включає різні контейнери, такі як вектори, списки, а також алгоритми, які дозволяють легко маніпулювати даними. Це значно спрощує розробку, дозволяючи програмістам зосередитися на логіці програми замість написання базових структур з нуля. [5]

Стандарт The C Language став основою для багатьох сучасних мов програмування, зокрема для C++. У цьому стандарті викладені базові принципи програмування на мові C, такі як контроль пам'яті, робота з вказівниками та операції на низькому рівні. Ці основні елементи мови C збереглися в C++ і є важливою частиною ефективної роботи з системними ресурсами. [8]

У курсовому проєкті мова програмування C++ дозволила структурувати програму за допомогою класів. Класи були створені для представлення різних доменних сутностей, що полегшує розуміння та підтримку коду. Для взаємодії з базою даних MySQL було використано функціонал C++, що забезпечує швидкий та надійний доступ до даних. Для роботи з динамічною пам’яттю використовувалися вказівники мови C++, низькорівневий доступ до пам'яті якої дозволив додатку працювати швидше.

1. Опис СКБД

Для виконання курсового проєкту було використано СУБД MySQL.

MySQL – це одна з найпопулярніших у світі систем управління реляційними базами даних з відкритим кодом. Її широко застосовують для створення веб-додатків, корпоративних систем та в багатьох інших сферах. MySQL здатна обробляти великі обсяги даних і одночасно обслуговувати тисячі запитів. Це досягається завдяки оптимізації запитів, ефективним алгоритмам сортування та індексування. [10]

MySQL забезпечує високий рівень доступності даних завдяки підтримці транзакцій, реплікації та механізмам відновлення після збоїв. Вона підтримує різні типи таблиць і механізми зберігання даних, що дозволяє адаптувати систему до конкретних потреб проєкту. СУБД сумісна зі стандартом SQL, що значно спрощує роботу з даними та дозволяє використовувати широкий спектр інструментів і бібліотек.

Вирішальними факторами для вибору СУБД MySQL для курсового проєкту стали її технології та можливості.

InnoDB – основний рушій зберігання даних у MySQL, який забезпечує високу швидкість роботи, підтримку транзакцій, цілісність даних і масштабованість. MyISAM – ще один рушій зберігання даних, оптимізований для швидкого читання. Реплікація дозволяє створювати копії бази даних на інших серверах для підвищення доступності й резервування. Партіціювання – це можливість розбивати великі таблиці на менші частини для ефективнішої обробки даних і покращення швидкодії. Тригери й збережені процедури автоматизують виконання певних дій при зміні даних у таблицях.[9]

Для реалізації курсового проєкту було використано такі можливості MySQL: збережені процедури для автоматизації часто виконуваних завдань, тригери для автоматичного виконання дій у відповідь на зміну даних, представлення (віртуальні таблиці) для об'єднання даних з кількох таблиць, індекси для прискорення пошуку, транзакції для забезпечення цілісності даних, реплікація для підвищення доступності та резервування, а також кластеризація для масштабування бази даних.

1. Опис основних принципів ООП

Об'єктно-орієнтоване програмування (ООП) – це парадигма програмування, яка фокусується на створенні програм за допомогою об'єктів, що взаємодіють між собою. Основні принципи ООП – інкапсуляція, наслідування, поліморфізм та абстракція.

Інкапсуляція – це об'єднання даних і методів, що працюють з ними, в єдиний об'єкт. Це приховує внутрішню структуру об'єкта від зовнішнього світу, забезпечуючи безпеку даних та спрощуючи підтримку коду. В C++ інкапсуляція досягається за допомогою модифікаторів доступу: private, protected та public. Приклад фрагменту коду що демонструє інкапсуляцію продемонстровано у лістингу 2.1.

Лістинг 2.1 – Фрагмент коду

class Neuron {

private:

double output;

std::vector<double> weights;

public:

double calculateOutput(const std::vector<double>& inputs) {

// Розрахунок виходу нейрона

// ...

return output;

}

};

Наслідування дозволяє створювати нові класи (похідні) на основі існуючих (базових), успадковуючи їхні властивості та методи. Це забезпечує повторне використання коду та створення ієрархій класів. Типи наслідування в C++: public, protected та private. При публічному наслідуванні публічні члени базового класу стають публічними членами похідного класу, а захищені – захищеними. Це означає, що об'єкти похідного класу можуть безпосередньо звертатися до цих членів. Публічне наслідування зазвичай використовується для створення ієрархій класів, де похідні класи є більш специфічними видами базового класу. При захищеному наслідуванні як публічні, так і захищені члени базового класу стають захищеними членами похідного класу. Це означає, що до них можуть звертатися лише члени похідного класу та інших похідних класів. Захищене наслідування часто використовується для створення ієрархій класів, де похідні класи можуть розширювати базовий клас, але приховувати деякі деталі реалізації від зовнішнього світу. При приватному наслідуванні публічні і захищені члени базового класу стають приватними членами похідного класу. Це означає, що до них можуть звертатися лише члени самого похідного класу. Приватне наслідування рідко використовується, оскільки обмежує можливості повторного використання коду. Зазвичай використовується для повного приховування реалізації базового класу і створення нового, більш абстрактного класу. Приклад фрагменту коду що демонструє наслідування продемонстровано у лістингу 2.2.

Лістинг 2.2 – Фрагмент коду

class NeuralNetwork {

public: virtual void train(const std::vector<std::pair<std::vector<double>, double>>& data) = 0;

};

class ConvolutionalNeuralNetwork : public NeuralNetwork { public: void train(const std::vector<std::pair<std::vector<double>, double>>& data) override {

// Тренування згорткової нейронної мережі // ...

}

};

Поліморфізм – це здатність об'єктів різних типів відповідати на один і той самий запит по-різному. Це досягається за допомогою перевантаження функцій та віртуальних функцій. Поліморфізм забезпечує гнучкість та розширюваність програм. Приклад фрагменту коду що демонструє поліморфізм продемонстровано у лістингу 2.3.

Лістинг 2.3 – Фрагмент коду

class ActivationFunction {

public:

virtual double activate(double z) = 0;

};

class Sigmoid : public ActivationFunction {

public:

double activate(double z) override {

// Функція сигмоїди

// ...

}

};

class ReLU : public ActivationFunction {

public:

double activate(double z) override {

// Функція ReLU

// ...

}

};

Абстракція – це процес фокусування на суттєвих характеристиках об'єкта, приховуючи деталі реалізації. Абстрактні класи та чисті віртуальні функції в C++ дозволяють створювати абстрактні інтерфейси, які не можуть бути інстанційовані безпосередньо. Приклад фрагменту коду що демонструє абстракцію продемонстровано у лістингу 2.4.

Лістинг 2.4 – Фрагмент коду

class Layer {

public:

virtual void forward(const std::vector<double>& input) = 0;

virtual void backward(const std::vector<double>& output\_error) = 0;};

1. Опис подібних програмних продуктів

Схожою за функціоналом є система my.testportal.gov.ua.

my.testportal.gov.ua — це веб-портал, призначений для організації та обліку результатів національних тестів. Система працює виключно онлайн, що означає, що будь-яка перерва в доступі до інтернету може спричинити затримку в роботі, оскільки програма не підтримує роботу в автономному режимі.

Платформа використовує власну базу даних для збереження інформації про учнів, навчальні заклади та результати тестувань. Проте, для більш гнучкої обробки даних немає можливості прямої інтеграції з СКБД на стороні користувача. Дані можна експортувати у форматі PDF, але це обмежує можливості подальшого аналізу та структуризації інформації.

Платформа також вимагає значного часу на налаштування, особливо для нових користувачів або навчальних закладів. Потрібен спеціаліст для розподілу ролей та доступів між адміністраторами, вчителями та учнями, оскільки самостійно це зробити складно без відповідного досвіду.

Висновок: my.testportal.gov.ua підходить для великих організацій, що мають відповідні ресурси для підтримки та адміністрування системи. Однак, для невеликих закладів або окремих користувачів платформа може виявитися занадто складною та вимагати значних зусиль для ефективного використання.

3 ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ ПРОГРАМНОГО ПРОДУКТУ

1. Опис бази даних

Для збереження великих об’ємів інформації потрібно розробити низку правил та відношень між цими даними. Відношення між таблицями показані на ER-діаграмі. Скрипт створення БД показаний у додатку Б. Зміст таблиць бази даних показаний у таблицях 3.2-3.8.

Таблиця 3.2 – Поля таблиці «Студент» (Student)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва поля | Опис | Тип | Розмір | Ключ |
| Student\_id | Унікальний ідентифікатор студента | Числовий |  | Первинний |
| Passport\_num | Номер паспорта | Символьний | До 20 символів |  |
| Passport\_type | Тип паспорта | Символьний | До 10 символів |  |
| PIB | Прізвище, ім'я, по-батькові | Символьний | До 100 символів |  |
| Birth\_date | Дата народження | DATE | Стандарт ISO |  |
| Gender | Стать | Символьний | 1 символ |  |
| E-mail | Електронна пошта | Символьний | До 100 символів |  |
| Phone\_num | Номер телефону | Символьний | До 15 символів |  |
| EduCerf\_num | Номер освітнього сертифіката | Символьний | До 30 символів |  |
| PN | Ідентифікаційний код | Символьний | 10 символів |  |
| Additional | Додаткова інформація | Текстовий |  |  |

Таблиця 3.3 – Поля таблиці «Користувачі» (Users)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва поля | Опис | Тип | Розмір | Ключ |
| Login | Логін користувача | Символьний | До 50 символів | Первинний |
| Password | Пароль користувача | Символьний | До 50 символів |  |
| Role | Роль користувача | Символьний | До 30 символів |  |
| PIB | ПІБ | Символьний | До 100 |  |
| Status | Статус | tinyint | 1 |  |

Таблиця 3.4 – Поля таблиці «Результат» (Result)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва поля | Опис | Тип | Розмір | Ключ |
| Res\_id | Унікальний код результату | Числовий |  | Первинний |
| Subj\_id | Код предмета | Числовий |  | Зовнішній |
| Condition\_id | Код умов | Числовий |  | Зовнішній |
| Student\_id | Код студента | Числовий |  | Зовнішній |
| Reached\_score | Досягнутий бал | Числовий |  |  |
| Status | Статус | Tinyint(1) | 1 символ |  |
| School\_id | Код навчального закладу | Числовий |  | Зовнішній |
| Attemp\_date | Дата спроби | DATE | Стандарт ISO |  |

Таблиця 3.5 – Поля таблиці «Сертифікат» (Certificate)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва поля | Опис | Тип | Розмір | Ключ |
| Cerf\_num | Номер сертифіката | Числовий |  | Первинний |
| Student\_id | ID студента | Числовий |  | Зовнішній |
| PIN | PIN-код | Символьний | До 15 символів |  |
| Creation\_date | Дата створення | DATE | Стандарт ISO |  |
| Efect\_time | Термін дії | DATE | Стандарт ISO |  |

Продовження таблиці “Certificate”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва поля | Опис | Тип | Розмір | Ключ |
| Status | Статус сертифіката | Tinyint(1) |  |  |
| Login | Логін | Символьний | До 50 символів | Зовнішній |

Таблиця 3.6 – Поля таблиці «Умови тестування» (Condition)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва поля | Опис | Тип | Розмір | Ключ |
| Condition\_id | Унікальний код умови | Числовий |  | Первинний |
| Subject\_id | Код предмета | Числовий |  | Зовнішній |
| Max\_point | Максимальний бал | Числовий |  |  |
| Min\_r\_point | Мінімальний бал для проходження | Числовий |  |  |
| Min\_point | Мінімальний бал | Числовий |  |  |
| Status | Обов'язкова умова | Tinyint(1) |  |  |
| Date | Дата | DATE | Стандарт ISO |  |

Таблиця 3.7 – Поля таблиці «Навчальний заклад» ()

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва поля | Опис | Тип | Розмір | Ключ |
| School\_id | Унікальний код навчального закладу | Числовий |  | Первинний |
| City | Місто | Символьний | До 50 символів |  |
| Region | Область | Символьний | До 50 символів |  |
| Type | Тип навчального закладу | Символьний | До 30 символів |  |
| E-mail | Електронна пошта | Символьний | До 100 символів |  |
| Teacher\_PIB | Прізвище, ім'я, по-батькові відповідального | Символьний | До 100 символів |  |

Таблиця 3.8– Поля таблиці «Предмети» (Subject)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва поля | Опис | Тип | Розмір | Ключ |
| Subject\_id | Унікальний код предмета | Числовий |  | Первинний |
| Name | Назва предмета | Символьний | До 100 символів |  |
| Description | Опис предмета | Текстовий |  |  |
| Image\_name | Шлях до піктограми завдань | Символьний | До 255 символів |  |
| Test\_sample | Шлях до прикладу завдань | Символьний | До 255 символів |  |

ER-діаграма зв’язку таблиць бази даних наведена нижче на рисунку 3.1.

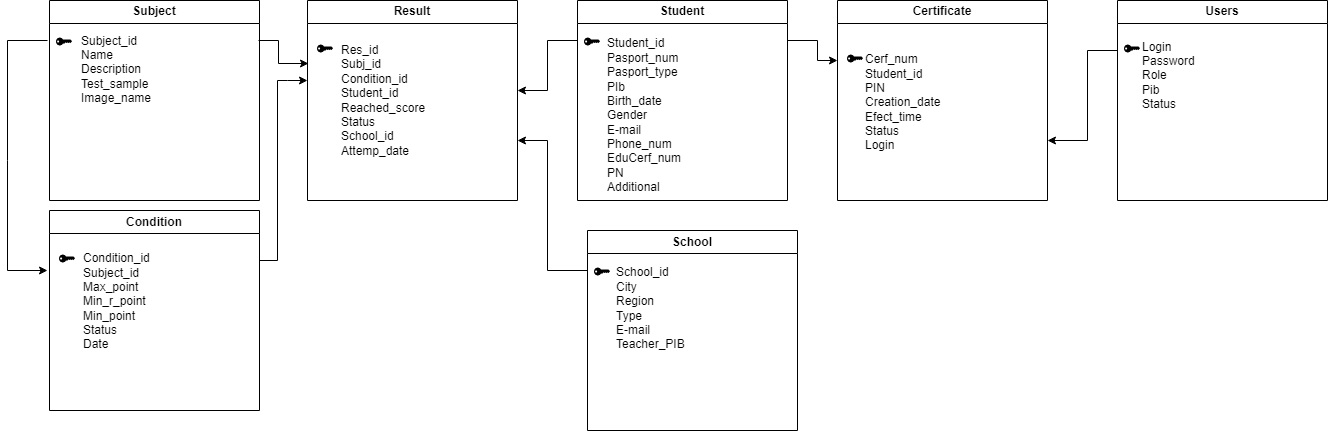


Рисунок 3.1 – ER діагарма бази даних

1. Проектування користувацького інтерфейсу програми

Для такого курсового проєкту потрібно грамотно спроєктувати взаємодію та ієрархію класів. Головною задачею проектування курсового проекту було створення логічних та програмних зв’язків між компонентами системи. Кожна підсистема виконує свою роль максимально абстраговано від інших та паралельно з ними.

Виходячи з цього можна спроектувати зручний інтерфейс. Його було створено саме під написані класи зв’язків з БД. Створено налаштування інтерфейсу для вдоволення потреб більшості користувачів(розмір інтерфейсу та кольорова палітра).

При проєктуванні додатку першим вікном, що користувач додатку побачить було обрано вікно авторизації, у якому за допомогою властивості passwordchar сховано пароль. У разі вдалої авторизації кооистувачу продемонстровано його роль у повідомленні ShowMessage().

У разі відстуності файлу config.ini відображено вікно створення шаблону назви сертифікату зі збереженням у ini файлі у папці запуску проєкту, якщо користувач авотризувався з роллю “головний адміністратор”. Для роботи цієї функції використано бібліотеку “IniFiles.cpp”. Фрагмент коду для створення ini файлу продемонстровано у лістингу 3.1.

Лістинг 3.1 – Фрагмент коду для створення ini файлу

void \_\_fastcall TForm13::SaveTemplateToIni()

{

TIniFile \*IniFile = new TIniFile(ExtractFilePath(Application->ExeName) + "config.ini");

try {

IniFile->WriteString("Certificate", "TemplateName", template\_name);

}

\_\_finally {

delete IniFile;

}

}

Для введення шаблону користувачу запропоновано об`єкт класу LabelEdit та кнопку для запуску створення ini файлу. Для перевірки введення користувача використано обробник подій OnExit для компоненту LabelEdit. Зовнішній вигляд вікна продемостровно на рисунку 3.1.

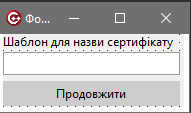


Рисунок 3.1 – Вікно введення шаблону для назви сертифікату

Після створення шаблону назви сертифікату користувачу програмного додатку буде відображено головне вікно програмного додатку. Інтерфейс головного вікна продемонстровано на рисунку 3.2.

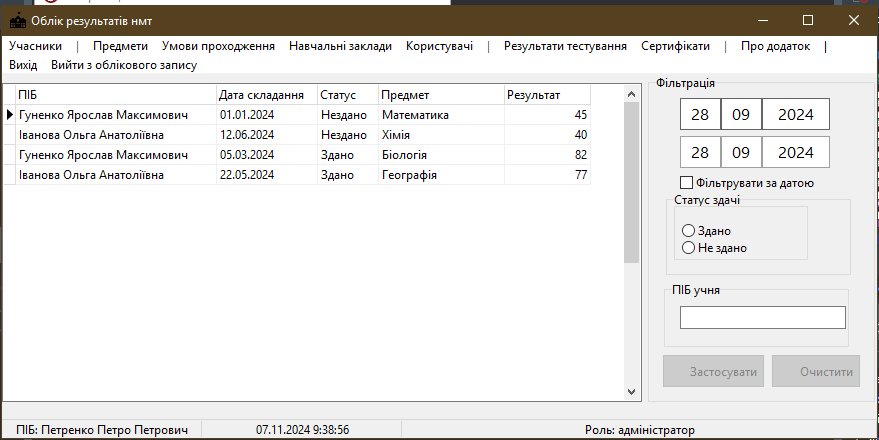


Рисунок 3.2 – Головне вікно програмного додатку

На головному вікні створено компонент DBGrid, у якому відображено стислу інформацію про результати тестування. Для зручної роботи з даними у таблиці створено сортування даних за стовпцями за допомогою обробника подій OnTitleClick. Лістинг фрагменту коду для сортування продемонстровано у лістингу 3.2.

Лістинг 3.2 – Фрагмент коду для сортування записів у таблиці

void \_\_fastcall TForm3::DBGrid1TitleClick(TColumn \*Column)

{

String columnName = Column->FieldName;

static bool sortAsc = true;

String sortOrder = sortAsc ? "ASC" : "DESC";

sortAsc = !sortAsc;

String query =

"SELECT s.PIB, "

"r.Attemp\_date, "

"CASE WHEN r.Status = 1 THEN 'Здано' ELSE 'Нездано' END AS Status, "

"subj.Name, "

"r.Reached\_score "

"FROM Result r "

"JOIN Student s ON r.Student\_id = s.Student\_id "

"JOIN Subject subj ON r.Subj\_id = subj.Subject\_id "

"ORDER BY " + columnName + " " + sortOrder;

try

{

DataModule1->MainQuery->Close();

DataModule1->MainQuery->SQL->Clear();

DataModule1->MainQuery->SQL->Add(query);

DataModule1->MainQuery->Open();

DBColumnSizes();

}

catch (Exception &e)

{

ShowMessage("Помилка сортування: " + e.Message);

}

}

Також для роботи з великою кількістю записів створено функціонал для фільтрації записів, винесений в окремий GroupBox з можливістю комбінації фільтрів. За замовчуванням кнопки для запуску фільтрації та її відміни заблоковані для користувача. Після вибору будь-якого пункту у групі перемикачів RadioGroup “Статус” або встановлення перемикача CheckBox “Фільтрувати за датою” чи введення піб учня кнопки активуються. Для фільтрації за датою використано об`єкти DatePicker.

Для зручної навігації використано компонент MainMenu. Інтерфейс головного меню продемонстровано на рисунку 3.3. Лістинг фрагменту коду для запуску іншого вікна на обробник подій OnClick для пунктів мею продемонстровано у лістингу 3.3.

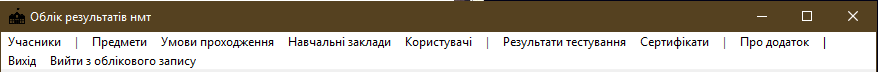


Рисунок 3.3 – Інтерфейс головного меню

Лістинг 3.3 – Запуск вікна по кліку на пункт головного меню

void \_\_fastcall TForm3::N2Click(TObject \*Sender)

{

Form14->ShowModal();

}

Для розмежовування доступу користувачів спроектовано три ролі: головний адміністратор (Доступ до усіх таблиць, роботи з ini файлом), адміністратор (доступ до усіх таблиць), відповідальний за сертифікати( доступ тільки до сертифікатів та роботи з ними). Інтерфейс вікна авторизації продемонтсровано на рисунку 3.4.

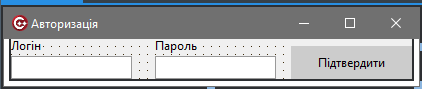


Рисунок 3.4 – Вікно авторизації

У разі успішної авторизації значення у StatusBar змінюється на назву авторизованої ролі та ПІБ працівника. У випадку неуспішної авторизації користувачу видано повідомлення “Спробуйте ще раз”. Щоб вийти з активної ролі розроблено підпункт “Вийти з облікового запису” в головному меню.

Для роботи з інформацією про учасників НМТ створено форму “Робота з нформацією про учасників НМТ”. Інтерфейс форми продемонстровано на рисунк 3.5.

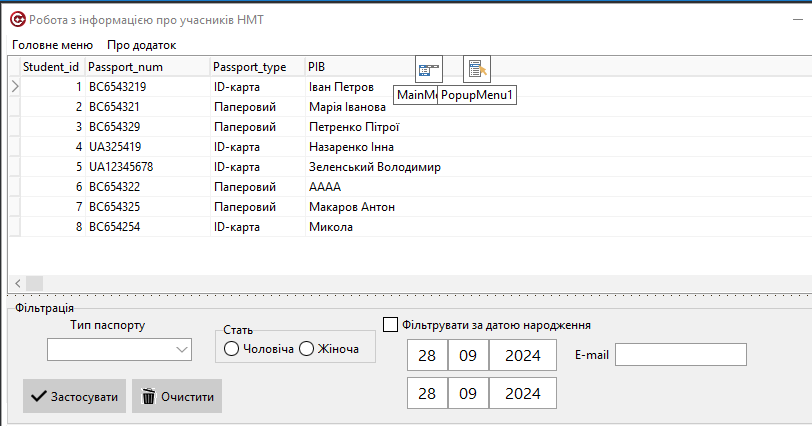


Рисунок 3.5 – Форма “Робота з нформацією про учасників НМТ”

Також для роботи з великою кількістю записів створено функціонал для фільтрації записів, винесений в окремий GroupBox з можливістю комбінації фільтрів. За замовчуванням кнопки для запуску фільтрації та її відміни заблоковані для користувача. Після вибору будь-якого пункту у групі перемикачів RadioGroup “Стать” або встановлення перемикача CheckBox “Фільтрувати за датою народження” чи введення E-mail або вибору типу паспорту з компоненту ComboBox кнопки активуються. Для фільтрації за датою використано об`єкти DatePicker. Також для сортування використано обробник події OnTitleClick для таблиці. Для додавання або редагування даних створено окрему форму та PopUpMenu для вибору над записом, яке прив`язане до об`єкту DBGrid. Інтерфейс вікна роботи з даними продемонстровано на рисунку 3.6. Інтерфейс контекстного меню продемонстровано на рисунку 3.7.

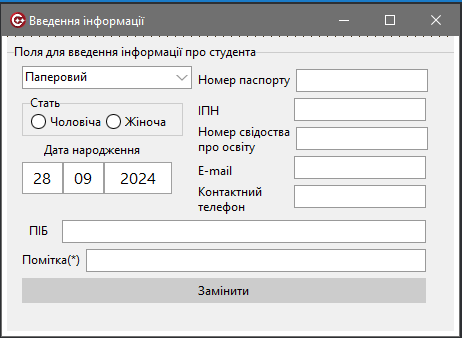


Рисунок 3.6 – Інтерфейс вікна длядодавання та редагування записів

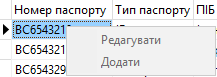


Рисунок 3.7 – Контекстне меню для DBGrid з роллю “відповідальний за сертифікати”

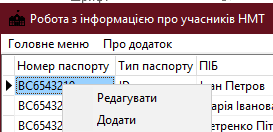


Рисунок 3.8 – Контекстне меню для DBGrid з роллю “адміністратор” або “головний адміністратор”

Для роботи з інформацією про навчальні дисципліни створено форму “Робота з інформацією про предмети”. Інтерфейс форми продемонстровано на рисунку 3.9. Для виведення зображення використано компонент TImage.

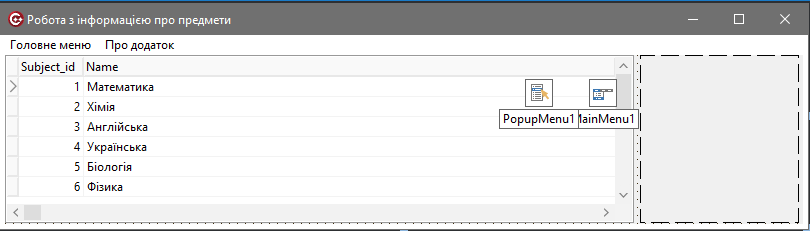


Рисунок 3.9 – Форма “ Робота з інформацією про предмети”

Для додавання та редагування або видалення даних створено окрему форму та компонент PopUpMenu для вибору над записом, яке прив`язане до об`єкту DBGrid. Інтерфейс вікна роботи з даними продемонстровано на рисунку 3.10. На формі використано OpenPictureDialog1 та OpenTextFileDialog1 для додавання зразків завдань та піктограм предметів. Також за замовчування доступний пункт контекстного меню “Подивитися зразки”. Інтерфейс контекстного меню продемонстровано на рисунку 3.11.

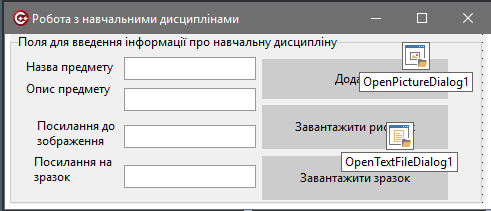


Рисунок 3.10 – Інтерфейс вікна для додавання та редагування записів

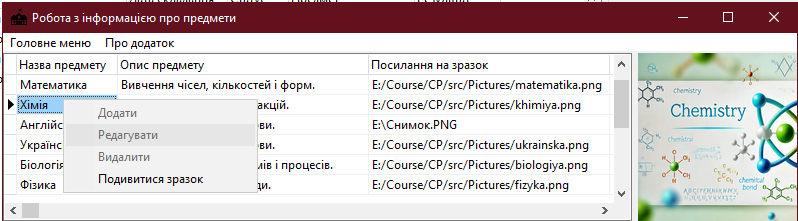


Рисунок 3.11 – Контекстне меню для DBGrid з роллю “відповідальний за сертифікати”

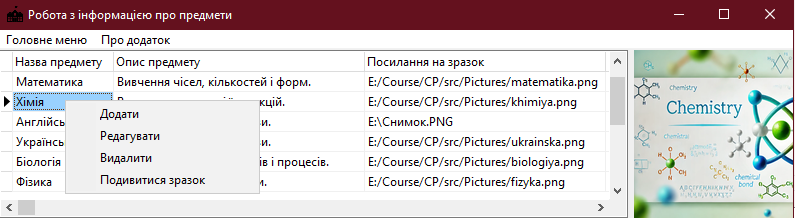


Рисунок 3.12 – Контекстне меню для DBGrid з роллю “адміністратор” або “головний адміністратор”

Для роботи з інформацією про навчальні дисципліни створено форму “Робота з інформацією про умови проходження НМТ”. Інтерфейс форми продемонстровано на рисунку 3.13.

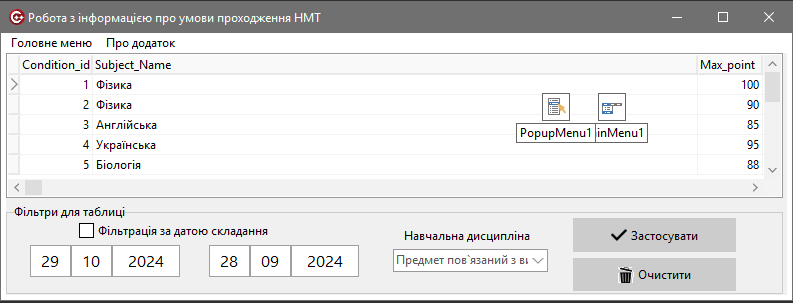


Рисунок 3.13 – Форма “ Робота з інформацією про умови проходження НМТ”

Також для роботи з великою кількістю записів створено функціонал для фільтрації записів, винесений в окремий GroupBox з можливістю комбінації фільтрів. За замовчуванням кнопки для запуску фільтрації та її відміни заблоковані для користувача. Після вибору будь-якої дисципліни у випадаючому списку ComboBox “Навчальні дисципліни” або встановлення перемикача CheckBox “Фільтрувати за датою складання” кнопки активуються. Для фільтрації за датою використано об`єкти DatePicker. Також для сортування використано обробник події OnTitleClick для таблиці.

Для додавання або редагування даних створено окрему форму та компонент PopUpMenu для вибору над записом, яке прив`язане до об`єкту DBGrid. Інтерфейс вікна роботи з даними продемонстровано на рисунку 3.14. Інтерфейс контекстного меню продемонстровано на рисунку 3.15.

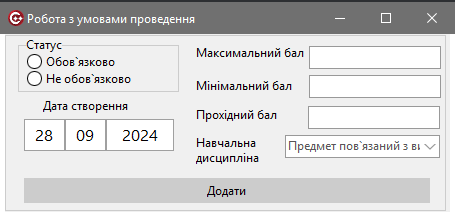


Рисунок 3.14 – Інтерфейс вікна для додавання та редагування записів

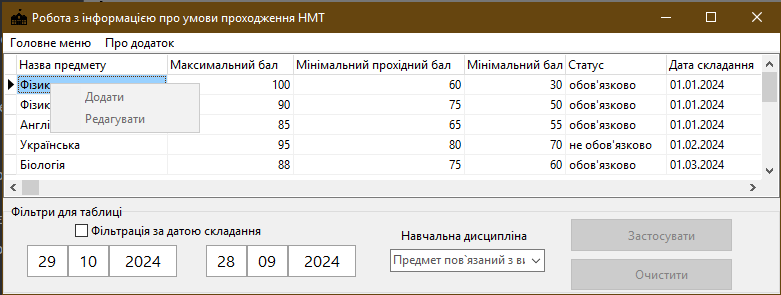


Рисунок 3.15 – Контекстне меню для DBGrid з роллю “відповідальний за сертифікати”

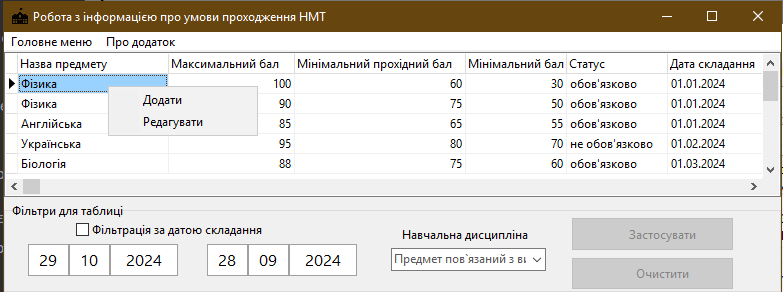


Рисунок 3.16 – Контекстне меню для DBGrid з роллю з роллю “адміністратор” або “головний адміністратор”

Для роботи з інформацією про навчальні дисципліни створено форму “Робота з навчальними закладами”. Інтерфейс форми продемонстровано на рисунку 3.17.

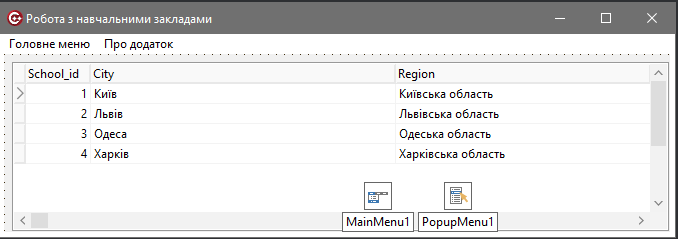


Рисунок 3.17 – Форма “Робота з навчальними закладами ”

Для сортування використано обробник події OnTitleClick для таблиці.

Для додавання або редагування даних, а також для видалення даних створено окрему форму та компонент PopUpMenu для вибору над записом, яке прив`язане до об`єкту DBGrid. Інтерфейс вікна роботи з даними продемонстровано на рисунку 3.18. Інтерфейс контекстного меню продемонстровано на рисунку 3.19.

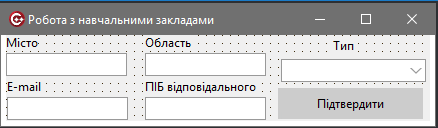


Рисунок 3.18 – Інтерфейс вікна для додавання та редагування записів

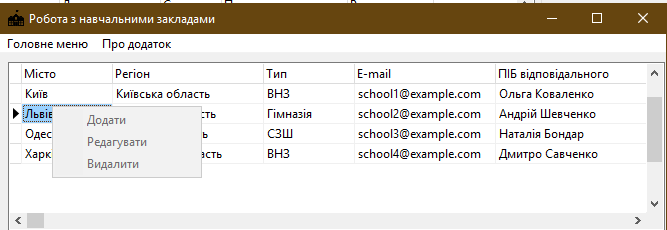


Рисунок 3.19 – Контекстне меню для DBGrid з роллю “відповідальний за сертифікати”

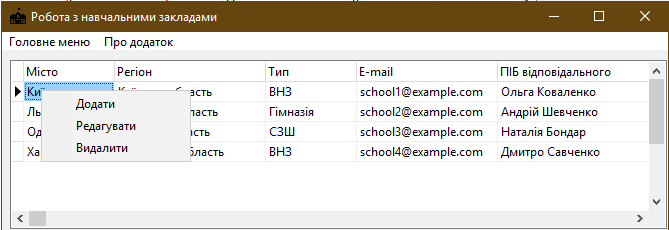


Рисунок 3.20 – Контекстне меню для DBGrid з роллю “адміністратор” або “головний адміністратор”

Для роботи з інформацією про результати тестування створено форму “Робота з інформацією про результати тестування”. Інтерфейс форми продемонстровано на рисунку 3.21.

