

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «бакалавр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Інженерія програмного забезпечення»)

на тему:

«Відеопрогравач»

Виконав студент групи ПЗ-21-3
ДАЦКЕВИЧ Владислав Вікторович

Керівник роботи:
ЧИЖМОТРЯ Олена Геннадіївна

Рецензент:
ЄФРЕМОВ Юрій Миколайович

Житомир – 2025

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

«ЗАТВЕРДЖУЮ»

Завідувач кафедри інженерії
програмного забезпечення

Тетяна ВАКАЛЮК

«26» лютого 2025 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **ДАЦКЕВИЧ Владислав Вікторович**

Керівник роботи: **Чижмотря Олена Геннадіївна**

Тема роботи: **«Відеопрогравач»,**

затверджена Наказом закладу вищої освіти від **«14» лютого 2025 р., №61/с**

Вихідні дані для роботи: програмне забезпечення для відтворення відеофайлів,
теоретичні та прикладні аспекти конструювання відеоплеєрів

Консультанти з бакалаврської кваліфікаційної роботи із зазначенням розділів,
що їх стосуються:

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Чижмотря О.Г.	20.02.2025р.	20.02.2025р.
2	Чижмотря О.Г.	20.02.2025р.	20.02.2025р.
3	Чижмотря О.Г.	20.02.2025р.	20.02.2025р.

РЕФЕРАТ

Випускна робота бакалавра складається з комплексної програмної реалізації відеопрогравача, а також пояснювальної записки. Пояснювальна записка до кваліфікаційної роботи містить 58 сторінок, 29 ілюстрацій, 2 таблиці, 5 додатків та 30 літературних джерел.

Основною метою даної роботи є розробка функціонального та зручного відеопрогравача, який забезпечує високу якість відтворення відео, підтримку широкого спектру відеоформатів, швидкість роботи та інтуїтивно зрозумілий інтерфейс.

В роботі визначено ключові завдання, що пов'язані з розробкою відеопрогравача. Проведено аналіз існуючого інструментарію, подібного до розроблюваного продукту, а також наведено сценарії роботи програмного продукту. Представлено огляд загальної структури додатку, описано алгоритми взаємодії між окремими компонентами.

В рамках випускної роботи проводились експерименти по оптимізації роботи з базою даних та інтерактивними елементами інтерфейсу, в результаті яких були визначені оптимальні налаштування для швидкої та надійної роботи сервісу.

КЛЮЧОВІ СЛОВА: ВІДЕО, АУДІО, ВІДЕОПРОГРАВАЧ, МУЛЬТИМЕДІЙНІ ТЕХНОЛОГІЇ, ПАУЗА, ЯКІСТЬ.

					ІПЗ.КР.Б – 121 – 25 – ПЗ					
Змн.	Арк.	№ докум.	Підпис	Дата	Відеопрогравач Пояснювальна записка			Літ.	Арк.	Аркушів
Розроб.		В.В. Дацкевич								
Керівник		О.Г. Чижмоторя							3	58
								Житомирська політехніка, група ІПЗ-21-3		
Реценз.		Ю.М. Єфремов								
Затверд.		Т.А. Вакалюк								

ABSTRACT

The bachelor's thesis consists of a complex software implementation of a video player, as well as an explanatory note. The explanatory note to the qualification paper contains 58 pages, 19 illustrations, 2 tables, 5 appendices and 30 literary sources.

The main goal of this work is the development of a functional and convenient video player that provides high quality video playback, support for a wide range of video formats, speed of operation and an intuitive interface.

The paper defines the key tasks related to the development of a video player. An analysis of the existing toolkit, similar to the product under development, has been carried out, and scenarios of the software product's operation have also been given. An overview of the general structure of the application is presented, the algorithms of interaction between individual components are described.

As part of the final work, experiments were carried out to optimize the work with the database and interactive elements of the interface, as a result of which the optimal settings for fast and reliable operation of the service were determined.

KEY WORDS: VIDEO, AUDIO, VIDEO PLAYER, MULTIMEDIA TECHNOLOGIES, PAUSE, QUALITY.

					ІІЗ.КР.Б – 121 – 25 – ІІЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		5

ЗМІСТ

РЕФЕРАТ.....	3
ABSTRACT.....	5
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ.....	7
ВСТУП.....	8
РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ.....	10
1.1. Постановка задачі.....	10
1.2. Аналіз існуючого програмного забезпечення.....	12
1.3. Обґрунтування вибору інструментальних засобів	19
Висновки до першого розділу	22
РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА МОДЕЛЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ	24
2.1. Технічне завдання на розробку системи	24
2.2. Аналіз вимог до програмного продукту	25
2.3. Алгоритм роботи та стани програмної системи	29
2.4. Взаємодія об'єктів системи	30
Висновки до другого розділу.....	32
РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З СИСТЕМОЮ ДОДАТКУ	34
3.1. Порядок встановлення та налаштування системи.....	34
3.2. Опис основних функцій додатку	35
3.3. Структура інтерфейсу застосунку	39
3.4. Тестування роботи програмного продукту	46
Висновки до третього розділу	48
ВИСНОВКИ	49
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	51
ДОДАТКИ	53

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

AVI - Audio Video Interleave;

HD - High Definition;

MKV - Matroska Video;

MOV - QuickTime Movie;

MP3 - MPEG-1 Audio Layer III;

MP4 - MPEG-4 Part 14;

MPC-NC - Media Player Classic - Home Cinema;

UML - Unified Modeling Language;

VLC - VideoLAN Client;

ОС – Операційна система;

ПЗ – програмне забезпечення.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		7

ВСТУП

Комп'ютерні технології за останні 25-30 років зазнали кардинальних змін, перетворюючи комп'ютер з рідкісного і дорогого пристрою на звичайний, доступний та мобільний інструмент. Ці зміни торкнулися не лише конструктивних особливостей комп'ютера, але й суттєво розширили сферу його застосування. Сьогодні важко знайти галузь, в якій не використовуються інформаційні технології. Особливо активне впровадження цих технологій відбувається в різних сферах людської діяльності, зокрема у сфері мультимедіа та розваг, що робить розробку програмного забезпечення для медіа контенту важливим завданням.

