Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Кафедра програмної інженерії

КУРСОВА РОБОТА

ПОЯСНЮВАЛЬНА ЗАПИСКА

з дисципліни “Об’єктно-орієнтоване програмування”

КАРТОТЕКА ІНТЕРПОЛУ

Керівник, проф. Бондарєв В.М.

Студент гр. ПЗПІ-22-1 Ємельяненко Є.О.

Комісія:

Проф. Бондарєв В.М.,

Ст. викл. Черепанова Ю.Ю.,

Ст. викл. Ляпота В.М.

Харків 2023

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Кафедра *програмної інженерії*

Рівень вищої освіти *перший (бакалаврський)*

Дисципліна *Об’єктно-орієнтоване програмування*

Спеціальність *121 Інженерія програмного забезпечення*

Освітня програма *Програмна інженерія*

Курс *1*. Група *ПЗПІ-22-1*. Семестр *2*.

*ЗАВДАННЯ*

*на курсовий проект студента*

*Ємельяненко Євгенія Олексійовича*

1 Тема проекту: Картотека інтерполу

2 Термін здачі студентом закінченого проекту: “16” - червня - 2023 р.

3 Вихідні дані до проекту:

*Специфікація програми, методичні вказівки до виконання курсової роботи.*

4 Зміст розрахунково-пояснювальної записки:

*Вступ, опис вимог, проектування програми, інструкція користувача, висновки.*

КАЛЕНДАРНЫЙ ПЛАН

|  |  |  |
| --- | --- | --- |
| № | Назва етапу | Термін виконання |
| 1 | Видача теми, узгодження і затвердження теми | 13.02.2023 - 14.03.2023 р. |
| 2 | Формулювання вимог до програми | 20.05.2023 - 21.05.2023 р. |
| 3 | Розробка моделей | 21.05.2023 р |
| 4 | Розробка переходу між формами | 30.05.2023 р. |
| 5 | Розробка функцій додавання, видалення. | 30.05.2023 р. |
| 6 | Розробка функції пошуку і сортування | 30.05.2023 р. |
| 7 | Розробка перенесення злочинців до архіву | 30.05.2023 р. |
| 8 | Тестування і доопрацювання розробленої програмної системи | 31.05.2023 - 01.06.2023 р. |
| 9 | Оформлення пояснювальної записки | 01.06.2023 - 02.06.2023 р. |
| 10 | Захист | 02.06.2023 р. |

Студент Ємельяненко Євгеній Олексійович

Керівник Бондарєв Володимир Михайлович

«31» травня 2023 року

**РЕФЕРАТ**

Пояснювальна записка до курсової роботи: 41 с., 29 рис.,1 табл., 2 джерела.

ДАНІ ПО КОЖНОМУ ЗАРЕЄСТРОВАНОМУ ЗЛОЧИНЦЮ, СПИСОК ЗЛОЧИНЦІВ, ПЕРЕНЕСЕННЯ, ВИДАЛЕННЯ, ЗЛОЧИНЕЦЬ, МОВА ПРОГРАМУВАННЯ С#, ОБ’ЕКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ, ПЛАТФОРМА .NET Framework 4.7.2, WINDOWS FORMS.

Метою роботи є розробка програми «Картотека інтерполу», яка дає можливість користувачу передивлятись інформацію про злочинців, переносити їх до архіву та видаляти. Також користувач може передивлятись інформацію про співучасників різних угрупувань.

В результаті отримана програма, що дозволяє керувати списком злочинців. Є можливість пошуку злодіїв за довільним шаблоном, перегляду інформації про кожного злочинця, а також перегляд інформації про співучасників угрупувань. Розроблено можливість перенесення злочинців до архіву і назад до загального списку, а також видалення, якщо злочинець мертвий.

В процесі розробки використано середовища Microsoft Visual Studio 2022, фреймворк Windows Forms, платформи .NET Framework 4.7.2, мова програмування C#.

**ЗМІСТ**

ВСТУП 5

1 ОПИС ВИМОГ 6

1.1 Сценарії використання 6

1.2 Функціональні вимоги 13

2 ПРОЕКТУВАННЯ ПРОГРАМИ 24

3 ІНСТРУКЦІЯ КОРИСТУВАЧА 26

3.1 Встановлення програми 26

3.2 Видалення програми 26

3.3 Робота з програмою 26

ВИСНОВКИ 38

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ 39

ДОДАТКИ 40

**ВСТУП**

Метою даної роботи є розробка програми з використанням фреймворку Windows Forms для обліку інформації про злочинців. Призначеною функцією цього програмного продукту є зручне управління списком злодіїв у відділах поліції, з обліком даних про злочинців.

Основна мета полягає в тому, щоб забезпечити користувача зручним інструментом для редагування списку злочинців та перегляду інформації про кожного, такої як ім'я, фамілія, злочинна професія та інші деталі.

Програма "Картотека інтерполу" призначена для широкого кола користувачів. Вона надає можливість видалення, перенесення і перегляду даних про злодіїв, а також функцію пошуку за будь-яким шаблоном.

**1 ОПИС ВИМОГ**

1.1 Сценарії використання

**Сценарій 1. Перегляд інформації про злочинців з основного списку**

*Передумова*

Користувач запускає програму, відкривається головне вікно, список злочинців не пустий.

*Основний сценарій*

1. Користувач натискає кнопку «Розпочати пошук».

2. Програма відкриває форму з списком усіх злочинців.

3. Користувач у полі пошуку вводить данні необхідного йому злодія.

4. Користувач натискає кнопку «Знайти».

5. Програма оновлює список та користувач отримує список усіх злочинців, що задовольняє умовам пошуку.

6. Користувач обирає будь-якого злочинця зі списку та натискає на нього лівою кнопкою миші.

7. Відкривається форма з усією інформацією про обраного злочинця.

*Додатковий сценарій*

1. Користувач натискає кнопку «Розпочати пошук».

2. Програма відкриває форму з списком усіх злочинців.

3. Користувач у полі пошуку нічого не вводить.

4. Користувач натискає кнопку «Знайти».

5. Програма оновлює список та користувач отримує список усіх злочинців.

6. Користувач обирає будь-якого злочинця зі списку та натискає на нього лівою кнопкою миші.

7. Відкривається форма з усією інформацією про обраного злочинця.

**Сценарій 2. Перенесення злочинця до архіву**

*Передумова*

Користувач запускає програму, відкривається головне вікно, список злочинців не пустий. Користувач відкрив форму з інформацією обраного злочинця.

*Основний сценарій*

1. Користувач обирає з випадаючого списку куди він хоче перенести обраного злочинця.

2. Користувач натискає кнопку «Перенести».

3. Відкривається діалогове вікно з підтвердженням.

4. Користувач підтверджує перенесення.

5. Програма переносить злочинця туди, куди обрав користувач, та виводить повідомлення що злочинець був перенесен

*Додатковий сценарій*

1. Користувач обирає з випадаючого списку куди він хоче перенести обраного злочинця.

2. Якщо злочинець вже є, там куди хоче перенести його користувач, програма виводить повідомлення що користувач вже є там.

3. Програма не виконує ніяких дій.

*Додатковий сценарій*

1. Користувач обирає з випадаючого списку куди він хоче перенести обраного злочинця.

2. Користувач натискає кнопку «Перенести».

3. Відкривається діалогове вікно з підтвердженням.

4. Користувач не підтверджує перенесення.

5. Програма не виконує ніяких дій.

**Сценарій 3. Видалення злочинця**

*Передумова*

Користувач запускає програму, відкривається головне вікно, список злочинців не пустий. Користувач відкрив форму з інформацією обраного злочинця.

*Основний сценарій*

1. Користувач натискає кнопку «Видалити».
2. Програма виводить діалогове вікно з підтвердженням.
3. Користувач підтверджує видалення.
4. Злочинець мертвий.
5. Програма видаляє злочинця.

*Додатковий сценарій*

1. Користувач натискає кнопку «Видалити».
2. Програма виводить діалогове вікно з підтвердженням.
3. Користувач не підтверджує видалення.
4. Програма не виконує ніяких дій.