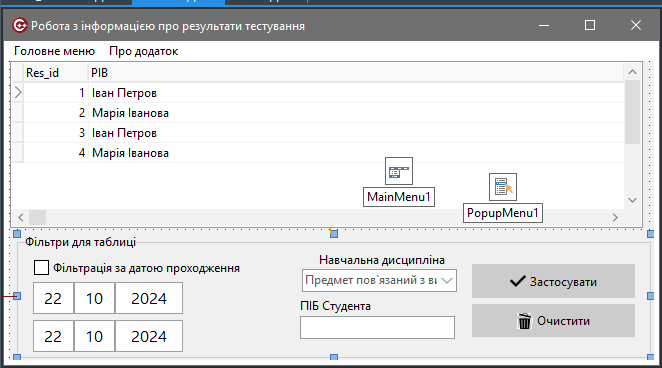


Рисунок 3.21 – Форма “Робота з інформацією про результати тестування ”

Також для роботи з великою кількістю записів створено функціонал для фільтрації записів, винесений в окремий GroupBox з можливістю комбінації фільтрів. За замовчуванням кнопки для запуску фільтрації та її відміни заблоковані для користувача. Після вибору будь-якої дисципліни у компоненті ComboBox “Навчальні дисципліни” або встановлення перемикача CheckBox “Фільтрувати за датою складання” або введення ПІБ студента кнопки активуються. Для фільтрації за датою використано об`єкти DatePicker. Також для сортування використано обробник події OnTitleClick для таблиці.

Для додавання або редагування даних створено окрему форму та компонент PopUpMenu для вибору над записом, яке прив`язане до об`єкту DBGrid. Інтерфейс вікна роботи з даними продемонстровано на рисунку 3.22. Інтерфейс контекстного меню продемонстровано на рисунку 3.23.

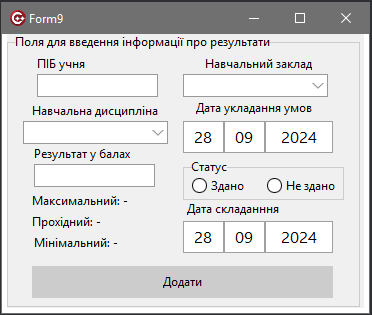


Рисунок 3.22 – Інтерфейс вікна для додавання та редагування записів

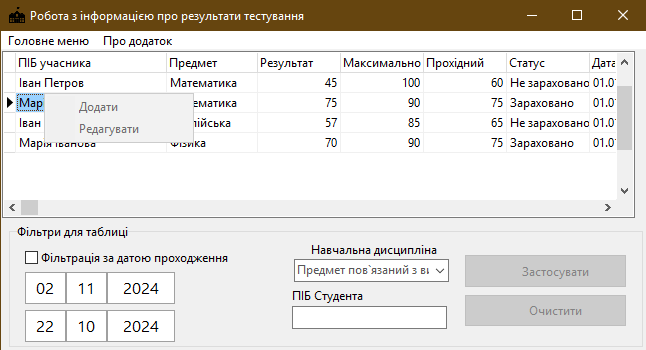


Рисунок 3.23 – Контекстне меню для DBGrid з роллю “відповідальний за сертифікати”

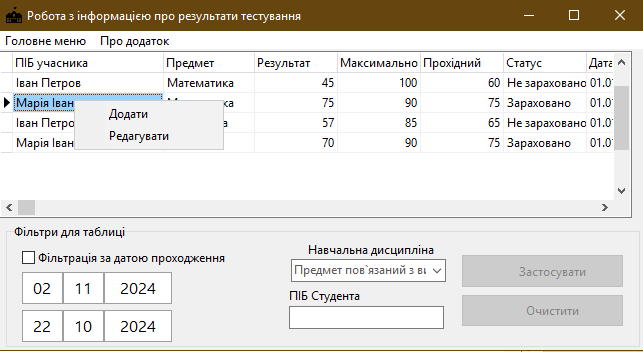


Рисунок 3.24 – Контекстне меню для DBGrid з роллю “адміністратор” або “головний адміністратор”

Для роботи з інформацією про результати тестування створено форму “Робота з інформацією про сертифікати НМТ”. Інтерфейс форми продемонстровано на рисунку 3.25.

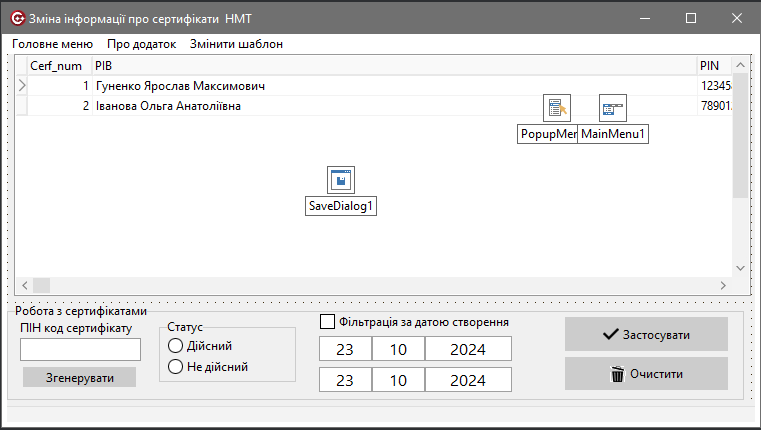


Рисунок 3.25 – Форма “Робота з інформацією про сертифікати НМТ ”

Також для роботи з великою кількістю записів створено функціонал для фільтрації записів, винесений в окремий GroupBox з можливістю комбінації фільтрів. За замовчуванням кнопки для запуску фільтрації та її відміни заблоковані для користувача. Після встановлення пермикача CheckBox “Фільтрувати за датою складання” або вибору пунту у RadioGroup1 кнопки активуються. Для фільтрації за датою використано об`єкти DatePicker. Для збереження сертифікату використано компонент SaveDialog. Також для сортування використано обробник події OnTitleClick для таблиці.

Для додавання або редагування даних створено окрему форму та PopUpMenu для вибору над записом, яке прив`язане до об`єкту DBGrid. Інтерфейс вікна роботи з даними продемонстровано на рисунку 3.26. Інтерфейс контекстного меню продемонстровано на рисунку 3.27.

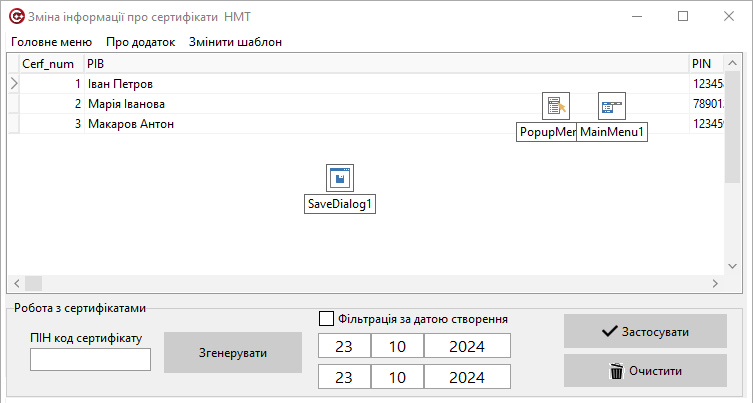


Рисунок 3.26 – Інтерфейс вікна для додавання та редагування записів

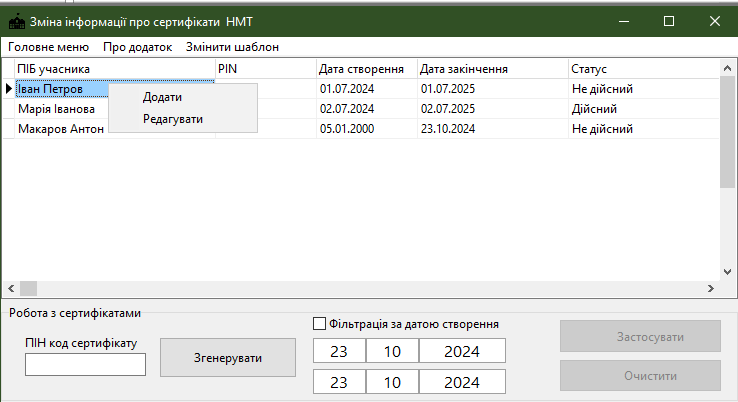


Рисунок 3.28 – Контекстне меню для DBGrid з роллю “адміністратор” або “головний адміністратор” та “відповідальний за сертифікати”

Для відображення кількості виданих сертифікатів на формі створено компонент StatusBar1, що оновлює текст після фільтрації.

Додатково реалізовано вікно “Про додаток”, що містить зображення TImage. Інтерфейс вікна продемонстровано на рисунку 3.29.

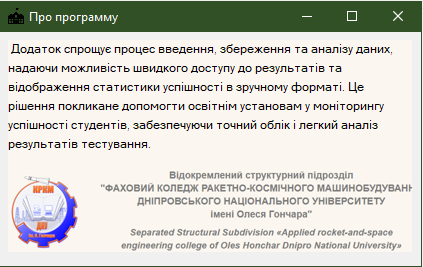


Рисунок 3.28 – Форма “Про додаток”

Для додавання нових користувачів розроблено форму користувачі, що відкривається за пунктом меню “Користувачі”, доступ до якого мають лише користувачі з ролями “головний адміністратор” та “адмністратор”. Інтерфейс форми продемонстровано на рисунку 3.29.

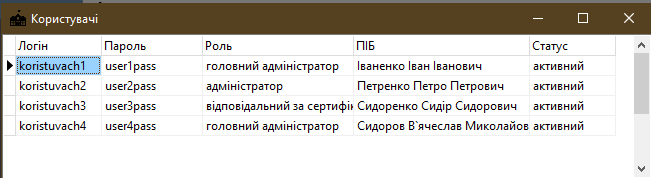


Рисунок 3.29 – Форма “Користувачі”

1. Контроль вхідних даних програми

У програмі для контролю вхідних даних було використано регулярні вираження класу std::regex, приклади регулярного виразів та створення валідаторів на введення даних продемонстровано у лістингах 3.4–3.5.

Лістинг 3.4 – Приклад регулярного виразу

void \_\_fastcall TForm11::LabeledEdit1Exit(TObject \*Sender)

{

String pib = LabeledEdit1->Text;

UnicodeString pattern = "^[А-ЯІЇЄҐа-яіїєґ' ]+$";

if (!TRegEx::IsMatch(pib, pattern)) {

ShowMessage("ПІБ повинен містити лише українські літери.");

LabeledEdit1->SetFocus();

return;

}

}

Лістинг 3.5 – Приклад регулярного виразу

void \_\_fastcall TForm14::Edit5Exit(TObject \*Sender)

{

String e-mail = Edit5->Text;

UnicodeString pattern = "^\\w+([-+.']\\w+)\*@\\w+([-.]\\w+)\*\\.\\w+([-.]\\w+)\*$";

if (!TRegEx::IsMatch(email, pattern)) {

ShowMessage("Введіть дійсний e-mail.");

Edit5->SetFocus();

}

}

Для уникнення помилок введення дати та вибору полів для заповнення таблиць, використано об`єкти класів TDatePicker для зручності введення часу з перевіркою на логічні межі дати та ComboBox для вибору полів з інших таблиць, наприклад вибір навчальної дисципліни у введенні результату. Об`єкти продемонстровано на рисунках 3.30-3.31.

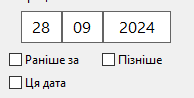


Рисунок 3.30 – Введення дати для фільтрації

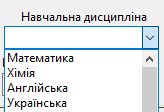


Рисунок 3.31 – Вибір навчальної дисципліни

Для коректного введення числових значень використано властивість NumbersOnly для LabeledEdit. Налаштування властивості продемонстровано на рисунку 3.32.



Рисунок 3.32 – Налаштування Edit

Для розмеження доступу до роботи з даними використано дезактивацію пунктів меню для роботи з даними за ролями. Блокування продемонстровано на рисунку 3.33.



Рисунок 3.33 – Заблоковане меню

Для повідомлення про некоректне введення використано повідомлення для користувача. Приклад помилки продемонстровано на рисунку 3.34.

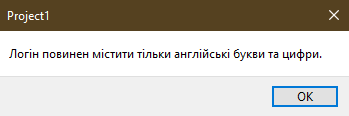


Рисунок 3.34 – Вікно помилки

Для підтвердження видалення використано діалогове вікно. Приклад вікна продемонстровано на рисунку 3.35.

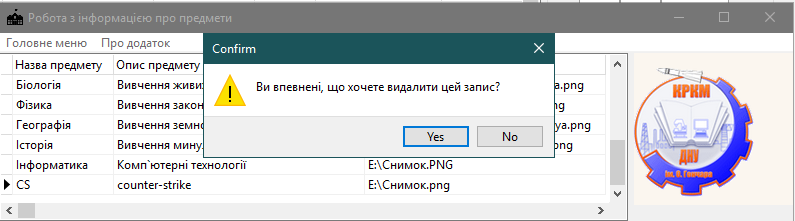


Рисунок 3.35 – Вікно помилки

4 ІНСТРУКЦІЯ З КОРИСТУВАННЯ ПРОГРАМНОЮ СИСТЕМОЮ

Для початку роботи з програмою необхідно ознайомитися з її основним інтерфейсом та функціоналом, щоб зрозуміти, як виконувати облік і керування даними. У програмі реалізовано декілька ролей з різними рівнями доступу до даних, що дозволяє забезпечити безпечне і ефективне управління інформацією. Кожна роль має свої права, що визначають доступ до певних функцій, таких як перегляд, редагування, додавання або видалення записів.

Програма створена для полегшення обліку результатів НМТ, керування даними про навчальні заклади, дисципліни та сертифікати. Користувач може легко отримувати доступ до необхідної інформації, використовувати фільтрацію, а також генерувати сертифікати у форматі HTML для подальшого використання. Інструкція покликана допомогти зрозуміти основні принципи роботи програми, а також полегшити використання доступних інструментів і функцій.

Головне правило введення даних у поля полягає в тому, що символи, які не мають сенсу для конкретного поля, вводити заборонено. Для всіх інших полів передбачені вбудовані перевірки та варіанти вибору.

При завантаженні додатка відображається вікно авторизації, в якому представлено поля логін та пароль для авторизації.

У разі першого запуску прогама вимагає авторизації користувача з роллю “головний адміністратор” для створення шаблону для сертифікатів. Уінші випадку передбачено повідомлення про помилку.

Наступним кроком відображається головне вікно, в якому представлено стислий зміст інформації, що зберігається у базі даних. У вікні містяться ПІБ особи, яка складала НМТ, дата складання, навчальна дисципліна та статус здачі або не здачі.

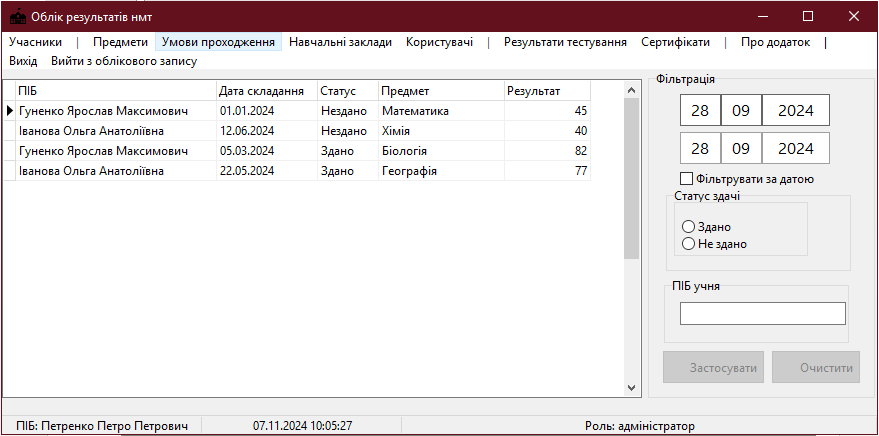


Рисунок 4.1 – Головне вікно

Щоб відсортувати записи у таблиці, достатньо натиснути на заголовок стовпця, щоб упорядкувати записи за цим стовпцем.

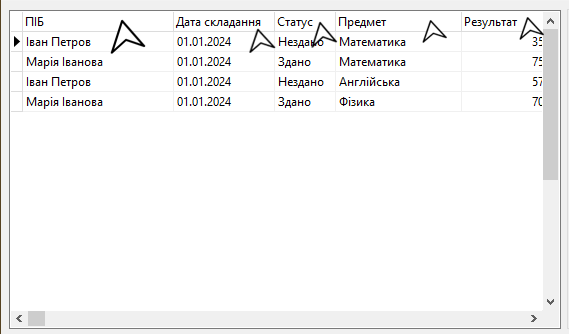


Рисунок 4.2 – Спосіб сортування

Також для зручного перегляду даних можна скористатися опцією фільтрації. За умови відсутності заповнених фільтрів кнопки запуску фільтрації та очищення фільтрів неактивні. Для їх активації необхідно заповнити хоча б один із запропонованих фільтрів, як показано на рисунку 4.3. Для фільтрації за датою потрібно вибрати дату у випадаючому списку та один із доступних варіантів у CheckBox. Для фільтрації за статусом здачі необхідно обрати потрібний варіант із двох кружечків. Для фільтрації за ПІБ учня потрібно ввести ПІБ у відповідне поле, при цьому дозволені лише українські літери, пробіли та апостроф. Фільтри можна комбінувати, обираючи декілька варіантів одночасно.

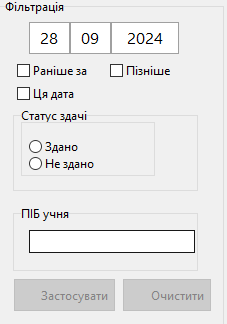


Рисунок 4.3 – Список можливих фільтрів

Для переходу до роботи з іншими таблицями передбачено головне меню, яке відкриває відповідне вікно залежно від обраного пункту. Пункти головного меню продемонстровані на рисунку 4.4. Для отримання розширеного доступу необхідно пройти авторизацію, вибравши пункт головного меню “Вийти з облікового запису”. Після цього відкриється форма авторизації з полями для введення логіна та пароля користувача (рисунок 4.6). У разі успішної авторизації за однією з чотирьох можливих ролей (головний адміністратор, адміністраторо, відповідаьний за сертифікати) статусбар оновлюється, відображаючи поточну роль користувача. Усі можливі ролі продемонстровані на рисунках 4.7–4.9.

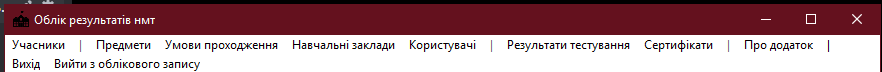


Рисунок 4.4 – Головне меню



Рисунок 4.5 – Статусбар

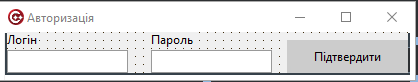


Рисунок 4.6 – Форма авторизації



Рисунок 4.7 – Роль “Адмінстратор”



Рисунок 4.8 – Роль “Відповідальний за сертифікати”



Рисунок 4.9 – Роль “Головний адміністратор”

Після вибору пункту меню "Учні" відкривається вікно, яке демонструє таблицю з інформацією про учнів. У таблиці відображаються такі поля: номер паспорта, тип паспорта, ПІБ, дата народження, стать, E-mail, номер телефону, номер посвідчення та ІПН. Ця інформація забезпечує зручний перегляд та швидкий доступ до основних даних учнів.

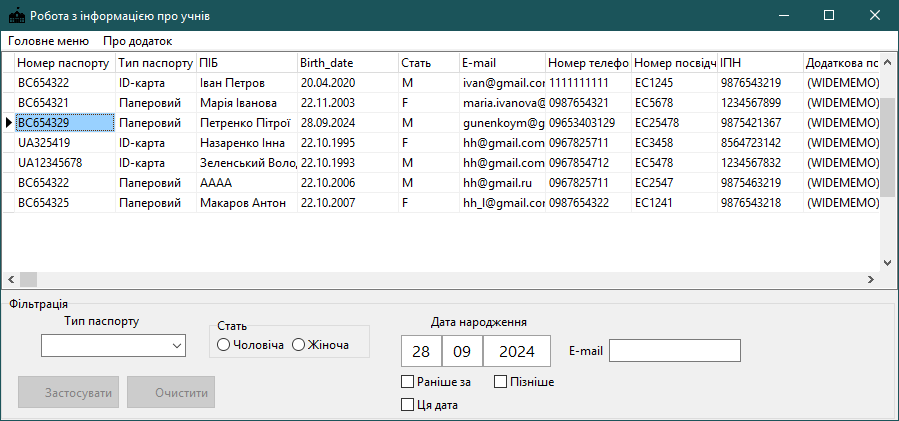


Рисунок 4.10 – Вікно з інформацією про учасників

Для сортування записів у таблиці потрібно натиснути на заголовок відповідного стовпця, що дозволяє впорядкувати записи за обраним критерієм.

Додаток також підтримує функцію фільтрації даних. Якщо фільтри не заповнені, кнопки для запуску фільтрації та очищення фільтрів залишаються неактивними. Для активації необхідно заповнити хоча б один із запропонованих фільтрів у нижній частині форми. Можливі фільтри за типом паспорта, статтю, датою народження та полем E-mail. Для фільтрації за датою народження можна вибрати конкретну дату та встановити один із варіантів у чекбоксах: "Раніше за," "Пізніше" або "Ця дата." Фільтри можна комбінувати для звуження пошуку. Тип паспорта обирається зі списку, стать — із двох варіантів, а для введення E-mail необхідно використовувати шаблон “*@*.\*”.

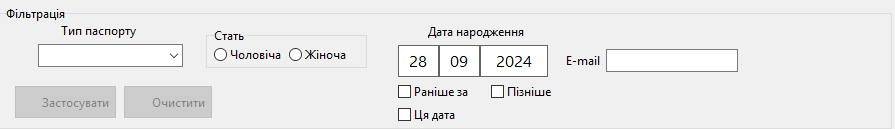


Рисунок 4.11 – Список можливих фільтрів

Окрім цього, користувачам з ролями "адміністратор" та “головний адміністратор” доступна можливість редагування або додавання записів через контекстне меню (PopupMenu) для таблиці, яке відкривається правою кнопкою миші. За допомогою цього меню можна обрати опції "Редагувати" або "Додати" для керування даними учнів.

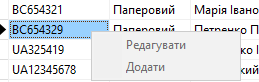


Рисунок 4.12 – PopupMenu для редагування/додавання записів без доступу

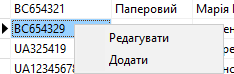


Рисунок 4.12 – PopupMenu для редагування/додавання записів з доступом

Після вибору пункту меню "Редагування" відкривається вікно з полями таблиці "Учні," заповненими відповідно до даних у базі. Кожне поле перевіряється на коректність введення, унікальність та логічний зміст. При виборі пункту меню "Додавання" відкривається вікно з порожніми полями для введення нових даних у таблицю "Учні." Кожне поле також перевіряється на коректність, унікальність та логічний зміст. У разі успішного додавання або зміни запису з'являється повідомлення про успішне виконання операції. У разі виникнення помилки з'являється повідомлення з її описом.

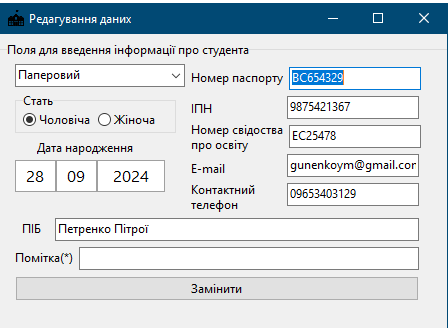


Рисунок 4.13 – Вікно редагування

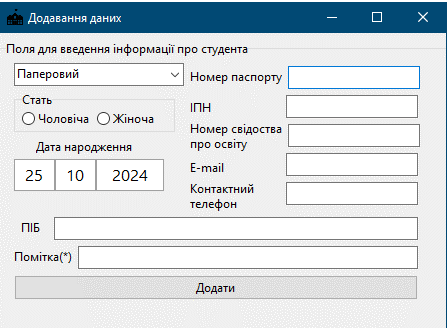


Рисунок 4.14 – Вікно додавання

Після натискання на пункт меню "Навчальні дисципліни" відкривається вікно, яке відображає таблицю з інформацією про навчальні предмети. У таблиці містяться такі поля: назва предмета, опис предмета, посилання на зображення та зразок завдань. Це забезпечує швидкий перегляд доступних дисциплін та надає доступ до візуального зразка для кожного предмета.

Для додавання або зміни інформації про предмет у вікні доступні функції додавання, редагування та видалення записів через контекстне меню, яке відкривається правою кнопкою миші. Ці дії доступні лише для користувачів із роллю "адміністратор" або “головний адмністратор”. Вони можуть керувати записами, редагуючи назви, опис, а також додаючи зображення для предметів.

Користувач може завантажити зображення для предмета двома способами: через вибір файлу з діалогу за допомогою кнопки "Обрати зображення" або вручну ввівши шлях до файлу зображення в рядок. Якщо файл не знайдено або він має неправильний формат, система повідомляє про помилку. Підтримуються файли з розширенням .jpg або .png.

Користувач також може завантажити зразок завдань двома способами: через вибір файлу з діалогу за допомогою кнопки "Обрати зразок" або вручну ввівши шлях до файлу у рядок. Якщо файл не знайдено або він має неправильний формат, система повідомляє про помилку. Підтримуються файли з розширенням .docx або .pdf.

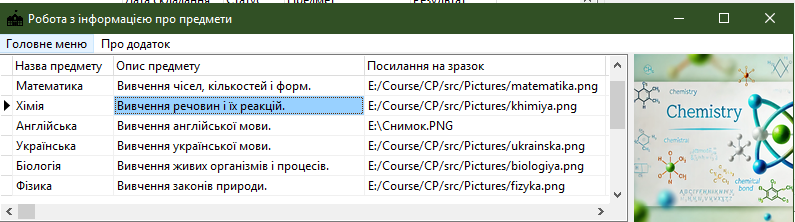


Рисунок 4.15 – Вікно з інформацією про навчальні дисципліни

Коли користувач натискає кнопку "Додати," відкривається вікно для введення нових даних: назви предмета, його опису та шляху до зображення. Усі поля мають бути заповнені коректно. Під час спроби збереження система перевіряє правильність введення даних і повідомляє про можливі помилки. Наприклад, назва предмета не повинна перевищувати 100 символів, а шлях до зображення та зразка має бути дійсним.

Після успішного додавання або редагування запису система оновлює дані в таблиці та виводить повідомлення про успішне збереження інформації.

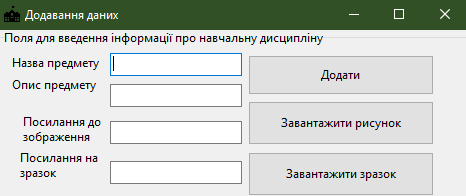


Рисунок 4.16 – Вікно додавання нового предмету

Після вибору пункту "Редагувати" система відкриває вікно з попередньо заповненими полями для редагування даних про предмет. За потреби користувач може оновити назву, опис або шлях до зображення, попередньо перевіривши його доступність перед збереженням. Усі зміни після збереження негайно відображаються в таблиці.

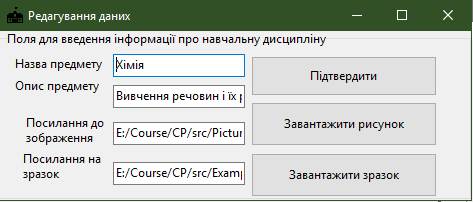


Рисунок 4.17 – Вікно редагування предмету

У разі видалення запису користувач отримає попередження, якщо предмет використовується в інших таблицях, і запис не може бути видалений.

Після натискання на пункт меню "Умови проходження" відкривається вікно з таблицею, що містить інформацію про умови проходження НМТ. У таблиці відображаються такі поля, як назва предмета, максимальний бал, мінімальний прохідний бал, мінімальний бал, статус та дата складання. Це дозволяє зручно переглядати та аналізувати умови складання для різних навчальних дисциплін.

Додаток підтримує фільтрацію даних. Користувач може обрати конкретну дату складання за допомогою селектора дати та вказати умову: "раніше за," "пізніше" або "ця дата," що дозволяє зосередитися на умовах, які діяли до або після певної дати, або точно на обрану дату. Також є можливість фільтрації за навчальною дисципліною, що дає змогу переглядати умови складання лише для конкретного предмета. Записи у таблиці автоматично оновлюються відповідно до вибраних критеріїв. Якщо жоден фільтр не обрано, кнопки для запуску фільтрації та очищення фільтрів залишаються неактивними.

Користувачі із роллю "адміністратор" або “головний адмністратор” можуть додавати та редагувати записи через контекстне меню, яке відкривається правою кнопкою миші. При додаванні нового запису відкривається вікно для введення даних: максимального балу, мінімального прохідного балу, мінімального балу, статусу (обов'язковий чи необов'язковий), дати складання, а також вибору навчальної дисципліни з випадаючого списку. Після введення даних користувач натискає кнопку "Додати," і система перевіряє коректність введених значень, зокрема перевіряється, щоб максимальний бал був більшим за мінімальний, а мінімальний прохідний бал не перевищував максимальний.

Якщо під час редагування запису користувач обирає запис з таблиці та натискає пункт "Редагувати," система відкриває вікно з попередньо заповненими даними, які можна змінити. Після успішного редагування запису система оновлює таблицю.

Якщо користувач бажає повернутися до перегляду всіх записів без застосованих фільтрів, він може скористатися кнопкою "Очистити," яка скидає всі фільтри і повертає повний список умов складання у таблиці.

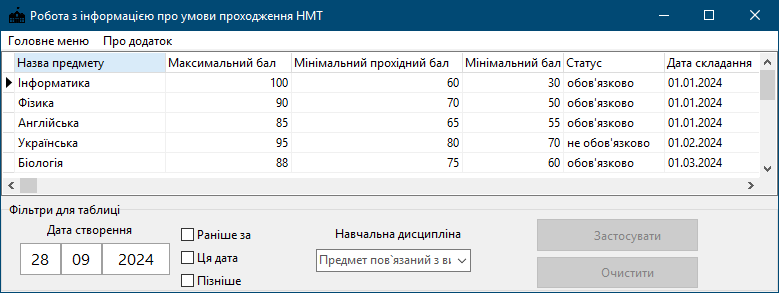


Рисунок 4.18 – Вікно з умовами проходження НМТ

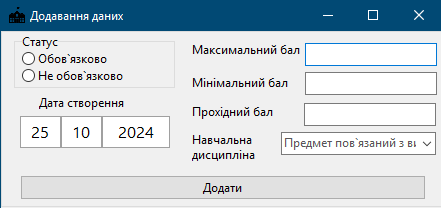


Рисунок 4.19 – Вікно додавання нових умов

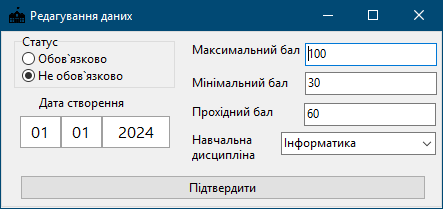


Рисунок 4.20 – Вікно редагування умов

Після натискання на пункт меню "Навчальні заклади" відкривається вікно з таблицею, що містить інформацію про навчальні заклади. Таблиця включає такі поля: місто, регіон, тип, E-mail та ПІБ відповідальної особи. Це дозволяє користувачам швидко переглядати список навчальних закладів з основною інформацією про кожен заклад.

Для додавання, редагування або видалення записів у таблиці доступне контекстне меню, яке відкривається правою кнопкою миші. Ці дії можуть виконувати лише користувачі із роллю "адміністратор" або “головний адмністратор”. Вони можуть додавати нові записи або редагувати наявні, зокрема оновлювати назву міста, регіон, тип, E-mail та ПІБ відповідальної особи.

Записи для кожного навчального закладу можна додати за допомогою кнопки "Додати," яка відкриває вікно введення даних: міста, регіону, типу закладу, E-mail та ПІБ відповідальної особи. Під час додавання або редагування запису система перевіряє коректність введених даних, наприклад, щоб E-mail мав дійсний формат, а введені символи відповідали українській мові.

У разі видалення запису користувач отримає попередження, якщо даний заклад використовується в інших таблицях, зокрема у таблиці "result." У такому випадку запис не може бути видалений.

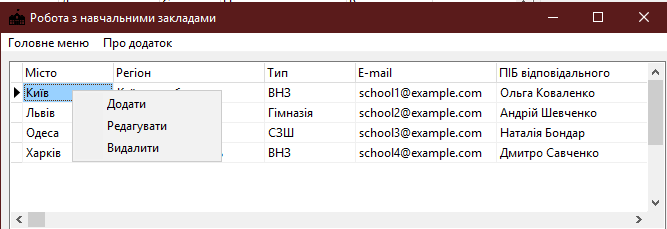


Рисунок 4.21 – Вікно роботи з навчальними закладами

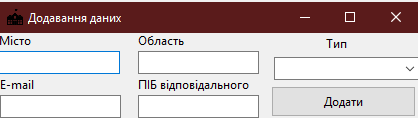


Рисунок 4.22 – Вікно додавання даних

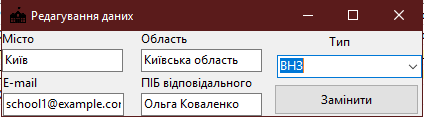


Рисунок 4.23 – Вікно редагування даних

Після натискання на пункт меню "Результати тестування" користувач із роллю "адміністратор" або “головний адмністратор” відкриває вікно з таблицею, що містить дані про результати тестування. У таблиці представлені такі поля: ПІБ студента, назва предмета, результат (бал), максимальний бал, прохідний бал, статус (зараховано/не зараховано), дата складання, дата укладання умов та E-mail навчального закладу. Це дозволяє користувачеві швидко переглядати результати тестів і фільтрувати їх за необхідними критеріями.

Для додавання, редагування та видалення записів доступне контекстне меню, яке відкривається правою кнопкою миші. Користувачі з роллю "Відповідальний за результати та сертифікати" можуть керувати записами, зокрема додавати нові результати тестування з вказанням ПІБ студента, предмета, балу, статусу та дати укладання умов, а також редагувати наявні записи з можливістю змінювати інформацію про студента, предмет або результат тесту. При видаленні запису система перевіряє наявність залежностей у таблиці результатів або сертифікатів і попереджає користувача, якщо запис не може бути видалений.