Сучасні мультимедійні технології забезпечують високий рівень якості відтворення відео та аудіо контенту. Це вимагає створення спеціалізованого програмного забезпечення, яке б могло забезпечити ефективне та зручне відтворення медіафайлів. Відеоплеєри є невід'ємною частиною цього процесу, оскільки вони дозволяють користувачам насолоджуватися відеоконтентом з різних джерел та форматів.

Метою даної кваліфікаційної роботи є розробка функціонального та зручного відеоплеєра, здатного відтворювати відеофайли різних форматів. Для досягнення цієї мети необхідно вирішити низку завдань, серед яких дослідження існуючих рішень у сфері програмного забезпечення для відтворення відеоконтенту, аналіз їх переваг і недоліків, вибір оптимальних технологій та інструментів для реалізації проєкту, розробка алгоритмів обробки відеофайлів та тестування розробленого програмного продукту.

Основними завданнями дослідження є аналіз ступеня розробленості питання в науковій літературі та практиці, обґрунтування складових і алгоритмів створення відеоплеєра, розробка структури програмного забезпечення, порівняльна характеристика існуючих програмних рішень, а також розробка

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		8

рекомендацій щодо практичного застосування інструментарію для створення відеоплеєра.

Предметом дослідження є програмне забезпечення для відтворення відеофайлів, а об'єктом - теоретичні та прикладні аспекти конструювання відеоплеєрів. Методами дослідження є термінологічний аналіз, методи аналізу та синтезу, метод аналогії та метод дедукції.

Структурно кваліфікаційна робота складається з чотирьох частин. Перша частина містить дослідження області застосування проєктного рішення, огляд існуючих рішень і постановку завдання. Друга частина присвячена обґрунтуванню проєктних рішень щодо системи програми, опису алгоритмів вирішення поставлених завдань. Третя частина займається проєктуванням, реалізацією та тестуванням програмного продукту, а четверта містить інструкцію для користувача.

Методами дослідження є аналіз, синтез, теоретичне узагальнення та системний підхід для визначення генеральної проблеми. Графічні методи використовуються для наочного відображення результатів дослідження. Інформаційною базою дослідження є підручники, навчальні посібники, навчально-методичні посібники та електронні ресурси.

Практична значущість даної кваліфікаційної роботи полягає в розробці функціонального відеоплеєра, який забезпечує зручне та якісне відтворення відеофайлів різних форматів. Цей відеоплеєр може бути використаний як кінцевими користувачами, так і розробниками програмного забезпечення для інтеграції в складніші мультимедійні системи. Розробка відеоплеєра сприяє професійному розвитку, надаючи цінний практичний досвід у програмуванні та мультимедійних технологіях, що особливо корисно для освітніх закладів та розважальних систем. Крім того, використання власного програмного забезпечення може знизити витрати на придбання комерційних продуктів.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		9

РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

1.1. Постановка задачі

На сьогоднішній день, стрімкий розвиток інформаційних технологій та повсюдне використання інтернету значно збільшили попит на мультимедійні додатки, зокрема відеопрогравачі. Існуючі рішення часто не задовольняють усіх потреб користувачів, такі як висока якість відтворення, підтримка різних відеоформатів, швидкість роботи та інтуїтивно зрозумілий інтерфейс. Крім того, багато відеопрогравачів не забезпечують належну кросплатформенність, що обмежує їх використання на різних пристроях та операційних системах.

Метою даного дослідження є розробка відеопрогравача, який забезпечував би високу якість відтворення відео, підтримку широкого спектру відеоформатів, швидкість роботи та інтуїтивно зрозумілий інтерфейс, а також забезпечував би належну кросплатформенність.

Основні вимоги до відеопрогравача полягають в тому, щоб гарантувати зручне та якісне відтворення відео для користувача. Насамперед, програвач повинен підтримувати високу якість відео — HD, Full HD, 4K — і при цьому функціонувати без затримок, зависань чи спотворень зображення. Це важливо, особливо коли мова йде про перегляд фільмів чи відео у високій роздільній здатності [25].

Також важливою є підтримка різноманітних форматів відеофайлів. Програвач мусить "розуміти" такі популярні формати, як MP4, AVI, MKV, MOV тощо, щоб користувач не мав потреби встановлювати додаткові програми чи конвертувати відео у необхідний формат [5].

Ще одна важлива вимога — швидкість роботи. Програма повинна швидко відкривати відеофайли, не змушуючи користувача довго чекати, а саме відтворення має бути плавним, без підвисань.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		10

Окрім цього, велике значення має зручний та зрозумілий інтерфейс. Навіть недосвідчені користувачі повинні легко розібратися, як користуватися програвачем — запускати відео, перемотувати, змінювати гучність, налаштовувати якість тощо.

І на останок, кросплатформенність. Сучасний відеопрогравач має працювати на різних операційних системах — Windows, macOS, Linux — щоб його можна було використовувати на будь-якому приладі без обмежень.

Для досягнення мети дослідження передбачено виконання декількох ключових завдань. Передусім здійснюється аналіз наявних відеопрогравачів з метою виявлення їхніх функціональних спроможностей, переваг і вад. Це дає змогу сформулювати уявлення про сучасні підходи до реалізації подібних програмних продуктів.

Далі визначається архітектура програмного забезпечення, яка має забезпечувати високу продуктивність, масштабованість та зручність у подальшій зміні. Ретельне опрацювання структури системи є важливою умовою її ефективного функціонування.

Окрему увагу приділено створенню інтерфейсу користувача. Він повинен бути інтуїтивно зрозумілим, зручним у користуванні та відповідати потребам як досвідчених, так і початківців. Наступним кроком є впровадження підтримки різних відеоформатів. Це забезпечує сумісність програвача з найбільш поширеними типами мультимедійних файлів, що є необхідною умовою для його практичного застосування. Завершальним етапом є оптимізація продуктивності. Проводиться тестування роботи програми з метою виявлення можливих помилок, поліпшення швидкодії та підвищення загальної стабільності системи.

Для втілення поставлених завдань у рамках дослідження передбачається застосування комплексу методів та інструментальних засобів.

На першому етапі буде проведено аналіз літературних джерел та наявних програмних рішень. Зокрема, планується вивчення наукових статей, технічної документації, а також аналіз функціональних можливостей вже розроблених

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		11

відеопрогравачів з метою виявлення їх сильних рис і обмежень. Наступним етапом є проєктування архітектури програмного забезпечення. Особлива увага приділяється побудові структури, яка відповідатиме сучасним вимогам до продуктивності, гнучкості та масштабованості системи.

