Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Кафедра Програмної інженерії

КУРСОВА РОБОТА

ПОЯСНЮВАЛЬНА ЗАПИСКА

з дисципліні “ Об’єктно -орієнтоване програмування”

Керівник , доцент Бондарєв В. М.

Студент гр. ПЗПІ-19-2 Омельченко Д. Г.

Комісія:

Ст. викл.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Черепанова Ю.Ю.,

Проф. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Бондарєв В. М.

Доц. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Побіженко І. О.

Харків 2020

**ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ**

Кафедра: ***Програмної інженерії***

Дисципліна: ***Об’єктно-орієнтоване програмування***

Спеціальність: ***121***  ***Інженерія програмного забезпечення***

Освітня програма: ***Програмна інженерія***

Курс 1 .Група *ПЗПІ-19 -2* . Семестр *2* .

***ЗАВДАННЯ***

***на курсовий проект студента***

*Омельченко Даніла Геннадійович* *\_*

1 Тема проекту: *Довідник меломана*

2 Термін здачі студентом закінченого проекту: ***“31” - травня - 2020 р***.

3 Вихідні дані до проекту:

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

4 Зміст розрахунково-пояснювальної записки:

*Вступ,* *специфікація програми, проектна специфікація, інструкція користувача, висновки \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5 Перелік графічного матеріалу:

*Схема об’єктної моделі, алгоритми, приклади екранних форм* \_\_\_\_*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |
| --- | --- | --- |
| *№* | *Назва етапу* | *Термін виконання* |
| 1 | Видача теми, узгодження і затвердження теми | 21-02-2020 р. |
| 2 | Формулювання вимог до програми | 01-03-2020 – 31-03-2020 р. |
| 3 | Розробка підсистеми зберігання та додавання товарів. | 02-04-2020 – 12-04-2020 р. |
| 4 | Розробка підсистеми зберігання та реєстрації клієнтів. | 12-04-2020 – 22-04-2020 р. |
| 5 | Розробка функцій продажу та повернення товарів. | 24-04-2020 – 30-04-2020 р. |
| 6 | Розробка функцій зберігання та завантаження даних. | 05-05-2020 – 11-05-2020 р. |
| 7 | Тестування і доопрацювання розробленої програмної системи. | 10-05-2020 – 15-05-2020 р. |
| 8 | Оформлення пояснювальної записки, додатків, графічного матеріалу | 25-05-2020 – 05-06-2020 р. |
| 9 | Захист | 01-06-2020 – 19-06-2020 р. |

Студент Омельченко Д. Г.

Керівник Бондарєв В. М.

« 21 »\_лютого\_\_\_\_\_\_\_\_\_\_ 2020 р.

**РЕФЕРАТ**

Пояснювальна записка до курсової роботи: 36 с., 22 рис., 1 додаток, 5 джерел.

МОВА ПРОГРАМУВАННЯ С#, WINDOWS FORMS, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, ДОВІДНИК, МУЗИКА, ПОШУК, КОРИСТУВАЧ, АДМІНІСТРАТОР.

Метою роботи є розробка програмного забезпечення “Довідник меломана”, базуючись на принципах об’єктно-орієнтованого програмування.

Інструменти, що використовувались для реалізації завдання: Microsoft Visual Studio 2019, Windows Forms, платформа .NET Framework, мови програмування C#.

В результаті створено програму, що має назву «Довідник меломана», яка є програмним забезпеченням для використання тими, хто захоплюються музикою. Програма має один додаток, який розбитий на дві підпрограми (користувача та адміністратора). Реалізовані наступні функції: реєстрація нового користувача в системі, перегляд існуючої музикальної бази, пошук музики за заданими параметрами. Для адміністратора додаються функції додавання/видалення інформації з бази, перегляд інформації користувачів. Реалізований запис даних до файлів.

ЗМІСТ

[Вступ 6](#_Toc532289188)

[1 Специфікація програми 7](#_Toc532289189)

[1.1 Інтерфейс користувача 7](#_Toc532289190)

[1.2 Окремі функції користувача](#_Toc532289191) 10

[2 Проектна специфікація 1](#_Toc532289192)4

[2.1 Об’єктна модель програми 1](#_Toc532289193)5

[2.2 Структура проекту  1](#_Toc532289194)5

[2.3 Об’єктна модель 1](#_Toc532289194)6

[2.4 Збереження та доступ до даних. Контроль версій](#_Toc532289194) 18

[3 Інструкція користувача](#_Toc532289195) 19

[3.1 Запуск програми  19](#_Toc532289194)

[3.2 Робота програми](#_Toc532289194) 19

[Висновки](#_Toc532289196) 26

[Перелік джерел посилання](#_Toc532289196) 27

[Додаток А Код програми 28](#_Toc532289197)

ВСТУП

Метою даного курсового проекту є створення спеціалізованого програмного забезпечення для довідника меломана. Реалізація проекту ґрунтується на принципах об’єктно-орієнтованого програмування, тому для роботи була обрана така мова програмування як С# та середа розробки Visual Studio 2019. За задачу було також поставлено створення графічного інтерфейсу користувача, що потребувало використання додаткового інструменту – Windows Forms.

Програма може бути використана абсолютно вільно будь-яким користувачем, який захоплюється музикою. Вона гарно підходить для роботи пов'язаної з пошуком і зберіганням інформації. Дані виводяться за запитом користувача в чітко структурованих списках, що дозволяють без зусиль сприймати, аналізувати і структурувати інформацію. Програма надає вільний і комфортний доступ до інформації.

**1 СПЕЦИФІКАЦІЯ ПРОГРАМИ**

* 1. Інтерфейс користувача

Для клієнтської версії додатку інтерфейс користувача складається з 4 вікон. При запуску програми з’являється перше - вікно входу. Можливий перехід на форму для реєстрації у системі. (Див. рис. 1.1 та 1.2)

