МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

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 НМТ»

(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)

Керівник   
викладач Гапоненко Н.В.

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

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

Члени комісії \_\_\_\_\_\_\_\_\_ Любохинець В.М.

(підпис) (прізвище та ініціали) .

\_\_\_\_\_\_\_\_\_ Ланська С.С. .

(підпис) (прізвище та ініціали) .

\_\_\_\_\_\_\_\_\_ Гапоненко Н.В. .

(підпис) (прізвище та ініціали) .

м. Дніпро - 2024 рік

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ФАХОВИЙ КОЛЕДЖ РАКЕТНО-КОСМІЧНОГО МАШИНОБУДУВАННЯ

ДНІПРОВСЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ ім. О. ГОНЧАРА

Цикловакомісія програмної інженерії

|  |  |
| --- | --- |
|  | **ЗАТВЕРДЖУЮ**  Голова комісії ПІ  \_\_\_\_\_\_\_\_\_\_\_\_ С.С.Ланська  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 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 НМТ» | | | | | |
|  | | | | | | | | | | |
|  | | | | | | | | | | |
| 2 Початкові дані | | | | | | Перелік даних про учня (ID учня, Номер паспорту, Тип , | | | | |
| Паспорту,ПІБ,Дата народження, Стать, E-mai, Номер телефону, Номер | | | | | | | | | | |
| свідоства про освіту,ІПН,Примітка), перелік навчальних закладів(місто, | | | | | | | | | | |
| область,тип, E-mail, ПІБ відповідального, Шифр навчального закладу) | | | | | | | | | | |
| перелік предметів(Код предмету, Назва предмету, Опис, Зразок завдань) | | | | | | | | | | |
| термін дії, статус) перелік сертифікатів(Номер сертифікату, ID учня, PIN, | | | | | | | | | | |
| Дата створення, Статус) перелік результатів тесту(Шифр результату, | | | | | | | | | | |
| ID умови, Отриманий бал, Статус здачі, Дата проходження, Шифр Статус | | | | | | | | | | |
| навчального закладу) Умови проходження(ID умови, Код предмету, | | | | | | | | | | |
| Максимальний бал, Мінімальний бал, Мінімальний бал для проходження, | | | | | | | | | | |
| предмету, Дата ухвалення умов) | | | | | | | | | | |
| Розглянуто і ухвалено на засіданні циклової комісії | | | | | | | | | програмної інженерії | |
| Протокол № 2 від 21.09.2021 р. | | | | | | | | | | |

Керівник КП \_\_\_\_\_\_\_\_\_\_\_\_\_ Н.В.Гапоненко

(підпис) (ініціали та прізвище)

Завдання до виконання

одержав студент \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (ініціали та прізвище)

Дата видачі 21 вересня 2021 р.

Термін виконання 22 листопада 2021 р.

ЗМІСТ

[ВСТУП 4](#_Toc180696039)

[1 ПОСТАНОВКА ЗАДАЧІ 5](#_Toc180696040)

[2 ХАРАКТЕРИСТИКА ПРОГРАМНИХ ЗАСОБІВ РЕАЛІЗАЦІЇ ПРОЕКТУ 7](#_Toc180696041)

[2.1 Опис середовища програмування 7](#_Toc180696042)

[2.2 Опис мови програмування 8](#_Toc180696043)

[2.3 Опис СКБД 9](#_Toc180696044)

[2.4 Опис основних принципів ООП 10](#_Toc180696045)

[2.5 Опис подібних програмних продуктів 13](#_Toc180696046)

[3 ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ ПРОГРАМНОГО ПРОДУКТУ 14](#_Toc180696047)

[3.1 Опис бази даних 14](#_Toc180696048)

[3.2 Проектування користувацького інтерфейсу програми 16](#_Toc180696049)

[3.3 Контроль вхідних даних програми 32](#_Toc180696050)

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.
* 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.

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 |
| 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. Як зазначає Б'ярн Страуструп, творець C++, "ООП дозволяє розробникам створювати більш складні та надійні системи, ніж це було б можливо за допомогою процедурного програмування".

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. Як зазначається в стандарті C++ (ISO/IEC 14882:2011), "шаблони є фундаментальною частиною мови C++ і дозволяють розробникам писати більш загальний та ефективний код".

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. Як зазначає Александре Степанов, один із розробників STL, "STL – це інструмент для створення ефективних та елегантних програм".

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. 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.

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 3.2-3.9.