У процесі реалізації функціоналу буде застосовано сучасні мови програмування та відповідні бібліотеки, які дозволяють ефективно працювати з мультимедійними даними та забезпечити високу якість відтворення відео. Окремий етап складатиме тестування та оптимізація програми. Передбачається проведення різнопланових перевірок для оцінки стабільності, швидкодії та надійності роботи програвача. Занаслідками тестування буде здійснено необхідні вдосконалення.

Завершальним етапом є розробка інтерфейсу користувача. Він має бути зручним, інтуїтивно зрозумілим і відповідати сучасним принципам UI/UX-дизайну, що забезпечить комфортну взаємодію користувача з програмою.

Розробка відеопрогравача, який задовольнятиме сучасні вимоги користувачів щодо якості відтворення, підтримки різних відеоформатів, швидкості роботи, зручності використання та кросплатформенності. Це дозволить забезпечити комфортне та якісне відтворення відеоконтенту на різних пристроях та операційних системах.

1.2. Аналіз існуючого програмного забезпечення

Для розробки ефективного відеопрогравача необхідно ретельно проаналізувати існуючі рішення на ринку. Це дозволить виявити їх переваги та недоліки, визначити основні функціональні та технічні вимоги, а також знайти можливості для покращення. У даному підрозділі буде проведено огляд найпопулярніших відеопрогравачів, їх особливостей, а також будуть визначені критерії оцінки якості відеопрогравачів [24].

VLC Media Player – <https://www.videolan.org/vlc/>

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		12

VLC Media Player має простий та функціональний дизайн, орієнтований на зручність користувача [12]. Центральне вікно призначене для відтворення відео, внизу розташована панель інструментів з основними кнопками керування, такими як "Відтворити/Пауза", "Зупинити", "Перемотка", регулювання гучності, повноекранний режим та доступ до додаткових опцій (рис. 1.1). У верхній частині вікна знаходиться горизонтальне меню з розділами "Файл", "Перегляд", "Відтворення", "Аудіо", "Відео", "Інструменти" та "Довідка". Окреме вікно плейлиста дозволяє організовувати та відтворювати кілька файлів поспіль. Контекстне меню, яке викликається правою кнопкою миші, забезпечує швидкий доступ до додаткових налаштувань.

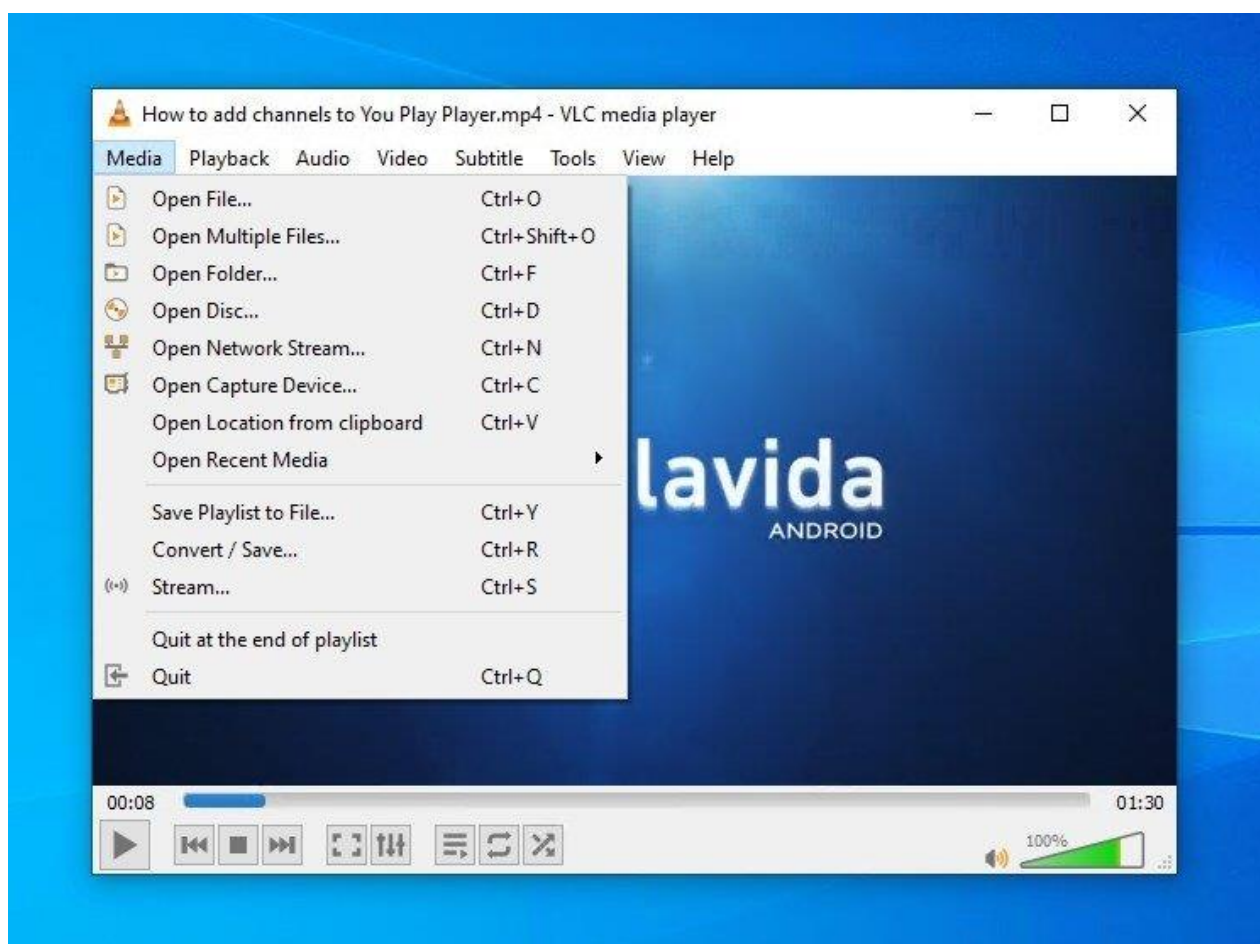


Рис. 1.1 – Вікно додатку VLC Media Player

Функціонал VLC Media Player відомий своєю багатofункціональністю. Програваач підтримує практично всі відео- та аудіоформати без необхідності встановлення додаткових кодеків, що забезпечує універсальність його

використання. Він працює на різних операційних системах, таких як Windows, macOS, Linux, а також на мобільних платформах iOS та Android, що робить його кросплатформеним. VLC також підтримує модулі та плагіни, дозволяючи розширювати функціональність програми. Додаткові функції включають відтворення потокового відео, запис екрану, конвертацію медіафайлів, підтримку субтитрів та аудіотреків, а також налаштування аудіо та відеоефектів. Висока продуктивність забезпечує плавне відтворення відео високої чіткості (HD, Full HD, 4K) без затримок (рис. 1.2). Загалом, VLC Media Player є потужним, універсальним та зручним інструментом для відтворення медіафайлів, що робить його одним із найкращих виборів для користувачів.