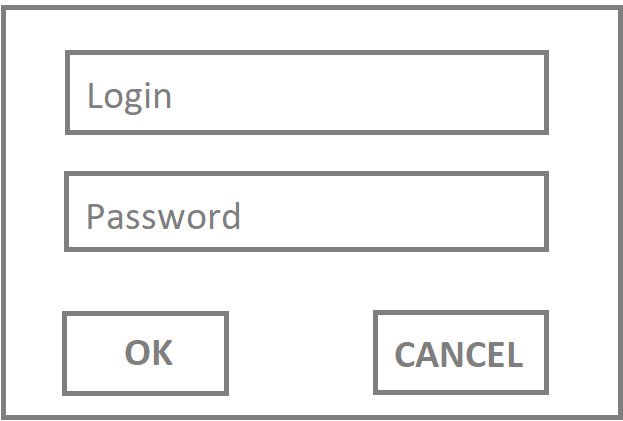


Рисунок 1.1 – Вікно входу

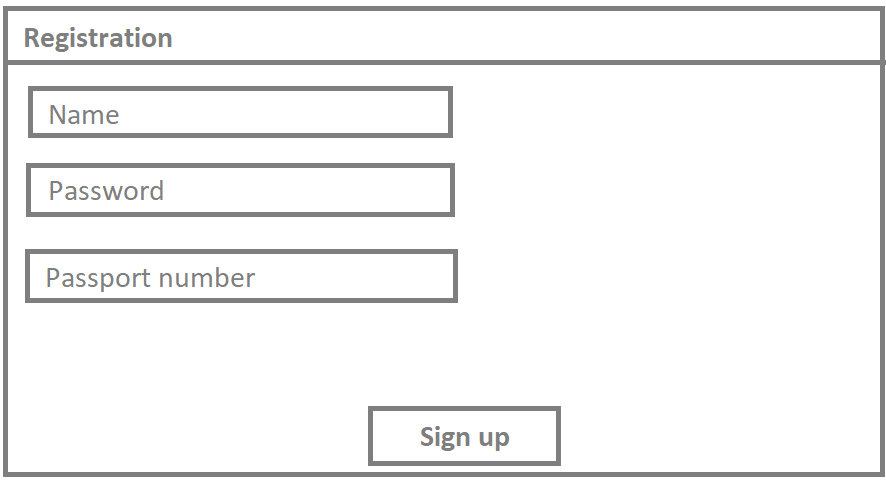


Рисунок 1.2 – Вікно реєстрації

Наступним вікном інтерфейсу користувача є основне вікно програми.

(Див. рис. 1.3)

З нього можливий перехід на вікно пошуку музики у базі. (Див. рис. 1.4)

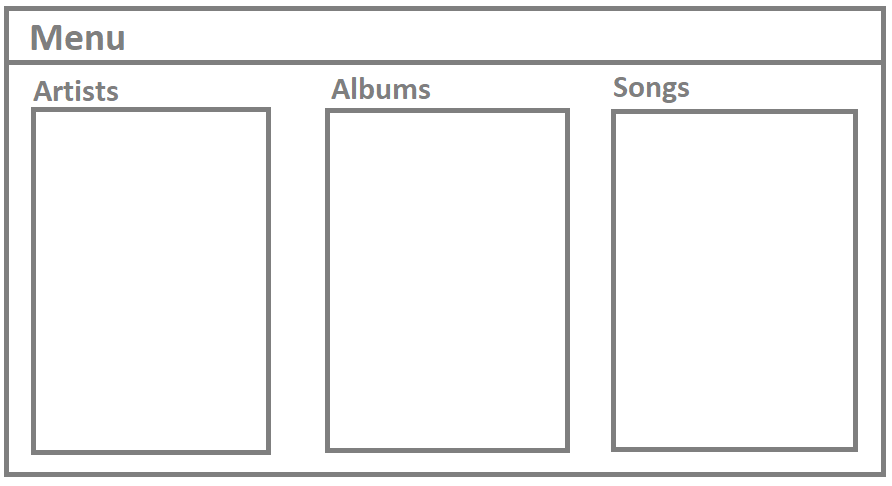


Рисунок 1.3 – Основне вікно

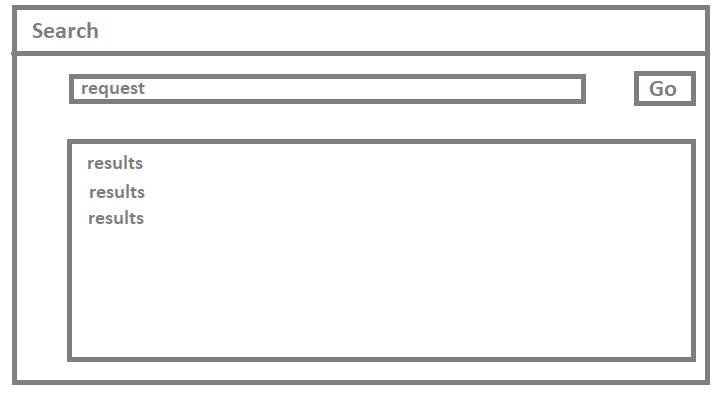


Рисунок 1.4 – Пошук музики

Для адміністраторської версії додатку інтерфейс складається з 6 вікон. При запуску програми відкривається перше – основне вікно програми. (Див. рис. 1.5)

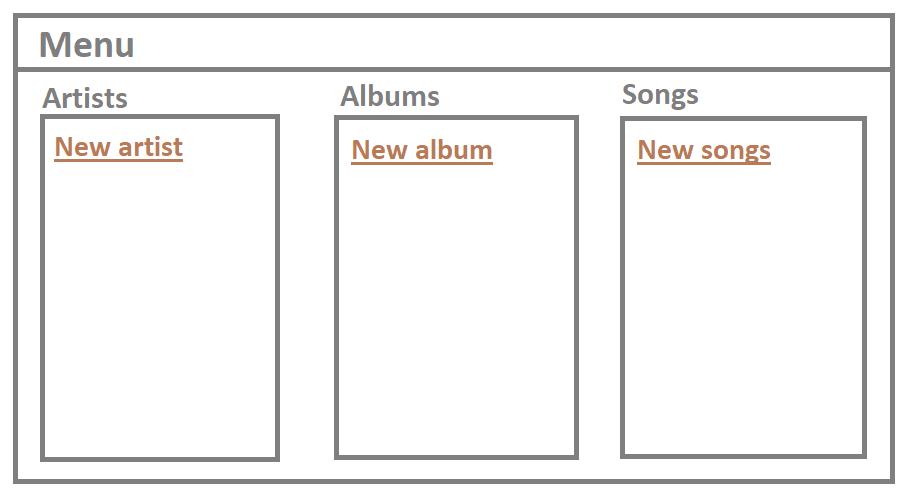


Рисунок 1.5 – Вікно адміністратора

З нього можливий перехід на наступні вікна:

а) Додати артиста (групу), додати альбом (диск), додати пісню (Див. рис. 1.6)

б) Пошук музики

в) Дивитись інформацію про користувачів (Див. рис. 1.7)