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 20 символів |  |

Таблиця 3.4 – Поля таблиці «Результат» (Result)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Назва поля** | **Опис** | **Тип** | **Розмір** | **Ключ** |
| Res\_id | Унікальний код результату | Числовий | Макс. значення – 2147483647 | 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 | Логічний |  |  |
| 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 | Макс. значення – 2147483647 | 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 |  |
| Status | Статус сертифіката | Логічний |  |  |

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Назва поля** | **Опис** | **Тип** | **Розмір** | **Ключ** |
| Condition\_id | Унікальний код умови | Числовий | Макс. значення – 2147483647 | 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 | Логічний |  |  |
| 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 | Макс. значення – 2147483647 | 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 | Макс. значення – 2147483647 | Первинний |
| Name | Назва предмета | Символьний | До 100 символів |  |
| Description | Опис предмета | Текстовий |  |  |
| Image\_name | Шлях до прикладу завдань | Символьний | До 255 символів |  |

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

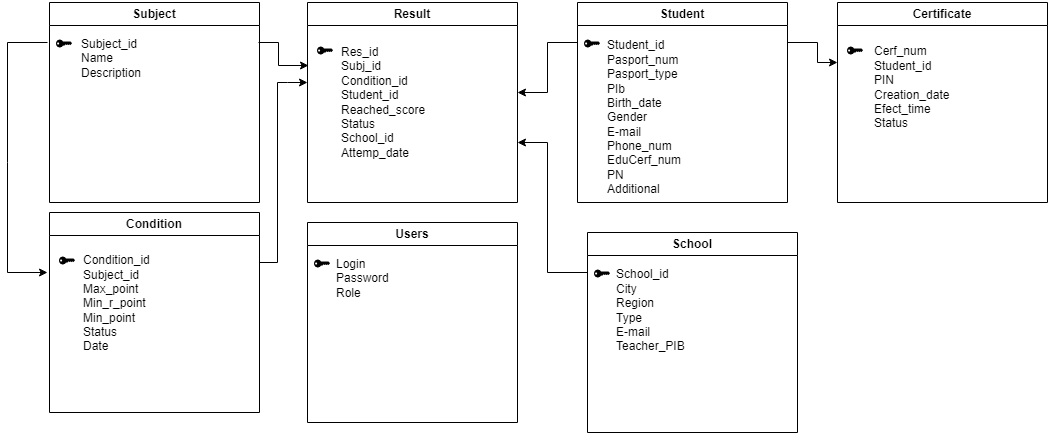


Рисунок 3.1 – ER діагарма бази даних

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).

Одразу після запуску додатку користувач побачить головне вікно.

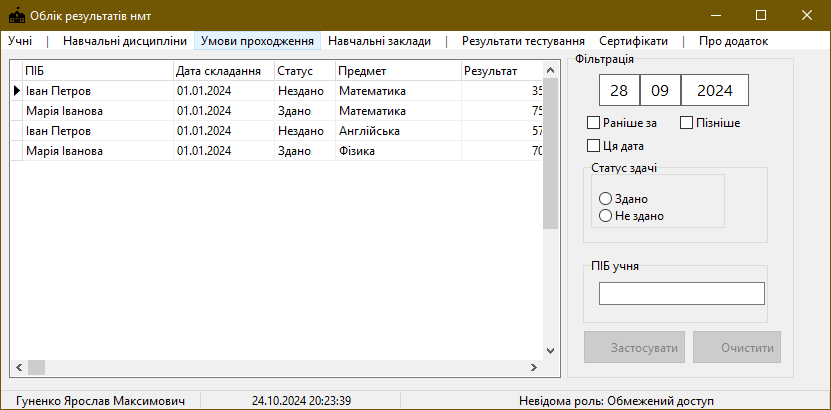


Рисунок 3.2 –Головне вікно

У цьому вікні користувач за замовчуванням може лише переглядати вміст таблиць для роботи з даними необхідно авторизуватись за допомогою відповідної форми, що відкривається за кліком по статусбару. Кожан роль розблоковує свої меню для форм, зокрема koristuvach1 – навчальні дисципліни, умови проходження, навчальні дисципліни, koristuvach2 – результати тестування, сертифікати, koristuvach3 – учні.

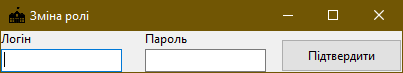


Рисунок 3.3 –Вікно авторизації

На головному вікні користувач бачить основні відомості з результатів тестування у стислому вигяді з можливістю фільтрації, зокрема користувач може фільтрувати інформацію у таблиці за ПІБ особи, що складала НМТ, що є зручним для перегляду інформації про конкретну особу. Також користувач може виконати фльтрацію за статусом здачі та датою складання, при тому фільтри можуть комбінуватися. Для роботи з конкретним типом даних реалізовано перехід між вікнами за допомогою головного меню з відповіними пунктами. Для сортування реалізовано метод для сортування будь-якого стовпця за кліком по назві стовця.