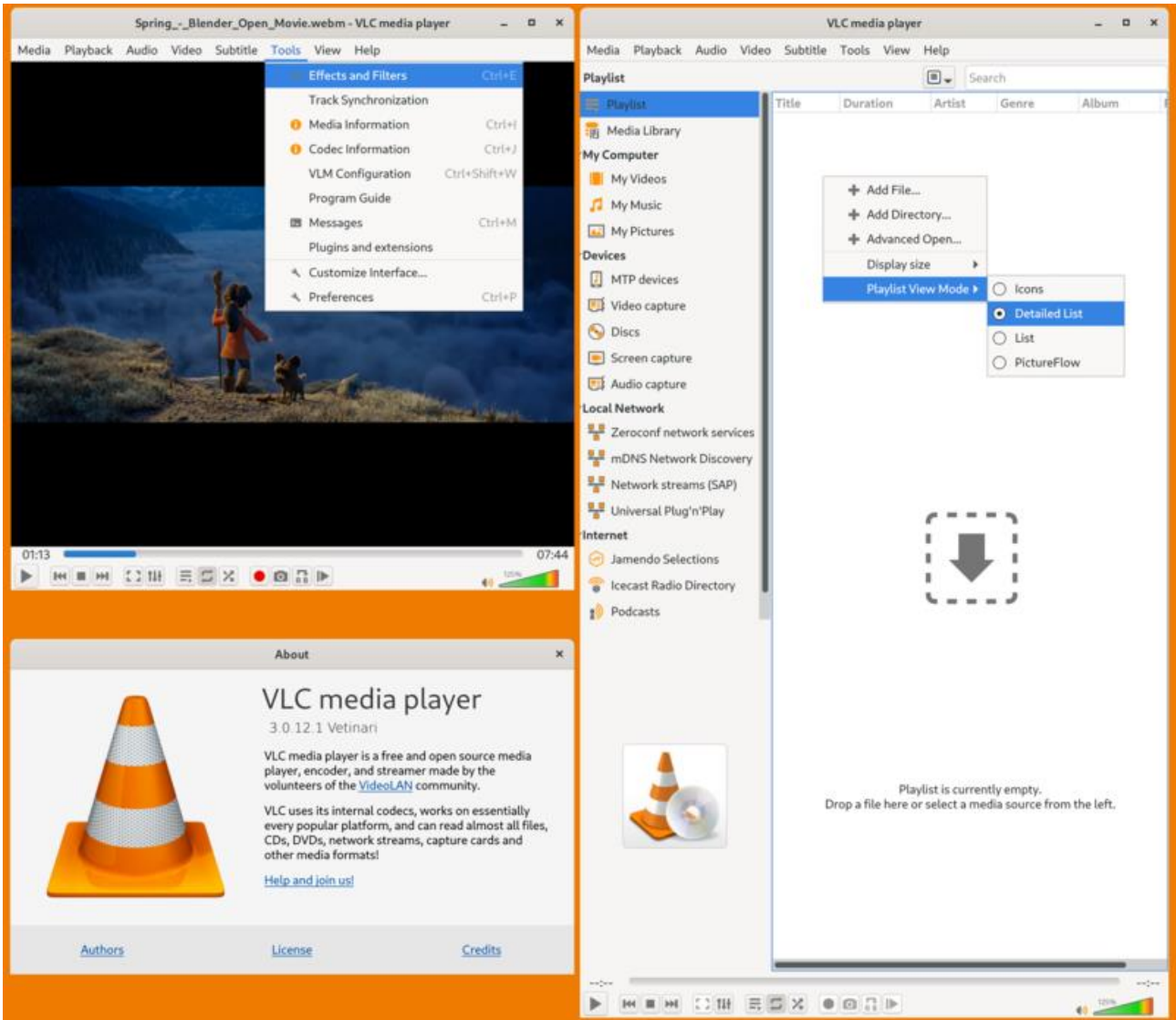


Рис 1.2 – Можливості VLC Media Player

PotPlayer – <https://potplayer.daum.net>

PotPlayer – це популярний мультимедійний програвач, відомий своєю потужністю та багатофункціональністю. Його дизайн є простим і зручним, що дозволяє користувачам легко орієнтуватися в інтерфейсі [8]. Центральне вікно відтворення займає більшу частину екрану, забезпечуючи великий та чіткий огляд відеоконтенту. Внизу розташована панель інструментів з основними елементами керування, такими як "Відтворити/Пауза", "Зупинити", "Перемотка", регулювання гучності та повноекранний режим (рис. 1.3). Меню налаштувань і додаткових опцій розташоване у верхній частині вікна, що забезпечує легкий доступ до різних функцій.

PotPlayer підтримує широкий спектр аудіо- та відеоформатів, таких як AVI, MKV, MP4, 3GP, FLAC, MP3 та багато інших, що робить його універсальним інструментом для відтворення медіафайлів. Програвач забезпечує високу продуктивність та плавне відтворення відео високої чіткості (HD, Full HD, 4K) без затримок, що робить його ідеальним для перегляду контенту високої якості.



Рис. 1.3 – Вікно додатку PotPlayer

Функціональність PotPlayer включає в себе підтримку субтитрів, аудіотреків, а також можливість налаштування відео- та аудіоефектів (рис. 1.4). Програваач дозволяє користувачам створювати та зберігати плейлисти, що зручно для організації великої кількості медіафайлів. Також PotPlayer підтримує функцію потокового відтворення відео з інтернету, що дозволяє дивитися онлайн-контент безпосередньо через програвач.