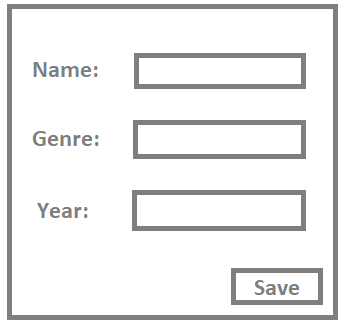


Рисунок 1.6 – Вікно для вводу інформації

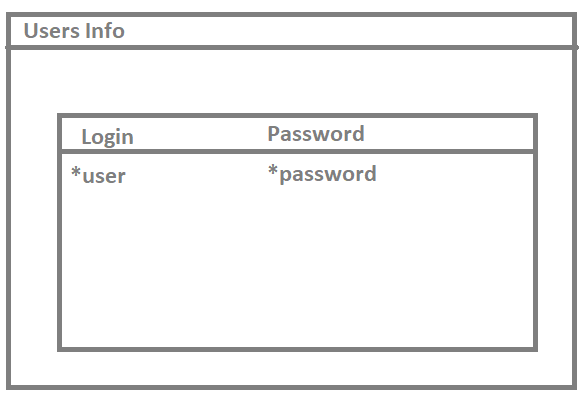


Рисунок 1.7 – Вікно з інформацією про користувача

* 1. Опис окремих функцій

Користувачу доступні такі функції:

* + 1. Вхід до системи за логіном та паролем

Користувач входить в програму за допомогою логіна і пароля.

Імена і паролі користувачів зберігаються в даних програми.

При правильному введенні пароля користувач потрапляє в меню юзера.

Основний сценарій

1. Користувач потрапляє на форму з авторизацією (рис. 1.1).

2. Користувач вводить логін і пароль і натискає кнопку "OK".

3. Якщо дані введені вірно - відбувається перехід в меню юзера.

Додатковий сценарій:

1. Якщо дані введені невірно - на формі введення з'являється повідомлення про помилку і можливий повторний вхід. Число спроб не обмежена.

2. Якщо користувач не зареєстрований, він може вибрати вікно «Реєстрація».

* + 1. Реєстрація у системі за логіном, паролем.

Меломан реєструється і вводить свої дані.

Основний сценарій:

1. Користувач вибирає вікно «Sing up».

2. Відкривається форма для заповнення даних про меломана.

3. Користувач заповнює форму.

4. Натискає кнопку «Sign up».

* + 1. Перегляд наявних груп, пісень, альбомів.

Сценарій:

1. На адмінпанелі у колонці артисти одним кліком вибираєте виконавця, у колонці альбоми з’являється всі його альбоми (диски).

2. На адмінпанелі у колонці альбоми одним кліком вибираєте альбом, у колонці пісні з’являється всі пісні з цього альбому.

* + 1. Перегляд додаткової інформації про групу, альбом, пісню.

Для цього необхідно зробити подвійний клік на будь-який вибраний альбом, групу чи пісню.

* + 1. Пошук музики

Основний сценарій:

1. Користувач вибирає вікно «Search music».

2. Відкривається форма для пошуку.

3. Користувач заповнює форму для введених даних.

4. Натискає кнопку «Go» та отримую результат свого запиту.

* + 1. Збереження даних

У процесі роботи користувач змінює дані, які знаходяться в оперативній пам'яті. Якщо дані змінені, але ще не введено постійної пам'яті, то перед закриттям програми користувач отримує пропозицію зберегти дані.

Адміністратору доступні всі функції, що і користувачу, а також ще додатково:

* + 1. Перегляд інформації про користувачів у базі.

1. На адмінпанелі вибираєте кнопку «Інформація користувачів».

2. Відкривається форма з даними про користувачів.

* + 1. Додавання одного артиста, альбом, пісні до бази

1. На адмінпанелі натискає кнопку Add.

2. Відкривається форма. Змінює поля форми.

3. Натискає кнопку «Зберегти».

* + 1. Видалення одного артиста, альбом, пісні з бази

1. Адміністратор вибирає артиста, альбом, пісню із загального переліку на адмінпанелі.

2. Потім натискає кнопку Delete.

* + 1. Можливість запису або зчитування інформації з файлу.

Якщо дані змінені, але ще не введено постійної пам'яті, то перед закриттям програми користувач отримує пропозицію зберегти дані.

1. **ПРОЕКТНА СПЕЦИФІКАЦІЯ**

2.1 Архітектура проекту

Програма буде створюватися на мові С # з використанням середовища розробки Microsoft Visual Studio Community 2019 на платформі .NET Framework 4.7.2, тип програми: Windows Forms.

 Курсовий проект передбачає об'єктно-орієнтований підхід, найпопулярнішим і зручним мовою програмування для даної мети є мова C #.

В процесі проектування вибір був між двома платформами .NЕТ Core і .NЕТ Framework. Вибір .NET Framework для написання даного курсового проекту заснований на зручному інтерфейсі і високий рівень розвитку даної платформи. Мінусом використання даної платформи є робота тільки на комп'ютерах з операційною системою Windows.

Для того, щоб реалізувати користувальницький графічний інтерфейс цього додатка буде використовуватися інтерфейс Windows Forms, який є однією з частин платформи .NET Framework.

2.2 Структура проекту

Проект, програмований в ході курсової роботи, буде розділений на 2 окремих додатки: перше - для адміна (адміністратора), друге - для меломана (користувач). Це рішення засноване на тому, що проект матиме два типи користувачів, для яких реалізуються різні функції і різний інтерфейс управління програмою.

Структура даного проекту буде виглядати наступним чином:

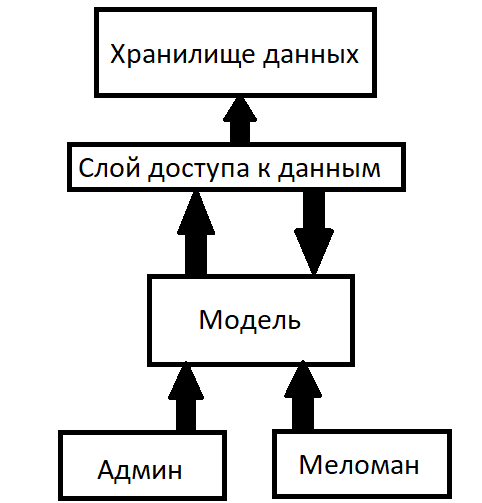


Рисунок 2.1 – Модель проекту

2.3 Об’єктна модель

2.3.1 Опис

База виконавців включає в себе список всіх доступних виконавців.

Виконавець включає в себе список всіх альбомів цього виконавця.

Альбом включають в себе список пісень з цього альбому.

Адмін додає виконавців, альбоми, пісні, вносить зміну, видаляє з бази (Робота з базою виконавців). Адмін додає, видаляє меломанів (Робота з базою користувачів).

Про виконавця відомо: назва, жанр, рік народження.

Про альбом відомо: назва, виконавець, рік випуску.