*Додатковий сценарій*

1. Користувач натискає кнопку «Видалити».
2. Програма виводить діалогове вікно з підтвердженням.
3. Користувач підтверджує видалення.
4. Злочинець живий.
5. Програма виводить повідомлення що злочинець живий і скасовує видалення.

**Сценарій 4. Огляд інформації про співучасників угрупувань**

*Передумова*

Користувач запускає програму, відкривається головне вікно, список злочинців не пустий. Користувач відкрив форму з інформацією обраного злочинця.

*Основний сценарій*

1. Користувач натискає на кнопку «…» біля назви угрупування.
2. Відкривається форма зі списком усіх співучасників цього угрупування, де користувач може перегляну інформацію про кожного злочинця окремо.

**Сценарій 5. Перегляд інформації про злочинців з архіву**

*Передумова*

Користувач запускає програму, відкривається головне вікно, список злочинців не пустий.

*Основний сценарій*

1. Користувач натискає кнопку «Архів».

2. Програма завантажує форму зі списком злочинців, які були перенесені до архіву.

3. Користувач обирає будь-якого злочинця натиснувши на нього лівою кнопкою миші.

4. Відкривається нова форма з інформацією про обраного злочинця.

1.2 Функціональні вимоги

**Функція 1. Перегляд інформації про злочинців з основного списку**

Головне вікно програми із якого відбувається навігація(рис. 1.1).



Рисунок 1.1 – Головне вікно програми

Функція перегляду здійснюється з пошукового вікна програми (рис. 1.2).

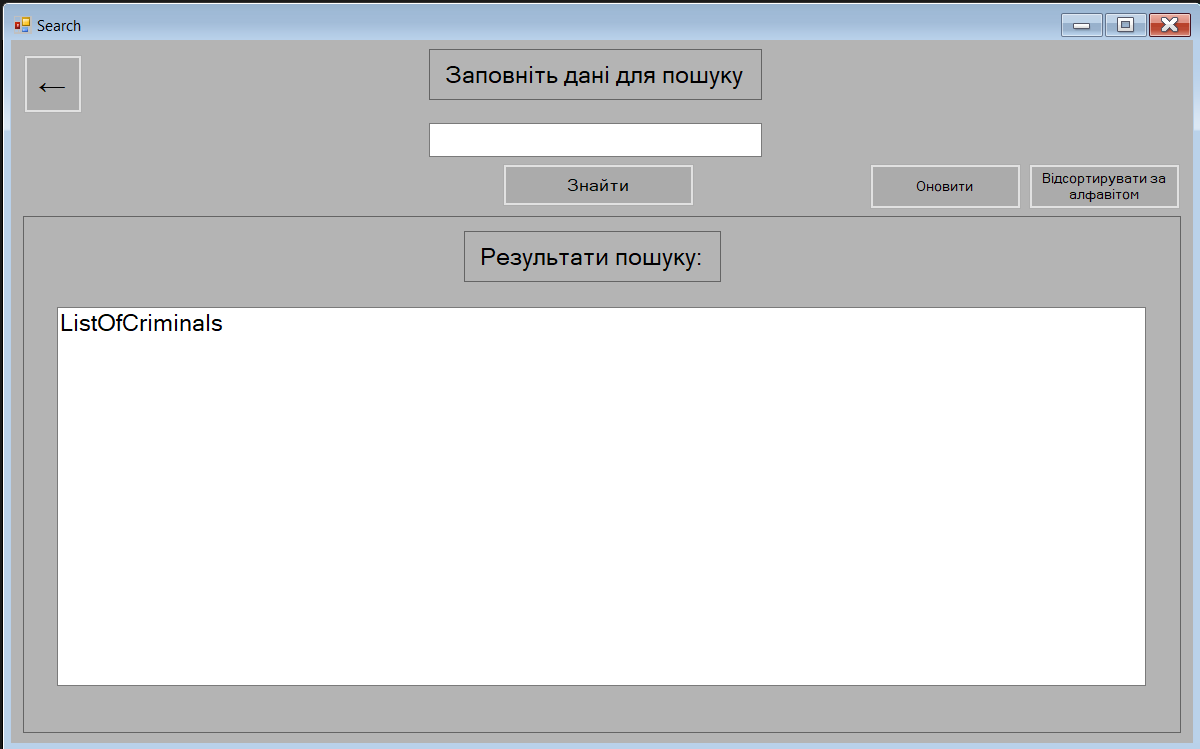


Рисунок 1.2 – Пошукове вікно програми

Інформація про працівників безпосередньо оновлюється під час роботи програми.

Функція перегляду здійснюється з інформаційного вікна програми (рис. 1.2). У відкритому пошуковому вікні(рис. 1.2) користувач натискає на обраного злочинця.

Програма відкриває інформаційну форму з полями, в які передаються дані про обрану людину.(рис. 1.3).

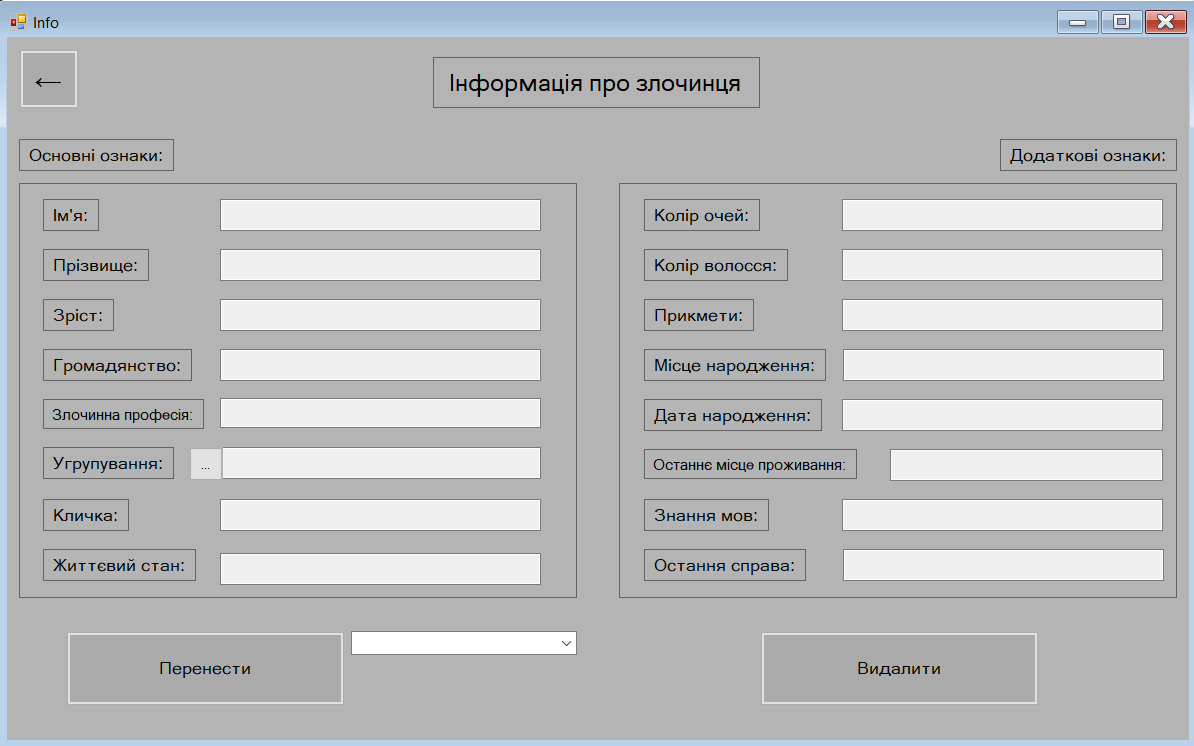


Рисунок 1.3 – Інформаційне вікно програми

**Функція 2. Перенесення злочинця до архіву**

Функція перенесення злочинця до архіву здійснюється у інформаційному вікні програми(рис. 1.3). Треба обрати місце з випадаючого списку(рис. 1.4).