Однією з ключових особливостей PotPlayer є його розширюваність завдяки підтримці модулів та плагінів. Це дозволяє користувачам додавати нові функції та розширювати можливості програми відповідно до своїх потреб. Крім того, програвач підтримує запис відео з екрану, що може бути корисним для створення навчальних відео або запису онлайн-трансляцій.

Загалом, PotPlayer є потужним та багатофункціональним мультимедійним програвачем, який поєднує в собі зручний дизайн, високу продуктивність та широкий спектр функцій. Це робить його відмінним вибором для користувачів, які шукають надійний інструмент для відтворення та керування медіафайлами.

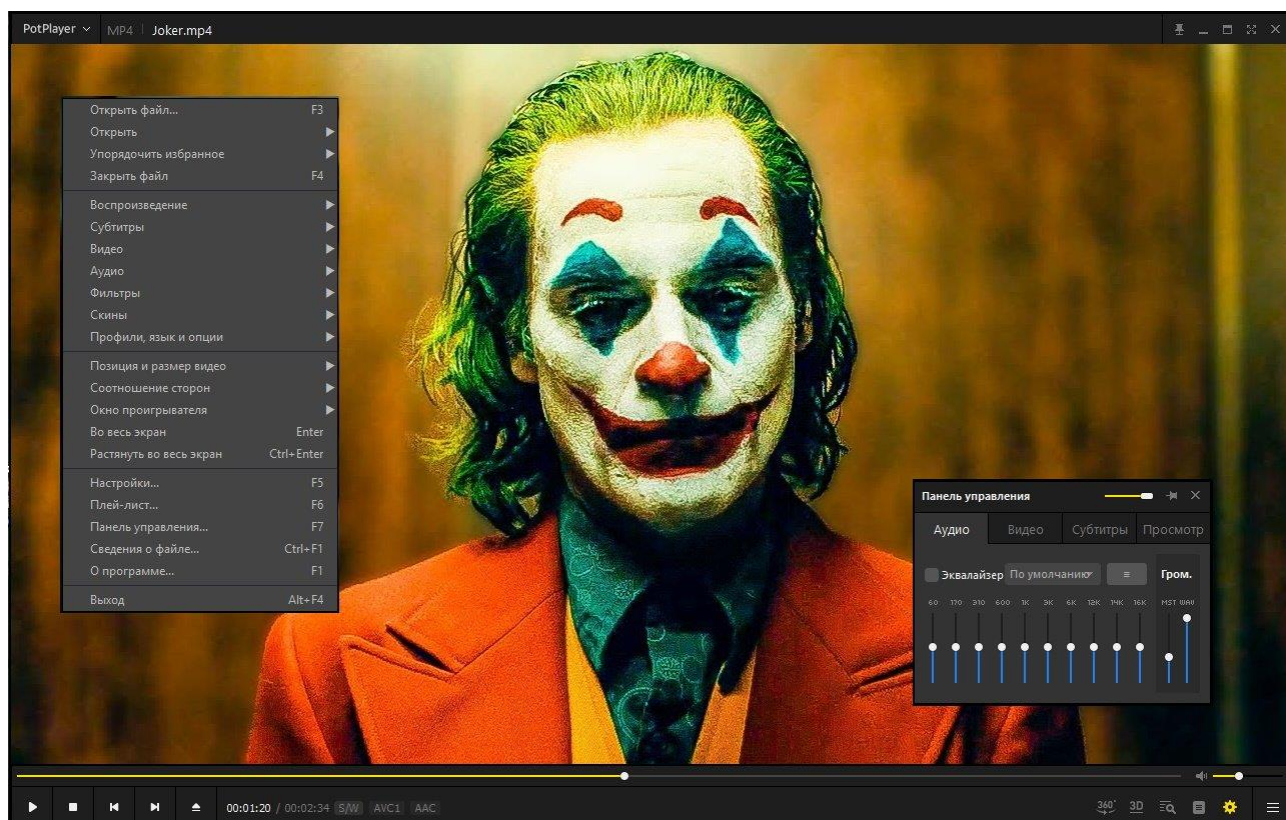


Рис. 1.4 – Можливості PotPlayer

Media Player Classic – <https://mpc-hc.org>

Media Player Classic - Home Cinema – це легкий та швидкий мультимедійний програвач з відкритим вихідним кодом, який надає користувачам простий і ефективний спосіб відтворення відео та аудіо. Дизайн програвача є мінімалістичним і нагадує ранні версії Windows Media Player, що робить його знайомим та зручним для багатьох користувачів [7].

Центральне місце в інтерфейсі займає велике вікно відтворення, що забезпечує чудовий перегляд відеоконтенту. Під вікном відтворення розташована панель інструментів з основними елементами керування: "Відтворити/Пауза", "Зупинити", "Перемотка вперед", "Перемотка назад", регулювання гучності та повноекранний режим. Верхнє меню програвача містить розділи "Файл", "Перегляд", "Відтворення", "Аудіо", "Відео", "Інструменти" та "Довідка", надаючи доступ до додаткових налаштувань та функцій (рис. 1.5).