Про пісню відомо: назва, виконавець, альбом, рік випуску.

Про меломана відомо: логін (ім'я та прізвище), пароль.

До простих типів не належать:

База виконавців, виконавці, альбоми, база користувачів.

З цих типів відокремлюємо ті, які не містять колекцій (простий тип):

Пісня, меломан (користувач), адмін.

Перші призначені для користувача типи:

Пісня - це назва + жанр + рік.

Користувач - це логін + пароль.

Адмін - це логін + пароль

Решта типів:

Список користувачів - це колекція користувачів.

Список виконавців - це колекція виконавців.

Список альбомів - це колекція альбомів.

Список пісень - це колекція пісень.

* + 1. Методи

Основні методи:

- Реєстрація нових користувачів програми (меломанів).

- Додавання виконавців, альбомів, пісень.

- Видалення виконавців, альбомів, пісень.

- Пошук по назвою виконавців і пісень

- Перегляд інформації про вподобаних пісень.

Зв’язки між класами представлені на рисунку 2.2 та 2.3

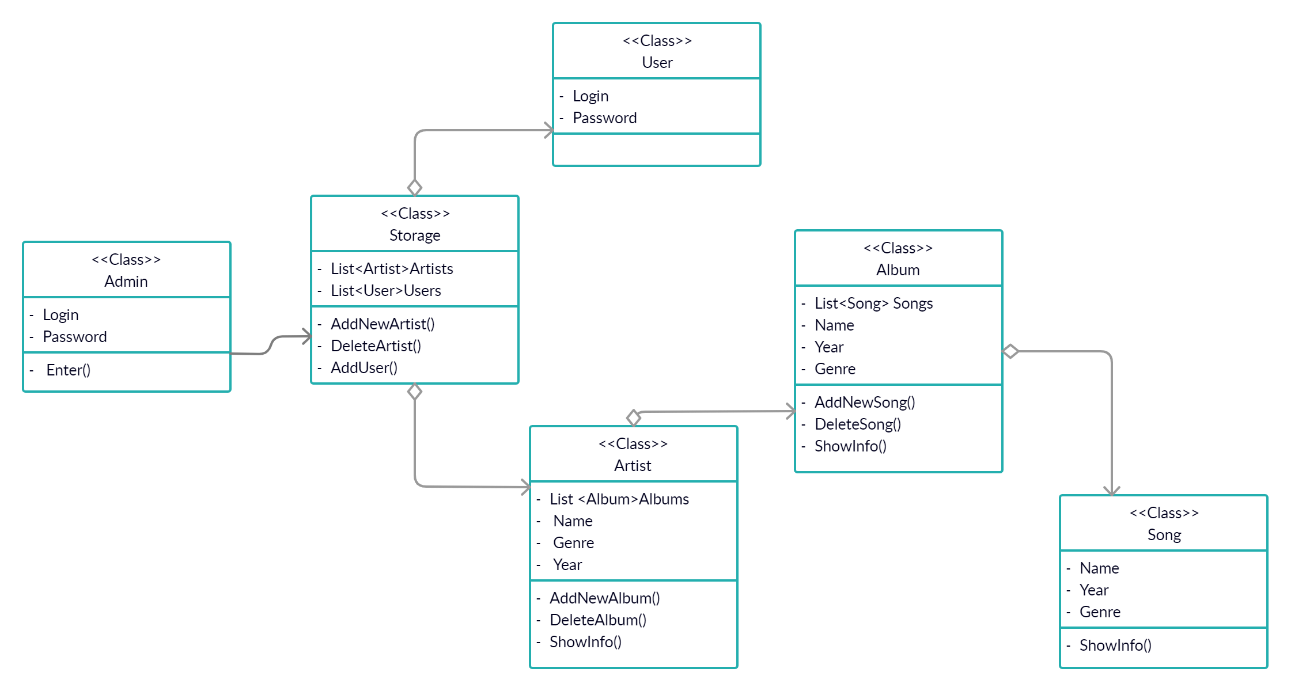


Рисунок 2.2 – Модель класів для адміна

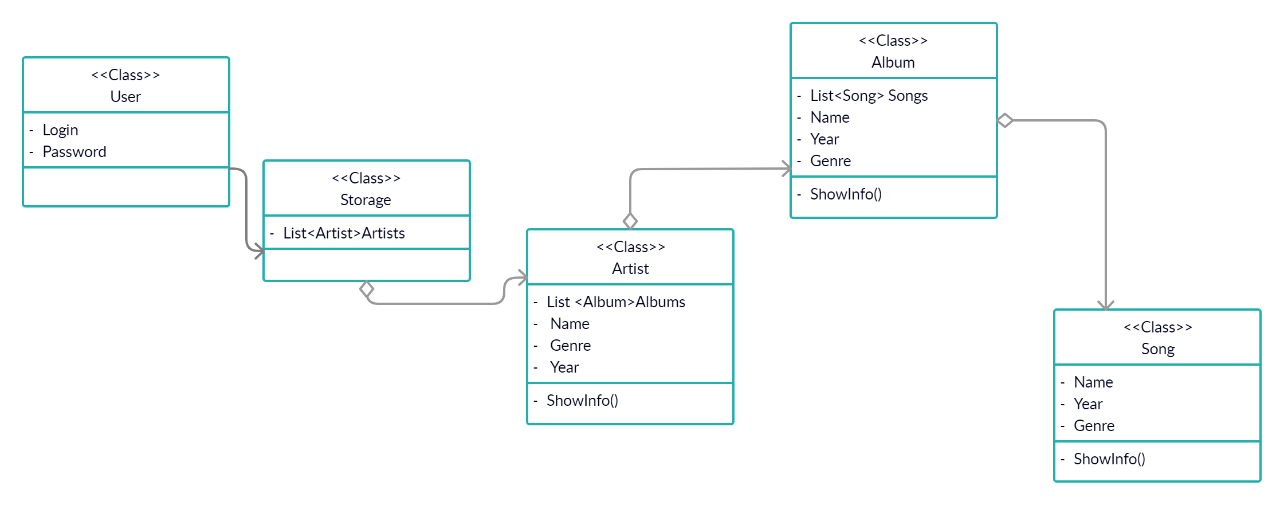


Рисунок 2.3 – Модель класів для юзера

* 1. Збереження та доступ до даних. Контроль версій

Дані переводяться з текстових в потік байтів за допомогою серіалізації з використанням вбудованої бібліотеки ВinaryFormatter. Захист даних в рамках даного проекту не розглядається.

Контроль версій проекту здійснюється за допомогою системи контроля версій GitHub.