Рисунок 1.4 – Випадаючий список

Після того як користувач обрав куди перенести злочинця, він повинен натиснути кнопку «Перенести», після цього програма виведе діалогове вікно з підтвердженням(рис. 1.5)

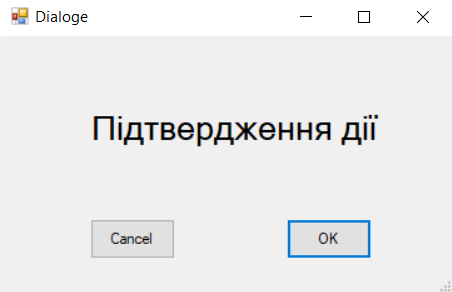


Рисунок 1.5 – Діалогове вікно підтвердження

Після успішного перенесення програма виведе повідомлення що злочинець був перенесен(рис. 1.6).

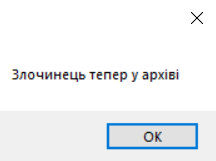


Рисунок 1.6 – Повідомлення про успішне перенесення

**Функція 3. Видалення злочинця**

Функція звільнення здійснюється з інформаційного вікна програми (рис. 1.3). Користувач натискає кнопку «Видалити». Програма виводить діалогове вікно з підтвердженням(рис. 1.5). Користувач підтверджує видалення і якщо злочинець мертвий, програма видаляє його, але якщо злочинець живий, то програма виводить повідомлення(рис. 1.7) що злочинець ще живий і скасовує видалення.

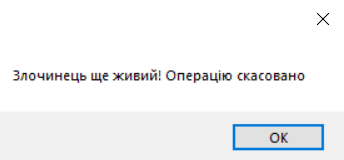


Рисунок 1.7 – Повідомлення про скасування видалення

**Функція 4. Перегляд інформації про співучасників угрупувань**

Функція перегляду інформації про співучасників угрупувань здійснюється з інформаційного вікна програми(рис. 1.3). Користувач натискає на кнопку «…»(рис. 1.8), що знаходиться біля назви угрупування обраного злочинця.



Рисунок 1.8 – Кнопка перегляду інформації про співучасників угрупування

Програма відкриває нове вікно програми(рис. 1.9) зі списком усіх злочинців цього угрупування.

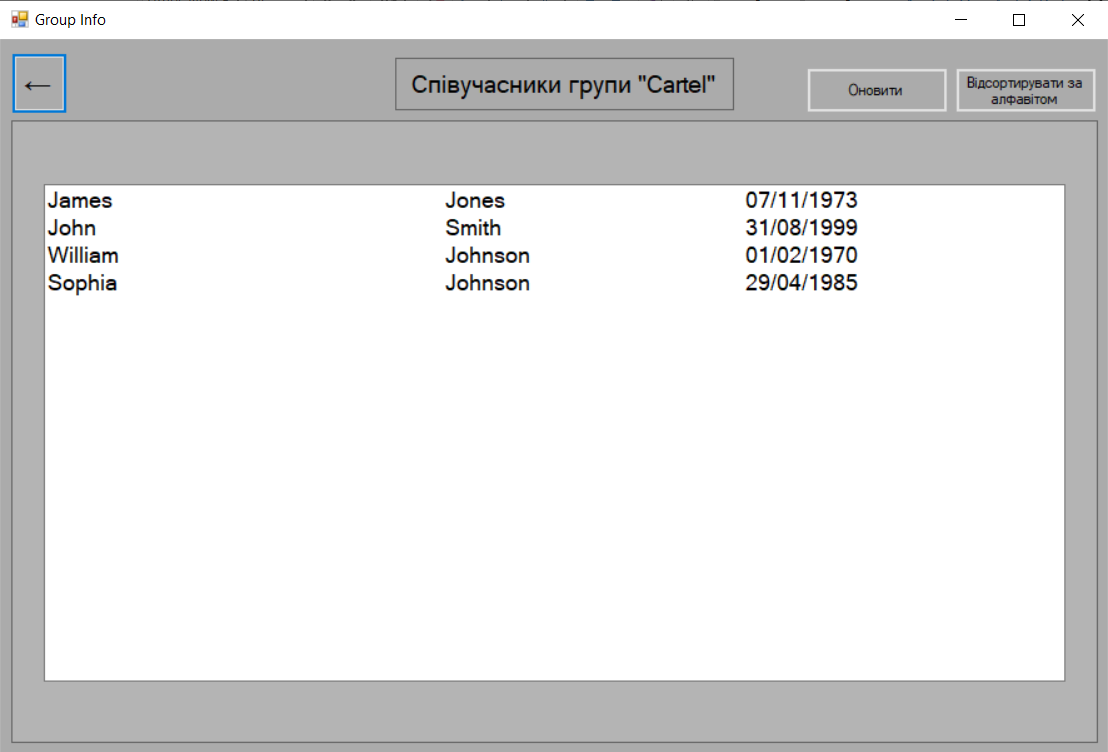


Рисунок 1.9 – Вікно перегляду інформації про співучасників угрупування

**Функція 5. Пошук злочинця по довільному шаблону**

Функція пошуку здійснюється у пошуковому вікні програми(рис. 1.2). Вибірка з будь-якої підмножини ознак:

1. Ім’я,
2. Прізвище,
3. Зріст,
4. Кличка,
5. Колір очей,
6. Колір волосся,
7. Громадянство,
8. Місце народження,
9. Дата народження(дд/мм//рррр),
10. Знання мов,
11. Злочинна професія,
12. Остання справа,
13. Особливі прикмети,
14. Останнє місце проживання.

Користувач вводить будь-яке одне з цих ознак у пошукове поле(рис. 1.10) і натискає кнопку «Знайти».

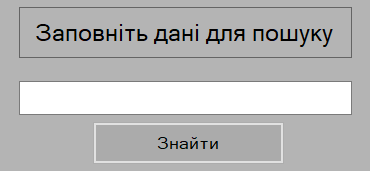


Рисунок 1.10 – Пошукове поле програми

**Функція 6. Збереження, завантаження працівників**

Данна функція працює автономно у коді програми. Після кожної дії над списком злочинців не помітно для користувача, данні автоматично оновлюються задля забезпечення цілісності файлів. Запис працівників відбувається списком.

**2 ПРОЕКТУВАННЯ ПРОГРАМИ**

Для реалізації програми була обрана настільна архітектура з Graphical User Interface. Для зручності розподілили усі файли проекту на три групи: Data, Forms, Models (рис. 2.1).

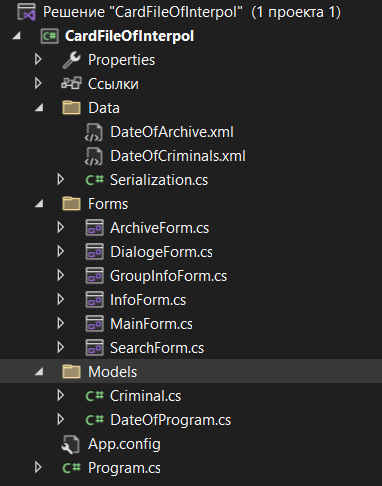


Рисунок 2.1 – Структура файлів проекту

Усі данні будуть зберігатись у форматі XML. Таке рішення обумовлене серіалізацією приватних полів а також більшою ефективністю стиснення. Також при пошкодженні файлів відновлення пошкодженої ділянки коду можна найкращим способом відновити данні (мова йде про переваги структурованого запису об’єктів).

У проекті є такі класи:

1. Клас "MainForm":

Це головне вікно програми, яке з'являється при запуску. З нього відбувається уся навігація по програмі.

2. Клас "SearchForm":

Це вікно відкривається з головної сторінки. Використовується для пошуку злочинців у списку. Використовується для сереалізації даних.

3. Клас "InfoForm":

Це вікно відкривається коли користувач натискає на обраного злочинця зі списку. Використовується для перегляду інформації про обраного злодія, для перенесення до архіву, для видалення та для перегляду інформації про співучасників унрупування.

4. Клас “DialogeForm”:

Це діалогове вікно відкривається для підвердження дії над обраним злочинцем. Використовується для видалення або перенесення злодія.

5. Клас "Employee":

Представляє об'єкт, що описує працівника в програмі.

6. Клас "ArchiveForm":

Це вікно, що містить список злодіїв яких було перенесено до архіву. Використовується для серіалізації даних. Використовується для видалення, перенесення до основного списку та для перегляду інформації про співучасників угрупування обраного злочинця.