Для роботи з інформацією про осіб які складали НМТ користувачу необхідно виконати клік по пункту меню “Учні”. Одразу після цього користувч побачить відповідне вікно.

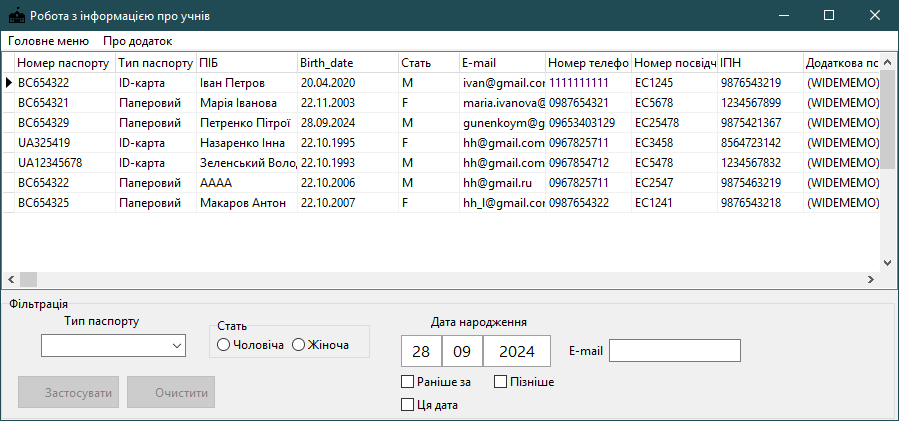


Рисунок 3.4 – Робота з інформацією про учнів

У створеному вікні користувач бачить таблицю з інформацією про особисті дані осіб, що складали НМТ з можливістю фільтрації за типом паспорту, що відповідно обирається у випада.чому списку, для уникнення можливих помилок введення користувачем. Додатково користувач може фільтурвати за статтю за допомогою перемикачів, датою народження з вибором дати у TDatePicker для уникнення помилок і вибором проміжку та за E-mail з перевіркою введення за допомогою регулярних виразів. Для сортування реалізовано метод для сортування будь-якого стовпця за кліком по назві стовця.

Якщо користувач хоче повернутися у головне меню реалізовано пункт меню “Головне меню”. Для роботи з додаванням та редагуванням даних реалізовано контекстне меню прив`язане до DBGrid.

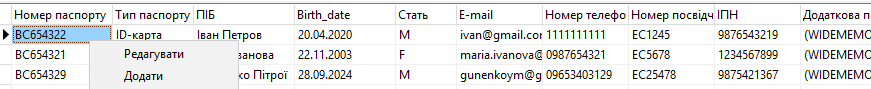


Рисунок 3.5 – Контекстне меню для роботи з записом

Відповідно для додавання даних у таблицю студенти у таблицю користувач обирає пункт меню є “Додати”.

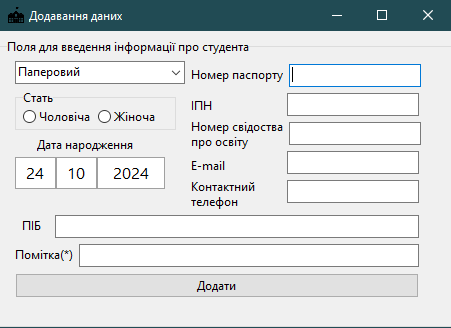


Рисунок 3.6 – Вікно для додавання записів у таблицю Студенти

Після вибору пункту меню користувач бачить форму з полями для вибору або введення інформації, що потребує таблиця. Для кожного поля реалізовані перевірки відповідно логічного змісту та правил збергіання у базі даних. У разі відстуності помилок та заповнення полів користувач може або натиснути додати, якщо потрібно підтвердити додавання, а якщо дії потрібно відмінити користувачу достатньо просто закрити вікно.

Для редагування даних у таблиці студенти користувач обирає пункт меню “Редагувати”.