1. **ІНСТРУКЦІЯ КОРИСТУВАЧА**

3.1 Запуск програми

Для запуску даної програми слід виконати такі дії:

1. Завантажити директорію з файлами курсового проекту.
2. У робочій директорії проекту запустити файл «MusicGuide.exe».
3. Програма запущена.
   1. Робота з програмою
      1. Додаток користувача

При запуску програми з’ являється вікно із входом до системи. Користувачу необхідно ввести свій логін та пароль, або натиснути на кнопку «Sing up», якщо він не є зареєстрованим у системі. У разі введення некоректних даних, поля підсвічуються червоним кольором. (Див. рис. 3.1 та 3.2)

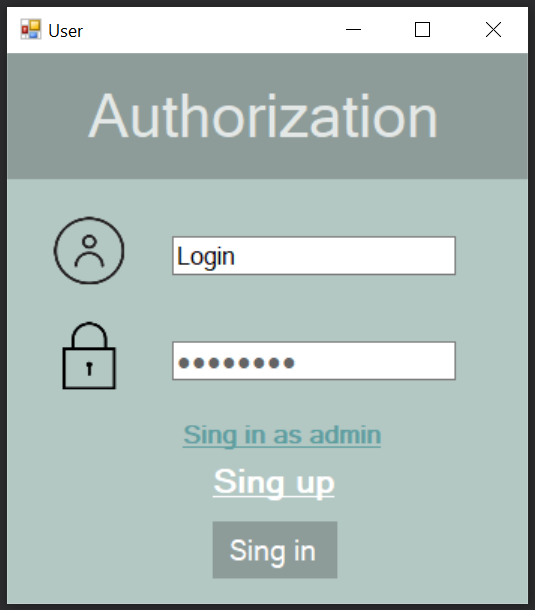


Рисунок 3.1 – Вікно входу

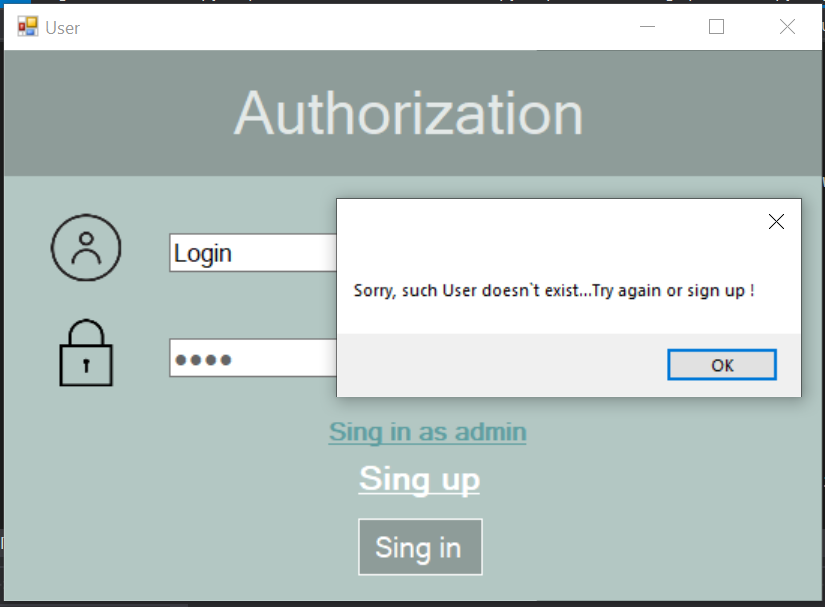


Рисунок 3.2 – Некоректні дані при вході

У разі реєстрації, відкривається форма для заповнення. Необхідно ввести логін, пароль та підтвердження паролю. Вимоги вказані справа від поля введення. У разі недотримання вимог з’являється діалогове вікно із повідомленням про помилку. (Див. рис. 3.3)

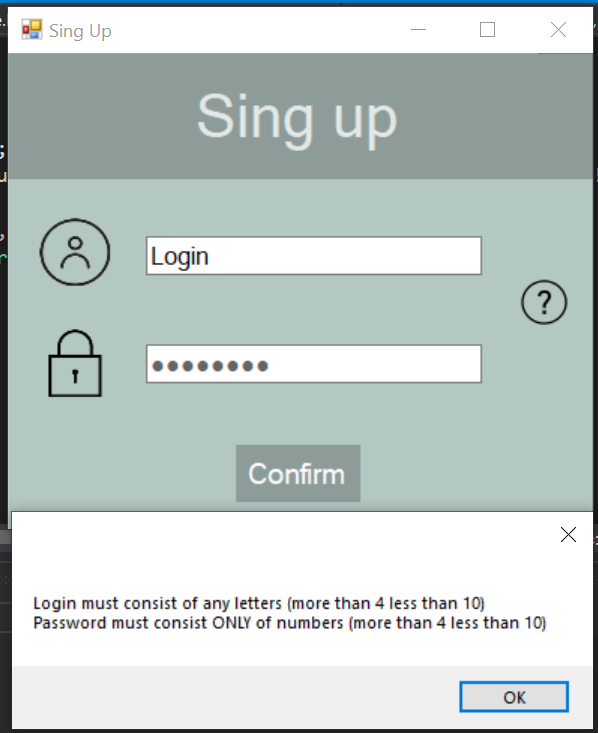


Рисунок 3.3 – Вікно реєстрації

Після входу/реєстрації відкривається вікно із списком наявних у базі виконавців. (Див. рис. 3.4) Потрібно одним кліком вибрати виконавця, зразу з’являться всі альбоми даного виконавця у вікні альбоми. Таким самим чином вибираємо альбом – з’являються пісні, у вікні пісні. (Див. рис. 3.5)