7. Клас "GroupInfoForm":

Це вікно, що містить список злодіїв які є співучасниками угрупування. Використовується для перегляду інформації обраного злочинця.

8. Клас "Criminal":

Це клас, що містить поля з даними о злодії, а також деякі методи для роботи програми.

9. Клас "DateOfProgram":

Це клас, що містить у собі основний список злодіїв та список архіву, а також деякі змінні для роботи програми. Використовується для сереалізаціїї даних.

**3 ІНСТРУКЦІЯ КОРИСТУВАЧА**

3.1 Встановлення програми

Процедура встановлення програми протікає таким чином:

1) Завантажити .zip файл.

2) Розархівувати його.

3) Далі шукаємо .exe у папках. CardFileOfInterpol > bin > Debug > CardFileOfInterpol.exe



Рисунок 3.1 – Вигляд у папці

4) Скопіювати цей файл до робочого столу.

5) Запустити програму з робочого столу.

3.2 Видалення програми

Для цього необхідно просто видалити папку проекту з Вашого комп’ютеру і також видалити ярлик з робочого столу.

3.3 Робота з програмою

Запускаємо програму. Відкривається головне вікно (рис. 3.2), з якого проводяться усі переходи до наступних вікон.



Рисунок 3.2 – Головне вікно програми

На головному екрані знаходяться три кнопки. Перша кнопка «Розпочати пошук» відкриває пошукове вікно зі списком злочинців з основного списку. Друга кнопка «Архів »відкриває вікно архіву зі списком злочинців з архіву. Кнопка «Вихід» завершує роботу програми.

Розглянемо роботу кнопки «Розпочати пошук». При натисненні відкривається пошукове вікно(рис. 3.3) зі списком злочинців з основного списку, поле пошуку, кнопка «Оновити»(рис. 3.4) та кнопка «Відсортувати за алфавітом»(рис. 3.5).

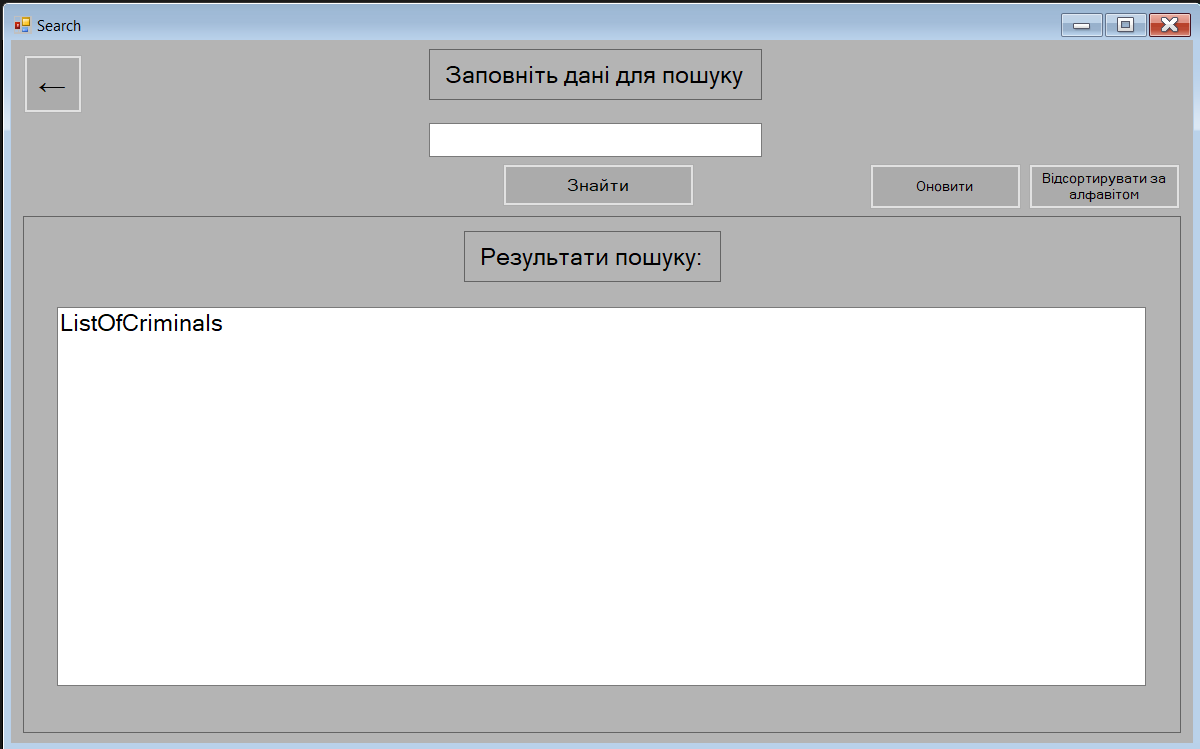


Рисунок 3.3 – Пошукове вікно програми



Рисунок 3.4 – Кнопка «Оновити»



Рисунок 3.5 – Кнопка «Відсортувати за алфавітом»

Натиснувши кнопку «Оновити» список злодіїв оновиться.

Натиснувши кнопку «Відсортувати за алфавітом» список злодіїв відсортується за алфавітом.

У пошуковому вікні розташовано панель пошуку(рис. 3.6):

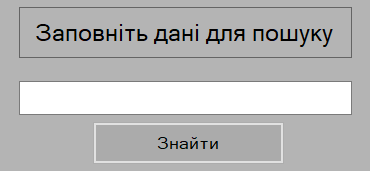


Рисунок 3.6 – Пошукове поле програми

У пошукове поле програми користувач обирає будь-яке одне з-під підмножини ознак:

1. Ім’я,
2. Прізвище,
3. Зріст,
4. Кличка,
5. Колір очей,
6. Колір волосся,
7. Громадянство,
8. Місце народження,
9. Дата народження(дд/мм//рррр),
10. Знання мов,
11. Злочинна професія,
12. Остання справа,
13. Особливі прикмети,
14. Останнє місце проживання.

Користувач вводить будь-яке одне з цих ознак у пошукове поле і натискає кнопку «Знайти».

Після обирання злочинця зі списку, користувач натискає на обраного злочинця лівою кнопкою миші. Програма виводить інформаційне вікно програми з усіма характеристиками обраного злодія(рис. 3.7).