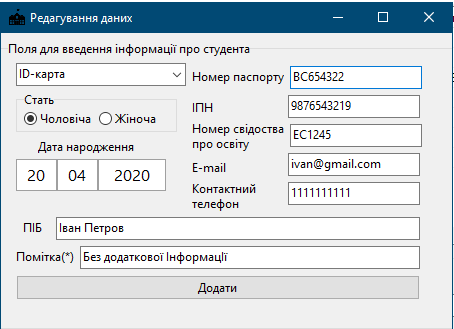


Рисунок 3.7 – Вікно для редагування записів у таблиці Студенти

Після вибору пункту меню користувач бачить форму з попередньо заповненими полями згідно значення полів таблиці для вибору або введення інформації, що потребує таблиця. Для кожного поля реалізовані перевірки відповідно логічного змісту та правил збергіання у базі даних. У разі відстуності помилок та заповнення полів користувач може або натиснути “ Додати ”, якщо потрібно підтвердити редагування, а якщо дії потрібно відмінити користувачу достатньо просто закрити вікно.

Для роботи з інформацією про навчальні дисипліни користувач має обрати відповідний пункт головного меню, а саме “Навчальні дисципліни”.

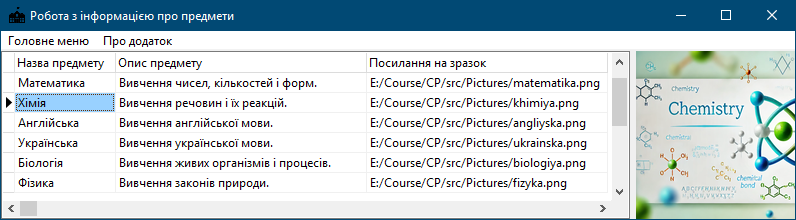


Рисунок 3.8 – Вікно Навчальні дисципліни

У створеному вікні користувач бачить таблицю з інформацією про навальні дисципліни для НМТ. Для сортування реалізовано метод для сортування будь-якого стовпця за кліком по назві стовця.

Якщо користувач хоче повернутися у головне меню реалізовано пункт меню “Головне меню”. Для роботи з додаванням та редагуванням даних реалізовано контекстне меню прив`язане до DBGrid.

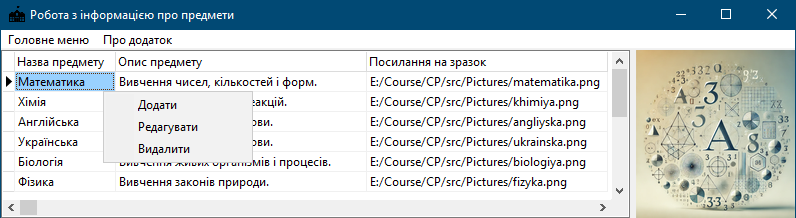


Рисунок 3.9 – Контекстне меню для роботи з вмістом таблиці навчальні дисципліни

Відповідно для додавання даних у таблицю Навчальні дисципліни користувач обирає пункт меню “Додати”.

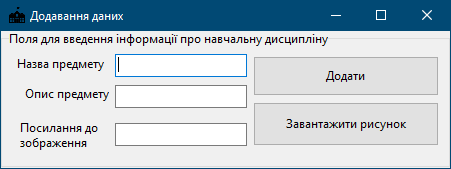


Рисунок 3.10 – Вікно для додавання записів у таблицю Навчальні дисципліни

Після вибору пункту меню користувач бачить форму з полями для вибору або введення інформації, що потребує таблиця. Для кожного поля реалізовані перевірки відповідно логічного змісту та правил збергіання у базі даних. Для вибору зображення для зразка завдань з предмету реалізовано і текстове поле для введення посилання на рисунок вручну і кнопку для завантаження рисунку через файловий менеджер. У разі відстуності помилок та заповнення полів користувач може або натиснути додати, якщо потрібно підтвердити додавання, а якщо дії потрібно відмінити користувачу достатньо просто закрити вікно.

Для редагування даних у таблиці навчальні дисципліни користувач обирає пункт меню “Редагувати”.

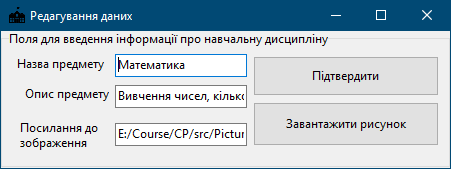


Рисунок 3.11 – Вікно для редагування записів у таблиці Навчальні дисципліни

Після вибору пункту меню користувач бачить форму з попередньо заповненими полями згідно значення полів таблиці для вибору або введення інформації, що потребує таблиця. Для кожного поля реалізовані перевірки відповідно логічного змісту та правил збергіання у базі даних. У разі відстуності помилок та заповнення полів користувач може або натиснути “ Підтвердити ”, якщо потрібно підтвердити редагування, а якщо дії потрібно відмінити користувачу достатньо просто закрити вікно.