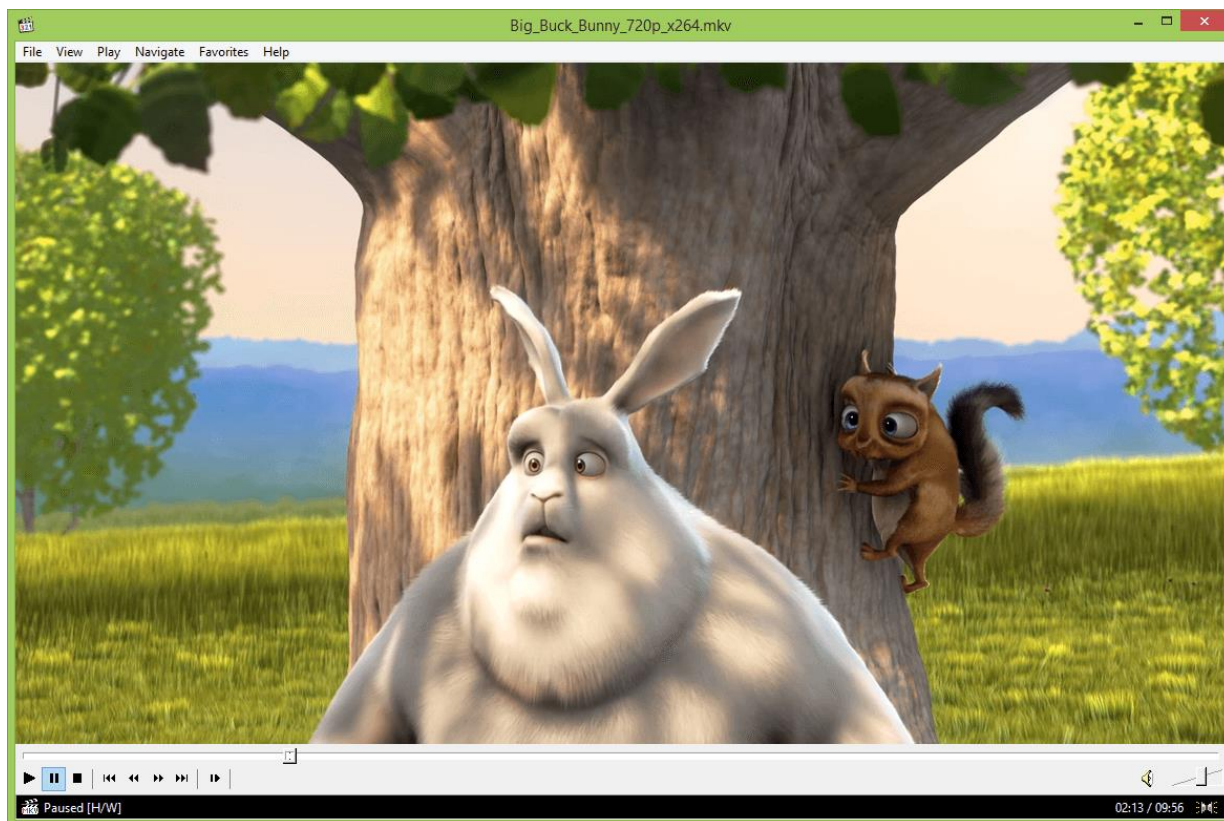


Рис. 1.5 – Вікно додатку Media Player Classic

MPC-NC підтримує широкий спектр відео- та аудіоформатів, включаючи AVI, MP4, MKV, FLAC, MP3 та багато інших, що робить його універсальним інструментом для відтворення медіафайлів. Програваач також підтримує субтитри та декілька аудіотреків, що особливо корисно для перегляду фільмів іноземною мовою або з перекладом (рис. 1.6).

Однією з головних переваг MPC-NC є його легкість та швидкість роботи. Програваач споживає мінімум системних ресурсів, що дозволяє йому працювати навіть на старих комп'ютерах без втрати продуктивності. Ця особливість робить його привабливим для користувачів з обмеженими апаратними можливостями.

MPC-NC також підтримує різні фільтри та декодери, що дозволяє покращити якість відтворення відео та аудіо. Користувачі можуть налаштовувати параметри відтворення відповідно до своїх уподобань, змінюючи яскравість, контрастність, насиченість кольорів та інші параметри відео.

Загалом, Media Player Classic - Home Cinema є надійним та ефективним мультимедійним програвачем, який поєднує в собі простий дизайн, високу продуктивність та широку підтримку форматів. Це робить його ідеальним вибором для користувачів, які шукають легкий і зручний інструмент для відтворення відео та аудіо.

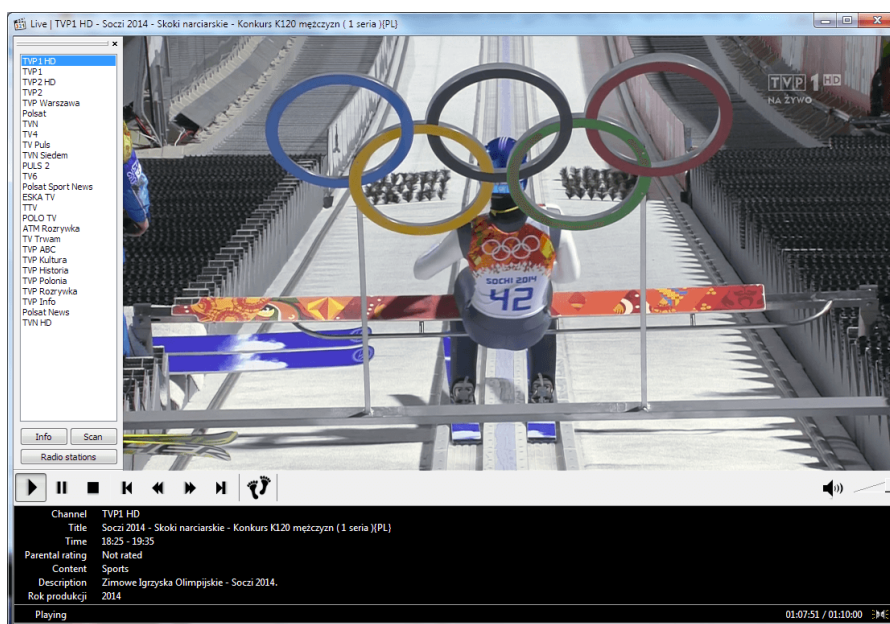


Рис. 1.6 – Вікно додатку Media Player Classic

Порівняння існуючих аналогів

Критерії	VLC Media Player	PotPlayer	Media Player Classic (MPC-NC)	FlikFlow
Підтримка форматів	Всі основні аудіо- та відеоформати	Всі основні аудіо- та відеоформати	Всі основні аудіо- та відеоформати	Всі основні аудіо- та відеоформати
Підтримка аудіотреків	+	+	+	+
Інтерфейс	Простий, але гнучкий	Проста та зручна	Зручний, налаштовуваний	Зручний, налаштовуваний
Збереження улюблених відео	–	+	+	+
Додавання альбомів з музикою	–	+	–	+

1.3. Обґрунтування вибору інструментальних засобів

Для розробки відеопрогравача було обрано набір інструментальних засобів, які забезпечують ефективне і зручне створення високоякісного програмного забезпечення. Зокрема, використовуються такі засоби:

- Мова програмування C#;
- Середовище розробки Visual Studio;
- Інструмент для прототипування Figma;
- Bunifu UI;
- Платформа Windows.