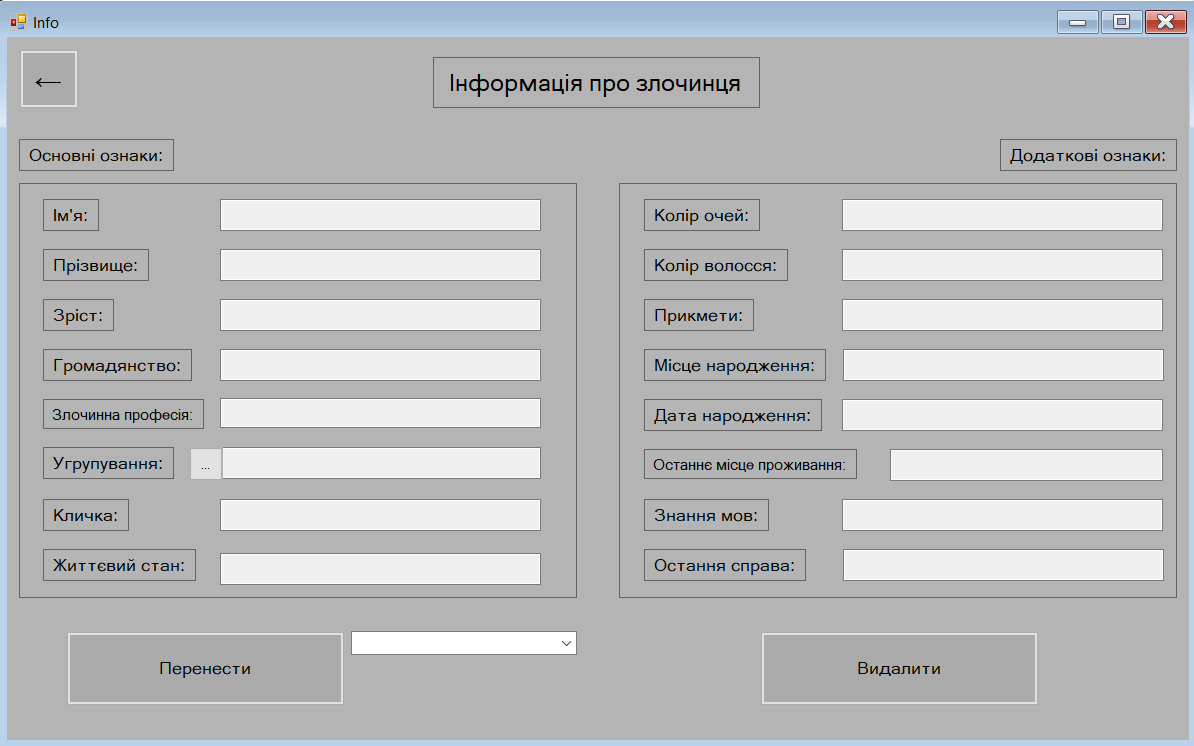


Рисунок 3.7 – Інформаційне вікно програми

Користувач може перенести обраного злодія до архіву натиснувши кнопку «Перенести»(рис. 3.8) та обравши куди він хоче перенести злодія у випадаючому списку(рис. 3.9).

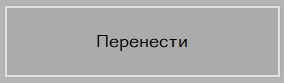


Рисунок 3.8 – Кнопка «Перенести»



Рисунок 3.9 – Випадаючий список

Після натискання кнопки «Перенести» програма виводить діалогове вікно з підтвердженням(рис. 3.10).

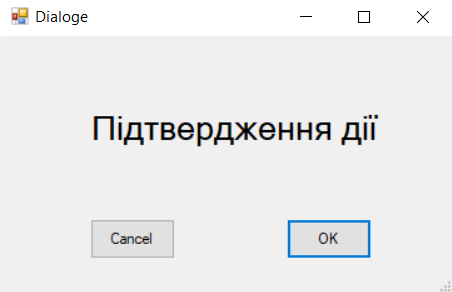


Рисунок 3.10 – Діалогове вікно підтвердження

Користувач натискає «ОК» і програма переносить обраного злочинця до архіву. Після успішного перенесення програма виводить повідомлення що злочинець був перенесен(рис. 3.11).

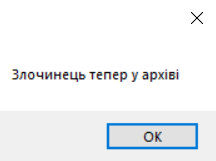


Рисунок 3.11 – Повідомлення про успішне перенесення

Натиснувши кнопку «Видалити» користувач повинен переконатися що злочинець мертвий. Цю інформацію можна дізнатись з поля «Життєвий стан»(рис. 3.12).



Рисунок 3.12 – Інформаційне поле «Життєвий стан»

Якщо злочинець живий, програма виводить повідомлення о скасуванні операції(рис. 3.12).

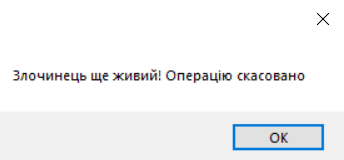


Рисунок 3.13 – Повідомлення про скасування видалення

Якщо злочинець мертвий, програма виводить діалогове вікно з підтвердженням(рис. 3.10). Користувач натискає «ОК», програма успішно видаляє злочинця зі списку.

Щоб перегляну список злочинців у архіві, користувач може натиснути кнопку «Повернутись»(рис. 3.14), яка розташована у верхньому лівому куті вікна програми.



Рисунок 3.14 – Кнопка «Повернутись»

Після натискання ми повертаємось у головне вікно програми(рис. 3.2).

Користувач натискає кнопку «Архів», програма виводить вікно програми зі списком злочинців які були перенесені до архіву(рис. 3.15).

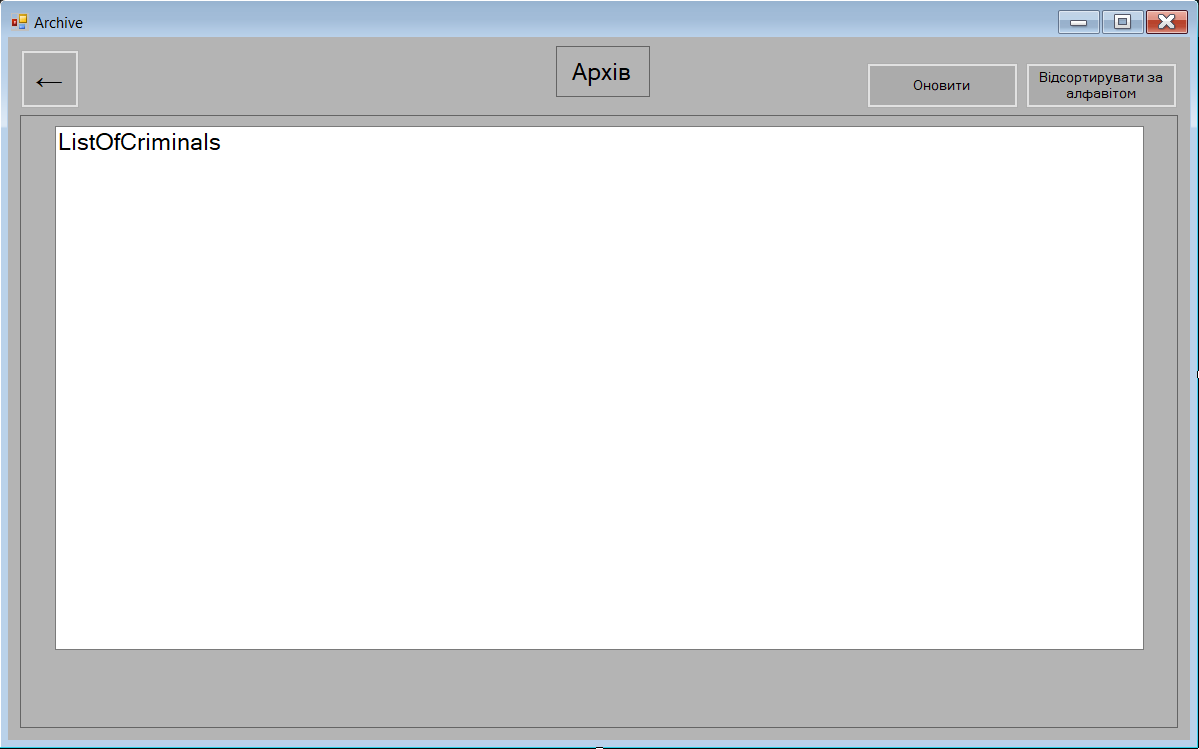


Рисунок 3.15 – Архів злочинців

Для того, щоб користувач міг переглянути інформацію про співучасників угрупувань, користувач повинен обрати злодія з основного списку злочинців, який розаташован у пошуковому вікні програми(рис. 3.3) або у архіві злочинців(рис. 3.15). Користувач обирає злодія і натискає на нього лівою кнопкою миші. Програма виводить інформаційне вікно програми(рис. 3.7) і користувач натискає на кнопку «…»(рис. 3.16).



Рисунок 3.16 – Кнопка перегляду інформації про співучасників угрупування

Після натиснення кнопки «…» програма виводить вікно зі списком співучасників обраного угрупування(рис. 3.17).