Функціонал фільтрації дозволяє обирати результати за різними критеріями, такими як ПІБ студента, назва предмета та дата складання (раніше, пізніше або обрана дата). При натисканні на кнопку "Застосувати" активується обраний користувачем фільтр, а при натисканні на кнопку "Очистити" всі параметри фільтрації скидаються, і таблиця відображає повний список записів.

Після додавання, редагування або видалення запису система автоматично оновлює дані в таблиці, щоб відобразити актуальну інформацію.

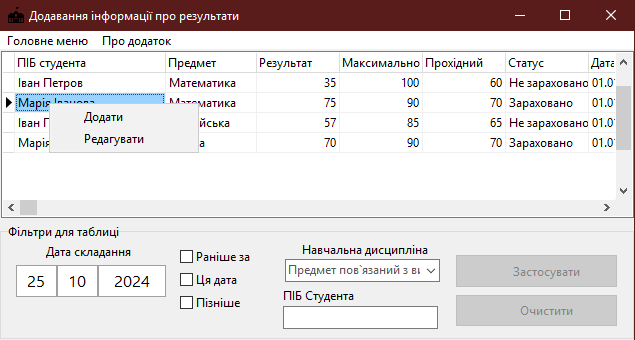


Рисунок 4.24 – Вікно роботи з результатами тестування



Рисунок 4.25 – Вікно додавання даних

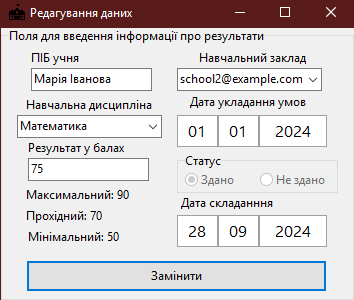


Рисунок 4.26 – Вікно редагування даних

Користувач із роллю "Відповідальний за сертифікати" або із роллю "адміністратор" або “головний адмністратор” має доступ до пункту меню "Сертифікати," який відкриває вікно для управління інформацією про сертифікати студентів. У таблиці відображаються такі дані: ПІБ студента, PIN-код сертифіката, дати створення та завершення дії сертифіката, а також його статус ("Дійсний" або "Не дійсний").

Користувач може фільтрувати сертифікати за датою створення, обираючи конкретну дату або інтервал (раніше чи пізніше обраної дати), а також сортувати дані в таблиці, натискаючи на заголовки стовпців.

Додатково користувач має можливість генерувати HTML-файл сертифіката. Для цього потрібно ввести PIN-код студента, натиснути кнопку "Згенерувати," і система створить HTML-документ, який містить особисті дані студента, статус сертифіката та результати тестів. Результати оцінюються за 200-бальною шкалою, відображаючи набрані студентом бали.

У меню "Сертифікати" є контекстне меню, яке дозволяє додавати нові записи або редагувати наявні. При виборі команди "Додати" відкривається вікно для введення даних нового сертифіката. Користувач вводить ПІБ студента, PIN-код, дати створення та завершення дії сертифіката та обирає статус. Якщо PIN-код вже існує, система виводить повідомлення про помилку. Після успішного введення даних, натискання кнопки "Додати" зберігає новий сертифікат у базі.

Для редагування сертифіката обирається команда "Редагувати," що відкриває вікно з уже заповненими полями. Користувач може змінити будь-які дані, після чого натискає кнопку "Зберегти," щоб зберегти зміни в базі. Також у додатку предебачено виведення кількості записів у разі фільтрації записів про сертифікатів.

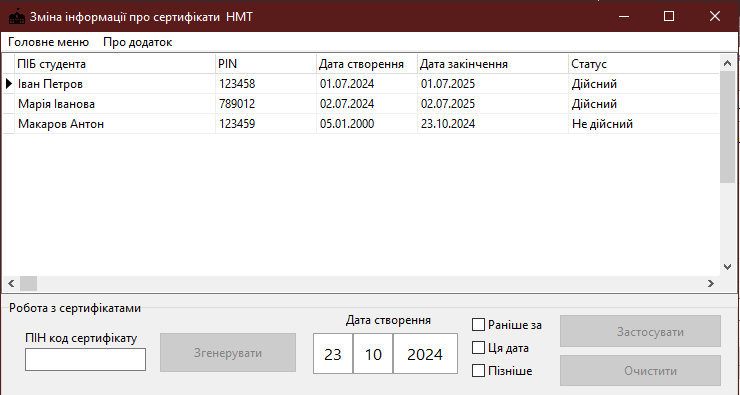


Рисунок 4.27 – Вікно роботи з сертифікатами



Рисунок 4.28 – Вікно додавання даних

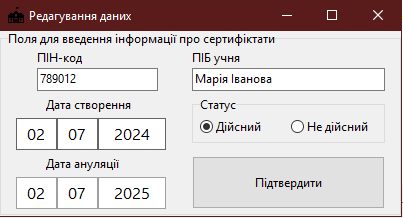


Рисунок 4.29 – Вікно редагування даних

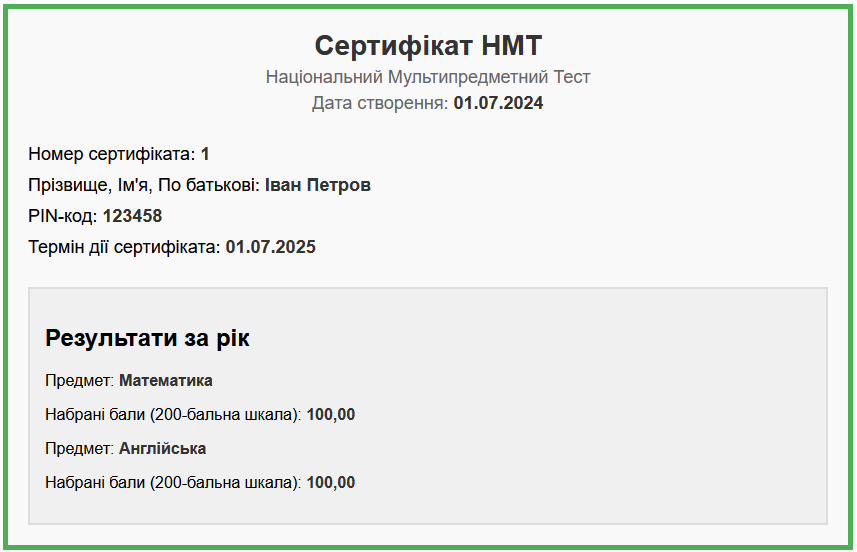


Рисунок 4.30 – Створений сертифікат

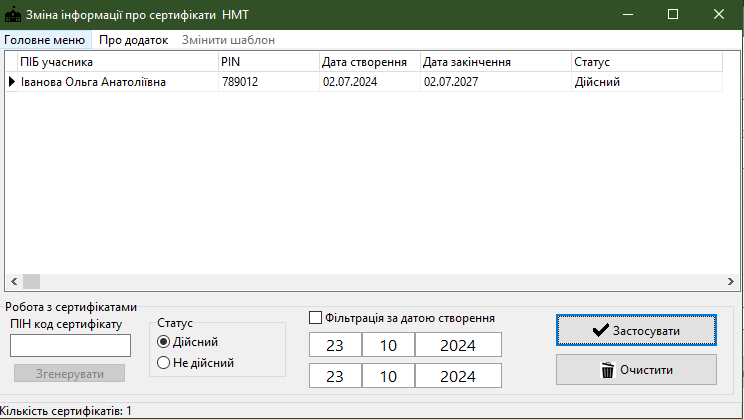


Рисунок 4.31 – Виведення кількості сертифікатів

Висновки

У процесі розробки курсового проєкту «Програма ведення обліку здачі НМТ» було спроектовано та реалізовано програмний додаток, який може використовуватися як великими центрами оцінювання, так і малими освітніми установами.

Розроблене програмне забезпечення є зручним для користувачів із різними рівнями навичок роботи з комп’ютером. Додаток має інтуїтивний інтерфейс, підтримує фільтрацію та сортування даних, а також можливість генерувати сертифікати у форматі HTML.

У результаті реалізації функцій додатка забезпечено:

* облік даних про студентів, їх результати тестування та навчальні дисципліни;
* редагування та видалення даних з перевіркою коректності введення та збереженням цілісності бази даних;
* пошук та фільтрацію за різними критеріями, що дозволяє швидко знаходити потрібну інформацію;
* управління доступом до даних, що забезпечує захист інформації.

Створений додаток допомагає організувати облік результатів НМТ та аналізувати отримані дані, що сприяє покращенню управління процесом тестування.

Список Використаних ДжЕРЕЛ

* + - 1. Страуструп, Б. Мова програмування C++ [Текст] / Б. Страуструп. - М. : Вильямс, 2002. - 1104 с.
      2. Шилдт, Г. Повний довідник з C++ [Текст] / Г. Шилдт. - К. : BHV, 2019. - 832 с.
      3. Ліппман, С. Основи програмування на C++ [Текст] / С. Ліппман, Б. Лахоуд. - М. : Пітер, 2015. - 672 с.
      4. Стандарт C++ [Текст] : ISO/IEC 14882:2011. Офіційна специфікація мови C++ / Міжнародна організація зі стандартизації. - Женева : ISO, 2011. - 1336 с.
      5. Документація до стандартної бібліотеки шаблонів (STL) [Текст] : Детальний опис контейнерів, алгоритмів та ітераторів. - Базується на офіційних специфікаціях.
      6. X3 Secretariat. Standard — The C Language [Текст] / X3J11/90-013. Computer and Business Equipment Manufactures Association, 311 First Street, NW, Suite 500, Washington, DC 20001, USA.
      7. Бауманская Википедия [Електронний ресурс] - Режим доступу: https://ru.bmstu.wiki.
      8. Офіційний сайт MySQL / Documentation [Електронний ресурс] - Режим доступу: https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html.
      9. MySQL [Електронний ресурс] - Режим доступу: http://www.mysql.ru.
      10. Wikipedia. MySQL [Електронний ресурс] - Режим доступу: https://ru.wikipedia.org/wiki/MySQL.

Додаток А

Use-case діаграму ролей створеного додатку продемонстрвано на рисунку А.1.

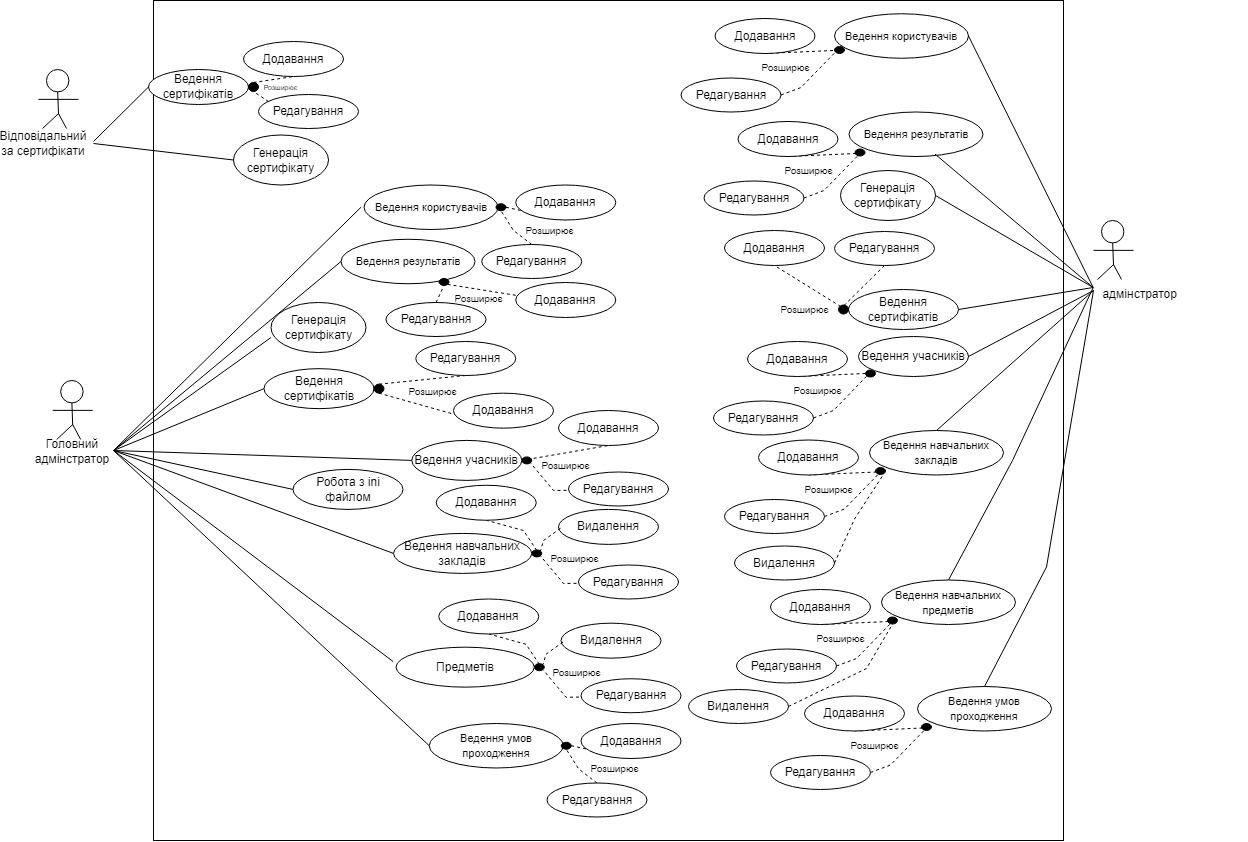


Рисунок А.1 – Use-case діаграма

Додаток Б

CREATE DATABASE NMT\_results;

USE NMT\_results;

-- Створення таблиці "Student"

CREATE TABLE Student (

Student\_id INT PRIMARY KEY AUTO\_INCREMENT,

Passport\_num VARCHAR(20),

Passport\_type VARCHAR(15) CHECK (Passport\_type IN ('ID-карта', 'Паперовий')),

PIB VARCHAR(100),

Birth\_date DATE,

Gender CHAR(1) CHECK (Gender IN ('M', 'F')),

Email VARCHAR(100),

Phone\_num VARCHAR(15) CHECK (Phone\_num REGEXP '^[0-9]+$'),

EduCerf\_num VARCHAR(30),

PN VARCHAR(10),

Additional TEXT

);

-- Створення таблиці "Users" з новими ролями

CREATE TABLE Users (

Login VARCHAR(50) PRIMARY KEY,

Password VARCHAR(50),

Role VARCHAR(30) CHECK (Role IN ('користувач', 'менеджер предметів', 'менеджер результатів', 'менеджер учнів'))

);

-- Створення таблиці "School" з новими типами

CREATE TABLE School (

School\_id INT PRIMARY KEY AUTO\_INCREMENT,

City VARCHAR(50),

Region VARCHAR(50),

Type VARCHAR(20) CHECK (Type IN ('ВНЗ', 'СЗШ', 'Гімназія', 'Ліцей')),

Email VARCHAR(100),

Teacher\_PIB VARCHAR(100)

);

-- Створення таблиці "Subject" з додатковим полем "Image\_name"

CREATE TABLE Subject (

Subject\_id INT PRIMARY KEY AUTO\_INCREMENT,

Name VARCHAR(100),

Description TEXT,

Image\_name VARCHAR(255)

);

CREATE TABLE Conditions (

Condition\_id INT PRIMARY KEY AUTO\_INCREMENT,

Subject\_id INT,

Max\_point INT,

Min\_r\_point INT,

Min\_point INT,

Status BOOLEAN,

Date DATE,

FOREIGN KEY (Subject\_id) REFERENCES Subject(Subject\_id),

CHECK (Max\_point > Min\_point AND Min\_r\_point < Max\_point AND Min\_r\_point > Min\_point)

);

-- Створення таблиці "Result"

CREATE TABLE Result (

Res\_id INT PRIMARY KEY AUTO\_INCREMENT,

Subj\_id INT,

Condition\_id INT,

Student\_id INT,

Reached\_score INT,

Status BOOLEAN,

School\_id INT,

Attemp\_date DATE,

FOREIGN KEY (Subj\_id) REFERENCES Subject(Subject\_id),

FOREIGN KEY (Condition\_id) REFERENCES Conditions(Condition\_id),

FOREIGN KEY (Student\_id) REFERENCES Student(Student\_id),

FOREIGN KEY (School\_id) REFERENCES School(School\_id)

);

-- Створення таблиці "Certificate"

CREATE TABLE Certificate (

Cerf\_num INT PRIMARY KEY AUTO\_INCREMENT,

Student\_id INT,

PIN VARCHAR(15) CHECK (PIN REGEXP '^[0-9]+$'),

Creation\_date DATE,

Effect\_time DATE,

Status BOOLEAN,

FOREIGN KEY (Student\_id) REFERENCES Student(Student\_id)

);

-- Заповнення таблиці "Student"

INSERT INTO Student (Passport\_num, Passport\_type, PIB, Birth\_date, Gender, Email, Phone\_num, EduCerf\_num, PN, Additional) VALUES

('AB123456', 'ID-карта', 'Іван Петров', '2002-04-15', 'M', 'ivan.petrov@example.com', '1234567890', 'EC1234', '9876543210', 'Без додаткової інформації'),

('BC654321', 'Паперовий', 'Марія Іванова', '2003-11-22', 'F', 'maria.ivanova@example.com', '0987654321', 'EC5678', '1234567891', 'Без додаткової інформації');

INSERT INTO Users (Login, Password, Role) VALUES

('koristuvach1', 'user1pass', 'менеджер предметів'),

('koristuvach2', 'user2pass', 'менеджер результатів'),

('koristuvach3', 'user3pass', 'менеджер учнів');

-- Заповнення таблиці "School"

INSERT INTO School (City, Region, Type, Email, Teacher\_PIB) VALUES

('Київ', 'Київська область', 'Ліцей', 'school1@example.com', 'Ольга Коваль'),

('Львів', 'Львівська область', 'СЗШ', 'school2@example.com', 'Андрій Шевченко'),

('Одеса', 'Одеська область', 'Гімназія', 'school3@example.com', 'Наталія Бондар'),

('Харків', 'Харківська область', 'ВНЗ', 'school4@example.com', 'Дмитро Савченко');

-- Заповнення таблиці "Subject" з новими предметами

INSERT INTO Subject (Name, Description, Image\_name) VALUES

('Математика', 'Вивчення чисел, кількостей і форм.', 'matematika.jpg'),

('Хімія', 'Вивчення речовин і їх реакцій.', 'khimiya.jpg'),

('Англійська', 'Вивчення англійської мови.', 'angliyska.jpg'),

('Українська', 'Вивчення української мови.', 'ukrainska.jpg'),

('Біологія', 'Вивчення живих організмів і процесів життя.', 'biologiya.jpg'),

('Фізика', 'Вивчення законів природи.', 'fizyka.jpg'),

('Географія', 'Вивчення земної поверхні і клімату.', 'geografiya.jpg'),

('Історія', 'Вивчення минулих подій і епох.', 'istoriya.jpg');

INSERT INTO Conditions (Subject\_id, Max\_point, Min\_r\_point, Min\_point, Status, Date) VALUES

(1, 100, 60, 30, TRUE, '2024-01-01'),

(2, 90, 70, 50, TRUE, '2024-01-01'),

(3, 85, 65, 55, TRUE, '2024-01-01'),

(4, 95, 80, 70, FALSE, '2024-02-01'),

(5, 88, 75, 60, TRUE, '2024-03-01');

-- Заповнення таблиці "Result"

INSERT INTO Result (Subj\_id, Condition\_id, Student\_id, Reached\_score, Status, School\_id, Attemp\_date) VALUES

(1, 1, 1, 85, TRUE, 1, '2024-06-15'),

(2, 2, 2, 75, TRUE, 2, '2024-06-16'),

(3, 3, 1, 60, FALSE, 3, '2024-06-17');

-- Заповнення таблиці "Certificate" з числовими значеннями в PIN

INSERT INTO Certificate (Student\_id, PIN, Creation\_date, Effect\_time, Status) VALUES

(1, '123456', '2024-07-01', '2025-07-01', TRUE),

(2, '789012', '2024-07-02', '2025-07-02', TRUE);

-- Зміна типу поля Description на VARCHAR(40)

ALTER TABLE Subject MODIFY COLUMN Description VARCHAR(40);

-- Оновлення таблиці Subject з укороченим описом

UPDATE Subject SET Description = 'Вивчення чисел, кількостей і форм.' WHERE Name = 'Математика';

UPDATE Subject SET Description = 'Вивчення речовин і їх реакцій.' WHERE Name = 'Хімія';

UPDATE Subject SET Description = 'Вивчення англійської мови.' WHERE Name = 'Англійська';

UPDATE Subject SET Description = 'Вивчення української мови.' WHERE Name = 'Українська';

UPDATE Subject SET Description = 'Вивчення живих організмів і процесів.' WHERE Name = 'Біологія';

UPDATE Subject SET Description = 'Вивчення законів природи.' WHERE Name = 'Фізика';

UPDATE Subject SET Description = 'Вивчення земної поверхні і клімату.' WHERE Name = 'Географія';

UPDATE Subject SET Description = 'Вивчення минулих подій і епох.' WHERE Name = 'Історія';

Запити:

SELECT s.PIB,

r.Attemp\_date,

CASE WHEN r.Status = 1 THEN 'Здано' ELSE 'Не здано' END,

subj.Name,

r.Reached\_score

FROM Result r

JOIN Student s ON r.Student\_id = s.Student\_id

JOIN Subject subj ON r.Subj\_id = subj.Subject\_id

UPDATE subject

SET Name = 'Математика', Description = 'Вивчення чисел, кількостей і форм.'

WHERE Subject\_id = 1;

UPDATE subject

SET Name = 'Хімія', Description = 'Вивчення речовин і їх реакцій.'

WHERE Subject\_id = 2;

UPDATE subject

SET Name = 'Англійська', Description = 'Вивчення англійської мови.'

WHERE Subject\_id = 3;

UPDATE subject

SET Name = 'Українська', Description = 'Вивчення української мови.'

WHERE Subject\_id = 4;

UPDATE subject

SET Name = 'Біологія', Description = 'Вивчення живих організмів і процесів.'

WHERE Subject\_id = 5;

UPDATE subject

SET Name = 'Фізика', Description = 'Вивчення законів природи.'

WHERE Subject\_id = 6;

UPDATE subject

SET Name = 'Географія', Description = 'Вивчення земної поверхні і клімату.'

WHERE Subject\_id = 7;

UPDATE subject

SET Name = 'Історія', Description = 'Вивчення минулих подій і епох.'

WHERE Subject\_id = 8;

UPDATE student

SET PIB = 'Іван Петров'

WHERE Student\_id = 1;

UPDATE student

SET PIB = 'Марія Іванова'

WHERE Student\_id = 2;

UPDATE student

SET PIB = 'Petrenko Petro'

WHERE Student\_id = 3;

UPDATE student

SET PIB = 'Петренко Петро'

WHERE Student\_id = 3;

ALTER TABLE school DROP CHECK school\_chk\_1;

INSERT INTO school (City, Region, Type, Email, Teacher\_PIB)

VALUES

('Київ', 'Київська область', 'Ліцей', 'school1@example.com', 'Ольга Коваль'),

('Львів', 'Львівська область', 'СЗШ', 'school2@example.com', 'Андрій Шевченко'),

('Одеса', 'Одеська область', 'Гімназія', 'school3@example.com', 'Наталія Бондар'),

('Харків', 'Харківська область', 'ВНЗ', 'school4@example.com', 'Дмитро Савченко');

Додаток В

Текст програми

Фрагменти програмного коду наведено в лістингах В.1-В.31.

Лістинг В.1 – Текст файлу Unit13.h

//---------------------------------------------------------------------------

#ifndef Unit13H

#define Unit13H

//---------------------------------------------------------------------------

#include <System.Classes.hpp>

#include <Vcl.Controls.hpp>

#include <Vcl.StdCtrls.hpp>

#include <Vcl.Forms.hpp>

#include <Vcl.ExtCtrls.hpp>

#include <Vcl.Mask.hpp>

//---------------------------------------------------------------------------

class TForm13 : public TForm

{

\_\_published: // IDE-managed Components

TLabeledEdit \*LabeledEdit1;

TButton \*Button1;

void \_\_fastcall Button1Click(TObject \*Sender);

void \_\_fastcall LabeledEdit1Exit(TObject \*Sender);

private:

bool \_\_fastcall LoadTemplateFromIni();

void \_\_fastcall SaveTemplateToIni();

bool fromCertificate; // User declarations

public:

String template\_name;

void setfromCertificate(bool k); // User declarations

\_\_fastcall TForm13(TComponent\* Owner);

};

//---------------------------------------------------------------------------

extern PACKAGE TForm13 \*Form13;

//---------------------------------------------------------------------------

#endif

Лістинг В.2 – Текст файлу Unit13.cpp

#include <System.RegularExpressions.hpp>

#include <Vcl.Dialogs.hpp>

#include <IniFiles.hpp>

#include <Data.Win.ADODB.hpp>

#include <System.RegularExpressions.hpp>

#include "Unit13.h"

#include "Main\_Window.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm13 \*Form13;

//---------------------------------------------------------------------------

\_\_fastcall TForm13::TForm13(TComponent\* Owner)

: TForm(Owner)

{

bool k = LoadTemplateFromIni();

}

bool \_\_fastcall TForm13::LoadTemplateFromIni()

{

TIniFile \*IniFile = new TIniFile(ExtractFilePath(Application->ExeName) + "config.ini");

bool success = false;

try {

if (IniFile->SectionExists("Certificate")) {

template\_name = IniFile->ReadString("Certificate", "TemplateName", "");

if (!template\_name.IsEmpty()) {

success = true;

}

}

}

\_\_finally {

delete IniFile;

}

return success;

}

void \_\_fastcall TForm13::Button1Click(TObject \*Sender)

{

if (!LabeledEdit1->Text.IsEmpty()) {

template\_name = LabeledEdit1->Text;

SaveTemplateToIni();

if (fromCertificate) {

this->Close();

} else {

Form3->Show();

fromCertificate = true;

this->Hide();

}

} else {

ShowMessage("Введіть шаблон для назви сертифікату");

}

}

void \_\_fastcall TForm13::SaveTemplateToIni()

{

TIniFile \*IniFile = new TIniFile(ExtractFilePath(Application->ExeName) + "config.ini");

try {

IniFile->WriteString("Certificate", "TemplateName", template\_name);

}

\_\_finally {

delete IniFile;

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm13::LabeledEdit1Exit(TObject \*Sender)

{

String text = LabeledEdit1->Text;

if (!TRegEx::IsMatch(text, L"^[A-Za-z0-9]+$")) {

ShowMessage(L"Дозволено тільки англійську мову та числа.");

LabeledEdit1->SetFocus();

}

}

void TForm13::setfromCertificate(bool k){

fromCertificate = k;

}

Лістинг В.3 – Текст файлу Main\_Window.h

//---------------------------------------------------------------------------

#ifndef Main\_WindowH

#define Main\_WindowH

//---------------------------------------------------------------------------

#include <System.Classes.hpp>

#include <Vcl.Controls.hpp>

#include <Vcl.StdCtrls.hpp>

#include <Vcl.Forms.hpp>

#include <Data.DB.hpp>

#include <Vcl.DBGrids.hpp>

#include <Vcl.ExtCtrls.hpp>

#include <Vcl.Grids.hpp>

#include <Vcl.Menus.hpp>

#include <Vcl.ComCtrls.hpp>

#include <Vcl.WinXPickers.hpp>

#include <Data.Win.ADODB.hpp>

#include <Vcl.Buttons.hpp>

#include <System.RegularExpressions.hpp>

#include "Data.h"

#include "Help.h"

#include "Conditions.h"

#include "Students.h"

#include "Subjects.h"

#include "Results.h"

#include "Certificate.h"

#include "School.h"

#include "Auth.h"

//---------------------------------------------------------------------------

class TForm3 : public TForm

{

\_\_published: // IDE-managed Components

TMainMenu \*MainMenu1;

TMenuItem \*N1;

TMenuItem \*N5;

TStatusBar \*StatusBar1;

TTimer \*Timer1;

TMenuItem \*Lj1;

TGroupBox \*GroupBox1;

TDatePicker \*DatePicker1;

TCheckBox \*ThisDate;

TGroupBox \*Status;

TGroupBox \*Surname\_student;

TRadioGroup \*Status\_check;

TEdit \*Edit1;

TMenuItem \*N3;

TBitBtn \*Clear;

TBitBtn \*Execute;

TMenuItem \*N2;

TMenuItem \*N4;

TMenuItem \*N6;

TMenuItem \*N7;

TMenuItem \*N8;

TMenuItem \*N9;

TDBGrid \*DBGrid1;

TMenuItem \*N10;

TMenuItem \*N11;

TMenuItem \*N12;

TMenuItem \*N13;

TDatePicker \*DatePicker2;

TMenuItem \*N14;

void \_\_fastcall Timer1Timer(TObject \*Sender);

void \_\_fastcall FormCreate(TObject \*Sender);

void \_\_fastcall FormShow(TObject \*Sender);

void \_\_fastcall CheckFiltersFilled(TObject \*Sender);

void \_\_fastcall ClearClick(TObject \*Sender);

void \_\_fastcall ExecuteClick(TObject \*Sender);

void \_\_fastcall Lj1Click(TObject \*Sender);

void \_\_fastcall N11Click(TObject \*Sender);

void \_\_fastcall N2Click(TObject \*Sender);

void \_\_fastcall N4Click(TObject \*Sender);

void \_\_fastcall N3Click(TObject \*Sender);

void \_\_fastcall DatePicker1CloseUp(TObject \*Sender);

void \_\_fastcall Edit1Exit(TObject \*Sender);

void \_\_fastcall N9Click(TObject \*Sender);

void \_\_fastcall N5Click(TObject \*Sender);

void \_\_fastcall DBGrid1TitleClick(TColumn \*Column);

void \_\_fastcall N1Click(TObject \*Sender);

void \_\_fastcall N12Click(TObject \*Sender);

void \_\_fastcall N13Click(TObject \*Sender);

void \_\_fastcall FormClose(TObject \*Sender, TCloseAction &Action);

void \_\_fastcall DatePicker2CloseUp(TObject \*Sender);

void \_\_fastcall N14Click(TObject \*Sender);

private:

public: // User declarations

\_\_fastcall TForm3(TComponent\* Owner);

void \_\_fastcall UpdateStatusBar(String role);

void ManageMenuItems(String role);

void DBColumnMainSizes();

};

//---------------------------------------------------------------------------

extern PACKAGE TForm3 \*Form3;

//---------------------------------------------------------------------------

#endif

Лістинг В.4 – Текст файлу Main\_Window.cpp

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include <SysUtils.hpp>

#include "Main\_Window.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm3 \*Form3;

//---------------------------------------------------------------------------

\_\_fastcall TForm3::TForm3(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::Timer1Timer(TObject \*Sender)

{

StatusBar1->Panels->Items[1]->Text = DateTimeToStr(Now());

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::FormCreate(TObject \*Sender)

{

StatusBar1->Panels->Items[1]->Text = DateTimeToStr(Now());

Clear->Enabled = false;

Execute->Enabled = false;

DatePicker1->OnChange = CheckFiltersFilled;

ThisDate->OnClick = CheckFiltersFilled;

Status\_check->OnClick = CheckFiltersFilled;

Edit1->OnChange = CheckFiltersFilled;

}

//---------------------------------------------------------------------------

void TForm3::DBColumnMainSizes(){

DBGrid1->Columns->Items[0]->Width = 200;

DBGrid1->Columns->Items[0]->Title->Caption = "ПІБ";

DBGrid1->Columns->Items[1]->Width = 100;

DBGrid1->Columns->Items[1]->Title->Caption = "Дата складання";

DBGrid1->Columns->Items[2]->Width = 60;

DBGrid1->Columns->Items[2]->Title->Caption = "Статус";

DBGrid1->Columns->Items[3]->Width = 125;

DBGrid1->Columns->Items[3]->Title->Caption = "Предмет";

DBGrid1->Columns->Items[4]->Width = 85;

DBGrid1->Columns->Items[4]->Title->Caption = "Результат";

}

void \_\_fastcall TForm3::CheckFiltersFilled(TObject \*Sender)

{

bool isAnyFilterFilled = false;

if (ThisDate->Checked) isAnyFilterFilled = true;

if (Status\_check->ItemIndex >= 0) isAnyFilterFilled = true;

if (!Edit1->Text.Trim().IsEmpty()) isAnyFilterFilled = true;

Execute->Enabled = isAnyFilterFilled;

Clear->Enabled = isAnyFilterFilled;

}

void \_\_fastcall TForm3::FormShow(TObject \*Sender)

{

DBColumnMainSizes();

UpdateStatusBar(Form12->get\_role(), Form12->get\_pib());

ManageMenuItems(Form12->get\_role());

Application->ProcessMessages();

}

void \_\_fastcall TForm3::ClearClick(TObject \*Sender)

{

DatePicker1->Date = Now();

ThisDate->Checked = false;

Status\_check->ItemIndex = -1;

Edit1->Text = "";

Execute->Enabled = false;

Clear->Enabled = false;

String query = "SELECT s.PIB, "

"r.Attemp\_date, "

"CASE WHEN r.Status = 1 THEN 'Здано' ELSE 'Не здано' END, "

"subj.Name, "

"r.Reached\_score "

"FROM Result r "

"JOIN Student s ON r.Student\_id = s.Student\_id "

"JOIN Subject subj ON r.Subj\_id = subj.Subject\_id "

"ORDER BY r.Attemp\_date DESC";

try

{

DataModule1->MainQuery->Close();

DataModule1->MainQuery->SQL->Clear();

DataModule1->MainQuery->SQL->Add(query);

DataModule1->MainQuery->Open();

DBColumnMainSizes();

}

catch (Exception &e)

{

ShowMessage("Помилка при скиданні фільтрів: " + e.Message);

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::ExecuteClick(TObject \*Sender)

{

String query = "SELECT s.PIB, r.Attemp\_date, CASE WHEN r.Status = 1 THEN 'Здано' ELSE 'Не здано' END AS Status, subj.Name, r.Reached\_score "

"FROM Result r "

"JOIN Student s ON r.Student\_id = s.Student\_id "

"JOIN Subject subj ON r.Subj\_id = subj.Subject\_id ";

String conditions = "";

if (ThisDate->Checked && DatePicker1->Date != TDateTime() && DatePicker2->Date != TDateTime())