У процесі розробки програмного забезпечення для відеопрогравача було прийнято рішення використовувати мову програмування C#. Такий вибір обумовлений її зручністю, широкими можливостями та активною підтримкою з боку спільноти. C# добре підходить для створення складних застосунків, оскільки дозволяє писати структурований, зрозумілий код і повторно використовувати вже створені модулі. Це значно спрощує як розробку, так і подальше обслуговування програми.

Мова має багатий набір вбудованих інструментів і бібліотек, які допомагають реалізовувати графічний інтерфейс, роботу з мультимедіа та інтеграцію з мережею. Завдяки платформі .NET можна досягти високої швидкодії програми, стабільності її роботи та сумісності з іншими системами.

Ще однією важливою перевагою є велика кількість навчальних матеріалів, документації та прикладів. Це дозволяє швидко знаходити відповіді на питання, які виникають у процесі роботи, та ефективно розв'язувати технічні проблеми.

У якості середовища розробки для реалізації програмного забезпечення було обрано Visual Studio, що є одним із найпотужніших та найзручніших інструментів для роботи з мовою програмування C#. Однією з ключових переваг Visual Studio є її глибока інтеграція з платформою .NET. Це дозволяє безперешкодно використовувати функціональні можливості фреймворку, застосовувати доступні бібліотеки та компоненти, що значно пришвидшує процес розробки.

Інтерфейс середовища інтуїтивно зрозумілий та добре організований. Він спрощує навігацію по проєкту, написання та редагування коду, а також керування різними частинами програмного рішення.

Крім того, Visual Studio має потужні інструменти для налагодження коду. Серед них — покрокова відладка, перегляд і моніторинг значень змінних у реальному часі, а також інструменти для аналізу продуктивності, що дозволяють своєчасно виявляти й усувати помилки.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		20

Також важливою перевагою Visual Studio є велика кількість доступних розширень та плагінів. Вони дозволяють розширювати функціональність середовища відповідно до конкретних потреб проєкту та індивідуальних уподобань розробника.

Для створення прототипів та розробки користувацького інтерфейсу було обрано інструмент Figma [3]. Це сучасне веб-застосування, яке забезпечує ефективну співпрацю в команді завдяки можливості роботи в режимі реального часу. Усі учасники проєкту можуть одночасно переглядати, редагувати та коментувати дизайн, що значно підвищує швидкість та злагодженість розробки.

Інтерфейс Figma є простим та інтуїтивно зрозумілим, що дозволяє швидко створювати макети, прототипи та оперативно вносити зміни у дизайн відповідно до вимог проєкту чи побажань користувачів. Figma підтримує створення інтерактивних прототипів, що дає змогу якісніше візуалізувати сценарії користування програмою. Це дозволяє ще до початку програмної реалізації перевірити зручність та логіку інтерфейсу, виявити можливі недоліки та своєчасно їх скоригувати.

Для реалізації графічного інтерфейсу користувача в межах проєкту було використано набір компонентів Bunifu UI [1]. Цей набір елементів керування призначений для створення сучасних, привабливих та зручних інтерфейсів у середовищі розробки на базі Windows Forms.

Bunifu UI надає широкий вибір готових елементів керування — таких як кнопки, слайдери, таблиці, панелі та інші — що дозволяє значно прискорити процес розробки інтерфейсу та зменшити кількість рутинного коду. Завдяки готовим стилізованим компонентам розробник може зосередитися безпосередньо на функціональній частині застосунку. Компоненти цього фреймворку легко налаштовуються і мають адаптивні властивості, що дозволяє коректно відображати інтерфейс на екранах із різною роздільною здатністю та на різних пристроях. Таким чином, використання Bunifu UI забезпечує як зручність розробки, так і естетичну привабливість кінцевого продукту.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		21

У рамках реалізації програмного забезпечення для відеопрогравача було обрано платформу Windows, що зумовлено низкою практичних міркувань.

Передусім, Windows є найбільш поширеною операційною системою для персональних комп'ютерів. Її популярність серед користувачів забезпечує широке охоплення цільової аудиторії та гарантує сумісність із переважною більшістю сучасних комп'ютерних систем. Ця платформа пропонує розробникам широкий набір інструментів, бібліотек і технологій для створення мультимедійних застосунків, зокрема для роботи з відео та аудіо. Це дає змогу реалізувати необхідну функціональність без потреби у складних сторонніх рішеннях. Ще однією важливою перевагою Windows є її підтримка великої кількості апаратного забезпечення й периферійних пристроїв, що є критично важливим для стабільної та якісної роботи мультимедійного програмного забезпечення.

Таким чином, вибір інструментальних засобів, таких як мова програмування C#, середовище розробки Visual Studio, інструмент для прототипування Figma, бібліотека компонентів Bunifu UI та платформа Windows, забезпечує ефективну і якісну розробку відеопрогравача з сучасним інтерфейсом і широкими можливостями.

Висновки до першого розділу

Стрімкий розвиток інформаційних технологій і повсюдне використання інтернету збільшили попит на мультимедійні додатки, зокрема відеопрогравачі. Існуючі рішення часто не задовольняють усіх потреб користувачів, таких як висока якість відтворення, підтримка різних відеоформатів, швидкість роботи та інтуїтивно зрозумілий інтерфейс. Крім того, багато відеопрогравачів не забезпечують належну кросплатформенність, що обмежує їх використання на різних пристроях та операційних системах. Метою дослідження є розробка відеопрогравача, який забезпечував би високу якість відтворення відео, підтримку широкого

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		22

спектра відеоформатів, швидкість роботи та інтуїтивно зрозумілий інтерфейс, а також забезпечував би належну кросплатформенність.

Основні вимоги до відеопрогравача: висока якість відтворення відео без затримок і спотворень, підтримка популярних відеоформатів (MP4, AVI, MKV, MOV), швидкість роботи, простий та зрозумілий інтерфейс.

Для досягнення мети необхідно вирішити наступні завдання: аналіз існуючих відеопрогравачів, визначення архітектури програмного забезпечення, розробка інтерфейсу користувача, реалізація підтримки різних відеоформатів і оптимізація продуктивності.

На основі аналізу популярних відеопрогравачів (VLC Media Player, PotPlayer, Media Player Classic) було визначено, що необхідно враховувати підтримку всіх основних аудіо- та відеоформатів, простий та зручний інтерфейс, підтримку аудіотреків, збереження улюблених відео та додавання альбомів з музикою. Для розробки відеопрогравача обрано мову програмування C#, середовище розробки Visual Studio, інструмент для прототипування Figma, бібліотеку компонентів Bunifu UI та платформу Windows.