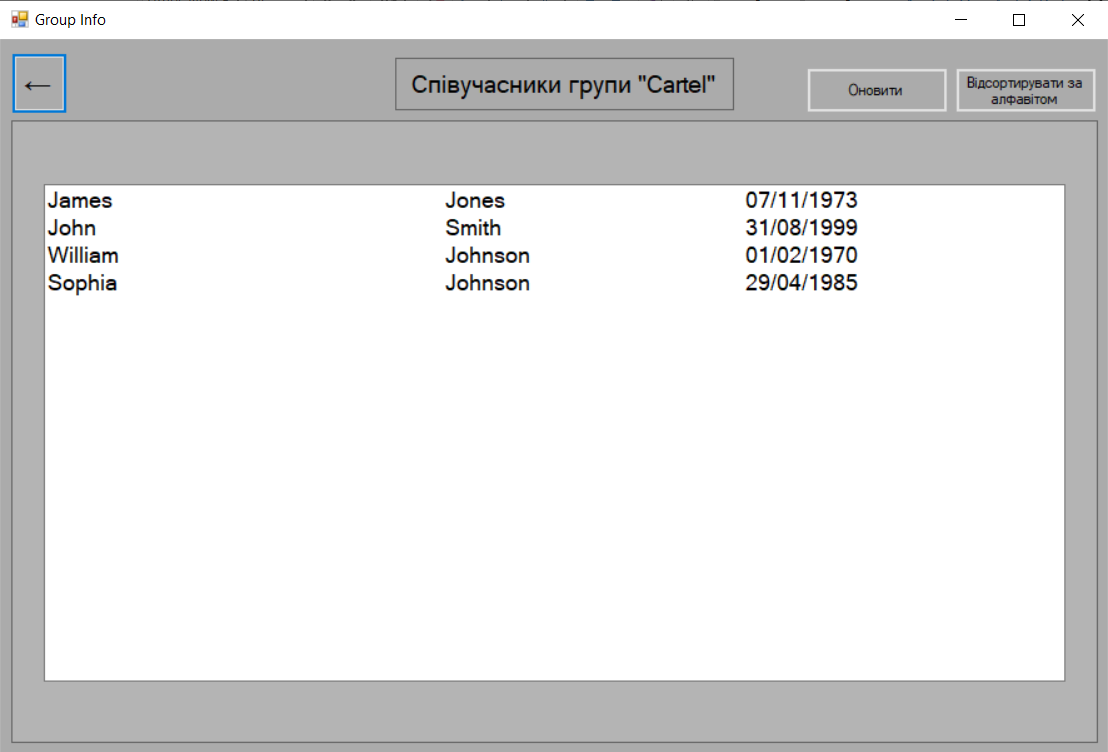


Рисунок 3.17 – Вікно перегляду інформації про співучасників угрупування

У цьому вікні користувач також може переглядати інформацію про будь-якого злодія.

**ВИСНОВКИ**

Розроблена програма успішно відповідає всім початковим вимогам і надає широкі можливості користувачам. За допомогою цієї програми користувачі можуть шукати злодіїв по довільному шаблону, передивлятись інформацію, переносити до архіву та видаляти злочинців. Також, за бажанням, користувач може подивитися інформацію про співучасників злочинних угрупувань.

Під час розробки застосунку, я вклав багато зусиль у покращення своїх навичок програмування мовою C#. Використання Windows Forms дало мені можливість дізнатись щось нове і цікаве для мене. За допомогою формату XML, я організував зберігання даних про працівників, забезпечуючи надійність та доступність цієї інформації.

У результаті моєї роботи, розроблена програма стала інструментом для керування картотекою інтерполу. Вона спрощує рутинні задачі, зменшує можливість помилок та забезпечує збереження даних в безпеці.

**ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ**

1. Бондарєв В.М. Електронний учбовий посібник з дисципліни «Основи програмування» URL: <http://tss.co.ua:5555/>(дата звернення: 30.06.2023)

2. Windows Forms documentation. URL[: https://learn.microsoft.com/en-us/dotnet/desktop/winforms/?view=netdesktop-7.0](:%20https:/learn.microsoft.com/en-us/dotnet/desktop/winforms/?view=netdesktop-7.0%20) (дата звернення: 21.05.2023)

**ДОДАТОК А**

Код програми

using CardFileOfInterpol.Data;

using System;

using System.Collections.Generic;

using System.Windows.Forms;

namespace CardFileOfInterpol

{

public partial class MainForm : Form

{

public MainForm()

{

InitializeComponent();

DateOfProgram.archive = Serialization.DeserializeObjects(Serialization.DATA\_PATH\_ARCHIVE);

Serialization.SerializeObjects(DateOfProgram.archive, Serialization.DATA\_PATH\_ARCHIVE);

DateOfProgram.criminals = Serialization.DeserializeObjects();

if (DateOfProgram.criminals.Count == 0)

{

DateOfProgram.criminals = GenerateCriminals();

}

Serialization.SerializeObjects(DateOfProgram.criminals);

}

private void SearchButton\_Click(object sender, EventArgs e)

{

SearchForm searchForm = new SearchForm();

searchForm.FormClosed += (eventSender, eventArgs) => Show();

searchForm.Show();

Hide();

}

private void ArchiveButton\_Click(object sender, EventArgs e)

{

ArchiveForm archiveForm = new ArchiveForm();

archiveForm.FormClosed += (eventSender, eventArgs) => Show();

archiveForm.Show();

Hide();

}

private void ExitButton\_Click(object sender, EventArgs e)

{

Application.Exit();

}

private List<Criminal> GenerateCriminals()

{

string[] names = { "John", "Emma", "Michael", "Olivia", "William", "Ava", "James", "Sophia", "Benjamin", "Isabella" };

string[] secondNames = { "Smith", "Johnson", "Williams", "Jones", "Brown", "Davis", "Miller", "Wilson", "Moore", "Taylor" };

string[] citizenships = { "United States", "Canada", "United Kingdom", "Australia", "Germany", "France", "Spain", "Italy", "Japan", "China" };

string[] criminalProfessions = { "Hitman", "Thief", "Hacker", "Smuggler", "Drug Dealer", "Blackmailer", "Gang Leader", "Forgery Expert", "Bank Robber", "Assassin" };

string[] aliases = { "Shadow", "Spike", "Viper", "Raven", "Wolf", "Phoenix", "Scar", "Ghost", "Ace", "Joker" };

string[] groups = { "Syndicate", "Mafia", "Cartel", "Chickens", "Dogs", "Nightfall", "Blood", "Cartel", "Wolves", "Phoenix" };

string[] colors = { "Red", "Blue", "Green", "Yellow", "Orange", "Purple", "Pink", "Black", "White", "Gray" };

string[] omens = { "Tattooed", "Scarred", "Mysterious", "Dangerous", "Deceptive", "Evasive", "Reckless", "Cunning", "Intimidating", "Elusive" };

string[] cities = { "London", "New York", "Paris", "Tokyo", "Sydney", "Berlin", "Rome", "Dubai", "Africa", "Toronto" };

string[] languages = { "English", "Spanish", "French", "German", "Italian", "Brazil", "Chinese", "Japanese", "Arabic", "Portuguese" };

string[] lastThings = { "Bank robbery", "Murder", "Drug trafficking", "Fraud", "Piracy", "Kidnapping", "Hacking", "vandalism", "Bribery of officials", "Terrorism" };

bool[] alive = { true, true, false, true, true};

List<Criminal> list = new List<Criminal>();

Random r = new Random();

for (int i = 0; i < 10; i++)

{

list.Add(new Criminal

{

Name = names[r.Next(0, 10)],

SecondName = secondNames[r.Next(0, 10)],

CitiezenShip = citizenships[r.Next(0, 10)],

CriminalProfession = criminalProfessions[r.Next(0, 10)],

GroupName = groups[r.Next(0, 10)],

Alias = aliases[r.Next(0, 10)],

EyeColor = colors[r.Next(0, 10)],

HairColor = colors[r.Next(0, 10)],

Omens = new string[] { omens[r.Next(0, 10)] },

PlaceOfBirth = cities[r.Next(0, 10)],

DateOfBirth = new DateTime(r.Next(1950, 2006), r.Next(1, 13), r.Next(1, 32)),

LastPlaceOfLiving = cities[r.Next(0, 10)],

Languages = new string[] { languages[r.Next(0, 10)] },

LastThing = lastThings[r.Next(0, 10)],

Alive = alive[r.Next(0, 5)],

Height = r.Next(150, 200),

});

}

return list;

}

}

}

using CardFileOfInterpol.Data;

using System;

using System.Windows.Forms;

namespace CardFileOfInterpol

{

public partial class ArchiveForm : Form

{

public ArchiveForm()

{

InitializeComponent();

DateOfProgram.archive = Serialization.DeserializeObjects(Serialization.DATA\_PATH\_ARCHIVE);

ListOfCriminals.Items.Clear();

foreach (Criminal criminal in DateOfProgram.archive)