{

if ((double)DatePicker1->Date <= (double)DatePicker2->Date)

{

conditions += "r.Attemp\_date BETWEEN :DateStart AND :DateEnd ";

}

else

{

ShowMessage("Дата кінця інтервалу має бути більша за початкову!");

return;

}

}

if (Status\_check->ItemIndex == 0)

{

if (!conditions.IsEmpty()) conditions += " AND ";

conditions += "r.Status = 1 ";

}

else if (Status\_check->ItemIndex == 1)

{

if (!conditions.IsEmpty()) conditions += " AND ";

conditions += "r.Status = 0 ";

}

if (!Edit1->Text.Trim().IsEmpty())

{

if (!conditions.IsEmpty()) conditions += " AND ";

conditions += "s.PIB LIKE :Surname ";

}

if (!conditions.IsEmpty())

{

query += " WHERE " + conditions;

}

query += " ORDER BY r.Attemp\_date DESC";

try

{

DataModule1->MainQuery->Close();

DataModule1->MainQuery->SQL->Clear();

DataModule1->MainQuery->SQL->Add(query);

if (ThisDate->Checked && DatePicker1->Date != TDateTime() && DatePicker2->Date != TDateTime())

{

DataModule1->MainQuery->Parameters->ParamByName("DateStart")->Value = DatePicker1->Date.FormatString("yyyy-mm-dd");

DataModule1->MainQuery->Parameters->ParamByName("DateEnd")->Value = DatePicker2->Date.FormatString("yyyy-mm-dd");

}

if (!Edit1->Text.Trim().IsEmpty())

{

DataModule1->MainQuery->Parameters->ParamByName("Surname")->Value = "%" + Edit1->Text.Trim() + "%";

}

DataModule1->MainQuery->Open();

DBColumnMainSizes();

}

catch (const Exception &e)

{

ShowMessage("Помилка при фільтрації: " + e.Message);

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::Lj1Click(TObject \*Sender)

{

Help\_m->ShowModal();

}

void \_\_fastcall TForm3::N11Click(TObject \*Sender)

{

Form3->Hide();

Form12->Show();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::N2Click(TObject \*Sender)

{

Form14->ShowModal();

DataModule1->MainQuery->Close();

DataModule1->MainQuery->Open();

DBColumnMainSizes();

}

void \_\_fastcall TForm3::N4Click(TObject \*Sender)

{

Form7->ShowModal();

DataModule1->MainQuery->Open();

DBColumnMainSizes();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::N3Click(TObject \*Sender)

{

Form4->ShowModal();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::DatePicker1CloseUp(TObject \*Sender)

{

try

{

TDateTime selectedDate = DatePicker1->Date;

TDateTime currentDate = Now();

if (selectedDate > currentDate)

{

throw Exception("Дата не може бути в майбутньому!");

}

else if(selectedDate < EncodeDate(1925, 1, 1))

{

throw Exception("Дата занадто стара! Виберіть пізнішу дату.");

}

}

catch (const Exception &e)

{

ShowMessage(e.Message);

DatePicker1->Date = Now();

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::Edit1Exit(TObject \*Sender)

{

String pib = Edit1->Text.Trim();

if (TRegEx::IsMatch(pib, L"[A-Za-z0-9ЁЫЭёыэ!\"#$%&()\*,./:;<=>?@[\\]^\_{|}~]")) {

ShowMessage("Будь ласка, вводьте ПІБ українською мовою!");

return;

Edit1->SetFocus();

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::N9Click(TObject \*Sender)

{

Form6->ShowModal();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::N5Click(TObject \*Sender)

{

Form11->ShowModal();

DataModule1->MainQuery->Open();

DBColumnMainSizes();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::DBGrid1TitleClick(TColumn \*Column)

{

String columnName = Column->FieldName;

static bool sortAsc = true;

String sortOrder = sortAsc ? "ASC" : "DESC";

sortAsc = !sortAsc;

String query =

"SELECT s.PIB, "

"r.Attemp\_date, "

"CASE WHEN r.Status = 1 THEN 'Здано' ELSE 'Нездано' END AS Status, "

"subj.Name, "

"r.Reached\_score "

"FROM Result r "

"JOIN Student s ON r.Student\_id = s.Student\_id "

"JOIN Subject subj ON r.Subj\_id = subj.Subject\_id "

"ORDER BY " + columnName + " " + sortOrder;

try

{

DataModule1->MainQuery->Close();

DataModule1->MainQuery->SQL->Clear();

DataModule1->MainQuery->SQL->Add(query);

DataModule1->MainQuery->Open();

DBColumnMainSizes();

}

catch (Exception &e)

{

ShowMessage("Помилка сортування: " + e.Message);

}

}

void \_\_fastcall TForm3::N1Click(TObject \*Sender)

{

Certificates->ShowModal();

}

//---------------------------------------------------------------------------

void TForm3::UpdateStatusBar(String role, String pib)

{

StatusBar1->Panels->Items[0]->Text = "ПІБ: " + pib;

StatusBar1->Panels->Items[1]->Text = DateTimeToStr(Now());

StatusBar1->Panels->Items[2]->Text = "Роль: " + role;

}

void TForm3::ManageMenuItems(String role)

{

Form7->N1->Enabled = false;

Form7->N2->Enabled = false;

Form7->N3->Enabled = false;

Form14->N1->Enabled = false;

Form14->N2->Enabled = false;

Form4->N3->Enabled = false;

Form4->N2->Enabled = false;

Form6->N1->Enabled = false;

Form6->N2->Enabled = false;

Form6->N3->Enabled = false;

Certificates->N3->Enabled = false;

Certificates->N2->Enabled = false;

Certificates->N4->Enabled = false;

Certificates->BitBtn1->Enabled = false;

Form11->N3->Enabled = false;

Form11->N2->Enabled = false;

Form3->N15->Enabled = false;

if (role == "головний адміністратор" || role == "адміністратор")

{

Form7->N1->Enabled = true;

Form7->N2->Enabled = true;

Form7->N3->Enabled = true;

Form14->N1->Enabled = true;

Form14->N2->Enabled = true;

Form4->N3->Enabled = true;

Form4->N2->Enabled = true;

Form6->N1->Enabled = true;

Form6->N2->Enabled = true;

Form6->N3->Enabled = true;

Certificates->N3->Enabled = true;

Certificates->N2->Enabled = true;

Certificates->N4->Enabled = true;

Certificates->BitBtn1->Enabled = true;

Form11->N3->Enabled = true;

Form11->N2->Enabled = true;

Form3->N15->Enabled = true;

}

else if (role == "відповідальний за сертифікати")

{

Certificates->N3->Enabled = true;

Certificates->N2->Enabled = true;

Certificates->BitBtn1->Enabled = true;

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::FormClose(TObject \*Sender, TCloseAction &Action)

{

Application->Terminate();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::DatePicker2CloseUp(TObject \*Sender)

{

try

{

TDateTime selectedDate = DatePicker2->Date;

TDateTime currentDate = Now();

if (selectedDate > currentDate)

{

throw Exception("Дата не може бути в майбутньому!");

}

else if(selectedDate < EncodeDate(1925, 1, 1))

{

throw Exception("Дата занадто стара! Виберіть пізнішу дату.");

}

}

catch (const Exception &e)

{

ShowMessage(e.Message);

DatePicker2->Date = Now();

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::N14Click(TObject \*Sender)

{

Application->Terminate();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::N15Click(TObject \*Sender)

{

Form15->ShowModal();

}

//-------------------------------------------------------------

Лістинг В.5 – Текст файлу Auth.h

//---------------------------------------------------------------------------

#ifndef AuthH

#define AuthH

//---------------------------------------------------------------------------

#include <System.Classes.hpp>

#include <Vcl.Controls.hpp>

#include <Vcl.StdCtrls.hpp>

#include <Vcl.Forms.hpp>

#include <Vcl.ExtCtrls.hpp>

#include <Vcl.Mask.hpp>

#include "Data.h"

#include "Main\_Window.h"

//---------------------------------------------------------------------------

class TForm12 : public TForm

{

\_\_published: // IDE-managed Components

TLabeledEdit \*LabeledEdit1;

TLabeledEdit \*LabeledEdit2;

TButton \*Button1;

void \_\_fastcall Button1Click(TObject \*Sender);

void \_\_fastcall FormShow(TObject \*Sender);

void \_\_fastcall LabeledEdit1Exit(TObject \*Sender);

void \_\_fastcall LabeledEdit2Exit(TObject \*Sender);

private:

String role\_s;

String pib\_s;

String login\_s; // User declarations

public: // User declarations

\_\_fastcall TForm12(TComponent\* Owner);

String get\_role();

String get\_pib();

String get\_login();

};

//---------------------------------------------------------------------------

extern PACKAGE TForm12 \*Form12;

//---------------------------------------------------------------------------

#endif

Лістинг В.6 – Текст файлу Auth.cpp

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include <System.RegularExpressions.hpp>

#include "Auth.h"

#include <IniFiles.hpp>

#include <System.IOUtils.hpp>

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm12 \*Form12;

//---------------------------------------------------------------------------

\_\_fastcall TForm12::TForm12(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm12::Button1Click(TObject \*Sender)

{

bool loginSuccess = false;

try

{

String login = LabeledEdit1->Text.Trim();

String password = LabeledEdit2->Text.Trim();

String query = "SELECT Role, PIB, Status FROM users WHERE Login = :Login AND Password = :Password";

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Clear();

DataModule1->ADOQuery1->SQL->Add(query);

DataModule1->ADOQuery1->Parameters->ParamByName("Login")->Value = login;

DataModule1->ADOQuery1->Parameters->ParamByName("Password")->Value = password;

DataModule1->ADOQuery1->Open();

if (DataModule1->ADOQuery1->RecordCount == 0)

{

ShowMessage("Невірний логін або пароль. Спробуйте ще раз.");

return;

}

int status = DataModule1->ADOQuery1->FieldByName("Status")->AsInteger;

if (status == 0)

{

ShowMessage("Обліковий запис не активний.");

return;

}

String role = DataModule1->ADOQuery1->FieldByName("Role")->AsString;

String pib = DataModule1->ADOQuery1->FieldByName("PIB")->AsString;

role\_s = role;

pib\_s = pib;

login\_s = login;

loginSuccess = true;

ShowMessage("Успішний вхід. Роль користувача: " + role);

}

catch (const Exception &e)

{

loginSuccess = false;

return;

}

if (loginSuccess)

{

try

{

String iniFilePath = TPath::Combine(ExtractFilePath(Application->ExeName), "config.ini");

TIniFile \*ini = new TIniFile(iniFilePath);

bool iniExists = ini->SectionExists("Certificate");

delete ini;

if (!iniExists)

{

if (role\_s != "головний адміністратор")

{

loginSuccess = false;

ShowMessage("Для першого запуску необхідно увійти як головний адміністратор.");

return;

}

else

{

if (Form13 == NULL)

{

Application->CreateForm(\_\_classid(TForm14), &Form14);

}

Form13->Show();

}

}

else

{

if (Form3 == NULL)

{

Application->CreateForm(\_\_classid(TForm3), &Form3);

}

Form3->Show();

}

this->Hide();

}

catch (const Exception &e)

{

loginSuccess = false;

}

}

}

String TForm12::get\_role(){

return role\_s;

}

String TForm12::get\_pib(){

return pib\_s;

}

String TForm12::get\_login(){

return login\_s;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm12::FormShow(TObject \*Sender)

{

LabeledEdit1->Text = "";

LabeledEdit2->Text = "";

}

//--------------------------------------------------------------------------

void \_\_fastcall TForm12::LabeledEdit1Exit(TObject \*Sender)

{

String login = LabeledEdit1->Text.Trim();

UnicodeString pattern = L"^[A-Za-z0-9]+$";

if (!TRegEx::IsMatch(login, pattern))

{

ShowMessage("Логін повинен містити тільки англійські букви та цифри.");

LabeledEdit1->SetFocus();

return;

}

}

void \_\_fastcall TForm12::LabeledEdit2Exit(TObject \*Sender)

{

String password = LabeledEdit2->Text.Trim();

UnicodeString pattern = L"^[A-Za-z0-9]+$";

if (!TRegEx::IsMatch(password, pattern))

{

ShowMessage("Пароль повинен містити тільки англійські букви та цифри.");

LabeledEdit2->SetFocus();

}

}

//-----------------------------------------------------------------------

Лістинг В.7 – Текст файлу Student.h

//---------------------------------------------------------------------------

#ifndef StudentsH

#define StudentsH

//---------------------------------------------------------------------------

#include <System.Classes.hpp>

#include <Vcl.Controls.hpp>

#include <Vcl.StdCtrls.hpp>

#include <Vcl.Forms.hpp>

#include <Vcl.ComCtrls.hpp>

#include <Vcl.ExtCtrls.hpp>

#include <Vcl.WinXPickers.hpp>

#include <Vcl.Menus.hpp>

#include <Data.DB.hpp>

#include <Vcl.DBGrids.hpp>

#include <Vcl.Grids.hpp>

#include <Vcl.Buttons.hpp>

#include <Vcl.DBCtrls.hpp>

#include "Main\_Window.h"

#include "Help.h"

#include <String.h>

#include <System.RegularExpressions.hpp>

#include "Data.h"

#include "Unit1.h"

//---------------------------------------------------------------------------

class TForm14 : public TForm

{

\_\_published: // IDE-managed Components

TMainMenu \*MainMenu1;

TMenuItem \*Lj1;

TDBGrid \*DBGrid1;

TMenuItem \*N6;

TPopupMenu \*PopupMenu1;

TMenuItem \*N1;

TMenuItem \*N2;

TGroupBox \*GroupBox1;

TLabel \*Label1;

TDatePicker \*DatePicker1;

TRadioGroup \*RadioGroup1;

TButton \*Button1;

TLabel \*Label3;

TLabel \*Label7;

TEdit \*Edit5;

TBitBtn \*Clear;

TBitBtn \*Execute;

TCheckBox \*ThisDate;

TComboBox \*ComboBox1;

TDatePicker \*DatePicker2;

void \_\_fastcall FormCreate(TObject \*Sender);

void \_\_fastcall N6Click(TObject \*Sender);

void \_\_fastcall Lj1Click(TObject \*Sender);

void \_\_fastcall N1Click(TObject \*Sender);

void \_\_fastcall DBGrid1TitleClick(TColumn \*Column);

void \_\_fastcall N2Click(TObject \*Sender);

void \_\_fastcall ExecuteClick(TObject \*Sender);

void \_\_fastcall ClearClick(TObject \*Sender);

void \_\_fastcall DatePicker1CloseUp(TObject \*Sender);

void \_\_fastcall Edit5Exit(TObject \*Sender);

void \_\_fastcall DatePicker2CloseUp(TObject \*Sender);

private:

bool SortAscending; // User declarations

void \_\_fastcall CheckFiltersFilled(TObject \*Sender);

public: // User declarations

\_\_fastcall TForm14(TComponent\* Owner);

void \_\_fastcall DBColumnSizes();

void ToggleView();

bool isMinimalView;

};

//---------------------------------------------------------------------------

extern PACKAGE TForm14 \*Form14;

//---------------------------------------------------------------------------

#endif

Лістинг В.8 – Текст файлу Student.cpp

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

#include "Students.h"

TForm14 \*Form14;

//---------------------------------------------------------------------------

\_\_fastcall TForm14::TForm14(TComponent\* Owner)

: TForm(Owner)

{

isMinimalView = false;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm14::FormCreate(TObject \*Sender)

{

if(!isMinimalView){

Clear->Enabled = false;

Execute->Enabled = false;

ComboBox1->Items->Clear();

ComboBox1->Items->Add("ID-карта");

ComboBox1->Items->Add("Паперовий");

DatePicker1->OnChange = CheckFiltersFilled;

ThisDate->OnClick = CheckFiltersFilled;

Edit5->OnChange = CheckFiltersFilled;

ComboBox1->OnChange = CheckFiltersFilled;

RadioGroup1->OnClick = CheckFiltersFilled;

}

ToggleView();

DBColumnSizes();

}

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

void \_\_fastcall TForm14::N6Click(TObject \*Sender)

{

this->Close();

Form3->Show();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm14::Lj1Click(TObject \*Sender)

{

Help\_m->ShowModal();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm14::DBColumnSizes(){

DBGrid1->Columns->Items[0]->Visible = false;

DBGrid1->Columns->Items[1]->Width = 100;

DBGrid1->Columns->Items[1]->Title->Caption = "Номер паспорту";

DBGrid1->Columns->Items[2]->Width = 80;

DBGrid1->Columns->Items[2]->Title->Caption = "Тип паспорту";

DBGrid1->Columns->Items[3]->Width = 100;

DBGrid1->Columns->Items[3]->Title->Caption = "ПІБ";

DBGrid1->Columns->Items[4]->Width = 100;

DBGrid1->Columns->Items[4]->Title->Caption = "Дата народження";

DBGrid1->Columns->Items[5]->Width = 60;

DBGrid1->Columns->Items[5]->Title->Caption = "Стать";

DBGrid1->Columns->Items[6]->Width = 85;

DBGrid1->Columns->Items[6]->Title->Caption = "E-mail";

DBGrid1->Columns->Items[7]->Width = 85;

DBGrid1->Columns->Items[7]->Title->Caption = "Номер телефону";

DBGrid1->Columns->Items[8]->Width = 85;

DBGrid1->Columns->Items[8]->Title->Caption = "Номер посвідчення";

DBGrid1->Columns->Items[9]->Width = 85;

DBGrid1->Columns->Items[9]->Title->Caption = "ІПН";

DBGrid1->Columns->Items[10]->Width = 85;

DBGrid1->Columns->Items[10]->Title->Caption = "Додаткова помітка";

}

void \_\_fastcall TForm14::N1Click(TObject \*Sender)

{

DataModule1->DataSource1->DataSet->Refresh();

String passport\_num = DBGrid1->DataSource->DataSet->FieldByName("Passport\_num")->AsString;

TADOQuery \*query = new TADOQuery(this);

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = "SELECT Student\_id FROM student WHERE Passport\_num = :Passport\_num";

query->Parameters->ParamByName("Passport\_num")->Value = passport\_num;

query->Open();

if (!query->Eof) {

int student\_id = query->FieldByName("Student\_id")->AsInteger;

Form1->Caption = "Редагування даних";

Form1->set\_id(student\_id);

Form1->ShowModal();

DataModule1->DataSource1->DataSet->Refresh();

}

else {

ShowMessage("Студента з таким паспортом не знайдено.");

}

delete query;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm14::DBGrid1TitleClick(TColumn \*Column)

{

String sortOrder = SortAscending ? " ASC" : " DESC";

DataModule1->ADOTable1->Sort = Column->Field->FieldName + sortOrder;

SortAscending = !SortAscending;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm14::N2Click(TObject \*Sender)

{

Form1->Caption = "Додавання даних";

Form1->set\_id(0);

Form1->ShowModal();

DataModule1->DataSource1->DataSet->Refresh();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm14::CheckFiltersFilled(TObject \*Sender)

{

bool isAnyFilterFilled = false;

if (ThisDate->Checked) isAnyFilterFilled = true;

if (RadioGroup1->ItemIndex >= 0) isAnyFilterFilled = true;

if (!Edit5->Text.Trim().IsEmpty()) isAnyFilterFilled = true;

if (ComboBox1->ItemIndex >= 0) isAnyFilterFilled = true;

Execute->Enabled = isAnyFilterFilled;

Clear->Enabled = isAnyFilterFilled;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm14::ExecuteClick(TObject \*Sender)

{

DataModule1->ADOTable1->Filtered = false;

if (ThisDate->Checked && DatePicker1->Date != TDateTime() && DatePicker2->Date != TDateTime())

{

if ((double)DatePicker1->Date <= (double)DatePicker2->Date)

{

String dateStart = "#" + DatePicker1->Date.FormatString("yyyy-mm-dd") + "#";

String dateEnd = "#" + DatePicker2->Date.FormatString("yyyy-mm-dd") + "#";

DataModule1->ADOTable1->Filter = "Birth\_date >= " + dateStart + " AND Birth\_date <= " + dateEnd;

}

else

{

ShowMessage("Дата початку повинна бути раніше дати завершення!");

return;

}

}

if (ComboBox1->ItemIndex >= 0)

{

if (!DataModule1->ADOTable1->Filter.IsEmpty())

DataModule1->ADOTable1->Filter += " AND ";

DataModule1->ADOTable1->Filter += "Passport\_type = '" + ComboBox1->Items->Strings[ComboBox1->ItemIndex] + "'";

}

if (!Edit5->Text.Trim().IsEmpty())

{

if (!DataModule1->ADOTable1->Filter.IsEmpty())

DataModule1->ADOTable1->Filter += " AND ";

DataModule1->ADOTable1->Filter += "Email LIKE '\*" + Edit5->Text.Trim() + "\*'";

}

if (RadioGroup1->ItemIndex >= 0)

{

String genderFilter = (RadioGroup1->ItemIndex == 0) ? "M" : "F";

if (!DataModule1->ADOTable1->Filter.IsEmpty())

DataModule1->ADOTable1->Filter += " AND ";

DataModule1->ADOTable1->Filter += "Gender = '" + genderFilter + "'";

}

DataModule1->ADOTable1->Filtered = true;

DBColumnSizes();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm14::ClearClick(TObject \*Sender)

{

ThisDate->Checked = false;

RadioGroup1->ItemIndex = -1;

Edit5->Clear();

ComboBox1->ItemIndex = -1;

DatePicker1->Date = TDateTime();

DataModule1->ADOTable1->Filtered = false;

DataModule1->ADOTable1->Filter = "";

Execute->Enabled = false;

Clear->Enabled = false;

}

void \_\_fastcall TForm14::DatePicker1CloseUp(TObject \*Sender)

{

try

{

TDateTime selectedDate = DatePicker1->Date;

TDateTime currentDate = Now();

if (selectedDate > currentDate)

{

throw Exception("Дата не може бути в майбутньому!");

}

else if(selectedDate < EncodeDate(1925, 1, 1))

{

throw Exception("Дата занадто стара! Виберіть пізнішу дату.");

}

}

catch (const Exception &e)

{

ShowMessage(e.Message);

DatePicker1->Date = Now();

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm14::Edit5Exit(TObject \*Sender)

{

String email = Edit5->Text;

UnicodeString pattern = "^\\w+([-+.']\\w+)\*@\\w+([-.]\\w+)\*\\.\\w+([-.]\\w+)\*$";

if (!TRegEx::IsMatch(email, pattern)) {

ShowMessage("Введіть дійсний e-mail.");

Edit5->SetFocus();

}

}

//---------------------------------------------------------------------------

void TForm14::ToggleView()

{

if (isMinimalView)

{

Form14->AutoSize = true;

for (int i = 0; i < Form14->MainMenu1->Items->Count; i++)

{

Form14->MainMenu1->Items->Items[i]->Enabled = false;

}

Form14->Caption = "Учасники НМТ";

for (int i = 0; i < Form14->ControlCount; i++)

{

if (Form14->Controls[i] != Form14->DBGrid1)

{

Form14->Controls[i]->Visible = false;

}

}

}

else

{

Form14->AutoSize = false;

for (int i = 0; i < Form14->MainMenu1->Items->Count; i++)

{

Form14->MainMenu1->Items->Items[i]->Enabled = true;

}

Form14->Caption = "Робота з інформацією про учасників НМТ";

for (int i = 0; i < Form14->ControlCount; i++)

{

Form14->Controls[i]->Visible = true;

}

}

}

void \_\_fastcall TForm14::DatePicker2CloseUp(TObject \*Sender)

{

try

{

TDateTime selectedDate = DatePicker2->Date;

TDateTime currentDate = Now();

if (selectedDate > currentDate)

{

throw Exception("Дата не може бути в майбутньому!");

}

else if(selectedDate < EncodeDate(1925, 1, 1))

{

throw Exception("Дата занадто стара! Виберіть пізнішу дату.");

}

}

catch (const Exception &e)

{

ShowMessage(e.Message);

DatePicker2->Date = Now();

}

}

//-------------------------------------------------------------

Лістинг В.9 – Текст файлу Subject.h

//---------------------------------------------------------------------------

#ifndef SubjectsH

#define SubjectsH

//---------------------------------------------------------------------------

#include <System.Classes.hpp>

#include <Vcl.Controls.hpp>

#include <Vcl.StdCtrls.hpp>

#include <Vcl.Forms.hpp>

#include <Data.DB.hpp>

#include <Vcl.ComCtrls.hpp>

#include <Vcl.DBGrids.hpp>

#include <Vcl.ExtCtrls.hpp>

#include <Vcl.Grids.hpp>

#include <Vcl.Menus.hpp>

#include <Vcl.WinXPickers.hpp>

#include "Data.h"

//---------------------------------------------------------------------------

class TForm7 : public TForm

{

\_\_published: // IDE-managed Components

TDBGrid \*DBGrid1;

TMainMenu \*MainMenu1;

TMenuItem \*N6;

TMenuItem \*Lj1;

TImage \*Image1;

TPopupMenu \*PopupMenu1;

TMenuItem \*N1;

TMenuItem \*N2;

TMenuItem \*N3;

TMenuItem \*N4;

void \_\_fastcall FormCreate(TObject \*Sender);

void \_\_fastcall DBGrid1TitleClick(TColumn \*Column);

void \_\_fastcall DBGrid1CellClick(TColumn \*Column);

void \_\_fastcall N1Click(TObject \*Sender);

void \_\_fastcall N2Click(TObject \*Sender);

void \_\_fastcall N3Click(TObject \*Sender);

void \_\_fastcall Lj1Click(TObject \*Sender);

void \_\_fastcall N6Click(TObject \*Sender);

void \_\_fastcall N4Click(TObject \*Sender);

private:

bool SortAscending; // User declarations

public:

void \_\_fastcall DBColumnSizes(); // User declarations

\_\_fastcall TForm7(TComponent\* Owner);

};

//---------------------------------------------------------------------------

extern PACKAGE TForm7 \*Form7;

//---------------------------------------------------------------------------

#endif

Лістинг В.10 – Текст файлу Subject.cpp

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Subjects.h"

#include "Unit2.h"

#include "Help.h"

#include "Main\_window.h"

#include <ShellAPI.h>

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm7 \*Form7;

//---------------------------------------------------------------------------

\_\_fastcall TForm7::TForm7(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm7::DBColumnSizes(){

DBGrid1->Columns->Items[0]->Visible = false;

DBGrid1->Columns->Items[1]->Width = 100;

DBGrid1->Columns->Items[1]->Title->Caption = "Назва предмету";

DBGrid1->Columns->Items[2]->Width = 250;

DBGrid1->Columns->Items[2]->Title->Caption = "Опис предмету";

DBGrid1->Columns->Items[3]->Width = 240;

DBGrid1->Columns->Items[3]->Title->Caption = "Посилання на зразок";

DBGrid1->Columns->Items[4]->Width = 240;

DBGrid1->Columns->Items[4]->Title->Caption = "Тестовий екземпляр";

}

void \_\_fastcall TForm7::FormCreate(TObject \*Sender)

{

DBColumnSizes();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm7::DBGrid1TitleClick(TColumn \*Column)

{

String sortOrder = SortAscending ? " ASC" : " DESC";

DataModule1->ADOTable2->Sort = Column->Field->FieldName + sortOrder;

SortAscending = !SortAscending;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm7::DBGrid1CellClick(TColumn \*Column)

{

try

{

String subjectName = DataModule1->DataSource2->DataSet->FieldByName("Name")->AsString;

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Text = "SELECT Image\_name FROM subject WHERE Name = :name";

DataModule1->ADOQuery1->Parameters->ParamByName("name")->Value = subjectName; // Використовуємо назву предмета

DataModule1->ADOQuery1->Open();

if (!DataModule1->ADOQuery1->FieldByName("Image\_name")->IsNull)

{

String imagePath = DataModule1->ADOQuery1->FieldByName("Image\_name")->AsString;

if (FileExists(imagePath))

{

Image1->Picture->LoadFromFile(imagePath);

}

else

{

ShowMessage("Файл не знайдено: " + imagePath);

Image1->Picture->Assign(NULL);

}

}

else

{

Image1->Picture->Assign(NULL);

}

}

catch (const Exception &e)

{

ShowMessage("Помилка завантаження зображення: " + e.Message);

Image1->Picture->Assign(NULL);

}

}

void \_\_fastcall TForm7::N1Click(TObject \*Sender)

{

Form2->set\_id(0);

Form2->Caption = "Додавання даних";

Form2->ShowModal();

DataModule1->DataSource2->DataSet->Refresh();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm7::N2Click(TObject \*Sender)

{

String name = DBGrid1->DataSource->DataSet->FieldByName("Name")->AsString;

TADOQuery \*query = new TADOQuery(this);

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = "SELECT Subject\_id FROM subject WHERE Name = :Name";

query->Parameters->ParamByName("Name")->Value = name;

query->Open();

int id = query->FieldByName("Subject\_id")->AsInteger;

Form2->Caption = "Редагування даних";

Form2->set\_id(id);

Form2->ShowModal();

DataModule1->DataSource2->DataSet->Refresh();

delete query;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm7::N3Click(TObject \*Sender)

{

try

{

int id = DataModule1->DataSource2->DataSet->FieldByName("Subject\_id")->AsInteger;

if (id <= 0)

{

ShowMessage("Виберіть запис для видалення.");

return;

}

// Перевіряємо на наявність залежностей в інших таблицях

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Text = "SELECT COUNT(\*) AS Count FROM result WHERE Subj\_id = :id";

DataModule1->ADOQuery1->Parameters->ParamByName("id")->Value = id;

DataModule1->ADOQuery1->Open();

int resultDependentCount = DataModule1->ADOQuery1->FieldByName("Count")->AsInteger;

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Text = "SELECT COUNT(\*) AS Count FROM conditions WHERE Subject\_id = :id";

DataModule1->ADOQuery1->Parameters->ParamByName("id")->Value = id;

DataModule1->ADOQuery1->Open();

int conditionsDependentCount = DataModule1->ADOQuery1->FieldByName("Count")->AsInteger;

DataModule1->ADOQuery1->Close();

// Якщо є залежні записи, показуємо повідомлення

if (resultDependentCount > 0 || conditionsDependentCount > 0)

{

ShowMessage("Цей запис не може бути видалений, оскільки він використовується в інших таблицях.");

return;

}

// Підтвердження видалення

if (MessageDlg("Ви впевнені, що хочете видалити цей запис?", mtConfirmation, TMsgDlgButtons() << mbYes << mbNo, 0) == mrYes)

{

try

{

// Видалення запису

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Text = "DELETE FROM subject WHERE Subject\_id = :id";

DataModule1->ADOQuery1->Parameters->ParamByName("id")->Value = id;

DataModule1->ADOQuery1->ExecSQL();

// Оновлюємо дані після видалення

DataModule1->DataSource2->DataSet->Close();

DataModule1->DataSource2->DataSet->Open();

ShowMessage("Запис успішно видалено.");

}

catch (const Exception &e)

{

ShowMessage("Помилка при видаленні запису: " + e.Message);

}

}

}

catch (const Exception &e)

{

ShowMessage("Помилка видалення запису: " + e.Message);

}

}

void \_\_fastcall TForm7::Lj1Click(TObject \*Sender)

{

Help\_m->ShowModal();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm7::N6Click(TObject \*Sender)

{

this->Close();

Form3->Show();

}

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

void \_\_fastcall TForm7::N4Click(TObject \*Sender)

{

try

{

String name = DBGrid1->DataSource->DataSet->FieldByName("Name")->AsString;

TADOQuery \*query = new TADOQuery(this);

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = "SELECT Subject\_id, Test\_sample FROM subject WHERE Name = :Name";

query->Parameters->ParamByName("Name")->Value = name;

query->Open();

if (!query->Eof)

{

int id = query->FieldByName("Subject\_id")->AsInteger;

String testSamplePath = query->FieldByName("Test\_sample")->AsString;

if (!testSamplePath.IsEmpty() && FileExists(testSamplePath))

{

ShellExecute(0, L"open", testSamplePath.w\_str(), NULL, NULL, SW\_SHOWNORMAL);

}

else

{

ShowMessage("Файл не вказано або він відсутній. Шлях: " + testSamplePath);

}

}

else

{

ShowMessage("Предмет не знайдено в базі даних.");

}

delete query;

}

catch (const Exception &e)

{

ShowMessage("Помилка при відкритті файлу: " + e.Message);

}

}

//-------------------------------------------------------------

Лістинг В.11 – Текст файлу Conditions.h

//---------------------------------------------------------------------------

#ifndef ConditionsH

#define ConditionsH

//---------------------------------------------------------------------------

#include <System.Classes.hpp>

#include <Vcl.Controls.hpp>

#include <Vcl.StdCtrls.hpp>

#include <Vcl.Forms.hpp>

#include <Data.DB.hpp>

#include <Vcl.Buttons.hpp>

#include <Vcl.DBGrids.hpp>

#include <Vcl.Grids.hpp>

#include <Vcl.Menus.hpp>

#include <Vcl.WinXPickers.hpp>

#include "Help.h"

#include <Data.DB.hpp>

#include <Vcl.ComCtrls.hpp>

#include <Vcl.DBGrids.hpp>

#include <Vcl.ExtCtrls.hpp>

#include <Vcl.Grids.hpp>

#include "Data.h"

#include "Unit5.h"

#include <Vcl.Menus.hpp>

#include <Vcl.WinXPickers.hpp>

#include <Vcl.Buttons.hpp>

#include "Main\_Window.h"

//---------------------------------------------------------------------------

class TForm4 : public TForm

{

\_\_published: // IDE-managed Components

TDBGrid \*DBGrid1;

TMainMenu \*MainMenu1;

TMenuItem \*Lj1;

TMenuItem \*N1;

TGroupBox \*GroupBox1;

TComboBox \*ComboBox2;

TLabel \*Label3;

TDatePicker \*DatePicker2;

TBitBtn \*Execute;

TBitBtn \*Clear;

TCheckBox \*ThisDate;

TPopupMenu \*PopupMenu1;

TMenuItem \*N2;

TMenuItem \*N3;