Для видалення запису з таблиці користувач відповідно обирає запис у тіблиці і за допомогою контекстного меню обирає пункт меню “Видалити”. Після перевірки на використання у інших таблицях користувач бачить вікно підтвердження видалення.

Для роботи з інформацією про умовами проходження користувачу необхідно виконати клік по пункту меню “Умови проходження”. Одразу після цього користувч побачить відповідне вікно.

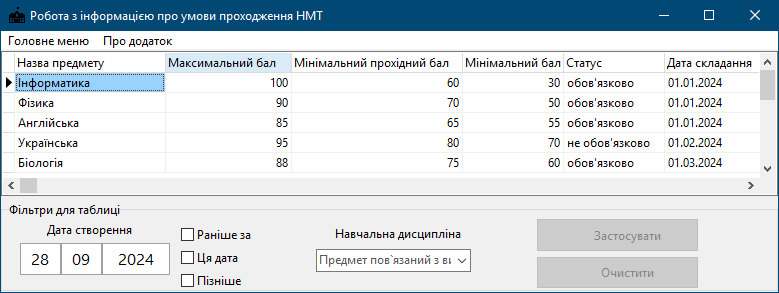


Рисунок 3.12 – Робота з інформацією про умови проходження

У створеному вікні користувач бачить таблицю з інформацією про умови проходження НМТ з можливістю фільтрації за навчальною дисципліною, що відповідно обирається у випадаючому списку, для уникнення можливих помилок введення користувачем. Додатково користувач може фільтрувати датою створення з вибором дати у TDatePicker для уникнення помилок. Для сортування реалізовано метод для сортування будь-якого стовпця за кліком по назві стовця.

Якщо користувач хоче повернутися у головне меню реалізовано пункт меню “Головне меню”. Для роботи з додаванням та редагуванням даних реалізовано контекстне меню прив`язане до DBGrid.

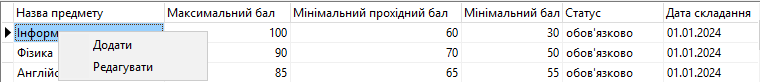


Рисунок 3.13 – Контекстне меню для роботи з записом

Відповідно для додавання даних у таблицю Умови проходження НМТ користувач обирає пункт меню “Додати”.

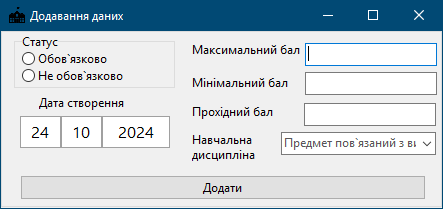


Рисунок 3.14 – Вікно для додавання записів у таблицю Умови проходження НМТ

Після вибору пункту меню користувач бачить форму з полями для вибору або введення інформації, що потребує таблиця. Для кожного поля реалізовані перевірки відповідно логічного змісту та правил збергіання у базі даних. У разі відстуності помилок та заповнення полів користувач може або натиснути додати, якщо потрібно підтвердити додавання, а якщо дії потрібно відмінити користувачу достатньо просто закрити вікно.

Для редагування даних у таблиці Умови проходження НМТ користувач обирає пункт меню “Редагувати”.

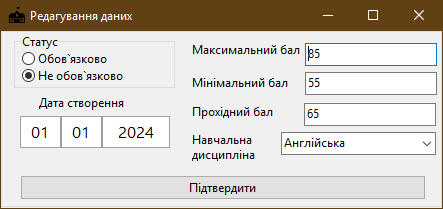


Рисунок 3.15 – Вікно для редагування записів у таблиці Умови проходження НМТ

Після вибору пункту меню користувач бачить форму з попередньо заповненими полями згідно значення полів таблиці для вибору або введення інформації, що потребує таблиця. Для кожного поля реалізовані перевірки відповідно логічного змісту та правил збергіання у базі даних. У разі відстуності помилок та заповнення полів користувач може або натиснути “ Підтвердити ”, якщо потрібно підтвердити редагування, а якщо дії потрібно відмінити користувачу достатньо просто закрити вікно.

Для роботи з інформацією про умовами проходження користувачу необхідно виконати клік по пункту меню “Навчальні заклади”. Одразу після цього користувач побачить відповідне вікно.