{

ListOfCriminals.Items.Add(criminal.GetInfo());

}

}

private void BackButton\_Click(object sender, EventArgs e)

{

this.Close();

}

private void ListOfCriminals\_SelectedIndexChanged(object sender, EventArgs e)

{

try

{

foreach (Criminal criminal in DateOfProgram.archive)

{

if (criminal.GetInfo() == ListOfCriminals.SelectedItem.ToString())

{

InfoForm infoForm = new InfoForm(criminal);

infoForm.FormClosed += (eventSender, eventArgs) => Show();

infoForm.Show();

Hide();

}

}

}

catch (Exception)

{

MessageBox.Show("Оберіть злочинця");

}

}

private void ButtonRefresh\_Click(object sender, EventArgs e)

{

DateOfProgram.archive = Serialization.DeserializeObjects(Serialization.DATA\_PATH\_ARCHIVE);

ListOfCriminals.Items.Clear();

foreach (Criminal criminal in DateOfProgram.archive)

{

ListOfCriminals.Items.Add(criminal.GetInfo());

}

}

}

}

using CardFileOfInterpol.Data;

using System;

using System.Collections.Generic;

using System.Windows.Forms;

namespace CardFileOfInterpol

{

public partial class SearchForm : Form

{

public SearchForm()

{

InitializeComponent();

}

public static List<Criminal> SearchCriminal(string line, List<Criminal> criminals)

{

List<Criminal> result = new List<Criminal>();

var t = line.ToLower();

foreach (Criminal criminal in criminals)

{

if (criminal.ToString().ToLower().IndexOf(t) > -1)

{

result.Add(criminal);

}

}

return result;

}

private void BackButton\_Click(object sender, EventArgs e)

{

this.Close();

}

private void SearchButton\_Click(object sender, EventArgs e)

{

if(SearchBox.Text == string.Empty)

{

ListOfCriminals.Items.Clear();

foreach (Criminal criminal in DateOfProgram.criminals)

{

ListOfCriminals.Items.Add(criminal.GetInfo());

}

}

else

{

DateOfProgram.SearchedCriminals = SearchCriminal(SearchBox.Text, DateOfProgram.criminals);

ListOfCriminals.Items.Clear();

foreach (Criminal criminal in DateOfProgram.SearchedCriminals)

{

ListOfCriminals.Items.Add(criminal.GetInfo());

}

}

}

private void ListOfCriminals\_SelectedIndexChanged(object sender, EventArgs e)

{

try

{

foreach (Criminal criminal in DateOfProgram.criminals)

{

if (criminal.GetInfo() == ListOfCriminals.SelectedItem.ToString())

{

InfoForm infoForm = new InfoForm(criminal);

infoForm.FormClosed += (eventSender, eventArgs) => Show();

infoForm.Show();

Hide();

}

}

}

catch (Exception)

{

MessageBox.Show("Оберіть злочинця");

}

}

private void ButtonSort\_Click(object sender, EventArgs e)

{

DateOfProgram.criminals.Sort((c1, c2) => string.Compare(c1.Name, c2.Name, StringComparison.Ordinal));

Serialization.SerializeObjects(DateOfProgram.criminals);

ListOfCriminals.Items.Clear();

foreach (Criminal criminal in DateOfProgram.criminals)

{

ListOfCriminals.Items.Add(criminal.GetInfo());

}

}

private void ButtonRefresh\_Click(object sender, EventArgs e)

{

ListOfCriminals.Items.Clear();

if (SearchBox.Text == string.Empty)

{

ListOfCriminals.Items.Clear();

foreach (Criminal criminal in DateOfProgram.criminals)

{

ListOfCriminals.Items.Add(criminal.GetInfo());

}

}

else

{

DateOfProgram.SearchedCriminals = SearchCriminal(SearchBox.Text, DateOfProgram.criminals);

ListOfCriminals.Items.Clear();

foreach (Criminal criminal in DateOfProgram.SearchedCriminals)

{

ListOfCriminals.Items.Add(criminal.GetInfo());

}

}

}

}

}

using CardFileOfInterpol.Data;

using System;

using System.Windows.Forms;

namespace CardFileOfInterpol

{

public partial class InfoForm : Form

{

string[] ways = new string[] { "Архів", "Список" };

string currentWay = "";

public InfoForm()

{

InitializeComponent();

}

public InfoForm(Criminal criminal)

{

InitializeComponent();

DateOfProgram.currentCriminal = criminal;

ShowInfo(criminal);

comboBoxWays.Items.AddRange(ways);

comboBoxWays.SelectedItem = ways[0];

}

public void ShowInfo(Criminal criminal)

{

textBoxName.Text = criminal.Name;

textBoxSecondName.Text = criminal.SecondName;

textBoxCitiezenShip.Text = criminal.CitiezenShip;

textBoxCriminalProfession.Text = criminal.CriminalProfession;

textBoxGroup.Text = criminal.GroupName;

textBoxAlias.Text = criminal.Alias;

textBoxEyeColor.Text = criminal.EyeColor;

textBoxHairColor.Text = criminal.HairColor;

textBoxOmens.Text = criminal.GetOmens();

textBoxPlaceOfBirth.Text = criminal.PlaceOfBirth;

textBoxDateOfBirth.Text = criminal.DateOfBirth.ToString("dd'/'MM'/'yyyy");

textBoxLastPlace.Text = criminal.LastPlaceOfLiving;

textBoxLanguages.Text = criminal.GetLanguages();

textBoxLastThing.Text = criminal.LastThing;

if(criminal.Alive == true)

{

textBoxAlive.Text = "Live";

}

else

{

textBoxAlive.Text = "Dead";

}

textBoxHeight.Text = criminal.Height.ToString();

}

private void BackButton\_Click(object sender, EventArgs e)

{

this.Close();

}

private void ButtonToArchive\_Click(object sender, EventArgs e)

{

if (currentWay == "Архів")//до архіву

{

DialogeForm dialogeForm = new DialogeForm();

dialogeForm.ShowDialog();

if(dialogeForm.DialogResult == DialogResult.OK)

{

if (DateOfProgram.criminals.Contains(DateOfProgram.currentCriminal) && !DateOfProgram.archive.Contains(DateOfProgram.currentCriminal))

{

DateOfProgram.archive.Add(DateOfProgram.currentCriminal);

Serialization.SerializeObjects(DateOfProgram.archive, Serialization.DATA\_PATH\_ARCHIVE);

DateOfProgram.criminals.Remove(DateOfProgram.currentCriminal);

Serialization.SerializeObjects(DateOfProgram.criminals);

MessageBox.Show("Злочинець тепер у архіві");

}

else

{

MessageBox.Show("Злочинець вже у архіві! Операцію скасовано");

}

}

}

else if(currentWay == "Список")//до списку

{

DialogeForm dialogeForm = new DialogeForm();

dialogeForm.ShowDialog();

if (dialogeForm.DialogResult == DialogResult.OK)

{

if (!DateOfProgram.criminals.Contains(DateOfProgram.currentCriminal) && DateOfProgram.archive.Contains(DateOfProgram.currentCriminal))

{

DateOfProgram.archive.Remove(DateOfProgram.currentCriminal);

Serialization.SerializeObjects(DateOfProgram.archive, Serialization.DATA\_PATH\_ARCHIVE);

DateOfProgram.criminals.Add(DateOfProgram.currentCriminal);

Serialization.SerializeObjects(DateOfProgram.criminals);

MessageBox.Show("Злочинець тепер у списку");

}

else

{

MessageBox.Show("Злочинець вже у списку! Операцію скасовано");

}

}

}

else

{

MessageBox.Show("Оберіть куди треба перенести злочинця!");

}

}

private void comboBoxWays\_SelectedIndexChanged(object sender, EventArgs e)

{

if (ways[0] == comboBoxWays.SelectedItem.ToString())

{

currentWay = ways[0];

}

else

{

currentWay = ways[1];

}

}

private void DeleteButton\_Click(object sender, EventArgs e)

{

if(DateOfProgram.currentCriminal.Alive == false)

{

DialogeForm dialogeForm = new DialogeForm();

dialogeForm.ShowDialog();

if (dialogeForm.DialogResult == DialogResult.OK)