TDatePicker \*DatePicker1;

void \_\_fastcall FormShow(TObject \*Sender);

void \_\_fastcall Lj1Click(TObject \*Sender);

void \_\_fastcall N1Click(TObject \*Sender);

void \_\_fastcall DBGrid1TitleClick(TColumn \*Column);

void \_\_fastcall DatePicker2CloseUp(TObject \*Sender);

void \_\_fastcall FormCreate(TObject \*Sender);

void \_\_fastcall ExecuteClick(TObject \*Sender);

void \_\_fastcall ClearClick(TObject \*Sender);

void \_\_fastcall N2Click(TObject \*Sender);

void \_\_fastcall N3Click(TObject \*Sender);

void \_\_fastcall DatePicker1CloseUp(TObject \*Sender);

private:

bool SortAscending;

void DBColumnSizes();

void \_\_fastcall CheckFiltersFilled(TObject \*Sender);

public: // User declarations

\_\_fastcall TForm4(TComponent\* Owner);

bool isMinimalView;

void TForm4::ToggleView();

};

//---------------------------------------------------------------------------

extern PACKAGE TForm4 \*Form4;

//---------------------------------------------------------------------------

#endif

Лістинг В.12 – Текст файлу Conditions.cpp

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Conditions.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm4 \*Form4;

//---------------------------------------------------------------------------

\_\_fastcall TForm4::TForm4(TComponent\* Owner)

: TForm(Owner)

{

isMinimalView = false;

}

//---------------------------------------------------------------------------

void TForm4::DBColumnSizes(){

DBGrid1->Columns->Items[0]->Visible = false;

DBGrid1->Columns->Items[1]->Width = 150;

DBGrid1->Columns->Items[1]->Title->Caption = "Назва предмету";

DBGrid1->Columns->Items[2]->Width = 125;

DBGrid1->Columns->Items[2]->Title->Caption = "Максимальний бал";

DBGrid1->Columns->Items[3]->Width = 170;

DBGrid1->Columns->Items[3]->Title->Caption = "Мінімальний прохідний бал";

DBGrid1->Columns->Items[4]->Width = 100;

DBGrid1->Columns->Items[4]->Title->Caption = "Мінімальний бал";

DBGrid1->Columns->Items[5]->Width = 100;

DBGrid1->Columns->Items[5]->Title->Caption = "Статус";

DBGrid1->Columns->Items[6]->Width = 100;

DBGrid1->Columns->Items[6]->Title->Caption = "Дата складання";

}

void \_\_fastcall TForm4::FormShow(TObject \*Sender)

{

DBColumnSizes();

TADOQuery \*Query = new TADOQuery(this);

Query->Connection = DataModule1->ADOConnection1;

Query->SQL->Text = "SELECT Subject\_id, Name FROM subject";

Query->Open();

ComboBox2->Items->Clear();

while (!Query->Eof)

{

ComboBox2->Items->AddObject(Query->FieldByName("Name")->AsString,

(TObject\*)Query->FieldByName("Subject\_id")->AsInteger);

Query->Next();

}

delete Query;

}

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

void \_\_fastcall TForm4::Lj1Click(TObject \*Sender)

{

Help\_m->ShowModal();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm4::N1Click(TObject \*Sender)

{

this->Close();

Form3->Show();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm4::DBGrid1TitleClick(TColumn \*Column)

{

String sortOrder = SortAscending ? " ASC" : " DESC";

DataModule1->ADOQuery2->Sort = Column->Field->FieldName + sortOrder;

SortAscending = !SortAscending;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm4::DatePicker2CloseUp(TObject \*Sender)

{

try

{

TDateTime selectedDate = DatePicker2->Date;

TDateTime currentDate = Now();

if (selectedDate > currentDate)

{

throw Exception("Дата не може бути в майбутньому!");

}

else if (selectedDate < EncodeDate(1990, 1, 1))

{

throw Exception("Дата занадто стара! Виберіть пізнішу дату.");

}

}

catch (const Exception &e)

{

ShowMessage(e.Message);

DatePicker2->Date = Now();

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm4::FormCreate(TObject \*Sender)

{

Execute->Enabled = false;

Clear->Enabled = false;

ThisDate->OnClick = CheckFiltersFilled;

ComboBox2->OnChange = CheckFiltersFilled;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm4::CheckFiltersFilled(TObject \*Sender)

{

bool isAnyFilterFilled = false;

if (ThisDate->Checked)

isAnyFilterFilled = true;

if (ComboBox2->ItemIndex >= 0)

isAnyFilterFilled = true;

Execute->Enabled = isAnyFilterFilled;

Clear->Enabled = isAnyFilterFilled;

}

void \_\_fastcall TForm4::ExecuteClick(TObject \*Sender)

{

String query = "SELECT c.Condition\_id, s.Name AS Subject\_Name, c.Max\_point, c.Min\_r\_point, c.Min\_point, "

"CASE WHEN c.Status = 1 THEN 1 ELSE 0 END AS Status, "

"c.Date "

"FROM conditions c "

"JOIN subject s ON c.Subject\_id = s.Subject\_id ";

String conditions = "";

if (ThisDate->Checked)

{

if (DatePicker1->Date > DatePicker2->Date)

{

ShowMessage("Дата початку повинна бути раніше дати завершення!");

return;

}

conditions += "c.Date BETWEEN :DateStart AND :DateEnd ";

}

if (ComboBox2->ItemIndex >= 0)

{

if (!conditions.IsEmpty()) conditions += " AND ";

conditions += "c.Subject\_id = :SubjectID ";

}

if (!conditions.IsEmpty())

{

query += " WHERE " + conditions;

}

query += " ORDER BY c.Date DESC";

try

{

DataModule1->ADOQuery2->Close();

DataModule1->ADOQuery2->SQL->Clear();

DataModule1->ADOQuery2->SQL->Add(query);

if (ThisDate->Checked)

{

DataModule1->ADOQuery2->Parameters->ParamByName("DateStart")->Value = DatePicker1->Date.FormatString("yyyy-mm-dd");

DataModule1->ADOQuery2->Parameters->ParamByName("DateEnd")->Value = DatePicker2->Date.FormatString("yyyy-mm-dd");

}

if (ComboBox2->ItemIndex >= 0)

{

DataModule1->ADOQuery2->Parameters->ParamByName("SubjectID")->Value = (int)ComboBox2->Items->Objects[ComboBox2->ItemIndex];

}

DataModule1->ADOQuery2->Open();

DBColumnSizes();

}

catch (const Exception &e)

{

ShowMessage("Помилка при фільтрації: " + e.Message);

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm4::ClearClick(TObject \*Sender)

{

DatePicker2->Date = Now();

ThisDate->Checked = false;

ComboBox2->ItemIndex = -1;

Execute->Enabled = false;

Clear->Enabled = false;

String query = "SELECT c.Condition\_id, s.Name AS Subject\_Name, c.Max\_point, c.Min\_r\_point, c.Min\_point, "

"CASE WHEN c.Status = 1 THEN 1 ELSE 0 END AS Status, "

"c.Date "

"FROM conditions c "

"JOIN subject s ON c.Subject\_id = s.Subject\_id "

"ORDER BY c.Date DESC";

try

{

DataModule1->ADOQuery2->Close();

DataModule1->ADOQuery2->SQL->Clear();

DataModule1->ADOQuery2->SQL->Add(query);

DataModule1->ADOQuery2->Open();

DBColumnSizes();

}

catch (const Exception &e)

{

ShowMessage("Помилка при очищенні фільтрів: " + e.Message);

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm4::N2Click(TObject \*Sender)

{

Form5->set\_id(0);

Form5->ShowModal();

DBColumnSizes();

}

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

void \_\_fastcall TForm4::N3Click(TObject \*Sender)

{

int selected\_id = 0;

if (!DataModule1->DataSource3->DataSet->FieldByName("Condition\_id")->IsNull) {

selected\_id = DataModule1->DataSource3->DataSet->FieldByName("Condition\_id")->AsInteger;

}

Form5->set\_id(selected\_id);

Form5->ShowModal();

DBColumnSizes();

}

//---------------------------------------------------------------------------

void TForm4::ToggleView()

{

if (isMinimalView)

{

Form4->AutoSize = true;

for (int i = 0; i < Form4->MainMenu1->Items->Count; i++)

{

Form4->MainMenu1->Items->Items[i]->Enabled = false;

}

Form4->Caption = "Умови складання НМТ";

for (int i = 0; i < Form4->ControlCount; i++)

{

if (Form4->Controls[i] != Form4->DBGrid1)

{

Form4->Controls[i]->Visible = false;

}

}

}

else

{

Form4->AutoSize = false;

for (int i = 0; i < Form4->MainMenu1->Items->Count; i++)

{

Form4->MainMenu1->Items->Items[i]->Enabled = true;

}

Form4->Caption = "Робота з інформацією про умови складання НМТ";

for (int i = 0; i < Form4->ControlCount; i++)

{

Form4->Controls[i]->Visible = true;

}

}

}

void \_\_fastcall TForm4::DatePicker1CloseUp(TObject \*Sender)

{

try

{

TDateTime selectedDate = DatePicker1->Date;

TDateTime currentDate = Now();

if (selectedDate > currentDate)

{

throw Exception("Дата не може бути в майбутньому!");

}

else if (selectedDate < EncodeDate(1990, 1, 1))

{

throw Exception("Дата занадто стара! Виберіть пізнішу дату.");

}

}

catch (const Exception &e)

{

ShowMessage(e.Message);

DatePicker1->Date = Now();

}

}

//---------------------------------------------------------------------------

Лістинг В.13 – Текст файлу School.h

//---------------------------------------------------------------------------

#ifndef SchoolH

#define SchoolH

//---------------------------------------------------------------------------

#include <System.Classes.hpp>

#include <Vcl.Controls.hpp>

#include <Vcl.StdCtrls.hpp>

#include <Vcl.Forms.hpp>

#include <Data.DB.hpp>

#include <Vcl.DBGrids.hpp>

#include <Vcl.Grids.hpp>

#include <Vcl.Menus.hpp>

#include "Data.h"

#include "Help.h"

#include "Unit8.h"

#include "Main\_Window.h"

//---------------------------------------------------------------------------

class TForm6 : public TForm

{

\_\_published: // IDE-managed Components

TMainMenu \*MainMenu1;

TMenuItem \*N6;

TMenuItem \*Lj1;

TPopupMenu \*PopupMenu1;

TMenuItem \*N1;

TMenuItem \*N2;

TMenuItem \*N3;

TDBGrid \*DBGrid1;

void \_\_fastcall Lj1Click(TObject \*Sender);

void \_\_fastcall N6Click(TObject \*Sender);

void \_\_fastcall FormShow(TObject \*Sender);

void \_\_fastcall N3Click(TObject \*Sender);

void \_\_fastcall N1Click(TObject \*Sender);

void \_\_fastcall N2Click(TObject \*Sender);

void \_\_fastcall DBGrid1TitleClick(TColumn \*Column);

private:

void \_\_fastcall DBColumnSizes();

bool SortAscending;// User declarations

public: // User declarations

\_\_fastcall TForm6(TComponent\* Owner);

};

//---------------------------------------------------------------------------

extern PACKAGE TForm6 \*Form6;

//---------------------------------------------------------------------------

#endif

Лістинг В.14 – Текст файлу School.cpp

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "School.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm6 \*Form6;

//---------------------------------------------------------------------------

\_\_fastcall TForm6::TForm6(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::Lj1Click(TObject \*Sender)

{

Help\_m->ShowModal();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::N6Click(TObject \*Sender)

{

this->Close();

Form3->Show();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::DBColumnSizes(){

DBGrid1->Columns->Items[0]->Visible = false;

DBGrid1->Columns->Items[1]->Width = 90;

DBGrid1->Columns->Items[1]->Title->Caption = "Місто";

DBGrid1->Columns->Items[2]->Width = 150;

DBGrid1->Columns->Items[2]->Title->Caption = "Регіон";

DBGrid1->Columns->Items[3]->Width = 90;

DBGrid1->Columns->Items[3]->Title->Caption = "Тип";

DBGrid1->Columns->Items[4]->Width = 140;

DBGrid1->Columns->Items[4]->Title->Caption = "E-mail";

DBGrid1->Columns->Items[5]->Width = 150;

DBGrid1->Columns->Items[5]->Title->Caption = "ПІБ відповідального";

}

void \_\_fastcall TForm6::FormShow(TObject \*Sender)

{

DBColumnSizes();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::N3Click(TObject \*Sender) {

try {

int id = DataModule1->ADOTable3->FieldByName("School\_id")->AsInteger;

if (id <= 0) {

ShowMessage("Виберіть запис для видалення.");

return;

}

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Text = "SELECT COUNT(\*) AS Count FROM result WHERE School\_id = :id";

DataModule1->ADOQuery1->Parameters->ParamByName("id")->Value = id;

DataModule1->ADOQuery1->Open();

int resultDependentCount = DataModule1->ADOQuery1->FieldByName("Count")->AsInteger;

DataModule1->ADOQuery1->Close();

if (resultDependentCount > 0) {

ShowMessage("Цей запис не може бути видалений, оскільки він використовується в таблиці 'result'.");

return;

}

if (MessageDlg("Ви впевнені, що хочете видалити цей запис?", mtConfirmation, TMsgDlgButtons() << mbYes << mbNo, 0) == mrYes) {

try {

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Text = "DELETE FROM school WHERE School\_id = :id";

DataModule1->ADOQuery1->Parameters->ParamByName("id")->Value = id;

DataModule1->ADOQuery1->ExecSQL();

ShowMessage("Запис успішно видалено.");

DataModule1->ADOTable3->Close();

DataModule1->ADOTable3->Open();

}

catch (const Exception &e) {

ShowMessage("Помилка при видаленні запису: " + e.Message);

}

}

}

catch (const Exception &e) {

ShowMessage("Помилка видалення запису: " + e.Message);

}

DBColumnSizes();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::N1Click(TObject \*Sender)

{

Form8->set\_id(0);

Form8->Caption = "Додавання даних";

Form8->ShowModal();

DBColumnSizes();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::N2Click(TObject \*Sender)

{

int school\_id = DataModule1->ADOTable3->FieldByName("School\_id")->AsInteger;

Form8->set\_id(school\_id);

Form8->Caption = "Редагування даних";

Form8->ShowModal();

DBColumnSizes();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::DBGrid1TitleClick(TColumn \*Column)

{

String sortOrder = SortAscending ? " ASC" : " DESC";

DataModule1->ADOTable3->Sort = Column->Field->FieldName + sortOrder;

SortAscending = !SortAscending;

}

//---------------------------------------------------------------------------

Лістинг В.15 – Текст файлу Results.h

//---------------------------------------------------------------------------

#ifndef ResultsH

#define ResultsH

//---------------------------------------------------------------------------

#include <System.Classes.hpp>

#include <Vcl.Controls.hpp>

#include <Vcl.StdCtrls.hpp>

#include <Vcl.Forms.hpp>

#include <Data.DB.hpp>

#include <Vcl.ComCtrls.hpp>

#include <Vcl.DBGrids.hpp>

#include <Vcl.Grids.hpp>

#include <Vcl.Menus.hpp>

#include <Vcl.ExtCtrls.hpp>

#include <Vcl.WinXPickers.hpp>

#include <Vcl.Mask.hpp>

#include <Vcl.Buttons.hpp>

#include "Data.h"

#include "Unit9.h"

#include "Main\_Window.h"

#include <System.RegularExpressions.hpp>

//---------------------------------------------------------------------------

class TForm11 : public TForm

{

\_\_published: // IDE-managed Components

TPopupMenu \*PopupMenu1;

TMenuItem \*N2;

TMenuItem \*N3;

TGroupBox \*GroupBox1;

TLabel \*Label3;

TComboBox \*ComboBox2;

TBitBtn \*Execute;

TBitBtn \*Clear;

TCheckBox \*ThisDate;

TDatePicker \*DatePicker1;

TLabeledEdit \*LabeledEdit1;

TDBGrid \*DBGrid1;

TMainMenu \*MainMenu1;

TMenuItem \*N6;

TMenuItem \*Lj1;

TDatePicker \*DatePicker2;

void \_\_fastcall FormShow(TObject \*Sender);

void \_\_fastcall N2Click(TObject \*Sender);

void \_\_fastcall DatePicker1CloseUp(TObject \*Sender);

void \_\_fastcall LabeledEdit1Exit(TObject \*Sender);

void \_\_fastcall ExecuteClick(TObject \*Sender);

void \_\_fastcall ClearClick(TObject \*Sender);

void \_\_fastcall FormCreate(TObject \*Sender);

void \_\_fastcall DBGrid1TitleClick(TColumn \*Column);

void \_\_fastcall N3Click(TObject \*Sender);

void \_\_fastcall N6Click(TObject \*Sender);

void \_\_fastcall Lj1Click(TObject \*Sender);

void \_\_fastcall DatePicker2CloseUp(TObject \*Sender);

private:

void \_\_fastcall DBColumnSizes();

//bool SortAscending;

void \_\_fastcall CheckFiltersFilled(TObject \*Sender); // User declarations

public: // User declarations

\_\_fastcall TForm11(TComponent\* Owner);

};

//---------------------------------------------------------------------------

extern PACKAGE TForm11 \*Form11;

//---------------------------------------------------------------------------

#endif

Лістинг В.16 – Текст файлу Results.cpp

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Results.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm11 \*Form11;

//---------------------------------------------------------------------------

\_\_fastcall TForm11::TForm11(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm11::DBColumnSizes() {

if (DBGrid1->Columns->Count >= 10) {

DBGrid1->Columns->Items[0]->Visible = false;

DBGrid1->Columns->Items[1]->Width = 150;

DBGrid1->Columns->Items[1]->Title->Caption = "ПІБ учасника";

DBGrid1->Columns->Items[2]->Width = 90;

DBGrid1->Columns->Items[2]->Title->Caption = "Предмет";

DBGrid1->Columns->Items[3]->Width = 82;

DBGrid1->Columns->Items[3]->Title->Caption = "Результат";

DBGrid1->Columns->Items[4]->Width = 82;

DBGrid1->Columns->Items[4]->Title->Caption = "Максимально";

DBGrid1->Columns->Items[5]->Width = 82;

DBGrid1->Columns->Items[5]->Title->Caption = "Прохідний";

DBGrid1->Columns->Items[6]->Width = 82;

DBGrid1->Columns->Items[6]->Title->Caption = "Статус";

DBGrid1->Columns->Items[7]->Width = 82;

DBGrid1->Columns->Items[7]->Title->Caption = "Дата складання";

DBGrid1->Columns->Items[8]->Width = 82;

DBGrid1->Columns->Items[8]->Title->Caption = "Дата укладання умов";

DBGrid1->Columns->Items[9]->Width = 175;

DBGrid1->Columns->Items[9]->Title->Caption = "E-mail навчального закладу";

} else {

ShowMessage("Недостатньо стовпців у таблиці.");

}

}

void \_\_fastcall TForm11::FormShow(TObject \*Sender)

{

DBColumnSizes();

ComboBox2->Clear();

TADOQuery \*query = new TADOQuery(this);

query->Connection = DataModule1->ADOConnection1;

try

{

query->SQL->Text = "SELECT Name FROM subject ORDER BY Subject\_id";

query->Open();

while (!query->Eof)

{

ComboBox2->Items->Add(query->FieldByName("Name")->AsString);

query->Next();

}

}

\_\_finally

{

query->Close();

delete query;

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm11::N2Click(TObject \*Sender)

{

Form9->set\_id(0);

Form9->ShowModal();

DBColumnSizes();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm11::DatePicker1CloseUp(TObject \*Sender)

{

try

{

TDateTime selectedDate = DatePicker1->Date;

TDateTime currentDate = Now();

if (selectedDate > currentDate)

{

throw Exception("Дата не може бути в майбутньому!");

}

else if (selectedDate < EncodeDate(1990, 1, 1))

{

throw Exception("Дата занадто стара! Виберіть пізнішу дату.");

}

}

catch (const Exception &e)

{

ShowMessage(e.Message);

DatePicker1->Date = Now();

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm11::LabeledEdit1Exit(TObject \*Sender)

{

String pib = LabeledEdit1->Text.Trim();

if (TRegEx::IsMatch(pib, L"[A-Za-z0-9ЁЫЭёыэ!\"#$%&()\*,./:;<=>?@[\\]^\_{|}~]")) {

ShowMessage("Будь ласка, вводьте ПІБ українською мовою!");

return;

LabeledEdit1->SetFocus();

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm11::CheckFiltersFilled(TObject \*Sender)

{

bool isAnyFilterFilled = false;

if (ThisDate->Checked) isAnyFilterFilled = true;

if (!LabeledEdit1->Text.Trim().IsEmpty()) isAnyFilterFilled = true;

if (ComboBox2->ItemIndex >= 0) isAnyFilterFilled = true;

Execute->Enabled = isAnyFilterFilled;

Clear->Enabled = isAnyFilterFilled;

}

void \_\_fastcall TForm11::ClearClick(TObject \*Sender)

{

LabeledEdit1->Clear();

ComboBox2->ItemIndex = -1;

DatePicker1->Date = Now();

ThisDate->Checked = false;

DataModule1->ADOQuery3->Filtered = false;

DataModule1->ADOQuery3->Filter = "";

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Add(

"SELECT r.Res\_id, "

"st.PIB, "

"sub.Name, "

"r.Reached\_score, "

"c.Max\_point, "

"c.Min\_r\_point, "

"CASE WHEN r.Status = 1 THEN 'Зараховано' ELSE 'Не зараховано' END AS Status, "

"r.Attemp\_date, "

"c.Date, "

"school.Email "

"FROM result r "

"JOIN student st ON r.Student\_id = st.Student\_id "

"JOIN subject sub ON r.Subj\_id = sub.Subject\_id "

"JOIN conditions c ON r.Condition\_id = c.Condition\_id "

"JOIN school ON r.School\_id = school.School\_id "

"ORDER BY r.Attemp\_date DESC");

DataModule1->ADOQuery3->Open();

Execute->Enabled = false;

Clear->Enabled = false;

}

void \_\_fastcall TForm11::ExecuteClick(TObject \*Sender)

{

String query = "SELECT r.Res\_id, "

"st.PIB, "

"sub.Name, "

"r.Reached\_score, "

"c.Max\_point, "

"c.Min\_r\_point, "

"CASE WHEN r.Status = 1 THEN 'Зараховано' ELSE 'Не зараховано' END AS Status, "

"r.Attemp\_date, "

"c.Date, "

"school.Email "

"FROM result r "

"JOIN student st ON r.Student\_id = st.Student\_id "

"JOIN subject sub ON r.Subj\_id = sub.Subject\_id "

"JOIN conditions c ON r.Condition\_id = c.Condition\_id "

"JOIN school ON r.School\_id = school.School\_id ";

String conditions = "";

if (ThisDate->Checked)

{

if (DatePicker1->Date > DatePicker2->Date)

{

ShowMessage("Дата початку повинна бути раніше дати завершення!");

return;

}

conditions += "r.Attemp\_date BETWEEN :DateStart AND :DateEnd ";

}

if (!LabeledEdit1->Text.Trim().IsEmpty())

{

if (!conditions.IsEmpty()) conditions += " AND ";

conditions += "st.PIB LIKE :PIB ";

}

if (ComboBox2->ItemIndex >= 0)

{

if (!conditions.IsEmpty()) conditions += " AND ";

conditions += "sub.Name = :Subject ";

}

if (!conditions.IsEmpty())

{

query += " WHERE " + conditions;

}

query += " ORDER BY r.Attemp\_date DESC";

try

{

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Add(query);

if (ThisDate->Checked)

{

DataModule1->ADOQuery3->Parameters->ParamByName("DateStart")->Value = DatePicker1->Date.FormatString("yyyy-mm-dd");

DataModule1->ADOQuery3->Parameters->ParamByName("DateEnd")->Value = DatePicker2->Date.FormatString("yyyy-mm-dd");

}

if (!LabeledEdit1->Text.Trim().IsEmpty())

{

String pibFilter = "%" + LabeledEdit1->Text.Trim() + "%";

DataModule1->ADOQuery3->Parameters->ParamByName("PIB")->Value = pibFilter;

}

if (ComboBox2->ItemIndex >= 0)

{

DataModule1->ADOQuery3->Parameters->ParamByName("Subject")->Value = ComboBox2->Text;

}

DataModule1->ADOQuery3->Open();

DBColumnSizes();

}

catch (Exception &e)

{

ShowMessage("Помилка при фільтрації: " + e.Message);

}

}

void \_\_fastcall TForm11::FormCreate(TObject \*Sender)

{

DBColumnSizes();

DatePicker1->Date = Now();

Clear->Enabled = false;

Execute->Enabled = false;

ComboBox2->OnChange = CheckFiltersFilled;

ThisDate->OnClick = CheckFiltersFilled;

LabeledEdit1->OnChange = CheckFiltersFilled;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm11::DBGrid1TitleClick(TColumn \*Column)

{

String columnName = Column->FieldName;

static bool sortAsc = true;

String sortOrder = sortAsc ? "ASC" : "DESC";

sortAsc = !sortAsc;

String query =

"SELECT r.Res\_id, st.PIB, sub.Name, r.Reached\_score, c.Max\_point, c.Min\_r\_point, "

"CASE WHEN r.Status = 1 THEN 'Зараховано' ELSE 'Не зараховано' END AS Status, "

"r.Attemp\_date, c.Date, school.Email "

"FROM result r "

"JOIN student st ON r.Student\_id = st.Student\_id "

"JOIN subject sub ON r.Subj\_id = sub.Subject\_id "

"JOIN conditions c ON r.Condition\_id = c.Condition\_id "

"JOIN school ON r.School\_id = school.School\_id "

"ORDER BY " + columnName + " " + sortOrder;

try

{

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Add(query);

DataModule1->ADOQuery3->Open();

DBColumnSizes();

}

catch (Exception &e)

{

ShowMessage("Помилка сортування: " + e.Message);

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm11::N3Click(TObject \*Sender)

{

int selected\_id = 0;

if (!DataModule1->DataSource5->DataSet->FieldByName("Res\_id")->IsNull) {

selected\_id = DataModule1->DataSource5->DataSet->FieldByName("Res\_id")->AsInteger;

}

Form9->set\_id(selected\_id);

Form9->ShowModal();

DBColumnSizes();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm11::N6Click(TObject \*Sender)

{

this->Close();

Form3->Show();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm11::Lj1Click(TObject \*Sender)

{

Help\_m->ShowModal();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm11::DatePicker2CloseUp(TObject \*Sender)

{

try

{

TDateTime selectedDate = DatePicker1->Date;

TDateTime currentDate = Now();

if (selectedDate > currentDate)

{

throw Exception("Дата не може бути в майбутньому!");

}

else if (selectedDate < EncodeDate(1990, 1, 1))

{

throw Exception("Дата занадто стара! Виберіть пізнішу дату.");

}

}

catch (const Exception &e)

{

ShowMessage(e.Message);

DatePicker1->Date = Now();

}

}

//-------------------------------------------------------------

Лістинг В.17 – Текст файлу Certificate.h

//---------------------------------------------------------------------------

#ifndef CertificateH

#define CertificateH

//---------------------------------------------------------------------------

#include <System.Classes.hpp>

#include <Vcl.Controls.hpp>

#include <Vcl.StdCtrls.hpp>

#include <Vcl.Forms.hpp>

#include <Data.DB.hpp>

#include <Vcl.DBGrids.hpp>

#include <Vcl.Grids.hpp>

#include <Vcl.Menus.hpp>

#include <Vcl.ComCtrls.hpp>

#include <Vcl.WinXPickers.hpp>

#include <Vcl.ExtCtrls.hpp>

#include <Vcl.Mask.hpp>

#include <Vcl.Buttons.hpp>

#include "Help.h"

#include "Main\_Window.h"

#include "Data.h"

#include "Unit10.h"

#include "Unit13.h"

#include <Vcl.Dialogs.hpp>

//---------------------------------------------------------------------------

class TCertificates : public TForm

{

\_\_published: // IDE-managed Components

TDBGrid \*DBGrid2;

TPopupMenu \*PopupMenu1;

TMenuItem \*N2;

TMenuItem \*N3;

TMainMenu \*MainMenu1;

TMenuItem \*N1;

TMenuItem \*Lj1;

TGroupBox \*GroupBox1;

TLabeledEdit \*LabeledEdit1;

TDatePicker \*DatePicker1;

TCheckBox \*ThisDate;

TBitBtn \*Execute;

TBitBtn \*Clear;

TBitBtn \*BitBtn1;

TSaveDialog \*SaveDialog1;

TMenuItem \*N4;

TDatePicker \*DatePicker2;

void \_\_fastcall N1Click(TObject \*Sender);

void \_\_fastcall Lj1Click(TObject \*Sender);

void \_\_fastcall FormShow(TObject \*Sender);

void \_\_fastcall DatePicker1CloseUp(TObject \*Sender);

void \_\_fastcall LabeledEdit1Exit(TObject \*Sender);

void \_\_fastcall FormCreate(TObject \*Sender);

void \_\_fastcall ExecuteClick(TObject \*Sender);

void \_\_fastcall ClearClick(TObject \*Sender);

void \_\_fastcall DBGrid2TitleClick(TColumn \*Column);

void \_\_fastcall N3Click(TObject \*Sender);

void \_\_fastcall N2Click(TObject \*Sender);

void \_\_fastcall BitBtn1Click(TObject \*Sender);

void \_\_fastcall N4Click(TObject \*Sender);

void \_\_fastcall DatePicker2CloseUp(TObject \*Sender);

private:

void DBColumnSizes();

bool SortAscending; // User declarations

public: // User declarations

\_\_fastcall TCertificates(TComponent\* Owner);

void \_\_fastcall CheckFiltersFilled(TObject \*Sender);

};

//---------------------------------------------------------------------------

extern PACKAGE TCertificates \*Certificates;

//---------------------------------------------------------------------------

#endif

Лістинг В.18 – Текст файлу Certificate.cpp

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Certificate.h"

#include <System.DateUtils.hpp>

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TCertificates \*Certificates;

//---------------------------------------------------------------------------

\_\_fastcall TCertificates::TCertificates(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TCertificates::N1Click(TObject \*Sender)

{

this->Close();

Form3->Show();

}

//---------------------------------------------------------------------------

void \_\_fastcall TCertificates::Lj1Click(TObject \*Sender)

{

Help\_m->ShowModal();

}

//---------------------------------------------------------------------------

void TCertificates::DBColumnSizes(){

DBGrid2->Columns->Items[0]->Visible = false;

DBGrid2->Columns->Items[1]->Width = 200;

DBGrid2->Columns->Items[1]->Title->Caption = "ПІБ учасника";

DBGrid2->Columns->Items[2]->Width = 100;

DBGrid2->Columns->Items[2]->Title->Caption = "PIN";

DBGrid2->Columns->Items[3]->Width = 100;

DBGrid2->Columns->Items[3]->Title->Caption = "Дата створення";

DBGrid2->Columns->Items[4]->Width = 150;

DBGrid2->Columns->Items[4]->Title->Caption = "Дата закінчення";

DBGrid2->Columns->Items[5]->Width = 150;

DBGrid2->Columns->Items[5]->Title->Caption = "Статус";

}

void \_\_fastcall TCertificates::FormShow(TObject \*Sender)

{

DBColumnSizes();

}

//---------------------------------------------------------------------------

void \_\_fastcall TCertificates::DatePicker1CloseUp(TObject \*Sender)

{

try

{

TDateTime selectedDate = DatePicker1->Date;

TDateTime currentDate = Now();

if (selectedDate > currentDate)

{

throw Exception("Дата не може бути в майбутньому!");

}

else if (selectedDate < EncodeDate(1990, 1, 1))

{

throw Exception("Дата занадто стара! Виберіть пізнішу дату.");

}

}

catch (const Exception &e)

{

ShowMessage("Помилка: " + e.Message);

DatePicker1->Date = Now();

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TCertificates::LabeledEdit1Exit(TObject \*Sender)

{

try

{

String enteredPIN = LabeledEdit1->Text.Trim();

if (enteredPIN.IsEmpty())

{

throw Exception("Поле PIN не може бути порожнім!");

}

TADOQuery \*query = new TADOQuery(NULL);

try

{

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = "SELECT COUNT(\*) AS Count FROM Certificate WHERE PIN = :EnteredPIN";

query->Parameters->ParamByName("EnteredPIN")->Value = enteredPIN;

query->Open();

int count = query->FieldByName("Count")->AsInteger;

if (count <= 0)

{

throw Exception("PIN не існує в базі даних!");

}

}

\_\_finally

{

delete query;

}

}

catch (const Exception &e)

{

ShowMessage(e.Message);

LabeledEdit1->SetFocus();

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TCertificates::FormCreate(TObject \*Sender)

{

Execute->Enabled = false;

Clear->Enabled = false;

ThisDate->OnClick = CheckFiltersFilled;

RadioGroup1->OnClick = CheckFiltersFilled;

}

//---------------------------------------------------------------------------

void \_\_fastcall TCertificates::CheckFiltersFilled(TObject \*Sender)

{

bool isAnyFilterFilled = ThisDate->Checked || (RadioGroup1->ItemIndex != -1);

Execute->Enabled = isAnyFilterFilled;

Clear->Enabled = isAnyFilterFilled;

}

void \_\_fastcall TCertificates::ExecuteClick(TObject \*Sender)

{

String query = "SELECT certificate.Cerf\_num, student.PIB, certificate.PIN, certificate.Creation\_date, certificate.Effect\_time, "

"CASE WHEN certificate.Status = 1 THEN 'Дійсний' ELSE 'Не дійсний' END AS Status "

"FROM certificate "

"JOIN student ON certificate.Student\_id = student.Student\_id ";

String conditions = "";

if (ThisDate->Checked)

{

if (DatePicker1->Date > DatePicker2->Date)

{

ShowMessage("Дата початку повинна бути раніше дати завершення!");

return;

}

conditions += "certificate.Creation\_date BETWEEN :DateStart AND :DateEnd ";

}

if (RadioGroup1->ItemIndex != -1)

{

if (!conditions.IsEmpty())

conditions += " AND ";

int statusValue = RadioGroup1->ItemIndex == 0 ? 1 : 0;

conditions += "certificate.Status = :Status";

}

if (!conditions.IsEmpty())

query += " WHERE " + conditions;

query += " ORDER BY certificate.Creation\_date DESC";

try

{

DataModule1->ADOQuery4->Close();

DataModule1->ADOQuery4->SQL->Clear();

DataModule1->ADOQuery4->SQL->Add(query);

if (ThisDate->Checked)

{

DataModule1->ADOQuery4->Parameters->ParamByName("DateStart")->Value = DatePicker1->Date.FormatString("yyyy-mm-dd");

DataModule1->ADOQuery4->Parameters->ParamByName("DateEnd")->Value = DatePicker2->Date.FormatString("yyyy-mm-dd");

}

if (RadioGroup1->ItemIndex != -1)