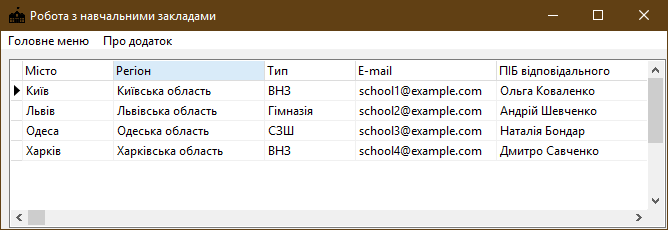


Рисунок 3.16 – Робота з інформацією про Навчальні заклади

У створеному вікні користувач бачить таблицю з інформацією про Навчальні заклади. Для сортування реалізовано метод для сортування будь-якого стовпця за кліком по назві стовця.

Якщо користувач хоче повернутися у головне меню реалізовано пункт меню “Головне меню”. Для роботи з додаванням та редагуванням даних реалізовано контекстне меню прив`язане до DBGrid.

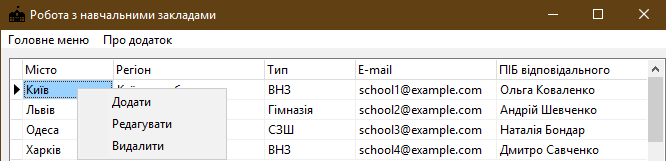


Рисунок 3.17 – Контекстне меню для роботи з записом

Відповідно для додавання даних у таблицю Навчальні заклади користувач обирає пункт меню “Додати”.

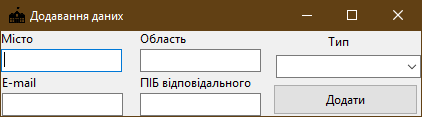


Рисунок 3.18 – Вікно для додавання записів у таблицю Навчальні заклади

Після вибору пункту меню користувач бачить форму з полями для вибору або введення інформації, що потребує таблиця. Для кожного поля реалізовані перевірки відповідно логічного змісту та правил збергіання у базі даних. У разі відстуності помилок та заповнення полів користувач може або натиснути додати, якщо потрібно підтвердити додавання, а якщо дії потрібно відмінити користувачу достатньо просто закрити вікно.

Для редагування даних у таблиці Навчальні заклади користувач обирає пункт меню “Редагувати”.

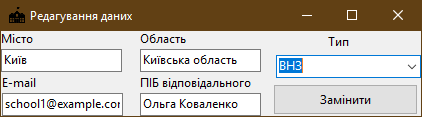


Рисунок 3.19 – Вікно для редагування записів у таблиці Навчальні заклади

Після вибору пункту меню користувач бачить форму з попередньо заповненими полями згідно значення полів таблиці для вибору або введення інформації, що потребує таблиця. Для кожного поля реалізовані перевірки відповідно логічного змісту та правил збергіання у базі даних. У разі відстуності помилок та заповнення полів користувач може або натиснути “ Замінити ”, якщо потрібно підтвердити редагування, а якщо дії потрібно відмінити користувачу достатньо просто закрити вікно.

Для видалення запису з таблиці користувач відповідно обирає запис у тіблиці і за допомогою контекстного меню обирає пункт меню “Видалити”. Після перевірки на використання у інших таблицях користувач бачить вікно підтвердження видалення.

Для роботи з інформацією про результати тестування користувачу необхідно виконати клік по пункту меню “Результати”. Одразу після цього користувач побачить відповідне вікно.

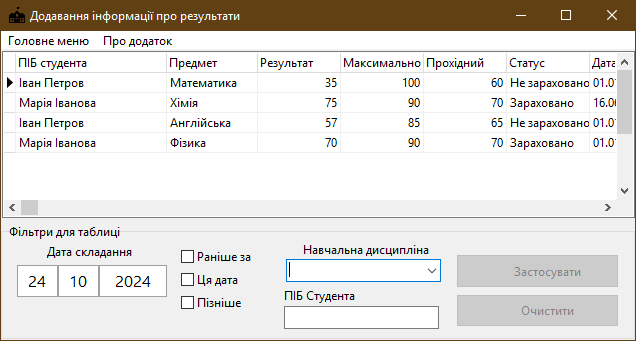


Рисунок 3.20 – Робота з інформацією про результати тестувань

У створеному вікні користувач бачить таблицю з інформацією про результати тестування з можливістю фільтрації за навчалною дисципліною, що відповідно обирається у випадаючому списку, для уникнення можливих помилок введення користувачем. Додатково користувач може фільтурвати за датою складання з вибором дати у TDatePicker для уникнення помилок і вибором проміжку та за ПІБ студента з перевіркою введення за допомогою регулярних виразів. Для сортування реалізовано метод для сортування будь-якого стовпця за кліком по назві стовця.