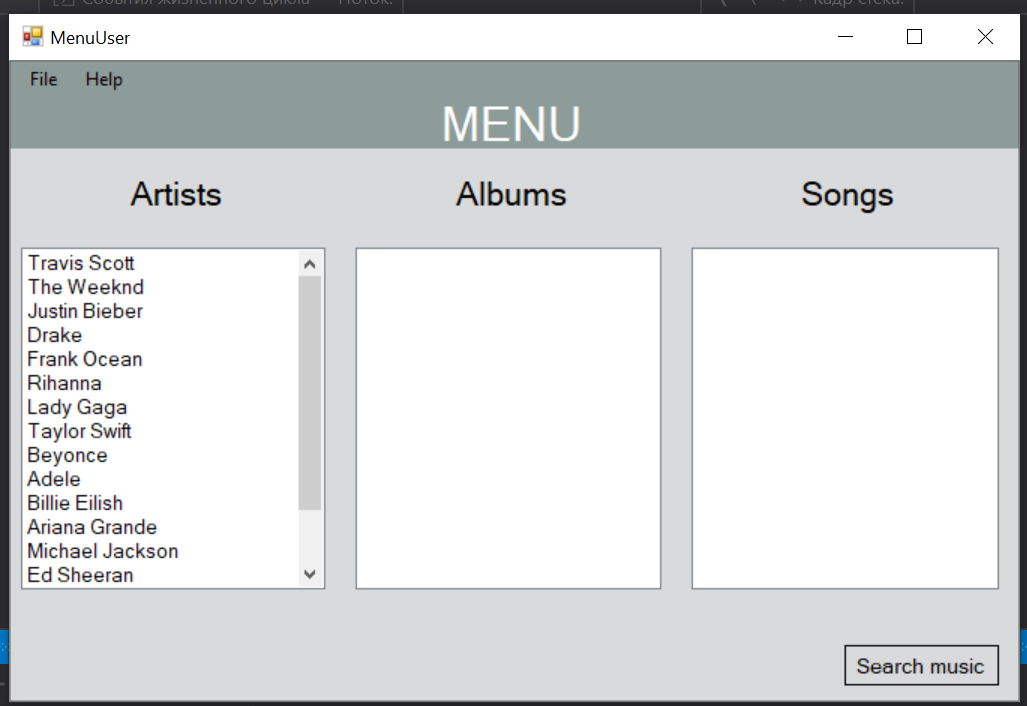


Рисунок 3.4 – Вікно основне

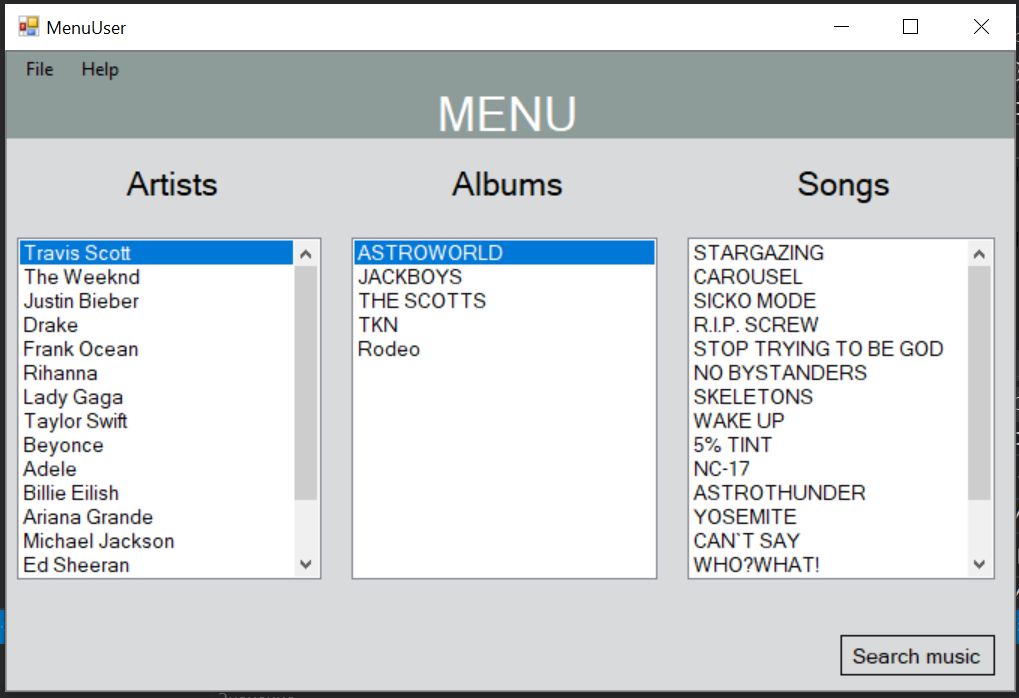


Рисунок 3.5 – Вибір виконавця та альбому

Для того, щоб переглянути додаткову інформацію про виконавця, альбом, пісню, треба зробити подвійний клік на бажаний елемент. (Див. рис. 3.6)

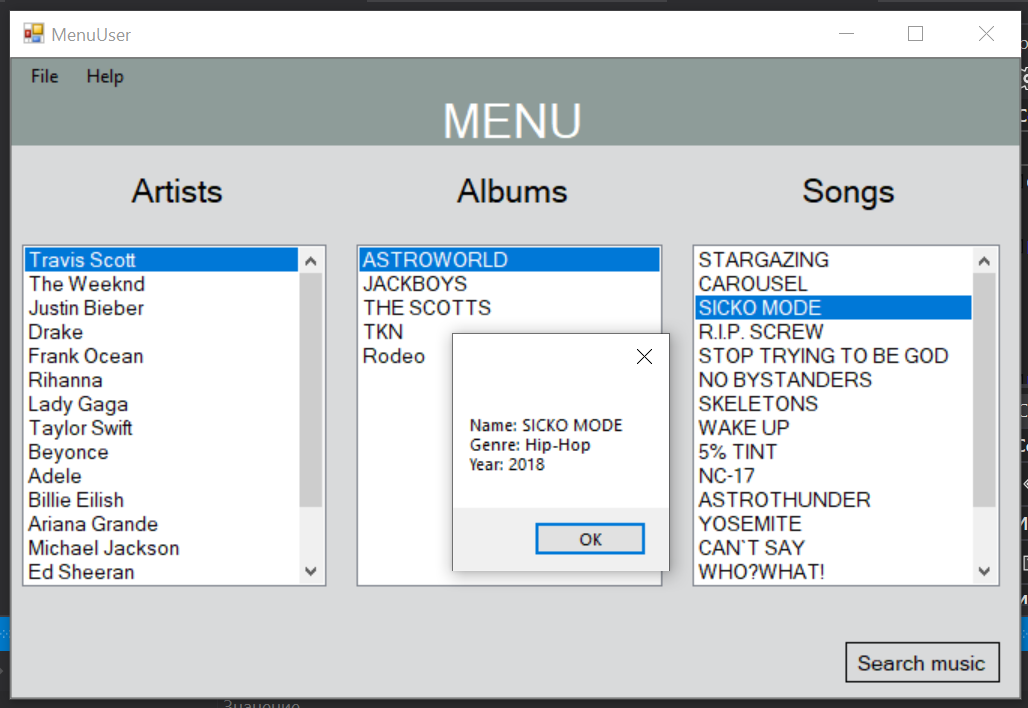


Рисунок 3.6 – Додаткова інформація

Вікно пошуку музики відкривається, після натискання кнопки Search music . У текстову форму вводимо данні для пошуку, а трохи нижче вибираємо параметри пошуку. Після цього натискаємо кнопку Go і в таблиці отримуємо результати пошуку.( Див. рис. 3.7)