{

int statusValue = RadioGroup1->ItemIndex == 0 ? 1 : 0;

DataModule1->ADOQuery4->Parameters->ParamByName("Status")->Value = statusValue;

}

DataModule1->ADOQuery4->Open();

DBColumnSizes();

int recordCount = DataModule1->ADOQuery4->RecordCount;

StatusBar1->Panels->Items[0]->Text = "Кількість сертифікатів: " + IntToStr(recordCount);

}

catch (const Exception &e)

{

ShowMessage("Помилка при фільтрації: " + e.Message);

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TCertificates::ClearClick(TObject \*Sender)

{

DatePicker1->Date = Now();

ThisDate->Checked = false;

RadioGroup1->ItemIndex = -1;

Execute->Enabled = false;

Clear->Enabled = false;

String query = "SELECT certificate.Cerf\_num, student.PIB, certificate.PIN, certificate.Creation\_date, certificate.Effect\_time, "

"CASE WHEN certificate.Status = 1 THEN 'Дійсний' ELSE 'Не дійсний' END AS Status "

"FROM certificate "

"JOIN student ON certificate.Student\_id = student.Student\_id "

"ORDER BY certificate.Creation\_date DESC";

try

{

DataModule1->ADOQuery4->Close();

DataModule1->ADOQuery4->SQL->Clear();

DataModule1->ADOQuery4->SQL->Add(query);

DataModule1->ADOQuery4->Open();

StatusBar1->Panels->Items[0]->Text = " ";

DBColumnSizes();

}

catch (const Exception &e)

{

ShowMessage("Помилка при очищенні фільтрів: " + e.Message);

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TCertificates::DBGrid2TitleClick(TColumn \*Column)

{

String sortOrder = SortAscending ? " ASC" : " DESC";

DataModule1->ADOQuery4->Sort = Column->Field->FieldName + sortOrder;

SortAscending = !SortAscending;

}

//---------------------------------------------------------------------------

void \_\_fastcall TCertificates::N3Click(TObject \*Sender)

{

int selected\_id = 0;

if (!DataModule1->DataSource6->DataSet->FieldByName("Cerf\_num")->IsNull) {

selected\_id = DataModule1->DataSource6->DataSet->FieldByName("Cerf\_num")->AsInteger;

}

Form10->set\_id(selected\_id);

Form10->ShowModal();

DBColumnSizes();

}

//---------------------------------------------------------------------------

void \_\_fastcall TCertificates::N2Click(TObject \*Sender)

{

Form10->set\_id(0);

Form10->ShowModal();

DBColumnSizes();

}

//---------------------------------------------------------------------------

void \_\_fastcall TCertificates::BitBtn1Click(TObject \*Sender)

{

try

{

String enteredPIN = LabeledEdit1->Text.Trim();

String query = "SELECT certificate.Cerf\_num, student.PIB, certificate.PIN, certificate.Creation\_date, certificate.Effect\_time, "

"certificate.Student\_id, "

"CASE WHEN certificate.Status = 1 THEN 'Дійсний' ELSE 'Не дійсний' END AS Status "

"FROM certificate "

"JOIN student ON certificate.Student\_id = student.Student\_id "

"WHERE certificate.PIN = :PIN";

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Clear();

DataModule1->ADOQuery1->SQL->Add(query);

DataModule1->ADOQuery1->Parameters->ParamByName("PIN")->Value = enteredPIN;

DataModule1->ADOQuery1->Open();

if (DataModule1->ADOQuery1->RecordCount > 0)

{

String cerf\_num = DataModule1->ADOQuery1->FieldByName("Cerf\_num")->AsString;

String pib = DataModule1->ADOQuery1->FieldByName("PIB")->AsString;

String pin = DataModule1->ADOQuery1->FieldByName("PIN")->AsString;

String creation\_date = DataModule1->ADOQuery1->FieldByName("Creation\_date")->AsString;

String effect\_time = DataModule1->ADOQuery1->FieldByName("Effect\_time")->AsString;

int student\_id = DataModule1->ADOQuery1->FieldByName("Student\_id")->AsInteger;

String resultQuery = "SELECT result.Reached\_score AS Reached\_score, result.Subj\_id AS SubjectID "

"FROM result "

"WHERE result.Student\_id = :StudentID";

DataModule1->ADOQuery2->Close();

DataModule1->ADOQuery2->SQL->Clear();

DataModule1->ADOQuery2->SQL->Add(resultQuery);

DataModule1->ADOQuery2->Parameters->ParamByName("StudentID")->Value = student\_id;

DataModule1->ADOQuery2->Open();

TStringList \*results = new TStringList();

while (!DataModule1->ADOQuery2->Eof)

{

int subject\_id = DataModule1->ADOQuery2->FieldByName("SubjectID")->AsInteger;

double reached\_score = DataModule1->ADOQuery2->FieldByName("Reached\_score")->AsFloat;

String subjectQuery = "SELECT subject.Name AS Subject, conditions.Max\_point AS Max\_point, conditions.Min\_r\_point AS Min\_r\_point "

"FROM subject "

"JOIN conditions ON subject.Subject\_id = conditions.Subject\_id "

"WHERE subject.Subject\_id = :SubjectID";

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Add(subjectQuery);

DataModule1->ADOQuery3->Parameters->ParamByName("SubjectID")->Value = subject\_id;

DataModule1->ADOQuery3->Open();

if (DataModule1->ADOQuery3->RecordCount > 0)

{

String subject = DataModule1->ADOQuery3->FieldByName("Subject")->AsString;

double max\_point = DataModule1->ADOQuery3->FieldByName("Max\_point")->AsFloat;

double min\_r\_point = DataModule1->ADOQuery3->FieldByName("Min\_r\_point")->AsFloat;

double score\_200;

if (reached\_score < min\_r\_point)

{

score\_200 = 100.0;

}

else if (reached\_score >= max\_point)

{

score\_200 = 200.0;

}

else

{

score\_200 = 100.0 + ((reached\_score - min\_r\_point) / (max\_point - min\_r\_point)) \* 100.0;

}

results->Add("<div class='result-item'>");

results->Add("<p>Предмет: <span class='highlight'>" + subject + "</span></p>");

results->Add("<p>Набрані бали (200-бальна шкала): <span class='highlight'>" + FloatToStrF(score\_200, ffFixed, 7, 2) + "</span></п>");

results->Add("</div>");

}

DataModule1->ADOQuery2->Next();

}

SaveDialog1->Filter = "HTML files (\*.html)|\*.html";

SaveDialog1->DefaultExt = "html";

String currentDateTime = FormatDateTime("yyyy\_mm\_dd\_hh\_nn\_ss", Now());

SaveDialog1->FileName = Form13->template\_name + "\_" + currentDateTime + ".html";

if (SaveDialog1->Execute())

{

String filePath = SaveDialog1->FileName;

TStringList \*htmlFile = new TStringList();

htmlFile->Text = "";

try

{

htmlFile->Add("<!DOCTYPE html>");

htmlFile->Add("<html lang='uk'>");

htmlFile->Add("<head>");

htmlFile->Add(" <meta charset='UTF-8'>");

htmlFile->Add(" <meta name='viewport' content='width=device-width, initial-scale=1.0'>");

htmlFile->Add(" <title>Сертифікат НМТ</title>");

htmlFile->Add(" <style>");

htmlFile->Add(" body { font-family: Arial, sans-serif; margin: 20px; }");

htmlFile->Add(" .certificate { border: 5px solid #4CAF50; padding: 20px; max-width: 800px; margin: 0 auto; background-color: #f9f9f9; }");

htmlFile->Add(" .certificate-header { text-align: center; margin-bottom: 30px; }");

htmlFile->Add(" .certificate-header h1 { margin: 0; font-size: 28px; color: #333; }");

htmlFile->Add(" .certificate-header p { margin: 5px 0; font-size: 18px; color: #666; }");

htmlFile->Add(" .certificate-body { margin-bottom: 30px; }");

htmlFile->Add(" .certificate-body p { font-size: 18px; margin: 10px 0; }");

htmlFile->Add(" .certificate-footer { text-align: center; font-size: 16px; color: #777; }");

htmlFile->Add(" .highlight { font-weight: bold; color: #333; }");

htmlFile->Add(" .results { margin-top: 20px; padding: 15px; border: 2px solid #ddd; background-color: #f0f0f0; }");

htmlFile->Add(" </style>");

htmlFile->Add("</head>");

htmlFile->Add("<body>");

htmlFile->Add(" <div class='certificate'>");

htmlFile->Add(" <div class='certificate-header'>");

htmlFile->Add(" <h1>Сертифікат НМТ</h1>");

htmlFile->Add(" <p>Національний Мультипредметний Тест</п>");

htmlFile->Add(" <p>Дата створення: <span class='highlight'>" + creation\_date + "</span></п>");

htmlFile->Add(" </div>");

htmlFile->Add(" <div class='certificate-body'>");

htmlFile->Add(" <p>Номер сертифіката: <span class='highlight'>" + cerf\_num + "</span></п>");

htmlFile->Add(" <p>Прізвище, Ім'я, По батькові: <span class='highlight'>" + pib + "</span></п>");

htmlFile->Add(" <p>PIN-код: <span class='highlight'>" + pin + "</span></п>");

htmlFile->Add(" <p>Термін дії сертифіката: <span class='highlight'>" + effect\_time + "</span></п>");

htmlFile->Add(" </div>");

htmlFile->Add(" <div class='results'>");

htmlFile->Add(" <h2>Результати за рік</h2>");

htmlFile->AddStrings(results);

htmlFile->Add(" </div>");

htmlFile->Add(" </div>");

htmlFile->Add("</body>");

htmlFile->Add("</html>");

htmlFile->SaveToFile(filePath, TEncoding::UTF8);

ShowMessage("HTML-файл створено за адресою: " + filePath);

}

\_\_finally

{

delete htmlFile;

delete results;

}

}

else

{

ShowMessage("Збереження було скасовано.");

}

}

else

{

ShowMessage("Дані про сертифікат не знайдено.");

}

}

catch (const Exception &e)

{

ShowMessage("Помилка при створенні HTML: " + e.Message);

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TCertificates::N4Click(TObject \*Sender)

{

bool k =true;

Form13->setfromCertificate(k);

Form13->ShowModal();

}

//---------------------------------------------------------------------------

void \_\_fastcall TCertificates::DatePicker2CloseUp(TObject \*Sender)

{

try

{

TDateTime selectedDate = DatePicker2->Date;

TDateTime currentDate = Now();

if (selectedDate > currentDate)

{

throw Exception("Дата не може бути в майбутньому!");

}

else if (selectedDate < EncodeDate(1990, 1, 1))

{

throw Exception("Дата занадто стара! Виберіть пізнішу дату.");

}

}

catch (const Exception &e)

{

ShowMessage("Помилка: " + e.Message);

DatePicker2->Date = Now();

}

}

//----------

Лістинг В.19 – Текст файлу Help.cpp

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Help.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

THelp\_m \*Help\_m;

//---------------------------------------------------------------------------

\_\_fastcall THelp\_m::THelp\_m(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

Лістинг В.20 – Текст файлу Data.cpp

//---------------------------------------------------------------------------

#pragma hdrstop

#include "Data.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma classgroup "Vcl.Controls.TControl"

#pragma resource "\*.dfm"

TDataModule1 \*DataModule1;

//---------------------------------------------------------------------------

\_\_fastcall TDataModule1::TDataModule1(TComponent\* Owner)

: TDataModule(Owner)

{

}

//---------------------------------------------------------------------------

Лістинг В.21 – Текст файлу Unit1.h

//---------------------------------------------------------------------------

#ifndef Unit1H

#define Unit1H

//---------------------------------------------------------------------------

#include <System.Classes.hpp>

#include <Vcl.Controls.hpp>

#include <Vcl.StdCtrls.hpp>

#include <Vcl.Forms.hpp>

#include <Vcl.ExtCtrls.hpp>

#include <Vcl.WinXPickers.hpp>

//---------------------------------------------------------------------------

class TForm1 : public TForm

{

\_\_published: // IDE-managed Components

TGroupBox \*GroupBox1;

TLabel \*Label1;

TLabel \*Label2;

TLabel \*Label4;

TLabel \*Label5;

TLabel \*Label6;

TLabel \*Label7;

TLabel \*Label8;

TEdit \*Edit2;

TComboBox \*ComboBox1;

TDatePicker \*DatePicker1;

TRadioGroup \*RadioGroup1;

TEdit \*Edit3;

TEdit \*Edit4;

TEdit \*Edit5;

TEdit \*Edit6;

TEdit \*Edit7;

TButton \*Button1;

TLabel \*Label3;

TEdit \*Edit1;

void \_\_fastcall FormShow(TObject \*Sender);

void \_\_fastcall Edit2Exit(TObject \*Sender);

void \_\_fastcall Edit4Exit(TObject \*Sender);

void \_\_fastcall Edit6Exit(TObject \*Sender);

void \_\_fastcall Edit5Exit(TObject \*Sender);

void \_\_fastcall Edit3Exit(TObject \*Sender);

void \_\_fastcall LoadStudentData(int student\_id);

void \_\_fastcall DatePicker1CloseUp(TObject \*Sender);

void \_\_fastcall ComboBox1CloseUp(TObject \*Sender);

void \_\_fastcall RadioGroup1Exit(TObject \*Sender);

void \_\_fastcall Button1Click(TObject \*Sender);

void \_\_fastcall Edit1Exit(TObject \*Sender);

private:

int id;

void ClearFields();

bool IsAllDigits(const String& str); // User declarations

public:

void set\_id(int k); // User declarations

\_\_fastcall TForm1(TComponent\* Owner);

};

//---------------------------------------------------------------------------

extern PACKAGE TForm1 \*Form1;

//---------------------------------------------------------------------------

#endif

Лістинг В.22 – Текст файлу Unit1.cpp

#include <vcl.h>

#pragma hdrstop

#include "Unit1.h"

#include "Data.h"

#include <DateUtils.hpp>

#include <System.RegularExpressions.hpp>

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm1 \*Form1;

//---------------------------------------------------------------------------

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner), previousPassportNum(""), previousPhoneNum(""), previousPN(""), previousEmail(""), previousPIB("")

{

}

//---------------------------------------------------------------------------

void TForm1::set\_id(int k) {

id = k;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::LoadStudentData(int student\_id)

{

TADOQuery \*query = new TADOQuery(this);

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = "SELECT \* FROM student WHERE Student\_id = :Student\_id";

query->Parameters->ParamByName("Student\_id")->Value = student\_id;

try {

query->Open();

if (!query->Eof) {

Edit2->Text = query->FieldByName("Passport\_num")->AsString;

ComboBox1->Text = query->FieldByName("Passport\_type")->AsString;

Edit1->Text = query->FieldByName("PIB")->AsString;

DatePicker1->Date = query->FieldByName("Birth\_date")->AsDateTime;

RadioGroup1->ItemIndex = (query->FieldByName("Gender")->AsString == "M") ? 0 : 1;

Edit5->Text = query->FieldByName("Email")->AsString;

Edit3->Text = query->FieldByName("Phone\_num")->AsString;

Edit6->Text = query->FieldByName("EduCerf\_num")->AsString;

Edit4->Text = query->FieldByName("PN")->AsString;

Edit7->Text = query->FieldByName("Additional")->AsString;

// Сохранение предыдущих значений

previousPassportNum = Edit2->Text;

previousPhoneNum = Edit3->Text;

previousPN = Edit4->Text;

previousEmail = Edit5->Text;

previousPIB = Edit1->Text;

} else {

ShowMessage("Студента з вказаним ID не знайдено.");

}

}

\_\_finally {

query->Close();

delete query;

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::FormShow(TObject \*Sender)

{

if (id == 0) {

Form1->Caption = "Додавання даних";

Button1->Caption = "Додати";

ClearFields();

} else {

Form1->Caption = "Редагування даних";

LoadStudentData(id);

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Edit2Exit(TObject \*Sender)

{

if (Edit2->Text.Length() < 8 || Edit2->Text.Length() > 20) {

ShowMessage("Номер паспорта повинен бути від 8 до 20 символів.");

Edit2->SetFocus();

return;

}

// Проверка уникальности, если значение изменилось

if (Edit2->Text != previousPassportNum) {

TADOQuery \*checkQuery = new TADOQuery(this);

checkQuery->Connection = DataModule1->ADOConnection1;

checkQuery->SQL->Text = "SELECT COUNT(\*) AS Cnt FROM student WHERE Passport\_num = :Passport\_num AND (Student\_id <> :Student\_id OR :Student\_id IS NULL)";

checkQuery->Parameters->ParamByName("Passport\_num")->Value = Edit2->Text;

checkQuery->Parameters->ParamByName("Student\_id")->Value = id;

checkQuery->Open();

if (checkQuery->FieldByName("Cnt")->AsInteger > 0)

{

ShowMessage("Цей номер паспорта вже існує.");

}

checkQuery->Close();

delete checkQuery;

previousPassportNum = Edit2->Text; // Обновление предыдущего значения

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Edit3Exit(TObject \*Sender)

{

if (Edit3->Text.Length() < 10 || Edit3->Text.Length() > 15) {

ShowMessage("Номер телефону повинен бути від 10 до 15 символів.");

Edit3->SetFocus();

return;

}

// Проверка уникальности, если значение изменилось

if (Edit3->Text != previousPhoneNum) {

TADOQuery \*checkQuery = new TADOQuery(this);

checkQuery->Connection = DataModule1->ADOConnection1;

checkQuery->SQL->Text = "SELECT COUNT(\*) AS Cnt FROM student WHERE Phone\_num = :Phone\_num AND (Student\_id <> :Student\_id OR :Student\_id IS NULL)";

checkQuery->Parameters->ParamByName("Phone\_num")->Value = Edit3->Text;

checkQuery->Parameters->ParamByName("Student\_id")->Value = id;

checkQuery->Open();

if (checkQuery->FieldByName("Cnt")->AsInteger > 0)

{

ShowMessage("Цей номер телефону вже існує.");

}

checkQuery->Close();

delete checkQuery;

previousPhoneNum = Edit3->Text; // Обновление предыдущего значения

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Edit4Exit(TObject \*Sender)

{

if (Edit4->Text.Length() != 10) {

ShowMessage("ІПН повинен містити 10 цифр.");

Edit4->SetFocus();

return;

}

// Проверка уникальности, если значение изменилось

if (Edit4->Text != previousPN) {

TADOQuery \*checkQuery = new TADOQuery(this);

checkQuery->Connection = DataModule1->ADOConnection1;

checkQuery->SQL->Text = "SELECT COUNT(\*) AS Cnt FROM student WHERE PN = :PN AND (Student\_id <> :Student\_id OR :Student\_id IS NULL)";

checkQuery->Parameters->ParamByName("PN")->Value = Edit4->Text;

checkQuery->Parameters->ParamByName("Student\_id")->Value = id;

checkQuery->Open();

if (checkQuery->FieldByName("Cnt")->AsInteger > 0)

{

ShowMessage("Цей ІПН вже існує.");

}

checkQuery->Close();

delete checkQuery;

previousPN = Edit4->Text; // Обновление предыдущего значения

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Edit5Exit(TObject \*Sender)

{

String email = Edit5->Text;

UnicodeString pattern = "^\\w+([-+.']\\w+)\*@\\w+([-.]\\w+)\*\\.\\w+([-.]\\w+)\*$";

if (!TRegEx::IsMatch(email, pattern)) {

ShowMessage("Введіть дійсний e-mail.");

Edit5->SetFocus();

return;

}

// Проверка уникальности, если значение изменилось

if (email != previousEmail) {

TADOQuery \*checkQuery = new TADOQuery(this);

checkQuery->Connection = DataModule1->ADOConnection1;

checkQuery->SQL->Text = "SELECT COUNT(\*) AS Cnt FROM student WHERE Email = :Email AND (Student\_id <> :Student\_id OR :Student\_id IS NULL)";

checkQuery->Parameters->ParamByName("Email")->Value = email;

checkQuery->Parameters->ParamByName("Student\_id")->Value = id;

checkQuery->Open();

if (checkQuery->FieldByName("Cnt")->AsInteger > 0)

{

ShowMessage("Цей e-mail вже існує.");

}

checkQuery->Close();

delete checkQuery;

previousEmail = email; // Обновление предыдущего значения

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Edit6Exit(TObject \*Sender)

{

if (Edit6->Text.IsEmpty()) {

ShowMessage("Поле сертифікату освіти не може бути пустим.");

return;

}

if (Edit6->Text != previousEduCerfNum) {

TADOQuery \*checkQuery = new TADOQuery(this);

checkQuery->Connection = DataModule1->ADOConnection1;

checkQuery->SQL->Text = "SELECT COUNT(\*) AS Cnt FROM student WHERE EduCerf\_num = :EduCerf\_num AND (Student\_id <> :Student\_id OR :Student\_id IS NULL)";

checkQuery->Parameters->ParamByName("EduCerf\_num")->Value = Edit6->Text;

checkQuery->Parameters->ParamByName("Student\_id")->Value = id;

checkQuery->Open();

if (checkQuery->FieldByName("Cnt")->AsInteger > 0)

{

ShowMessage("Цей номер сертифіката про освіту вже існує.");

Edit6->SetFocus();

}

checkQuery->Close();

delete checkQuery;

previousEduCerfNum = Edit6->Text; // Обновление предыдущего значения

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::DatePicker1CloseUp(TObject \*Sender)

{

try

{

TDateTime selectedDate = DatePicker1->Date;

TDateTime currentDate = Now();

if (selectedDate > currentDate)

{

throw Exception("Дата не може бути в майбутньому!");

}

else if (selectedDate < EncodeDate(1925, 1, 1))

{

throw Exception("Дата занадто стара! Виберіть пізнішу дату.");

}

int birthYear = YearOf(selectedDate);

int currentYear = YearOf(currentDate);

int yearsDiff = currentYear - birthYear;

if (currentDate < EncodeDate(currentYear, MonthOf(selectedDate), DayOf(selectedDate)))

{

yearsDiff--;

}

if (yearsDiff < 16)

{

throw Exception("Для здачі НМТ потрібно мати мінімум 16 років.");

}

}

catch (const Exception &e)

{

ShowMessage(e.Message);

DatePicker1->Date = Now();

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::ComboBox1CloseUp(TObject \*Sender)

{

if (ComboBox1->Text.IsEmpty() || ComboBox1->ItemIndex == -1) {

ShowMessage("Будь ласка, оберіть тип паспорта.");

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::RadioGroup1Exit(TObject \*Sender)

{

if (RadioGroup1->ItemIndex == -1) {

ShowMessage("Будь ласка, оберіть стать.");

}

}

//---------------------------------------------------------------------------

bool TForm1::IsAllDigits(const String& str)

{

for (int i = 1; i <= str.Length(); ++i)

{

if (!isdigit(str[i]))

return false;

}

return true;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button1Click(TObject \*Sender)

{

if (Edit2->Text.IsEmpty() || ComboBox1->Text.IsEmpty() || Edit1->Text.IsEmpty() ||

DatePicker1->Date > Date() || Edit5->Text.IsEmpty() || Edit3->Text.IsEmpty() ||

Edit6->Text.IsEmpty() || Edit4->Text.IsEmpty() || RadioGroup1->ItemIndex == -1)

{

ShowMessage("Не всі обов'язкові поля заповнені.");

return;

}

if (ComboBox1->Text != "ID-карта" && ComboBox1->Text != "Паперовий")

{

ShowMessage("Невірний тип паспорта. Допустимі значення: 'ID-карта' або 'Паперовий'.");

return;

}

if (!Edit3->Text.IsEmpty() && !IsAllDigits(Edit3->Text))

{

ShowMessage("Телефонний номер повинен містити тільки цифри.");

return;

}

TADOQuery \*query = new TADOQuery(this);

query->Connection = DataModule1->ADOConnection1;

try {

if(id == 0) {

query->SQL->Text = "INSERT INTO student (Passport\_num, Passport\_type, PIB, Birth\_date, Gender, Email, Phone\_num, EduCerf\_num, PN, Additional) "

"VALUES (:Passport\_num, :Passport\_type, :PIB, :Birth\_date, :Gender, :Email, :Phone\_num, :EduCerf\_num, :PN, :Additional)";

} else {

query->SQL->Text = "UPDATE student SET Passport\_num = :Passport\_num, Passport\_type = :Passport\_type, PIB = :PIB, "

"Birth\_date = :Birth\_date, Gender = :Gender, Email = :Email, Phone\_num = :Phone\_num, "

"EduCerf\_num = :EduCerf\_num, PN = :PN, Additional = :Additional WHERE Student\_id = :Student\_id";

query->Parameters->ParamByName("Student\_id")->Value = id;

}

query->Parameters->ParamByName("Passport\_num")->Value = Edit2->Text;

query->Parameters->ParamByName("Passport\_type")->Value = ComboBox1->Text;

query->Parameters->ParamByName("PIB")->Value = Edit1->Text;

TDate birthDate = DatePicker1->Date;

String birthDateString = birthDate.FormatString("yyyy-mm-dd");

query->Parameters->ParamByName("Birth\_date")->Value = birthDateString;

query->Parameters->ParamByName("Gender")->Value = (RadioGroup1->ItemIndex == 0) ? "M" : "F";

query->Parameters->ParamByName("Email")->Value = Edit5->Text;

query->Parameters->ParamByName("Phone\_num")->Value = Edit3->Text;

query->Parameters->ParamByName("EduCerf\_num")->Value = Edit6->Text;

query->Parameters->ParamByName("PN")->Value = Edit4->Text;

query->Parameters->ParamByName("Additional")->Value = Edit7->Text;

query->ExecSQL();

ShowMessage(id == 0 ? "Дані успішно додані!" : "Дані успішно оновлені!");

}

\_\_finally {

delete query;

}

DataModule1->DataSource1->DataSet->Close();

DataModule1->DataSource1->DataSet->Open();

Close();

}

//---------------------------------------------------------------------------

void TForm1::ClearFields()

{

Edit1->Text = "";

Edit2->Text = "";

Edit3->Text = "";

Edit4->Text = "";

Edit5->Text = "";

Edit6->Text = "";

Edit7->Text = "";

ComboBox1->ItemIndex = -1;

DatePicker1->Date = Now();

RadioGroup1->ItemIndex = -1;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Edit1Exit(TObject \*Sender)

{

String pib = Edit1->Text.Trim();

if (TRegEx::IsMatch(pib, L"[A-Za-z0-9ЁЫЭёыэ!\"#$%&()\*,./:;<=>?@[\\]^\_{|}~]")) {

ShowMessage("Будь ласка, вводьте ПІБ українською мовою!");

return;

Edit1->SetFocus();

}

if (pib != previousPIB) {

TADOQuery \*checkQuery = new TADOQuery(this);

checkQuery->Connection = DataModule1->ADOConnection1;

checkQuery->SQL->Text = "SELECT COUNT(\*) AS Cnt FROM student WHERE PIB = :PIB AND (Student\_id <> :Student\_id OR :Student\_id IS NULL)";

checkQuery->Parameters->ParamByName("PIB")->Value = pib;

checkQuery->Parameters->ParamByName("Student\_id")->Value = id;

try {

checkQuery->Open();

if (checkQuery->FieldByName("Cnt")->AsInteger > 0) {

ShowMessage("Студент з таким ПІБ вже існує.");

Edit1->SetFocus();

}

}

\_\_finally {

checkQuery->Close();

delete checkQuery;

}

previousPIB = pib; // Обновление предыдущего значения

}

}

//---------------------------------------------------------------------------

Лістинг В.23 – Текст файлу Unit2.h

//---------------------------------------------------------------------------

#ifndef Unit2H

#define Unit2H

//---------------------------------------------------------------------------

#include <System.Classes.hpp>

#include <Vcl.Controls.hpp>

#include <Vcl.StdCtrls.hpp>

#include <Vcl.Forms.hpp>

#include <Vcl.Dialogs.hpp>

#include <Vcl.ExtDlgs.hpp>

//---------------------------------------------------------------------------

class TForm2 : public TForm

{

\_\_published: // IDE-managed Components

TGroupBox \*GroupBox1;

TLabel \*Label4;

TEdit \*Edit2;

TEdit \*Edit4;

TButton \*Button1;

TLabel \*Label1;

TEdit \*Edit1;

TButton \*Button2;

TOpenPictureDialog \*OpenPictureDialog1;

TLabel \*Label2;

TButton \*Button3;

TEdit \*Edit3;

TLabel \*Label3;

TOpenTextFileDialog \*OpenTextFileDialog1;

void \_\_fastcall Edit2Exit(TObject \*Sender);

void \_\_fastcall Edit4Exit(TObject \*Sender);

void \_\_fastcall Edit1Exit(TObject \*Sender);

void \_\_fastcall Button2Click(TObject \*Sender);

void \_\_fastcall Button1Click(TObject \*Sender);

void \_\_fastcall FormShow(TObject \*Sender);

void \_\_fastcall Button3Click(TObject \*Sender);

void \_\_fastcall Edit3Exit(TObject \*Sender);

private:

int id; // User declarations

public:

void \_\_fastcall set\_id(int k); // User declarations

\_\_fastcall TForm2(TComponent\* Owner);

};

//---------------------------------------------------------------------------

extern PACKAGE TForm2 \*Form2;

//---------------------------------------------------------------------------

#endif

Лістинг В.24 – Текст файлу Unit2.cpp

#include <vcl.h>

#pragma hdrstop

#include "Unit2.h"

#include "Data.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm2 \*Form2;

//---------------------------------------------------------------------------

\_\_fastcall TForm2::TForm2(TComponent\* Owner) : TForm(Owner) {}

bool TForm2::IsValidUkrainianText(const String &text) {

for (int i = 1; i <= text.Length(); i++) {

wchar\_t c = text[i];

if (!((c >= L'А' && c <= L'Я') || (c >= L'а' && c <= L'я') ||

c == L'І' || c == L'і' || c == L'Ї' || c == L'ї' ||

c == L'Є' || c == L'є' || c == L' ' || c == L'\'' || c == L'-')) {

return false;

}

}

return true;

}

void \_\_fastcall TForm2::Edit1Exit(TObject \*Sender)

{

if (Edit1->Text.Trim().IsEmpty())

{

ShowMessage("Поле 'Шлях до зображення' не може бути порожнім.");

}

else if (!FileExists(Edit1->Text))

{

ShowMessage("Файл зображення не знайдено за вказаним шляхом.");

}

else if (Edit1->Text.SubString(Edit1->Text.Length() - 3, 4).LowerCase() != ".jpg" &&

Edit1->Text.SubString(Edit1->Text.Length() - 3, 4).LowerCase() != ".png")

{

ShowMessage("Файл зображення повинен мати розширення .jpg або .png.");

}

}

void \_\_fastcall TForm2::Edit2Exit(TObject \*Sender)

{

if (Edit2->Text.Trim().IsEmpty())

{

ShowMessage("Поле 'Назва предмету' не може бути порожнім.");

Edit2->SetFocus();

}

else if (Edit2->Text.Length() > 100)

{

ShowMessage("Назва предмету не може перевищувати 100 символів.");

Edit2->SetFocus();

}

else if (!IsValidUkrainianText(Edit2->Text))

{

ShowMessage("Назва предмету повинна містити лише українські літери, пробіл, апостроф або дефіс.");

Edit2->SetFocus();

}

else

{

try

{

String subjectName = Edit2->Text.Trim();

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Text = "SELECT COUNT(\*) AS Count FROM subject WHERE Name = :subjectName";

DataModule1->ADOQuery1->Parameters->ParamByName("subjectName")->Value = subjectName;

DataModule1->ADOQuery1->Open();

int count = DataModule1->ADOQuery1->FieldByName("Count")->AsInteger;

if (count > 0 && id == 0)

{

ShowMessage("Назва предмету вже існує. Будь ласка, виберіть іншу назву.");

Edit2->SetFocus();

}

}

catch (const Exception &e)

{

ShowMessage("Помилка перевірки унікальності: " + e.Message);

}

}

}

void \_\_fastcall TForm2::Edit4Exit(TObject \*Sender)

{

if (Edit4->Text.Length() > 40)

{

ShowMessage("Опис не може перевищувати 40 символів.");

Edit4->SetFocus();

}

else if (!IsValidUkrainianText(Edit4->Text))

{

ShowMessage("Опис повинен містити лише українські літери, пробіл, апостроф або дефіс.");

Edit4->SetFocus();

}

}

void \_\_fastcall TForm2::Button2Click(TObject \*Sender)

{

if (OpenPictureDialog1->Execute())

{

Edit1->Text = OpenPictureDialog1->FileName;

if (!FileExists(Edit1->Text))

{

ShowMessage("Файл зображення не знайдено за вказаним шляхом.");

}

}

}

void \_\_fastcall TForm2::set\_id(int k)

{

id = k;

if (id > 0)

{

try

{

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Text = "SELECT Image\_name, Name, Description, Test\_sample FROM subject WHERE Subject\_id = :id";

DataModule1->ADOQuery1->Parameters->ParamByName("id")->Value = id;

DataModule1->ADOQuery1->Open();

if (!DataModule1->ADOQuery1->Eof)

{

Edit1->Text = DataModule1->ADOQuery1->FieldByName("Image\_name")->AsString;

Edit2->Text = DataModule1->ADOQuery1->FieldByName("Name")->AsString;

Edit4->Text = DataModule1->ADOQuery1->FieldByName("Description")->AsString;

Edit3->Text = DataModule1->ADOQuery1->FieldByName("Test\_sample")->AsString;

}

else

{

ShowMessage("Запис з вказаним ID не знайдено.");

}

}

catch (const Exception &e)

{

ShowMessage("Помилка завантаження даних для редагування: " + e.Message);

}

}

else

{

Edit1->Clear();

Edit2->Clear();

Edit4->Clear();

Edit3->Clear();

}

}

void \_\_fastcall TForm2::Button1Click(TObject \*Sender)

{

try

{

if (Edit1->Text.Trim().IsEmpty() || Edit2->Text.Trim().IsEmpty() || Edit4->Text.Trim().IsEmpty())

{

ShowMessage("Усі поля повинні бути заповнені перед додаванням або оновленням.");

return;

}

String imagePath = Edit1->Text.Trim();

String subjectName = Edit2->Text.Trim();

String description = Edit4->Text.Trim();

String testSamplePath = Edit3->Text.Trim();

if (id > 0)

{

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Text = "UPDATE subject SET Image\_name = :imagePath, Name = :subjectName, Description = :description, Test\_sample = :testSamplePath WHERE Subject\_id = :id";

DataModule1->ADOQuery1->Parameters->ParamByName("imagePath")->Value = imagePath;