{

if (DateOfProgram.archive.Contains(DateOfProgram.currentCriminal))

{

DateOfProgram.archive.Remove(DateOfProgram.currentCriminal);

Serialization.SerializeObjects(DateOfProgram.archive, Serialization.DATA\_PATH\_ARCHIVE);

}

else if (DateOfProgram.criminals.Contains(DateOfProgram.currentCriminal))

{

DateOfProgram.criminals.Remove(DateOfProgram.currentCriminal);

Serialization.SerializeObjects(DateOfProgram.criminals);

}

MessageBox.Show("Злочинець був видалений");

}

}

else

{

MessageBox.Show("Злочинець ще живий! Операцію скасовано");

}

}

private void buttonGroupInfo\_Click(object sender, EventArgs e)

{

GroupInfoForm groupInfoForm = new GroupInfoForm(DateOfProgram.currentCriminal.GroupName);

groupInfoForm.FormClosed += (eventSender, eventArgs) => Show();

groupInfoForm.Show();

Hide();

}

}

}

using System;

namespace CardFileOfInterpol

{

public class Criminal

{

public string Name;

public string SecondName;

public int Height;

public string CitiezenShip;

public string CriminalProfession;

public string GroupName;

public string Alias;

public string EyeColor;

public string HairColor;

public string[] Omens;

public string PlaceOfBirth;

public DateTime DateOfBirth;

public string LastPlaceOfLiving;

public string[] Languages;

public string LastThing;

public bool? Alive;

public Criminal()

{

Name = string.Empty;

SecondName = string.Empty;

CitiezenShip = string.Empty;

CriminalProfession = string.Empty;

GroupName = string.Empty;

Alias = string.Empty;

EyeColor = string.Empty;

HairColor = string.Empty;

Omens = new string[] { };

PlaceOfBirth = string.Empty;

DateOfBirth = DateTime.Now;

LastPlaceOfLiving = string.Empty;

Languages = new string[] { };

LastThing = string.Empty;

Alive = null;

}

public Criminal(string name, string secondName, string citiezenShip, string criminalProfession, string groupName, string alias, string eyeColor,

string hairColor, string[] omens, string placeOfBirth, DateTime dateOfBirth, string lastPlaceOfLiving, string[] languages,

string lastThing, bool alive, int height)

{

Name = name;

SecondName = secondName;

CitiezenShip = citiezenShip;

CriminalProfession = criminalProfession;

GroupName = groupName;

Alias = alias;

EyeColor = eyeColor;

HairColor = hairColor;

Omens = omens;

PlaceOfBirth = placeOfBirth;

DateOfBirth = dateOfBirth;

LastPlaceOfLiving = lastPlaceOfLiving;

Languages = languages;

LastThing = lastThing;

Alive = alive;

Height = height;

}

public string GetOmens()

{

string str = "";

for (int i = 0; i < Omens.Length; i++)

{

if(i == Omens.Length - 1)

{

str += Omens[i];

}

else

{

str += Omens[i] + ", ";

}

}

return str;

}

public string GetLanguages()

{

string str = "";

for (int i = 0; i < Languages.Length; i++)

{

if (i == Omens.Length - 1)

{

str += Languages[i];

}

else

{

str += Languages[i] + ", ";

}

}

return str;

}

public override string ToString()

{

return $"{Name}, {SecondName}, {Height}, {CitiezenShip}, {CriminalProfession}, {GroupName}, {Alias}, {EyeColor}, {HairColor}," +

$"{GetOmens()}, {PlaceOfBirth}, {DateOfBirth.ToString("dd'/'MM'/'yyyy")}, {LastPlaceOfLiving}, {GetLanguages()}, {LastThing}, {Alive}";

}

public string GetInfo()

{

return $"{Name}\t\t\t\t{SecondName}\t\t\t{DateOfBirth.ToString("dd'/'MM'/'yyyy")}";

}

}

}

using System;

using System.Collections.Generic;

using System.Drawing;

using System.Windows.Forms;

namespace CardFileOfInterpol

{

public partial class GroupInfoForm : Form

{

List<Criminal> criminalGroup = new List<Criminal>();

public GroupInfoForm()

{

InitializeComponent();

}

public GroupInfoForm(string groupName)

{

InitializeComponent();

MainPanel.Text = $"Співучасники групи \"{groupName}\"";

MainPanel.Location = new Point(this.Width / 2 - MainPanel.Width / 2, 15);

foreach (Criminal criminal in DateOfProgram.criminals)

{

if (criminal.GroupName == groupName)

{

criminalGroup.Add(criminal);

ListOfCriminals.Items.Add(criminal.GetInfo());

}

}

foreach (Criminal criminal in DateOfProgram.archive)

{

if (criminal.GroupName == groupName)

{

criminalGroup.Add(criminal);

ListOfCriminals.Items.Add(criminal.GetInfo());

}

}

}

private void ListOfCriminals\_SelectedIndexChanged(object sender, EventArgs e)

{

try

{

foreach (Criminal criminal in DateOfProgram.criminals)

{

if (criminal.GetInfo() == ListOfCriminals.SelectedItem.ToString())

{

InfoForm infoForm = new InfoForm(criminal);

infoForm.buttonGroupInfo.Visible = false;

infoForm.FormClosed += (eventSender, eventArgs) => Show();

infoForm.Show();

Hide();

}

}

foreach (Criminal criminal in DateOfProgram.archive)

{

if (criminal.GetInfo() == ListOfCriminals.SelectedItem.ToString())

{

InfoForm infoForm = new InfoForm(criminal);

infoForm.buttonGroupInfo.Visible = false;

infoForm.FormClosed += (eventSender, eventArgs) => Show();

infoForm.Show();

Hide();

}

}

}

catch (Exception)

{

MessageBox.Show("Оберіть злочинця");

}

}

private void BackButton\_Click(object sender, EventArgs e)

{

this.Close();

}

private void ButtonRefresh\_Click(object sender, EventArgs e)

{

ListOfCriminals.Items.Clear();

foreach (Criminal criminal in criminalGroup)

{

ListOfCriminals.Items.Add(criminal.GetInfo());

}

}

private void ButtonSort\_Click(object sender, EventArgs e)

{

criminalGroup.Sort((c1, c2) => string.Compare(c1.Name, c2.Name, StringComparison.Ordinal));

ListOfCriminals.Items.Clear();

foreach (Criminal criminal in criminalGroup)

{

ListOfCriminals.Items.Add(criminal.GetInfo());

}

}

}

}

using System.Collections.Generic;

namespace CardFileOfInterpol

{

internal static class DateOfProgram

{

public static List<Criminal> criminals = new List<Criminal>();

public static List<Criminal> archive = new List<Criminal>();

public static Criminal currentCriminal = new Criminal();

public static List<Criminal> SearchedCriminals = new List<Criminal>();

}

}

using System.Collections.Generic;

using System.IO;

using System.Xml.Serialization;

namespace CardFileOfInterpol.Data

{

public static class Serialization

{

public const string DATA\_PATH\_CRIMINALS = "C:\\Users\\emely\\source\\repos\\CardFileOfInterpol\\CardFileOfInterpol\\Data\\DateOfCriminals.xml";

public const string DATA\_PATH\_ARCHIVE = "C:\\Users\\emely\\source\\repos\\CardFileOfInterpol\\CardFileOfInterpol\\Data\\DateOfArchive.xml";

public static void SerializeObjects(List<Criminal> criminals, string DATA\_PATH = DATA\_PATH\_CRIMINALS)

{

XmlSerializer serializer = new XmlSerializer(typeof(List<Criminal>));

using (FileStream fileStream = new FileStream(DATA\_PATH, FileMode.Create))

{

serializer.Serialize(fileStream, criminals);

}

}

public static List<Criminal> DeserializeObjects(string DATA\_PATH = DATA\_PATH\_CRIMINALS)

{

XmlSerializer serializer = new XmlSerializer(typeof(List<Criminal>));

using (FileStream fileStream = new FileStream(DATA\_PATH, FileMode.Open))

{

return (List<Criminal>)serializer.Deserialize(fileStream);

}

}

}

}