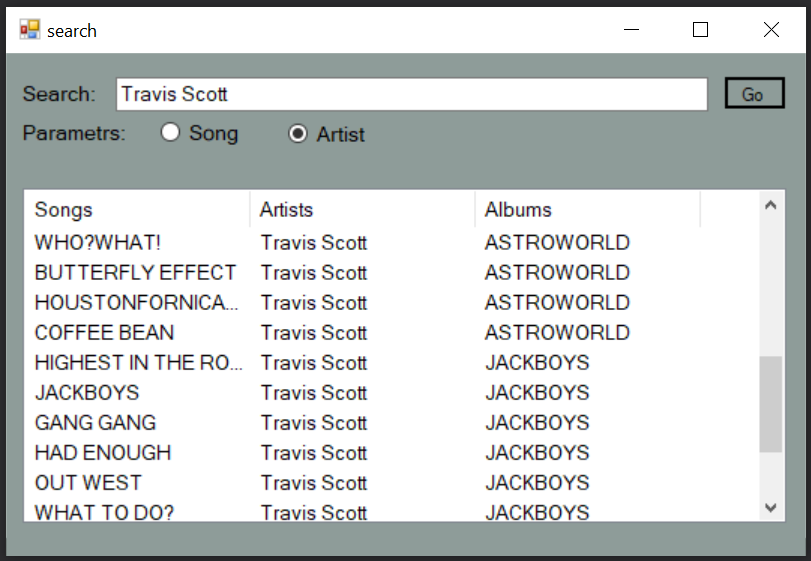


Рисунок 3.7 – Вікно пошуку музики

* + 1. Додаток для адміністратора

При запуску програми вибираємо Sing in as admin. Відкривається вікно для входу адміна, натискаємо sing in. З’являється основне вікно для адміністратора. (Див. рис. 3.8)

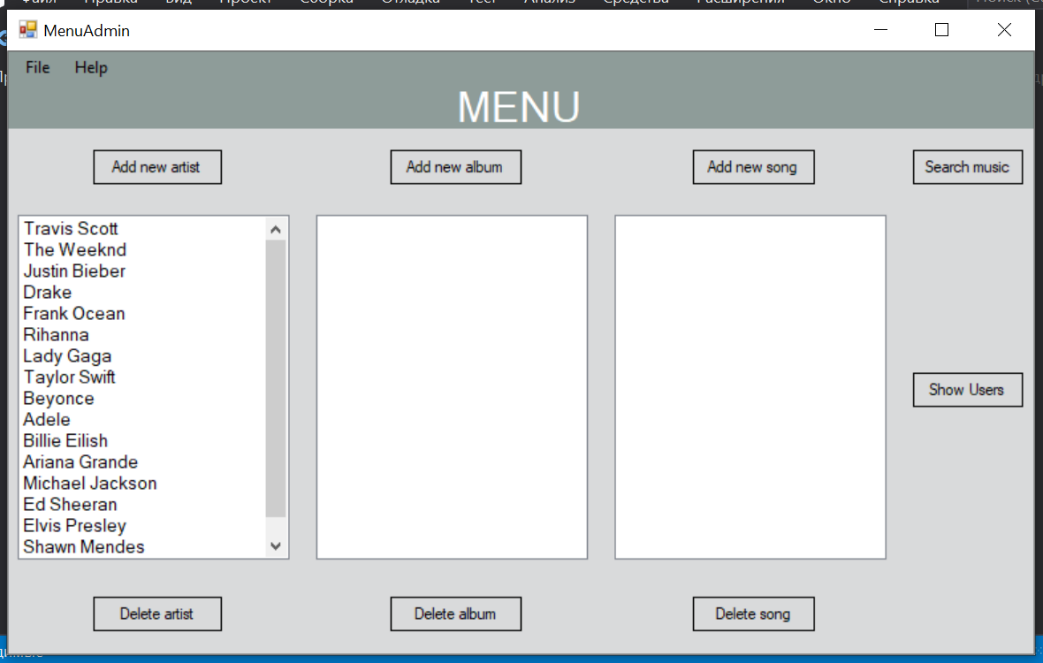


Рисунок 3.8 – Вікно для адміна

Додавання виконавця виконується натискання кнопки add new artist. Відкривається форма для додавання інформації. Після її заповнення можна її зберегти кнопкою save або натиснути хрестик для закриття форми без застосування змін. (Див. рис. 3.9)

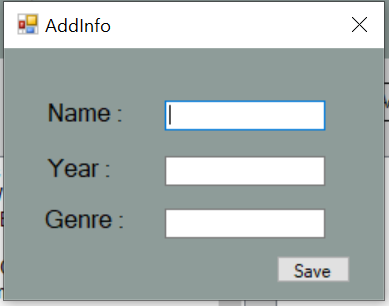


Рисунок 3.9 – Вікно для додавання інформації

Для видалення виконавця необхідно вибрати кліком виконавця зі списку та натиснути кнопку delete artist. (Див. рис. 3.10)

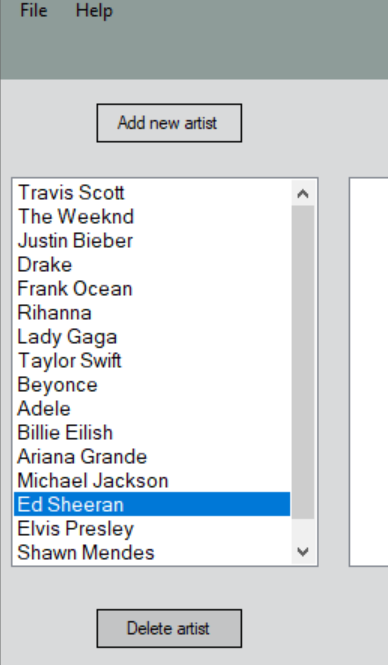


Рисунок 3.10 – Видалення виконавця

Інші можливості роботи з базою пісень та альбомів зрозумілі користувачеві за аналогом роботи з базою виконавців.