DataModule1->ADOQuery1->Parameters->ParamByName("subjectName")->Value = subjectName;

DataModule1->ADOQuery1->Parameters->ParamByName("description")->Value = description;

if (!testSamplePath.IsEmpty())

DataModule1->ADOQuery1->Parameters->ParamByName("testSamplePath")->Value = testSamplePath;

else

DataModule1->ADOQuery1->Parameters->ParamByName("testSamplePath")->Value = Variant();

DataModule1->ADOQuery1->Parameters->ParamByName("id")->Value = id;

DataModule1->ADOQuery1->ExecSQL();

ShowMessage("Запис успішно оновлено.");

}

else

{

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Text = "INSERT INTO subject (Image\_name, Name, Description, Test\_sample) VALUES (:imagePath, :subjectName, :description, :testSamplePath)";

DataModule1->ADOQuery1->Parameters->ParamByName("imagePath")->Value = imagePath;

DataModule1->ADOQuery1->Parameters->ParamByName("subjectName")->Value = subjectName;

DataModule1->ADOQuery1->Parameters->ParamByName("description")->Value = description;

if (!testSamplePath.IsEmpty())

DataModule1->ADOQuery1->Parameters->ParamByName("testSamplePath")->Value = testSamplePath;

else

DataModule1->ADOQuery1->Parameters->ParamByName("testSamplePath")->Value = Variant();

DataModule1->ADOQuery1->ExecSQL();

ShowMessage("Новий запис успішно додано до бази даних.");

}

DataModule1->DataSource2->DataSet->Close();

DataModule1->DataSource2->DataSet->Open();

Close();

}

catch (const Exception &e)

{

ShowMessage("Помилка обробки запису: " + e.Message);

}

}

void \_\_fastcall TForm2::FormShow(TObject \*Sender)

{

if (id > 0) Button1->Caption = "Підтвердити";

else Button1->Caption = "Додати";

}

void \_\_fastcall TForm2::Button3Click(TObject \*Sender)

{

TOpenDialog \*OpenDialog = new TOpenDialog(this);

OpenDialog->Filter = "Документи Word або PDF|\*.docx;\*.pdf";

if (OpenDialog->Execute())

{

Edit3->Text = OpenDialog->FileName;

}

delete OpenDialog;

}

void \_\_fastcall TForm2::Edit3Exit(TObject \*Sender)

{

String testSamplePath = Edit3->Text.Trim();

if (!testSamplePath.IsEmpty() && !FileExists(testSamplePath))

{

ShowMessage("Файл не знайдено за вказаним шляхом.");

}

else if (!testSamplePath.IsEmpty() &&

!(testSamplePath.SubString(testSamplePath.Length() - 3, 4).LowerCase() == ".docx" ||

testSamplePath.SubString(testSamplePath.Length() - 3, 4).LowerCase() == ".pdf"))

{

ShowMessage("Файл має бути у форматі .docx або .pdf.");

}

}

Лістинг В.25 – Текст файлу Unit5.h

//---------------------------------------------------------------------------

#ifndef Unit5H

#define Unit5H

//---------------------------------------------------------------------------

#include <System.Classes.hpp>

#include <Vcl.Controls.hpp>

#include <Vcl.StdCtrls.hpp>

#include <Vcl.Forms.hpp>

#include <Vcl.ExtCtrls.hpp>

#include <Vcl.WinXPickers.hpp>

#include "Data.h"

//---------------------------------------------------------------------------

class TForm5 : public TForm

{

\_\_published: // IDE-managed Components

TGroupBox \*GroupBox1;

TLabel \*Label2;

TLabel \*Label4;

TLabel \*Label5;

TLabel \*Label6;

TEdit \*Edit2;

TDatePicker \*DatePicker1;

TRadioGroup \*RadioGroup1;

TEdit \*Edit4;

TEdit \*Edit6;

TButton \*Button1;

TComboBox \*ComboBox1;

TLabel \*Label1;

void \_\_fastcall DatePicker1CloseUp(TObject \*Sender);

void \_\_fastcall Button1Click(TObject \*Sender);

private:

int id; // User declarations

public: // User declarations

\_\_fastcall TForm5(TComponent\* Owner);

void set\_id(int k);

};

//---------------------------------------------------------------------------

extern PACKAGE TForm5 \*Form5;

//---------------------------------------------------------------------------

#endif

Лістинг В.26 – Текст файлу Unit5.cpp

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit5.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm5 \*Form5;

//---------------------------------------------------------------------------

\_\_fastcall TForm5::TForm5(TComponent\* Owner)

: TForm(Owner)

{

}

void TForm5::set\_id(int k) {

if (k == 0) {

Form5->Caption = "Додавання даних";

Button1->Caption = "Додати";

RadioGroup1->ItemIndex = -1;

Edit2->Text = "";

Edit4->Text = "";

Edit6->Text = "";

DatePicker1->Date = Date();

ComboBox1->ItemIndex = -1;

ComboBox1->Text = "";

TADOQuery \*Query = new TADOQuery(this);

Query->Connection = DataModule1->ADOConnection1;

Query->SQL->Text = "SELECT Subject\_id, Name FROM subject";

Query->Open();

ComboBox1->Items->Clear();

while (!Query->Eof)

{

ComboBox1->Items->AddObject(Query->FieldByName("Name")->AsString,

(TObject\*)Query->FieldByName("Subject\_id")->AsInteger);

Query->Next();

}

delete Query;

} else {

Form5->Caption = "Редагування даних";

Button1->Caption = "Підтвердити";

TADOQuery \*query = new TADOQuery(this);

TADOQuery \*Query = new TADOQuery(this);

Query->Connection = DataModule1->ADOConnection1;

Query->SQL->Text = "SELECT Subject\_id, Name FROM subject";

Query->Open();

ComboBox1->Items->Clear();

while (!Query->Eof)

{

ComboBox1->Items->AddObject(Query->FieldByName("Name")->AsString,

(TObject\*)Query->FieldByName("Subject\_id")->AsInteger);

Query->Next();

}

delete Query;

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = "SELECT Status, Max\_point, Min\_r\_point, Min\_point, Date, Subject\_id FROM conditions WHERE Condition\_id = :Condition\_id";

query->Parameters->ParamByName("Condition\_id")->Value = k;

query->Open();

if (!query->Eof) {

RadioGroup1->ItemIndex = query->FieldByName("Status")->AsInteger;

Edit2->Text = query->FieldByName("Max\_point")->AsString;

Edit6->Text = query->FieldByName("Min\_r\_point")->AsString;

Edit4->Text = query->FieldByName("Min\_point")->AsString;

DatePicker1->Date = query->FieldByName("Date")->AsDateTime;

int subject\_id = query->FieldByName("Subject\_id")->AsInteger;

for (int i = 0; i < ComboBox1->Items->Count; i++) {

if ((int)ComboBox1->Items->Objects[i] == subject\_id) {

ComboBox1->ItemIndex = i;

break;

}

}

}

query->Close();

delete query;

}

id = k;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm5::DatePicker1CloseUp(TObject \*Sender)

{

try

{

TDateTime selectedDate = DatePicker1->Date;

TDateTime currentDate = Now();

if (selectedDate > currentDate)

{

ShowMessage("Дата не може бути в майбутньому!");

DatePicker1->SetFocus();

throw Exception("");

}

else if (selectedDate < EncodeDate(1999, 1, 1))

{

ShowMessage("Дата занадто стара! Виберіть пізнішу дату.");

DatePicker1->SetFocus();

throw Exception("");

}

}

catch (const Exception &e)

{

DatePicker1->Date = Now();

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm5::Button1Click(TObject \*Sender) {

if (RadioGroup1->ItemIndex == -1) {

ShowMessage("Будь ласка, виберіть статус.");

RadioGroup1->SetFocus();

return;

}

if (Edit2->Text.IsEmpty()) {

ShowMessage("Будь ласка, введіть максимальний бал.");

Edit2->SetFocus();

return;

}

if (Edit4->Text.IsEmpty()) {

ShowMessage("Будь ласка, введіть мінімальний бал.");

Edit4->SetFocus();

return;

}

if (Edit6->Text.IsEmpty()) {

ShowMessage("Будь ласка, введіть мінімальний рейтинг.");

Edit6->SetFocus();

return;

}

if (ComboBox1->ItemIndex == -1) {

ShowMessage("Будь ласка, виберіть предмет.");

ComboBox1->SetFocus();

return;

}

int max\_point = StrToInt(Edit2->Text);

int min\_r\_point = StrToInt(Edit6->Text);

int min\_point = StrToInt(Edit4->Text);

int status = RadioGroup1->ItemIndex;

String formattedDate = FormatDateTime("yyyy-mm-dd", DatePicker1->Date);

String subject\_name = ComboBox1->Items->Strings[ComboBox1->ItemIndex];

int subject\_id = 0;

TADOQuery \*subjectQuery = new TADOQuery(this);

subjectQuery->Connection = DataModule1->ADOConnection1;

subjectQuery->SQL->Text = "SELECT Subject\_id FROM subject WHERE Name = :Name";

subjectQuery->Parameters->ParamByName("Name")->Value = subject\_name;

subjectQuery->Open();

if (!subjectQuery->Eof) {

subject\_id = subjectQuery->FieldByName("Subject\_id")->AsInteger;

} else {

ShowMessage("Обраний предмет не знайдено.");

ComboBox1->SetFocus();

subjectQuery->Close();

delete subjectQuery;

return;

}

subjectQuery->Close();

delete subjectQuery;

if (!(max\_point > min\_point && min\_r\_point < max\_point && min\_r\_point > min\_point)) {

ShowMessage("Помилка: значення не відповідають умовам. Переконайтеся, що: \nMax\_point > Min\_point\nMin\_r\_point < Max\_point\nMin\_r\_point > Min\_point.");

Edit2->SetFocus();

return;

}

if (id == 0) {

TADOQuery \*query = new TADOQuery(this);

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = "INSERT INTO conditions (Max\_point, Min\_r\_point, Min\_point, Status, Date, Subject\_id) VALUES (:Max\_point, :Min\_r\_point, :Min\_point, :Status, :Date, :Subject\_id)";

query->Parameters->ParamByName("Max\_point")->Value = max\_point;

query->Parameters->ParamByName("Min\_r\_point")->Value = min\_r\_point;

query->Parameters->ParamByName("Min\_point")->Value = min\_point;

query->Parameters->ParamByName("Status")->Value = status;

query->Parameters->ParamByName("Date")->Value = formattedDate;

query->Parameters->ParamByName("Subject\_id")->Value = subject\_id;

query->ExecSQL();

delete query;

ShowMessage("Новий запис успішно додано.");

} else {

TADOQuery \*query = new TADOQuery(this);

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = "UPDATE conditions SET Max\_point = :Max\_point, Min\_r\_point = :Min\_r\_point, Min\_point = :Min\_point, Status = :Status, Date = :Date, Subject\_id = :Subject\_id WHERE Condition\_id = :Condition\_id";

query->Parameters->ParamByName("Max\_point")->Value = max\_point;

query->Parameters->ParamByName("Min\_r\_point")->Value = min\_r\_point;

query->Parameters->ParamByName("Min\_point")->Value = min\_point;

query->Parameters->ParamByName("Status")->Value = status;

query->Parameters->ParamByName("Date")->Value = formattedDate;

query->Parameters->ParamByName("Subject\_id")->Value = subject\_id;

query->Parameters->ParamByName("Condition\_id")->Value = id;

query->ExecSQL();

delete query;

ShowMessage("Запис успішно відредаговано.");

}

DataModule1->DataSource3->DataSet->Close();

DataModule1->DataSource3->DataSet->Open();

Form5->Close();

}

//-------------------------------------------------------------

Лістинг В.27 – Текст файлу Unit8.h

#ifndef Unit8H

#define Unit8H

#include <System.Classes.hpp>

#include <Vcl.Controls.hpp>

#include <Vcl.StdCtrls.hpp>

#include <Vcl.Forms.hpp>

#include <Vcl.ExtCtrls.hpp>

#include <Vcl.Mask.hpp>

#include "Data.h"

//---------------------------------------------------------------------------

class TForm8 : public TForm

{

\_\_published: // IDE-managed Components

TLabeledEdit \*LabeledEdit1;

TLabeledEdit \*LabeledEdit2;

TLabeledEdit \*LabeledEdit4;

TLabeledEdit \*LabeledEdit5;

TButton \*Button1;

TComboBox \*ComboBox1;

TLabel \*Label1;

void \_\_fastcall LabeledEdit1Exit(TObject \*Sender);

void \_\_fastcall LabeledEdit2Exit(TObject \*Sender);

void \_\_fastcall LabeledEdit4Exit(TObject \*Sender);

void \_\_fastcall LabeledEdit5Exit(TObject \*Sender);

void \_\_fastcall Button1Click(TObject \*Sender);

void \_\_fastcall FormShow(TObject \*Sender);

private:

bool IsUniqueValue(String field, String value);

bool IsValidUkrainianInput(String input);

int id;

public:

\_\_fastcall TForm8(TComponent\* Owner);

void set\_id(int k);

};

//---------------------------------------------------------------------------

extern PACKAGE TForm8 \*Form8;

//---------------------------------------------------------------------------

#endif

Лістинг В.28 – Текст файлу Unit8.cpp

#include <vcl.h>

#pragma hdrstop

#include "Unit8.h"

#pragma package(smart\_init)

#include <System.RegularExpressions.hpp>

#pragma resource "\*.dfm"

TForm8 \*Form8;

\_\_fastcall TForm8::TForm8(TComponent\* Owner)

: TForm(Owner)

{

}

void TForm8::set\_id(int k)

{

id = k;

if (id > 0)

{

Button1->Caption = "Замінити";

AnsiString queryStr = "SELECT \* FROM school WHERE School\_id = :id";

TADOQuery \*query = new TADOQuery(NULL);

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = queryStr;

query->Parameters->ParamByName("id")->Value = id;

query->Open();

if (!query->Eof)

{

LabeledEdit1->Text = query->FieldByName("City")->AsString;

LabeledEdit2->Text = query->FieldByName("Region")->AsString;

LabeledEdit4->Text = query->FieldByName("Email")->AsString;

LabeledEdit5->Text = query->FieldByName("Teacher\_PIB")->AsString;

originalEmail = LabeledEdit4->Text;

String schoolType = query->FieldByName("Type")->AsString;

schoolType = schoolType.Trim();

ComboBox1->ItemIndex = ComboBox1->Items->IndexOf(schoolType);

}

query->Close();

delete query;

}

else

{

Button1->Caption = "Додати";

LabeledEdit1->Text = "";

LabeledEdit2->Text = "";

LabeledEdit4->Text = "";

LabeledEdit5->Text = "";

ComboBox1->ItemIndex = -1;

originalEmail = "";

}

}

bool TForm8::IsValidUkrainianInput(String input)

{

for (int i = 1; i <= input.Length(); i++)

{

wchar\_t c = input[i];

if (!((c >= L'А' && c <= L'Я') || (c >= L'а' && c <= L'я') ||

c == L'І' || c == L'і' || c == L'Ї' || c == L'ї' ||

c == L'Є' || c == L'є' || c == L' '))

{

return false;

}

}

return true;

}

bool TForm8::IsUniqueValue(String field, String value)

{

AnsiString queryStr = "SELECT COUNT(\*) FROM school WHERE " + field + " = :value";

TADOQuery \*query = new TADOQuery(NULL);

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = queryStr;

query->Parameters->ParamByName("value")->Value = value;

query->Open();

bool isUnique = query->Fields->Fields[0]->AsInteger == 0;

query->Close();

delete query;

return isUnique;

}

void \_\_fastcall TForm8::LabeledEdit1Exit(TObject \*Sender)

{

String input = LabeledEdit1->Text;

if (!IsValidUkrainianInput(input))

{

ShowMessage("Будь ласка, використовуйте тільки українські символи для назви міста.");

LabeledEdit1->SetFocus();

return;

}

}

void \_\_fastcall TForm8::LabeledEdit2Exit(TObject \*Sender)

{

String input = LabeledEdit2->Text;

if (!IsValidUkrainianInput(input))

{

ShowMessage("Будь ласка, використовуйте тільки українські символи для назви області.");

LabeledEdit2->SetFocus();

return;

}

}

void \_\_fastcall TForm8::LabeledEdit4Exit(TObject \*Sender)

{

String input = LabeledEdit4->Text;

UnicodeString pattern = "^\\w+([-+.']\\w+)\*@\\w+([-.]\\w+)\*\\.\\w+([-.]\\w+)\*$";

if (!TRegEx::IsMatch(input, pattern))

{

ShowMessage("Введіть дійсний e-mail.");

LabeledEdit4->SetFocus();

return;

}

if (input != originalEmail && !IsUniqueValue("Email", input))

{

ShowMessage("Такий E-mail вже існує в базі даних.");

LabeledEdit4->SetFocus();

return;

}

}

void \_\_fastcall TForm8::LabeledEdit5Exit(TObject \*Sender)

{

String input = LabeledEdit5->Text;

if (!IsValidUkrainianInput(input))

{

ShowMessage("Будь ласка, використовуйте тільки українські символи для ПІБ.");

LabeledEdit5->SetFocus();

return;

}

}

void \_\_fastcall TForm8::Button1Click(TObject \*Sender)

{

if (LabeledEdit1->Text.IsEmpty())

{

ShowMessage("Будь ласка, введіть назву міста.");

LabeledEdit1->SetFocus();

return;

}

if (LabeledEdit2->Text.IsEmpty())

{

ShowMessage("Будь ласка, введіть назву області.");

LabeledEdit2->SetFocus();

return;

}

if (LabeledEdit4->Text.IsEmpty())

{

ShowMessage("Будь ласка, введіть E-mail.");

LabeledEdit4->SetFocus();

return;

}

if (LabeledEdit5->Text.IsEmpty())

{

ShowMessage("Будь ласка, введіть ПІБ.");

LabeledEdit5->SetFocus();

return;

}

if (ComboBox1->ItemIndex == -1)

{

ShowMessage("Будь ласка, виберіть тип навчального закладу.");

ComboBox1->SetFocus();

return;

}

AnsiString queryStr;

if (id > 0)

{

queryStr = "UPDATE school SET City = :City, Region = :Region, Type = :Type, Email = :Email, Teacher\_PIB = :Teacher\_PIB WHERE School\_id = :id";

}

else

{

queryStr = "INSERT INTO school (City, Region, Type, Email, Teacher\_PIB) VALUES (:City, :Region, :Type, :Email, :Teacher\_PIB)";

}

TADOQuery \*query = new TADOQuery(NULL);

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = queryStr;

query->Parameters->ParamByName("City")->Value = LabeledEdit1->Text;

query->Parameters->ParamByName("Region")->Value = LabeledEdit2->Text;

query->Parameters->ParamByName("Type")->Value = ComboBox1->Text;

query->Parameters->ParamByName("Email")->Value = LabeledEdit4->Text;

query->Parameters->ParamByName("Teacher\_PIB")->Value = LabeledEdit5->Text;

if (id > 0)

query->Parameters->ParamByName("id")->Value = id;

query->ExecSQL();

delete query;

DataModule1->ADOTable3->Close();

DataModule1->ADOTable3->Open();

Close();

}

void \_\_fastcall TForm8::FormShow(TObject \*Sender)

{

if (id > 0) ComboBox1->SetFocus();

}

Лістинг В.29 – Текст файлу Unit9.h

//---------------------------------------------------------------------------

#ifndef Unit9H

#define Unit9H

//---------------------------------------------------------------------------

#include <System.Classes.hpp>

#include <Vcl.Controls.hpp>

#include <Vcl.StdCtrls.hpp>

#include <Vcl.Forms.hpp>

#include <Vcl.ExtCtrls.hpp>

#include <Vcl.Mask.hpp>

#include <Vcl.WinXPickers.hpp>

#include "Data.h"

#include "Students.h"

#include "Conditions.h"

#include "Main\_Window.h"

//---------------------------------------------------------------------------

class TForm9 : public TForm

{

\_\_published: // IDE-managed Components

TGroupBox \*GroupBox1;

TLabel \*Label1;

TLabel \*Label2;

TLabel \*Label3;

TButton \*Button1;

TLabeledEdit \*LabeledEdit1;

TLabeledEdit \*LabeledEdit4;

TComboBox \*ComboBox1;

TDatePicker \*DatePicker2;

TRadioGroup \*RadioGroup1;

TDatePicker \*DatePicker1;

TComboBox \*ComboBox2;

TLabel \*Label4;

TLabel \*Label5;

TLabel \*Label6;

TLabel \*Label7;

void \_\_fastcall FormCreate(TObject \*Sender);

void \_\_fastcall LabeledEdit1Exit(TObject \*Sender);

void \_\_fastcall DatePicker2CloseUp(TObject \*Sender);

void \_\_fastcall DatePicker1CloseUp(TObject \*Sender);

void \_\_fastcall LabeledEdit4Exit(TObject \*Sender);

void \_\_fastcall ComboBox1CloseUp(TObject \*Sender);

void \_\_fastcall Button1Click(TObject \*Sender);

void \_\_fastcall LabeledEdit1SubLabelClick(TObject \*Sender);

void \_\_fastcall Label2Click(TObject \*Sender);

private:

int id;

void \_\_fastcall ShowForm14Modal();

public: // User declarations

\_\_fastcall TForm9(TComponent\* Owner);

void set\_id(int k);

};

//---------------------------------------------------------------------------

extern PACKAGE TForm9 \*Form9;

//---------------------------------------------------------------------------

#endif

Лістинг В.30 – Текст файлу Unit9.cpp

#include <vcl.h>

#pragma hdrstop

#include "Unit9.h"

#include <System.RegularExpressions.hpp>

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm9 \*Form9;

\_\_fastcall TForm9::TForm9(TComponent\* Owner)

: TForm(Owner)

{

}

void TForm9::set\_id(int k) {

id = k;

if (id > 0) {

Button1->Caption = "Замінити";

AnsiString queryStr = "SELECT r.Subj\_id, r.Condition\_id, r.Status, st.PIB, r.Reached\_score, r.Attemp\_date, "

"school.Email, c.Date, c.Max\_point, c.Min\_r\_point, c.Min\_point "

"FROM result r "

"JOIN student st ON r.Student\_id = st.Student\_id "

"JOIN school ON r.School\_id = school.School\_id "

"JOIN conditions c ON r.Condition\_id = c.Condition\_id "

"WHERE r.Res\_id = :id";

TADOQuery \*query = new TADOQuery(NULL);

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = queryStr;

query->Parameters->ParamByName("id")->Value = id;

query->Open();

if (!query->Eof) {

LabeledEdit1->Text = query->FieldByName("PIB")->AsString;

LabeledEdit4->Text = query->FieldByName("Reached\_score")->AsString;

int subject\_id = query->FieldByName("Subj\_id")->AsInteger;

ComboBox1->ItemIndex = subject\_id - 1;

DatePicker2->Date = query->FieldByName("Date")->AsDateTime;

String email = query->FieldByName("Email")->AsString;

ComboBox2->ItemIndex = ComboBox2->Items->IndexOf(email);

int status = query->FieldByName("Status")->AsInteger;

RadioGroup1->ItemIndex = (status == 1) ? 0 : 1;

int maxPoint = query->FieldByName("Max\_point")->AsInteger;

int minPassPoint = query->FieldByName("Min\_r\_point")->AsInteger;

int minPoint = query->FieldByName("Min\_point")->AsInteger;

Label5->Caption = "Максимальний: " + IntToStr(maxPoint);

Label7->Caption = "Прохідний: " + IntToStr(minPassPoint);

Label6->Caption = "Мінімальний: " + IntToStr(minPoint);

}

query->Close();

delete query;

Form9->Caption = "Редагування даних";

} else {

Button1->Caption = "Додати";

LabeledEdit1->Text = "";

LabeledEdit4->Text = "";

ComboBox1->ItemIndex = -1;

ComboBox2->ItemIndex = -1;

DatePicker2->Date = EncodeDate(1990, 1, 1);

RadioGroup1->ItemIndex = -1;

Label5->Caption = "Максимальний: -";

Label7->Caption = "Прохідний: -";

Label6->Caption = "Мінімальний: -";

Form9->Caption = "Додавання даних";

}

}

void \_\_fastcall TForm9::FormCreate(TObject \*Sender) {

if (id == 0) {

TDateTime checkDate = EncodeDate(1990, 1, 1);

DatePicker2->Date = checkDate;

}

RadioGroup1->Enabled = false;

ComboBox1->Clear();

ComboBox2->Clear();

TADOQuery \*query = new TADOQuery(this);

query->Connection = DataModule1->ADOConnection1;

try {

query->SQL->Text = "SELECT Name FROM subject ORDER BY Subject\_id";

query->Open();

while (!query->Eof) {

ComboBox1->Items->Add(query->FieldByName("Name")->AsString);

query->Next();

}

query->Close();

query->SQL->Text = "SELECT Email FROM school ORDER BY School\_id";

query->Open();

while (!query->Eof) {

ComboBox2->Items->Add(query->FieldByName("Email")->AsString);

query->Next();

}

} \_\_finally {

query->Close();

delete query;

}

}

void \_\_fastcall TForm9::LabeledEdit1Exit(TObject \*Sender) {

String pib = LabeledEdit1->Text;

UnicodeString pattern = "^[А-ЯІЇЄҐа-яіїєґ' ]+$";

if (!TRegEx::IsMatch(pib, pattern)) {

ShowMessage("ПІБ повинен містити лише українські літери.");

return;

}

TADOQuery \*query = new TADOQuery(NULL);

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = "SELECT COUNT(\*) AS count FROM student WHERE PIB = :pib";

query->Parameters->ParamByName("pib")->Value = pib;

query->Open();

int count = query->FieldByName("count")->AsInteger;

if (count == 0) {

ShowMessage("ПІБ не знайдений у базі даних.");

query->Close();

delete query;

return;

}

query->Close();

delete query;

}

void \_\_fastcall TForm9::DatePicker2CloseUp(TObject \*Sender) {

int subject\_id = ComboBox1->ItemIndex + 1;

TDateTime selectedDate = DatePicker2->Date;

String formattedDate = selectedDate.FormatString("yyyy-mm-dd");

TADOQuery \*query = new TADOQuery(NULL);

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = "SELECT Max\_point, Min\_r\_point, Min\_point FROM conditions WHERE Date = :date AND Subject\_id = :subject\_id";

query->Parameters->ParamByName("date")->Value = formattedDate;

query->Parameters->ParamByName("subject\_id")->Value = subject\_id;

query->Open();

if (!query->Eof) {

int maxPoint = query->FieldByName("Max\_point")->AsInteger;

int minPassPoint = query->FieldByName("Min\_r\_point")->AsInteger;

int minPoint = query->FieldByName("Min\_point")->AsInteger;

Label5->Caption = "Максимальний: " + IntToStr(maxPoint);

Label7->Caption = "Прохідний: " + IntToStr(minPassPoint);

Label6->Caption = "Мінімальний: " + IntToStr(minPoint);

} else {

ShowMessage("Для вибраної дати та предмета немає даних у таблиці.");

Label5->Caption = "Максимальний: -";

Label7->Caption = "Прохідний: -";

Label6->Caption = "Мінімальний: -";

}

query->Close();

delete query;

}

void \_\_fastcall TForm9::DatePicker1CloseUp(TObject \*Sender) {

try {

TDateTime selectedDate = DatePicker1->Date;

TDateTime currentDate = Now();

if (selectedDate > currentDate) {

throw Exception("Дата не може бути в майбутньому!");

} else if (selectedDate < EncodeDate(1990, 1, 1)) {

throw Exception("Дата занадто стара! Виберіть пізнішу дату.");

}

} catch (const Exception &e) {

ShowMessage(e.Message);

DatePicker1->Date = Now();

}

}

void \_\_fastcall TForm9::LabeledEdit4Exit(TObject \*Sender) {

TDateTime checkDate = EncodeDate(1990, 1, 1);

if (ComboBox1->ItemIndex == -1 || DatePicker2->Date == checkDate) {

ShowMessage("Будь ласка, виберіть предмет та дату укладання вимог.");

return;

}

int subject\_id = ComboBox1->ItemIndex + 1;

TDateTime conditionDate = DatePicker2->Date;

String enteredScoreStr = LabeledEdit4->Text;

int enteredScore = enteredScoreStr.ToIntDef(-1);

if (enteredScore == -1) {

ShowMessage("Будь ласка, введіть коректне значення балу.");

return;

}

String formattedDate = conditionDate.FormatString("yyyy-mm-dd");

TADOQuery \*query = new TADOQuery(NULL);

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = "SELECT Max\_point, Min\_r\_point, Min\_point FROM conditions WHERE Subject\_id = :subject\_id AND Date = :date";

query->Parameters->ParamByName("subject\_id")->Value = subject\_id;

query->Parameters->ParamByName("date")->Value = formattedDate;

query->Open();

if (!query->Eof) {

int maxPoint = query->FieldByName("Max\_point")->AsInteger;

int minPassPoint = query->FieldByName("Min\_r\_point")->AsInteger;

int minPoint = query->FieldByName("Min\_point")->AsInteger;

if (enteredScore < minPoint) {

ShowMessage("Введений бал менший за мінімальний.");

} else if (enteredScore > maxPoint) {

ShowMessage("Введений бал більший за максимальний.");

} else {

if (enteredScore >= minPassPoint) {

RadioGroup1->ItemIndex = 0;

} else {

RadioGroup1->ItemIndex = 1;

}

}

} else {

ShowMessage("Вимоги для цього предмета та дати не знайдені.");

}

query->Close();

delete query;

}

void \_\_fastcall TForm9::ComboBox1CloseUp(TObject \*Sender) {

if (ComboBox1->ItemIndex == -1) {

ShowMessage("Будь ласка, виберіть предмет.");

return;

}

int subject\_id = ComboBox1->ItemIndex + 1;

TDateTime selectedDate = DatePicker2->Date;

String formattedDate = selectedDate.FormatString("yyyy-mm-dd");

TADOQuery \*query = new TADOQuery(NULL);

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = "SELECT Max\_point, Min\_r\_point, Min\_point FROM conditions WHERE Date = :date AND Subject\_id = :subject\_id";

query->Parameters->ParamByName("date")->Value = formattedDate;

query->Parameters->ParamByName("subject\_id")->Value = subject\_id;

query->Open();

if (!query->Eof) {

int maxPoint = query->FieldByName("Max\_point")->AsInteger;

int minPassPoint = query->FieldByName("Min\_r\_point")->AsInteger;

int minPoint = query->FieldByName("Min\_point")->AsInteger;

Label5->Caption = "Максимальний: " + IntToStr(maxPoint);

Label7->Caption = "Прохідний: " + IntToStr(minPassPoint);

Label6->Caption = "Мінімальний: " + IntToStr(minPoint);

} else {

ShowMessage("Для вибраної дати та предмета немає даних у таблиці.");

Label5->Caption = "Максимальний: -";

Label7->Caption = "Прохідний: -"; #include <vcl.h>

#pragma hdrstop

#include "Unit9.h"

#include <System.RegularExpressions.hpp>

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm9 \*Form9;

\_\_fastcall TForm9::TForm9(TComponent\* Owner)

: TForm(Owner)

{

}

void TForm9::set\_id(int k) {

id = k;

if (id > 0) {

Button1->Caption = "Замінити";

AnsiString queryStr = "SELECT r.Subj\_id, r.Condition\_id, r.Status, st.PIB, r.Reached\_score, r.Attemp\_date, "

"school.Email, c.Date, c.Max\_point, c.Min\_r\_point, c.Min\_point "

"FROM result r "

"JOIN student st ON r.Student\_id = st.Student\_id "

"JOIN school ON r.School\_id = school.School\_id "

"JOIN conditions c ON r.Condition\_id = c.Condition\_id "

"WHERE r.Res\_id = :id";

TADOQuery \*query = new TADOQuery(NULL);

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = queryStr;

query->Parameters->ParamByName("id")->Value = id;

query->Open();

if (!query->Eof) {

LabeledEdit1->Text = query->FieldByName("PIB")->AsString;

LabeledEdit4->Text = query->FieldByName("Reached\_score")->AsString;

int subject\_id = query->FieldByName("Subj\_id")->AsInteger;

ComboBox1->ItemIndex = subject\_id - 1;

DatePicker2->Date = query->FieldByName("Date")->AsDateTime;

String email = query->FieldByName("Email")->AsString;

ComboBox2->ItemIndex = ComboBox2->Items->IndexOf(email);

int status = query->FieldByName("Status")->AsInteger;

RadioGroup1->ItemIndex = (status == 1) ? 0 : 1;

int maxPoint = query->FieldByName("Max\_point")->AsInteger;

int minPassPoint = query->FieldByName("Min\_r\_point")->AsInteger;

int minPoint = query->FieldByName("Min\_point")->AsInteger;

Label5->Caption = "Максимальний: " + IntToStr(maxPoint);

Label7->Caption = "Прохідний: " + IntToStr(minPassPoint);

Label6->Caption = "Мінімальний: " + IntToStr(minPoint);

}

query->Close();

delete query;

Form9->Caption = "Редагування даних";

} else {

Button1->Caption = "Додати";

LabeledEdit1->Text = "";

LabeledEdit4->Text = "";

ComboBox1->ItemIndex = -1;

ComboBox2->ItemIndex = -1;

DatePicker2->Date = EncodeDate(1990, 1, 1);

RadioGroup1->ItemIndex = -1;

Label5->Caption = "Максимальний: -";

Label7->Caption = "Прохідний: -";

Label6->Caption = "Мінімальний: -";

Form9->Caption = "Додавання даних";

}

}

void \_\_fastcall TForm9::FormCreate(TObject \*Sender) {

if (id == 0) {

TDateTime checkDate = EncodeDate(1990, 1, 1);

DatePicker2->Date = checkDate;

}

RadioGroup1->Enabled = false;

ComboBox1->Clear();

ComboBox2->Clear();

TADOQuery \*query = new TADOQuery(this);

query->Connection = DataModule1->ADOConnection1;

try {

query->SQL->Text = "SELECT Name FROM subject ORDER BY Subject\_id";

query->Open();

while (!query->Eof) {

ComboBox1->Items->Add(query->FieldByName("Name")->AsString);

query->Next();

}

query->Close();

query->SQL->Text = "SELECT Email FROM school ORDER BY School\_id";

query->Open();

while (!query->Eof) {

ComboBox2->Items->Add(query->FieldByName("Email")->AsString);

query->Next();

}

} \_\_finally {

query->Close();

delete query;