Якщо користувач хоче повернутися у головне меню реалізовано пункт меню “Головне меню”. Для роботи з додаванням та редагуванням даних реалізовано контекстне меню прив`язане до DBGrid.

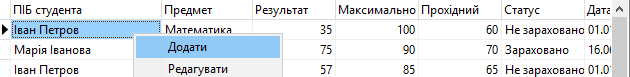


Рисунок 3.21 – Контекстне меню для роботи з записом

Відповідно для додавання даних у таблицю студенти у таблицю користувач обирає пункт меню є “Додати”.

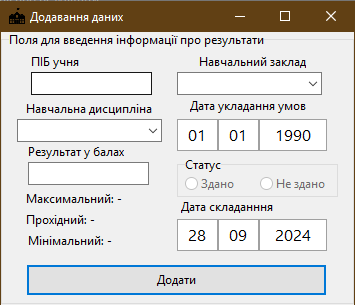


Рисунок 3.22 – Вікно для додавання записів у таблицю Результати

Після вибору пункту меню користувач бачить форму з полями для вибору або введення інформації, що потребує таблиця. Для кожного поля реалізовані перевірки відповідно логічного змісту та правил збергіання у базі даних. Після введення дати укладання умов та предмету користувач побачить ліміти балів. У разі відстуності помилок та заповнення полів користувач може або натиснути додати, якщо потрібно підтвердити додавання, а якщо дії потрібно відмінити користувачу достатньо просто закрити вікно.

Для редагування даних у таблиці результати тестування користувач обирає пункт меню “Редагувати”.

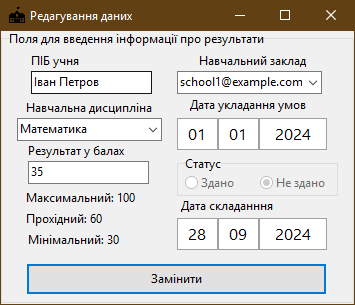


Рисунок 3.23 – Вікно для редагування записів у таблиці Результати тестування

Після вибору пункту меню користувач бачить форму з попередньо заповненими полями згідно значення полів таблиці для вибору або введення інформації, що потребує таблиця. Для кожного поля реалізовані перевірки відповідно логічного змісту та правил збергіання у базі даних. У разі відстуності помилок та заповнення полів користувач може або натиснути “ Замінити ”, якщо потрібно підтвердити редагування, а якщо дії потрібно відмінити користувачу достатньо просто закрити вікно.

Для роботи з інформацією про сертифікату НМТ користувачу необхідно виконати клік по пункту меню “Сертифікати”. Одразу після цього користувач побачить відповідне вікно.

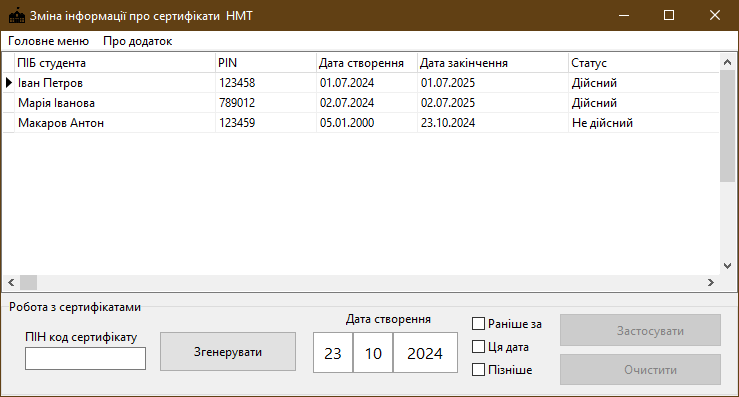


Рисунок 3.24 – Робота з інформацією про результати тестувань

У створеному вікні користувач бачить таблицю з інформацією про сертифікати НМТ з можливістю фільтрації за датою створення з вибором дати у TDatePicker для уникнення помилок і вибором проміжку. Для сортування реалізовано метод для сортування будь-якого стовпця за кліком по назві стовця. Для генерації сертифікату користувач має ввести ПІН-код сертифікату та після перевірки додатком натиснути кнопку “Згенерувати”.

Якщо користувач хоче повернутися у головне меню реалізовано пункт меню “Головне меню”. Для роботи з додаванням та редагуванням даних реалізовано контекстне меню прив`язане до DBGrid.

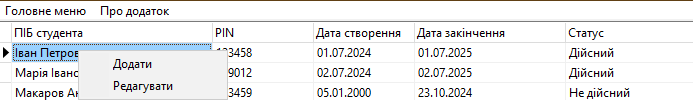


Рисунок 3.25 – Контекстне меню для роботи з записом

Відповідно для додавання даних у таблицю студенти у таблицю користувач обирає пункт меню є “Додати”.