Для перегляду інформації про користувачів треба натиснути на головному меню кнопку users info. Відкриється вікно з інформацією. (Див. рис. 3.11)

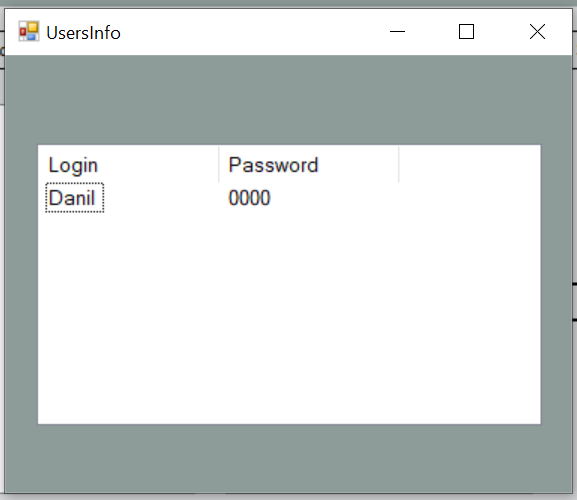


Рисунок 3.11 – Інформація про користувачів

Адміністратор може також записати або вивести інформацію з файлу про стан бази, натиснувши File -> Save/Load. (Див. рис. 3.12)

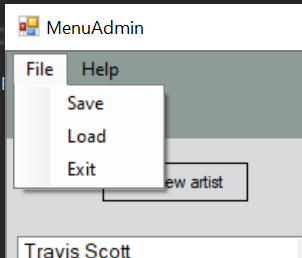


Рисунок 3.12 – Запис до файлу

**ВИСНОВКИ**

За результатами виконання курсового проекту було створену програмне забезпечення «Довідник меломана», який дозволяє регіструватись новим користувачам, переглядати та шукати інформацію про виконавців та їх пісні.. Відповідно адміністратор має змогу редагувати та оновлювати інформацію.

Під час розробки даного проекту були глибоко вивчені основи об’єктно- орієнтованого програмування, такі як наслідування та поліморфізм. Також було розглянуто особливості мови програмування С#, створення інтерфейсу за допомогою Windows Forms. У сукупності це допомогло зрозуміти тонкощі розробки програмних продуктів, від самого початку до кінця, розуміння дедлайнів та вміння налагодити власний тайм менеджмент.

Для подальшого розвитку запропонованої програми є декілька шляхів, такі як: розширення функціоналу, розробка швидших алгоритмів пошуку музики, відтворення більше інформації про артистів та пісні, під’єднання до бази даних.

**ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ**

1. Документація по C#. URL: https://metanit.com/sharp/ (дата звернення: 17.05.2020).
2. Документація по C# від Microsoft. URL: https://docs.microsoft.com/en-us/dotnet/csharp/ (дата звернення: 17.05.2020).
3. Эндрю Стиллмен, Дженнифер Грин. Изучаем C#. 3-издание. Санкт-Петербург. Питер, 2016 г.
4. Документація по Windows Forms. URL: https://docs.microsoft.com/en-us/dotnet/framework/winforms/ (дата звернення: 17.05.2020).
5. Эрик Фримен, Элизабет Робинсон. Head First. Паттерны проектирования. Санкт-Петербург. Питер, 2012 г.

**ДОДАТОК А**

Код програми

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MusicGuide.Models

{

//Базовий абстрактний клас, який містить основну інформацію

[Serializable]

public abstract class Info

{

//ім’я

public string Name { private set; get; }

//рік

public string Year { private set; get; }

//жанр

public string Genre { private set; get; }

//конструктор

public Info(string name = "", string year = "", string genre = "")

{

this.Name = name;

this.Year = year;

this.Genre = genre;

}

//виводить інформацію

public virtual void ShowInfo()

{

Console.WriteLine("Name : " + Name);

Console.WriteLine("Year : " + Year);

Console.WriteLine("Genre : " + Genre);

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MusicGuide.Models

{

//класс для пісень, наслідує клас info

[Serializable]

public class Song : Info

{

public Song(string name = "", string year = "", string genre = "") : base(name, year, genre) {}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MusicGuide.Models

{

//класс для альбомів, наслідує клас info

[Serializable]

public class Album : Info

{

// колекція пісень

public List<Song> songs = new List<Song>();

//конструктор

public Album(string name = "", string year = "", string genre = "") : base(name, year, genre) {}

//додати нову пісню

public void AddNewSong(Song song)

{

songs.Add(song);

}

//видалити пісню

public void DeleteSong(string name)

{

if (songs.Count == 0) Console.WriteLine("Empty");

else

{

foreach (Song element in songs)

{

if (element.Name == name) { songs.Remove(element); break; }

}

}

}

//метод перевизначений для виводу інформації

public override void ShowInfo()

{

base.ShowInfo();

Console.WriteLine("Number of songs in album : " + songs.Count);

if (songs.Count == 0) Console.WriteLine("Empty");

else

{

Console.WriteLine("List of songs : ");

foreach (Song element in songs)

{

Console.WriteLine(element.Name);

}

}

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MusicGuide.Models

{

[Serializable]

//клас для виконавця, наслідує info

public class Artist : Info

{

//колекція альбомів

public List<Album> albums = new List<Album>();

//конструктор

public Artist(string name = "", string year = "", string genre = "") : base(name, year, genre){}

//додати новий альбом

public void AddNewAlbum(Album album)

{

albums.Add(album);

}

//видалити альбом

public void DeleteAlbum(string name)

{

if (albums.Count == 0) Console.WriteLine("Empty");

else

{

foreach (Album element in albums)

{

if (element.Name == name) { albums.Remove(element); break; }

}

}

}

//вивід інформації

public override void ShowInfo()

{

base.ShowInfo();

Console.WriteLine("Number of artist albums : " + albums.Count);

if (albums.Count == 0) Console.WriteLine("Empty");

else

{

Console.WriteLine("List of albums : ");

foreach (Album element in albums)

{

Console.WriteLine(element.Name);

}

}

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MusicGuide.MusicGuideLibrary.Models

{

[Serializable]

//клас для користувача

public class User

{

//поле для логіна

public string Login { set; get; }

//поле для пароля

public string Password { set; get; }

//конструктор

public User(string login, string password)

{

Login = login;

Password = password;

}

}

}

using MusicGuide.Models;

using MusicGuide.MusicGuideLibrary.DAL;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MusicGuide.MusicGuideLibrary.Models

{

[Serializable]

//головний клас, для зберігання

public class Storage

{

//колекція виконавців

public List<Artist> artists = new List<Artist>();

//колекція користувачів

public List<User> users = new List<User>();

//додати виконавця

public void AddNewArtist(Artist artist)

{

artists.Add(artist);

}

//видалити виконавця

public void DeleteArtist(String artist)

{

if (artists.Count == 0) Console.WriteLine("Empty");

else

{

foreach (Artist element in artists)

{

if (element.Name == artist) { artists.Remove(element); break; }

}

}

}

//додати користувача

public void AddNewUser(User user)

{

users.Add(user);

}

//зберегти дані

public void Save()

{

new Dao(this).Save();

}

//завантажити дані

public void Load()

{

new Dao(this).Load();

}

}

}