}

}

void \_\_fastcall TForm9::LabeledEdit1Exit(TObject \*Sender) {

String pib = LabeledEdit1->Text;

UnicodeString pattern = "^[А-ЯІЇЄҐа-яіїєґ' ]+$";

if (!TRegEx::IsMatch(pib, pattern)) {

ShowMessage("ПІБ повинен містити лише українські літери.");

return;

}

TADOQuery \*query = new TADOQuery(NULL);

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = "SELECT COUNT(\*) AS count FROM student WHERE PIB = :pib";

query->Parameters->ParamByName("pib")->Value = pib;

query->Open();

int count = query->FieldByName("count")->AsInteger;

if (count == 0) {

ShowMessage("ПІБ не знайдений у базі даних.");

query->Close();

delete query;

return;

}

query->Close();

delete query;

}

void \_\_fastcall TForm9::DatePicker2CloseUp(TObject \*Sender) {

int subject\_id = ComboBox1->ItemIndex + 1;

TDateTime selectedDate = DatePicker2->Date;

String formattedDate = selectedDate.FormatString("yyyy-mm-dd");

TADOQuery \*query = new TADOQuery(NULL);

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = "SELECT Max\_point, Min\_r\_point, Min\_point FROM conditions WHERE Date = :date AND Subject\_id = :subject\_id";

query->Parameters->ParamByName("date")->Value = formattedDate;

query->Parameters->ParamByName("subject\_id")->Value = subject\_id;

query->Open();

if (!query->Eof) {

int maxPoint = query->FieldByName("Max\_point")->AsInteger;

int minPassPoint = query->FieldByName("Min\_r\_point")->AsInteger;

int minPoint = query->FieldByName("Min\_point")->AsInteger;

Label5->Caption = "Максимальний: " + IntToStr(maxPoint);

Label7->Caption = "Прохідний: " + IntToStr(minPassPoint);

Label6->Caption = "Мінімальний: " + IntToStr(minPoint);

} else {

ShowMessage("Для вибраної дати та предмета немає даних у таблиці.");

Label5->Caption = "Максимальний: -";

Label7->Caption = "Прохідний: -";

Label6->Caption = "Мінімальний: -";

}

query->Close();

delete query;

}

void \_\_fastcall TForm9::DatePicker1CloseUp(TObject \*Sender) {

try {

TDateTime selectedDate = DatePicker1->Date;

TDateTime currentDate = Now();

if (selectedDate > currentDate) {

throw Exception("Дата не може бути в майбутньому!");

} else if (selectedDate < EncodeDate(1990, 1, 1)) {

throw Exception("Дата занадто стара! Виберіть пізнішу дату.");

}

} catch (const Exception &e) {

ShowMessage(e.Message);

DatePicker1->Date = Now();

}

}

void \_\_fastcall TForm9::LabeledEdit4Exit(TObject \*Sender) {

TDateTime checkDate = EncodeDate(1990, 1, 1);

if (ComboBox1->ItemIndex == -1 || DatePicker2->Date == checkDate) {

ShowMessage("Будь ласка, виберіть предмет та дату укладання вимог.");

return;

}

int subject\_id = ComboBox1->ItemIndex + 1;

TDateTime conditionDate = DatePicker2->Date;

String enteredScoreStr = LabeledEdit4->Text;

int enteredScore = enteredScoreStr.ToIntDef(-1);

if (enteredScore == -1) {

ShowMessage("Будь ласка, введіть коректне значення балу.");

return;

}

String formattedDate = conditionDate.FormatString("yyyy-mm-dd");

TADOQuery \*query = new TADOQuery(NULL);

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = "SELECT Max\_point, Min\_r\_point, Min\_point FROM conditions WHERE Subject\_id = :subject\_id AND Date = :date";

query->Parameters->ParamByName("subject\_id")->Value = subject\_id;

query->Parameters->ParamByName("date")->Value = formattedDate;

query->Open();

if (!query->Eof) {

int maxPoint = query->FieldByName("Max\_point")->AsInteger;

int minPassPoint = query->FieldByName("Min\_r\_point")->AsInteger;

int minPoint = query->FieldByName("Min\_point")->AsInteger;

if (enteredScore < minPoint) {

ShowMessage("Введений бал менший за мінімальний.");

} else if (enteredScore > maxPoint) {

ShowMessage("Введений бал більший за максимальний.");

} else {

if (enteredScore >= minPassPoint) {

RadioGroup1->ItemIndex = 0;

} else {

RadioGroup1->ItemIndex = 1;

}

}

} else {

ShowMessage("Вимоги для цього предмета та дати не знайдені.");

}

query->Close();

delete query;

}

void \_\_fastcall TForm9::ComboBox1CloseUp(TObject \*Sender) {

if (ComboBox1->ItemIndex == -1) {

ShowMessage("Будь ласка, виберіть предмет.");

return;

}

int subject\_id = ComboBox1->ItemIndex + 1;

TDateTime selectedDate = DatePicker2->Date;

String formattedDate = selectedDate.FormatString("yyyy-mm-dd");

TADOQuery \*query = new TADOQuery(NULL);

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = "SELECT Max\_point, Min\_r\_point, Min\_point FROM conditions WHERE Date = :date AND Subject\_id = :subject\_id";

query->Parameters->ParamByName("date")->Value = formattedDate;

query->Parameters->ParamByName("subject\_id")->Value = subject\_id;

query->Open();

if (!query->Eof) {

int maxPoint = query->FieldByName("Max\_point")->AsInteger;

int minPassPoint = query->FieldByName("Min\_r\_point")->AsInteger;

int minPoint = query->FieldByName("Min\_point")->AsInteger;

Label5->Caption = "Максимальний: " + IntToStr(maxPoint);

Label7->Caption = "Прохідний: " + IntToStr(minPassPoint);

Label6->Caption = "Мінімальний: " + IntToStr(minPoint);

} else {

ShowMessage("Для вибраної дати та предмета немає даних у таблиці.");

Label5->Caption = "Максимальний: -";

Label7->Caption = "Прохідний: -";

Label6->Caption = "Мінімальний: -";

}

query->Close();

delete query;

}

void \_\_fastcall TForm9::Button1Click(TObject \*Sender)

{

if (LabeledEdit1->Text.Trim() == "") {

ShowMessage("Будь ласка, введіть ПІБ.");

return;

}

if (LabeledEdit4->Text.Trim() == "") {

ShowMessage("Будь ласка, введіть результат.");

return;

}

if (ComboBox1->ItemIndex == -1) {

ShowMessage("Будь ласка, виберіть навчальну дисципліну.");

return;

}

if (DatePicker2->Date == EncodeDate(1990, 1, 1)) {

ShowMessage("Будь ласка, виберіть дату укладання умов.");

return;

}

int subject\_id = ComboBox1->ItemIndex + 1;

TDateTime selectedDate = DatePicker2->Date;

String formattedDate = selectedDate.FormatString("yyyy-mm-dd");

int reachedScore = LabeledEdit4->Text.ToIntDef(-1);

int status = RadioGroup1->ItemIndex == 0 ? 1 : 0;

TADOQuery \*studentQuery = new TADOQuery(NULL);

studentQuery->Connection = DataModule1->ADOConnection1;

studentQuery->SQL->Text = "SELECT Student\_id FROM student WHERE PIB = :pib";

studentQuery->Parameters->ParamByName("pib")->Value = LabeledEdit1->Text;

studentQuery->Open();

int student\_id = -1;

if (!studentQuery->Eof) {

student\_id = studentQuery->FieldByName("Student\_id")->AsInteger;

} else {

ShowMessage("Студента з таким ПІБ не знайдено.");

studentQuery->Close();

delete studentQuery;

return;

}

studentQuery->Close();

delete studentQuery;

TADOQuery \*schoolQuery = new TADOQuery(NULL);

schoolQuery->Connection = DataModule1->ADOConnection1;

schoolQuery->SQL->Text = "SELECT School\_id FROM school WHERE Email = :email";

schoolQuery->Parameters->ParamByName("email")->Value = ComboBox2->Text;

schoolQuery->Open();

int school\_id = -1;

if (!schoolQuery->Eof) {

school\_id = schoolQuery->FieldByName("School\_id")->AsInteger;

} else {

ShowMessage("Школу з таким Email не знайдено.");

schoolQuery->Close();

delete schoolQuery;

return;

}

schoolQuery->Close();

delete schoolQuery;

TADOQuery \*conditionQuery = new TADOQuery(NULL);

conditionQuery->Connection = DataModule1->ADOConnection1;

conditionQuery->SQL->Text = "SELECT Condition\_id FROM conditions WHERE Subject\_id = :subject\_id AND Date = :date";

conditionQuery->Parameters->ParamByName("subject\_id")->Value = subject\_id;

conditionQuery->Parameters->ParamByName("date")->Value = formattedDate;

conditionQuery->Open();

int condition\_id = -1;

if (!conditionQuery->Eof) {

condition\_id = conditionQuery->FieldByName("Condition\_id")->AsInteger;

} else {

ShowMessage("Не знайдено умов для вибраного предмету та дати.");

conditionQuery->Close();

delete conditionQuery;

return;

}

conditionQuery->Close();

delete conditionQuery;

TADOQuery \*query = new TADOQuery(NULL);

query->Connection = DataModule1->ADOConnection1;

if (id > 0) {

query->SQL->Text = "UPDATE result SET Subj\_id = :subj\_id, Reached\_score = :reached\_score, Status = :status, Attemp\_date = :attemp\_date "

"WHERE Res\_id = :id";

query->Parameters->ParamByName("id")->Value = id;

} else {

query->SQL->Text = "INSERT INTO result (Subj\_id, Condition\_id, Student\_id, Reached\_score, Status, School\_id, Attemp\_date) "

"VALUES (:subj\_id, :condition\_id, :student\_id, :reached\_score, :status, :school\_id, :attemp\_date)";

query->Parameters->ParamByName("student\_id")->Value = student\_id;

query->Parameters->ParamByName("school\_id")->Value = school\_id;

query->Parameters->ParamByName("condition\_id")->Value = condition\_id;

}

query->Parameters->ParamByName("subj\_id")->Value = subject\_id;

query->Parameters->ParamByName("reached\_score")->Value = reachedScore;

query->Parameters->ParamByName("status")->Value = status;

query->Parameters->ParamByName("attemp\_date")->Value = DatePicker2->Date.FormatString("yyyy-mm-dd");

try {

query->ExecSQL();

ShowMessage("Запис успішно збережено.");

this->Close();

} catch (Exception &e) {

ShowMessage("Помилка збереження запису: " + e.Message);

}

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->Open();

DataModule1->MainQuery->Close();

DataModule1->MainQuery->Open();

query->Close();

delete query;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm9::LabeledEdit1SubLabelClick(TObject \*Sender)

{

Form14->isMinimalView = true;

Form14->ToggleView();

Form14->Update();

Form14->ShowModal();

Form14->isMinimalView = false;

Form14->ToggleView();

Form14->AutoSize = true;

Form14->Update();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm9::Label2Click(TObject \*Sender)

{

Form4->isMinimalView = true;

Form4->ToggleView();

Form4->Update();

Form4->ShowModal();

Form4->isMinimalView = false;

Form4->ToggleView();

Form4->AutoSize = true;

Form4->Update();

}

//-----------------------------------------------------------------------------------------

Лістинг В.31 – Текст файлу Unit10.h

//---------------------------------------------------------------------------

#ifndef Unit10H

#define Unit10H

//---------------------------------------------------------------------------

#include <System.Classes.hpp>

#include <Vcl.Controls.hpp>

#include <Vcl.StdCtrls.hpp>

#include <Vcl.Forms.hpp>

#include <Vcl.ExtCtrls.hpp>

#include <Vcl.Mask.hpp>

#include <Vcl.WinXPickers.hpp>

#include "Students.h"

#include "Data.h"

//---------------------------------------------------------------------------

class TForm10 : public TForm

{

\_\_published: // IDE-managed Components

TGroupBox \*GroupBox1;

TLabel \*Label2;

TDatePicker \*DatePicker1;

TButton \*Button1;

TRadioGroup \*RadioGroup1;

TLabeledEdit \*LabeledEdit1;

TLabeledEdit \*LabeledEdit2;

TLabel \*Label1;

TDatePicker \*DatePicker2;

void \_\_fastcall LabeledEdit1SubLabelClick(TObject \*Sender);

void \_\_fastcall LabeledEdit1Exit(TObject \*Sender);

void \_\_fastcall LabeledEdit2Exit(TObject \*Sender);

void \_\_fastcall DatePicker1CloseUp(TObject \*Sender);

void \_\_fastcall DatePicker2CloseUp(TObject \*Sender);

void \_\_fastcall FormCreate(TObject \*Sender);

void \_\_fastcall Button1Click(TObject \*Sender);

private:

int id; // User declarations

public: // User declarations

\_\_fastcall TForm10(TComponent\* Owner);

void set\_id(int k);

};

//---------------------------------------------------------------------------

extern PACKAGE TForm10 \*Form10;

//---------------------------------------------------------------------------

#endif

Лістинг В.32 – Текст файлу Unit10.cpp

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit10.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm10 \*Form10;

//---------------------------------------------------------------------------

\_\_fastcall TForm10::TForm10(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm10::LabeledEdit1SubLabelClick(TObject \*Sender)

{

Form14->isMinimalView = true;

Form14->ToggleView();

Form14->Update();

Form14->ShowModal();

Form14->isMinimalView = false;

Form14->ToggleView();

Form14->AutoSize = true;

Form14->Update();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm10::LabeledEdit1Exit(TObject \*Sender)

{

String pib = LabeledEdit1->Text.Trim();

UnicodeString pattern = "^[А-ЯІЇЄҐа-яіїєґ' ]+$";

if (!TRegEx::IsMatch(pib, pattern)) {

ShowMessage("ПІБ повинен містити лише українські літери.");

LabeledEdit1->SetFocus();

return;

}

if (pib != previousPIB) {

TADOQuery \*query = new TADOQuery(NULL);

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = "SELECT COUNT(\*) AS count FROM student WHERE PIB = :pib";

query->Parameters->ParamByName("pib")->Value = pib;

query->Open();

int count = query->FieldByName("count")->AsInteger;

if (count == 0) {

ShowMessage("ПІБ не знайдений у базі даних.");

LabeledEdit1->SetFocus();

query->Close();

delete query;

return;

}

previousPIB = pib;

query->Close();

delete query;

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm10::LabeledEdit2Exit(TObject \*Sender)

{

String enteredPIN = LabeledEdit2->Text.Trim();

if (enteredPIN.IsEmpty()) {

ShowMessage("Поле PIN не може бути порожнім!");

LabeledEdit2->SetFocus();

return;

}

if (enteredPIN != previousPIN) {

TADOQuery \*query = new TADOQuery(NULL);

try {

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = "SELECT COUNT(\*) AS Count FROM certificate WHERE PIN = :EnteredPIN";

query->Parameters->ParamByName("EnteredPIN")->Value = enteredPIN;

query->Open();

int count = query->FieldByName("Count")->AsInteger;

if (count > 0) {

ShowMessage("Цей PIN вже існує в базі даних! Введіть інший.");

LabeledEdit2->SetFocus();

}

}

\_\_finally {

delete query;

}

previousPIN = enteredPIN;

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm10::DatePicker2CloseUp(TObject \*Sender)

{

try

{

TDateTime creationDate = DatePicker1->Date;

TDateTime registrationDate = DatePicker2->Date;

if (registrationDate < EncodeDate(1990, 1, 1))

{

ShowMessage("Дата реєстрації повинна бути не ранішою за 01.01.1990!");

DatePicker2->SetFocus();

return;

}

if (registrationDate <= creationDate)

{

ShowMessage("Дата реєстрації повинна бути пізніше за дату створення!");

DatePicker2->SetFocus();

return;

}

}

catch (const Exception &e)

{

ShowMessage("Помилка: " + e.Message);

DatePicker2->SetFocus();

}

}

//---------------------------------------------------------------------------

void TForm10::set\_id(int k)

{

id = k;

if (id == 0)

{

LabeledEdit1->Text = "";

LabeledEdit2->Text = "";

RadioGroup1->ItemIndex = -1;

TDateTime defaultDate = EncodeDate(1990, 1, 1);

DatePicker1->Date = defaultDate;

DatePicker2->Date = defaultDate;

Form10->Caption = "Додавання даних";

}

else

{

TADOQuery \*query = new TADOQuery(NULL);

try

{

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = "SELECT c.PIN, c.Status, c.Creation\_date, c.Effect\_time, s.PIB "

"FROM certificate c "

"JOIN student s ON c.Student\_id = s.Student\_id "

"WHERE c.Cerf\_num = :id";

query->Parameters->ParamByName("id")->Value = id;

query->Open();

if (!query->Eof)

{

LabeledEdit1->Text = query->FieldByName("PIB")->AsString;

LabeledEdit2->Text = query->FieldByName("PIN")->AsString;

int status = query->FieldByName("Status")->AsInteger;

if (status == 1)

{

RadioGroup1->ItemIndex = 0;

}

else

{

RadioGroup1->ItemIndex = 1;

}

DatePicker2->Date = query->FieldByName("Effect\_time")->AsDateTime;

DatePicker1->Date = query->FieldByName("Creation\_date")->AsDateTime;

previousPIB = LabeledEdit1->Text;

previousPIN = LabeledEdit2->Text;

}

}

\_\_finally

{

delete query;

}

Form10->Caption = "Редагування даних";

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm10::DatePicker1CloseUp(TObject \*Sender)

{

try

{

TDateTime selectedDate = DatePicker1->Date;

TDateTime currentDate = Now();

if (selectedDate > currentDate)

{

ShowMessage("Дата не може бути в майбутньому!");

DatePicker1->SetFocus();

return;

}

else if (selectedDate < EncodeDate(1990, 1, 1))

{

ShowMessage("Дата занадто стара! Виберіть пізнішу дату.");

DatePicker1->SetFocus();

return;

}

}

catch (const Exception &e)

{

ShowMessage("Помилка: " + e.Message);

DatePicker1->SetFocus();

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm10::FormCreate(TObject \*Sender)

{

TDateTime date = EncodeDate(1990, 1, 1);

DatePicker1->Date = date;

DatePicker2->Date = date;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm10::Button1Click(TObject \*Sender)

{

try

{

String pib = LabeledEdit1->Text.Trim();

String pin = LabeledEdit2->Text.Trim();

TDateTime creationDate = DatePicker1->Date;

TDateTime effectTime = DatePicker2->Date;

int status = RadioGroup1->ItemIndex == 0 ? 1 : 0;

if (pib.IsEmpty())

{

ShowMessage("Поле ПІБ не може бути порожнім!");

LabeledEdit1->SetFocus();

return;

}

if (pin.IsEmpty())

{

ShowMessage("Поле PIN не може бути порожнім!");

LabeledEdit2->SetFocus();

return;

}

if (effectTime < creationDate)

{

ShowMessage("Дата реєстрації повинна бути пізніше за дату створення!");

DatePicker2->SetFocus();

return;

}

TADOQuery \*query = new TADOQuery(NULL);

try

{

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = "SELECT Student\_id FROM student WHERE PIB = :pib";

query->Parameters->ParamByName("pib")->Value = pib;

query->Open();

if (query->Eof)

{

ShowMessage("ПІБ не знайдений у базі даних.");

LabeledEdit1->SetFocus();

query->Close();

delete query;

return;

}

int student\_id = query->FieldByName("Student\_id")->AsInteger;

query->Close();

String creationDateStr = creationDate.FormatString("yyyy-mm-dd");

String effectTimeStr = effectTime.FormatString("yyyy-mm-dd");

if (id == 0)

{

query->SQL->Text = "INSERT INTO certificate (Student\_id, PIN, Creation\_date, Effect\_time, Status) "

"VALUES (:student\_id, :pin, :creationDate, :effectTime, :status)";

}

else

{

query->SQL->Text = "UPDATE certificate SET Student\_id = :student\_id, PIN = :pin, Creation\_date = :creationDate, "

"Effect\_time = :effectTime, Status = :status WHERE Cerf\_num = :id";

query->Parameters->ParamByName("id")->Value = id;

}

query->Parameters->ParamByName("student\_id")->Value = student\_id;

query->Parameters->ParamByName("pin")->Value = pin;

query->Parameters->ParamByName("creationDate")->Value = creationDateStr;

query->Parameters->ParamByName("effectTime")->Value = effectTimeStr;

query->Parameters->ParamByName("status")->Value = status;

query->ExecSQL();

ShowMessage("Операція успішно виконана!");

Form10->Close();

}

\_\_finally

{

delete query;

}

}

catch (const Exception &e)

{

ShowMessage("Помилка: " + e.Message);

}

DataModule1->ADOQuery4->Close();

DataModule1->ADOQuery4->Open();

]

--------------

Лістинг В.33 – Текст файлу Unit15.h

//---------------------------------------------------------------------------

#ifndef Unit15H

#define Unit15H

//---------------------------------------------------------------------------

#include <System.Classes.hpp>

#include <Vcl.Controls.hpp>

#include <Vcl.StdCtrls.hpp>

#include <Vcl.Forms.hpp>

#include <Vcl.ExtCtrls.hpp>

#include <Vcl.Mask.hpp>

#include <Data.DB.hpp>

#include <Vcl.DBGrids.hpp>

#include <Vcl.Grids.hpp>

#include <Vcl.Menus.hpp>

#include "Data.h"

#include "Unit16.h"

//---------------------------------------------------------------------------

class TForm15 : public TForm

{

\_\_published: // IDE-managed Components

TDBGrid \*DBGrid1;

TPopupMenu \*PopupMenu1;

TMenuItem \*N1;

TMenuItem \*N2;

void \_\_fastcall FormShow(TObject \*Sender);

void \_\_fastcall DBGrid1TitleClick(TColumn \*Column);

void \_\_fastcall N1Click(TObject \*Sender);

void \_\_fastcall N2Click(TObject \*Sender);

private:

bool SortAscending;

void DBColumnSizes(); // User declarations

public: // User declarations

\_\_fastcall TForm15(TComponent\* Owner);

};

//---------------------------------------------------------------------------

extern PACKAGE TForm15 \*Form15;

//---------------------------------------------------------------------------

#endif

Лістинг В.34 – Текст файлу Unit15.cpp

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit15.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm15 \*Form15;

//---------------------------------------------------------------------------

\_\_fastcall TForm15::TForm15(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void TForm15::DBColumnSizes(){

DBGrid1->Columns->Items[0]->Width = 85;

DBGrid1->Columns->Items[0]->Title->Caption = "Логін";

DBGrid1->Columns->Items[1]->Width = 100;

DBGrid1->Columns->Items[1]->Title->Caption = "Пароль";

DBGrid1->Columns->Items[2]->Width = 150;

DBGrid1->Columns->Items[2]->Title->Caption = "Роль";

DBGrid1->Columns->Items[3]->Width = 175;

DBGrid1->Columns->Items[3]->Title->Caption = "ПІБ";

DBGrid1->Columns->Items[4]->Width = 85;

DBGrid1->Columns->Items[4]->Title->Caption = "Статус";

}

void \_\_fastcall TForm15::FormShow(TObject \*Sender)

{

DBColumnSizes();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm15::DBGrid1TitleClick(TColumn \*Column)

{

String sortOrder = SortAscending ? " ASC" : " DESC";

DataModule1->ADOQuery5->Sort = Column->Field->FieldName + sortOrder;

SortAscending = !SortAscending;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm15::N1Click(TObject \*Sender)

{

Form16->set\_login("");

Form16->ShowModal();

DataModule1->ADOQuery5->Close();

DataModule1->ADOQuery5->Open();

DBColumnSizes();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm15::N2Click(TObject \*Sender)

{

DataModule1->DataSource1->DataSet->Refresh();

String login = DBGrid1->DataSource->DataSet->FieldByName("Login")->AsString;

TADOQuery \*query = new TADOQuery(this);

query->Connection = DataModule1->ADOConnection1;

query->SQL->Text = "SELECT Login FROM users WHERE Login = :Login";

query->Parameters->ParamByName("Login")->Value = login;

query->Open();

if (!query->Eof)

{

Form16->set\_login(login);

Form16->ShowModal();

DataModule1->ADOQuery5->Close();

DataModule1->ADOQuery5->Open();

DBColumnSizes();

}

else

{

ShowMessage("Користувача з таким логіном не знайдено.");

}

delete query;

}

//---------------------------------------------------------------------------

Лістинг В.35 – Текст файлу Unit16.h

//---------------------------------------------------------------------------

#ifndef Unit16H

#define Unit16H

//---------------------------------------------------------------------------

#include <System.Classes.hpp>

#include <Vcl.Controls.hpp>

#include <Vcl.StdCtrls.hpp>

#include <Vcl.Forms.hpp>

#include <Vcl.ExtCtrls.hpp>

#include <Vcl.Mask.hpp>

#include "Data.h"

//---------------------------------------------------------------------------

class TForm16 : public TForm

{

\_\_published: // IDE-managed Components

TLabeledEdit \*LabeledEdit1;

TLabeledEdit \*LabeledEdit2;

TLabeledEdit \*LabeledEdit3;

TButton \*Button1;

TLabel \*Label1;

TComboBox \*ComboBox1;

TRadioGroup \*RadioGroup1;

void \_\_fastcall LabeledEdit3Exit(TObject \*Sender);

void \_\_fastcall LabeledEdit1Exit(TObject \*Sender);

void \_\_fastcall LabeledEdit2Exit(TObject \*Sender);

void \_\_fastcall FormShow(TObject \*Sender);

void \_\_fastcall Button1Click(TObject \*Sender);

private:

String originalLogin;

bool isEditMode;

// User declarations

public: // User declarations

\_\_fastcall TForm16(TComponent\* Owner);

void \_\_fastcall set\_login(String login);

};

//---------------------------------------------------------------------------

extern PACKAGE TForm16 \*Form16;

//---------------------------------------------------------------------------

#endif

Лістинг В.36 – Текст файлу Unit16.cpp

#include <vcl.h>

#include <System.RegularExpressions.hpp>

#pragma hdrstop

#include "Unit16.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm16 \*Form16;

\_\_fastcall TForm16::TForm16(TComponent\* Owner)

: TForm(Owner)

{

}

void \_\_fastcall TForm16::set\_login(String login)

{

originalLogin = login;

if (login.IsEmpty())

{

LabeledEdit1->Text = "";

LabeledEdit2->Text = "";

LabeledEdit3->Text = "";

ComboBox1->ItemIndex = -1;

RadioGroup1->ItemIndex = -1;

isEditMode =false;

}

else

{

String query = "SELECT Login, Password, Role, PIB, Status FROM users WHERE Login = :Login";

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Clear();

DataModule1->ADOQuery1->SQL->Add(query);

DataModule1->ADOQuery1->Parameters->ParamByName("Login")->Value = login;

DataModule1->ADOQuery1->Open();

if (DataModule1->ADOQuery1->RecordCount > 0)

{

LabeledEdit1->Text = DataModule1->ADOQuery1->FieldByName("Login")->AsString;

LabeledEdit2->Text = DataModule1->ADOQuery1->FieldByName("Password")->AsString;

String role = DataModule1->ADOQuery1->FieldByName("Role")->AsString;

for (int i = 0; i < ComboBox1->Items->Count; i++)

{

if (ComboBox1->Items->Strings[i] == role)

{

ComboBox1->ItemIndex = i;

break;

}

}

LabeledEdit3->Text = DataModule1->ADOQuery1->FieldByName("PIB")->AsString;

int status = DataModule1->ADOQuery1->FieldByName("Status")->AsInteger;

RadioGroup1->ItemIndex = (status == 1) ? 0 : 1;

isEditMode = true;

}

else

{

LabeledEdit1->Text = "";

LabeledEdit2->Text = "";

LabeledEdit3->Text = "";

ComboBox1->ItemIndex = -1;

RadioGroup1->ItemIndex = -1;

}

DataModule1->ADOQuery1->Close();

}

}

void \_\_fastcall TForm16::LabeledEdit3Exit(TObject \*Sender)

{

String pib = LabeledEdit3->Text.Trim();

if (TRegEx::IsMatch(pib, L"[A-Za-z0-9ЁЫЭёыэ!\"#$%&()\*,./:;<=>?@[\\]^\_{|}~]")) {

ShowMessage("Будь ласка, вводьте ПІБ українською мовою!");

return;

LabeledEdit3->SetFocus();

}

}

void \_\_fastcall TForm16::LabeledEdit1Exit(TObject \*Sender)

{

String login = LabeledEdit1->Text.Trim();

UnicodeString pattern = L"^[A-Za-z0-9]+$";

if (!TRegEx::IsMatch(login, pattern))

{

ShowMessage("Логін повинен містити тільки англійські букви та цифри.");

LabeledEdit1->SetFocus();

return;

}

if (login == originalLogin)

{

return;

}

String query = "SELECT COUNT(\*) AS UserCount FROM users WHERE Login = :Login";

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Clear();

DataModule1->ADOQuery1->SQL->Add(query);

DataModule1->ADOQuery1->Parameters->ParamByName("Login")->Value = login;

DataModule1->ADOQuery1->Open();

if (DataModule1->ADOQuery1->FieldByName("UserCount")->AsInteger > 0)

{

ShowMessage("Цей логін вже існує. Будь ласка, виберіть інший.");

LabeledEdit1->SetFocus();

}

DataModule1->ADOQuery1->Close();

}

void \_\_fastcall TForm16::LabeledEdit2Exit(TObject \*Sender)

{

String password = LabeledEdit2->Text.Trim();

UnicodeString pattern = L"^[A-Za-z0-9]+$";

if (!TRegEx::IsMatch(password, pattern))

{

ShowMessage("Пароль повинен містити тільки англійські букви та цифри.");

LabeledEdit2->SetFocus();

}

}

void \_\_fastcall TForm16::FormShow(TObject \*Sender)

{

RadioGroup1->SetFocus();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm16::Button1Click(TObject \*Sender)

{

String login = LabeledEdit1->Text.Trim();

String password = LabeledEdit2->Text.Trim();

String pib = LabeledEdit3->Text.Trim();

String role = ComboBox1->Text;

int status = (RadioGroup1->ItemIndex == 0) ? 1 : 0;

if (login.IsEmpty() || password.IsEmpty() || pib.IsEmpty() || role.IsEmpty())

{

ShowMessage("Будь ласка, заповніть всі обов'язкові поля.");

return;

}

try

{

if (isEditMode)

{

String query = "UPDATE users SET Login = :Login, Password = :Password, Role = :Role, PIB = :PIB, Status = :Status WHERE Login = :OriginalLogin";

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Clear();

DataModule1->ADOQuery1->SQL->Add(query);

DataModule1->ADOQuery1->Parameters->ParamByName("Login")->Value = login;

DataModule1->ADOQuery1->Parameters->ParamByName("Password")->Value = password;

DataModule1->ADOQuery1->Parameters->ParamByName("Role")->Value = role;

DataModule1->ADOQuery1->Parameters->ParamByName("PIB")->Value = pib;

DataModule1->ADOQuery1->Parameters->ParamByName("Status")->Value = status;

DataModule1->ADOQuery1->Parameters->ParamByName("OriginalLogin")->Value = originalLogin;

DataModule1->ADOQuery1->ExecSQL();

ShowMessage("Користувач успішно оновлений.");

}

else

{

String query = "INSERT INTO users (Login, Password, Role, PIB, Status) VALUES (:Login, :Password, :Role, :PIB, :Status)";

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Clear();

DataModule1->ADOQuery1->SQL->Add(query);

DataModule1->ADOQuery1->Parameters->ParamByName("Login")->Value = login;

DataModule1->ADOQuery1->Parameters->ParamByName("Password")->Value = password;

DataModule1->ADOQuery1->Parameters->ParamByName("Role")->Value = role;

DataModule1->ADOQuery1->Parameters->ParamByName("PIB")->Value = pib;

DataModule1->ADOQuery1->Parameters->ParamByName("Status")->Value = status;

DataModule1->ADOQuery1->ExecSQL();

ShowMessage("Користувач успішно доданий.");

}

this->Close();

}

catch (const Exception &e)

{

ShowMessage("Помилка при збереженні користувача: " + e.Message);

}

}

//---------------------------------------------------------------------------

Додаток Г

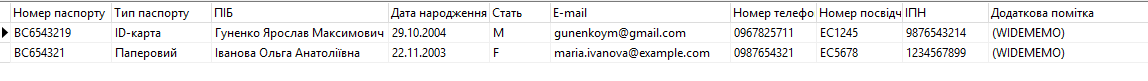


Рисунок Г.1 – Початкові дані таблиці “Student”

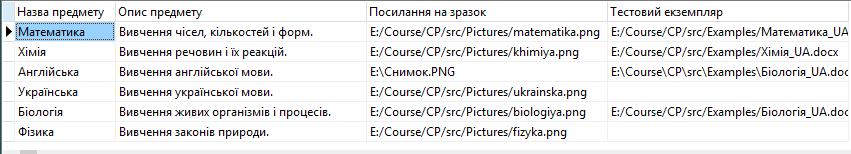


Рисунок Г.2 – Початкові дані таблиці “Subject”

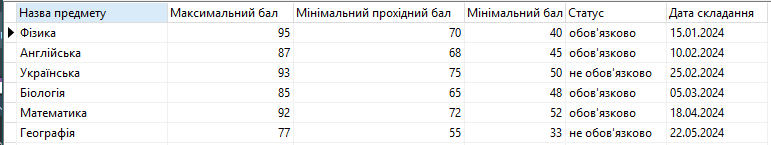


Рисунок Г.3 – Початкові дані таблиці “Conditions”

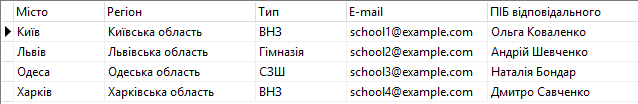


Рисунок Г.4 – Початкові дані таблиці “School”

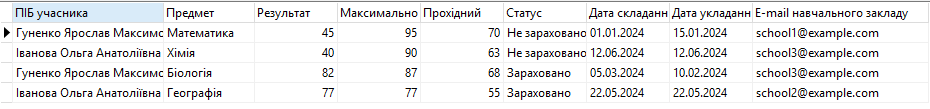


Рисунок Г.5 – Початкові дані таблиці “Result”



Рисунок Г.6 – Початкові дані таблиці “Certificate”

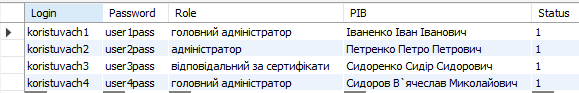


Рисунок Г.7 – Початкові дані таблиці “Users”