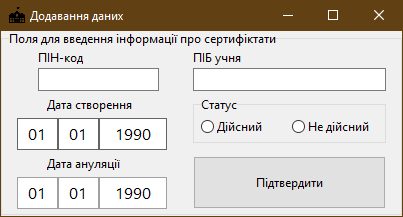


Рисунок 3.26 – Вікно для додавання записів у таблицю Сертифікати

Після вибору пункту меню користувач бачить форму з полями для вибору або введення інформації, що потребує таблиця. Для кожного поля реалізовані перевірки відповідно логічного змісту та правил збергіання у базі даних. У разі відстуності помилок та заповнення полів користувач може або натиснути “Підтвердити”, якщо потрібно підтвердити додавання, а якщо дії потрібно відмінити користувачу достатньо просто закрити вікно.

Для редагування даних у таблиці результати тестування користувач обирає пункт меню “Редагувати”.

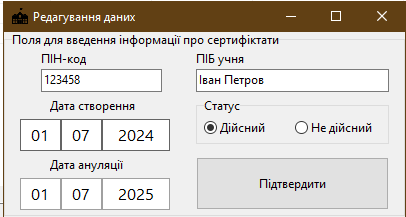


Рисунок 3.27 – Вікно для редагування записів у таблиці Сертифікати НМТ

Після вибору пункту меню користувач бачить форму з попередньо заповненими полями згідно значення полів таблиці для вибору або введення інформації, що потребує таблиця. Для кожного поля реалізовані перевірки відповідно логічного змісту та правил збергіання у базі даних. У разі відстуності помилок та заповнення полів користувач може або натиснути “ Підтвердити ”, якщо потрібно підтвердити редагування, а якщо дії потрібно відмінити користувачу достатньо просто закрити вікно.

До того ж, з будь-якого меню можно викликати довідкове вікно кліком по пункту меню “Про додаток”.

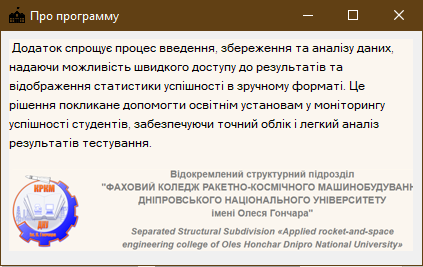


Рисунок 3.28 – Довідкове вікно

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 3.1–3.2.

Лістинг 3.1 – Приклад регулярного вираження

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 3.2 – Приклад регулярного вираження

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();

}

}

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 3.29-3.30.

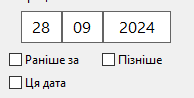


Рисунок 3.29 – Введення дати для фільтрації

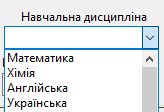


Рисунок 3.30 – Вибір навчальної дисципліни

Для коректного введення числових значень використано властивість NumbersOnly для LabeledEdit. Налаштування властивості продемонстровано на рисунку 3.31.



Рисунок 3.31 – Налаштування Edit

Для розмеження доступу до роботи з даними використано дезактивацію пунктів меню для роботи з даними для неавторизованих користувачів. Блокування продемонстровано на рисунку 3.32.

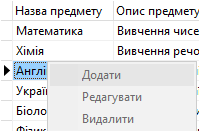


Рисунок 3.32 – Заблоковане меню

Для повідомлення про некоректне введення використано повідомлення для користувача. Приклад помилки продемонстровано на рисунку 3.33.

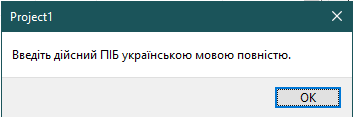


Рисунок 3.33 – Вікно помилки

Для підтвердження видалення використано діалогове вікно. Приклад вікна продемонстровано на рисунку 3.34.

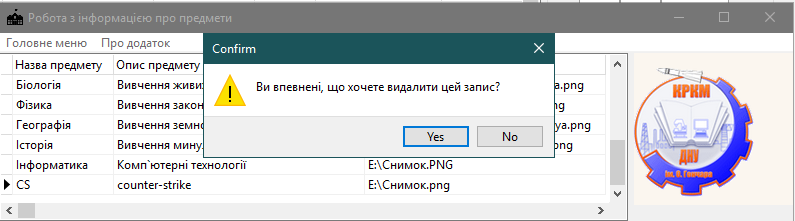


Рисунок 3.34 – Вікно помилки

Додаток А

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');