C# дозволяє створювати складні програми з високим рівнем організації коду, пропонує багатий набір бібліотек і фреймворків для розробки графічного інтерфейсу, роботи з мультимедіа та мережевими технологіями. Visual Studio забезпечує тісну інтеграцію з платформою .NET, має інтуїтивно зрозумілий інтерфейс і потужні засоби для відладки коду. Figma дозволяє швидко створювати прототипи і макети, працювати над проєктом спільно в режимі реального часу, підтримує створення інтерактивних прототипів. Bunifu UI забезпечує просте і швидке створення стильних і функціональних інтерфейсів. Платформа Windows, як найпоширеніша операційна система для настільних ПК, пропонує багатий набір інструментів і бібліотек для розробки мультимедійних додатків.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		23

РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА МОДЕЛЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1. Технічне завдання на розробку системи

Призначення системи відеопрогравача полягає у забезпеченні користувачів засобом для високоякісного відтворення відео та аудіо. Відеопрогравач має надавати такі основні функції: відтворення відеофайлів високої чіткості (HD, Full HD, 4K) без затримок та спотворень, підтримка широкого спектра популярних відеоформатів (MP4, AVI, MKV, MOV), швидке завантаження відеофайлів і плавне відтворення без затримок.

Передусім, система повинна забезпечувати високу якість відтворення відео, включаючи підтримку форматів високої роздільної здатності. Однією з ключових характеристик є також сумісність із різними відеоформатами, що дозволяє користувачеві переглядати більшість поширених типів відеофайлів без додаткових конвертацій чи встановлення сторонніх кодеків. Програмне забезпечення має працювати швидко та стабільно — відеофайли повинні завантажуватись оперативно, а саме відтворення має бути плавним, без затримок і збоїв.

Окрему увагу приділено інтерфейсу користувача, який має бути інтуїтивно зрозумілим і містити основні елементи керування: запуск, пауза, перемотування, регулювання гучності тощо. Додатково передбачається можливість створення плейлистів і додавання музичних альбомів, що розширює функціональність програми та підвищує її зручність. Також передбачається функція збереження улюблених відео, що дозволяє користувачам формувати персоналізовані добірки та швидко отримувати до них доступ у майбутньому.

Програмний продукт повинен мати наступні вимоги до інтерфейсу. Інтерфейс повинен бути інтуїтивно зрозумілим і легким у навігації навіть для недосвідчених користувачів. Основні елементи керування, такі як кнопки "Відтворити/Пауза", "Зупинити", "Перемотка вперед/назад", регулювання гучності та

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		24

повноекранний режим, повинні бути легко доступні. Інтерфейс має забезпечувати зручну організацію плейлистів, включаючи створення, редагування та відтворення. Дизайн інтерфейсу має бути мінімалістичним та естетичним, з акцентом на зручність та функціональність. Інтерфейс повинен адаптуватися до різних розмірів екрану та роздільної здатності, забезпечуючи зручність використання на різних пристроях.

2.2. Аналіз вимог до програмного продукту

Розглянемо високорівневі вимоги, яким повинен володіти відеопрогравач, щоб мати можливість безперервно виконувати покладені на нього функції, тим самим відповідно задовольнивши всі зазначені стандарти, специфікації та інші формальні документи. Зазначені умови, яких повинен дотримуватись користувач для роботи з даною інформаційною системою та вказані основні ділові переваги. Приведені можливості системи, обмеження та показники якості.

Бізнес-вимоги: основна мета проєкту полягає у створенні програмного продукту, який надаватиме користувачам доступ до повної та структурованої інформації про функціонування системи відеопрогравача, його можливості та переваги.

Проєкт спрямований на розробку сучасного відеопрогравача, який відповідатиме актуальним вимогам користувачів. Передбачається забезпечення високої якості відтворення відео, підтримки широкого спектра відеоформатів, а також швидкої та стабільної роботи застосунку.

Особливу увагу приділено створенню інтуїтивно зрозумілого та зручного інтерфейсу, що дозволить користувачам легко взаємодіяти з програмою незалежно від рівня технічної підготовки. Таким чином, реалізація цього програмного рішення має сприяти підвищенню загального рівня комфорту при перегляді відеоконтенту на персональних комп'ютерах.

Вимоги користувачів: серед основних вимог, які користувачі висувають до сучасного відеопрогравача, передусім виділяється якість відтворення.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		25

Очікується, що програма забезпечуватиме стабільне й бездоганне відтворення відео у високій роздільній здатності — включаючи HD, Full HD та 4K — без затримок, ривків або спотворень зображення. Не менш важливою є підтримка широкого спектра відеоформатів. Користувачі хочуть мати змогу відкривати найпопулярніші типи файлів, такі як MP4, AVI, MKV, MOV та інші, без необхідності встановлення додаткових кодеків чи сторонніх програм.

Швидкодія також залишається критично важливим фактором. Відеопрогравач має швидко реагувати на дії користувача, оперативно відкривати файли та забезпечувати плавне відтворення навіть на комп'ютерах із середніми технічними характеристиками. Значну роль відіграє й інтерфейс користувача — він повинен бути простим, зрозумілим і зручним у використанні, щоб ним могли ефективно користуватися як досвідчені, так і нові користувачі без технічної підготовки. Важливою є можливість зручної навігації у відео — передбачається наявність інструментів для перемотування, зупинки, продовження відтворення з місця зупинки та регулювання гучності. Ще одна очікувана функція — це створення та збереження плейлистів, що дозволяє упорядковувати відеоконтент, формувати добірки та швидко повертатися до улюблених файлів.

Нефункціональні вимоги: у межах реалізації проєкту визначено низку нефункціональних вимог, які мають забезпечити зручність, продуктивність і надійність роботи відеопрогравача. Насамперед ідеться про сприйняття та зручність для користувача. Система повинна бути простою у використанні й не вимагати тривалого навчання: звичайному користувачеві має вистачити 10–15 хвилин для ознайомлення з основними функціями, а досвідченому — близько 5 хвилин. Інтерфейс програми має бути інтуїтивно зрозумілим і не вимагати спеціальних технічних знань. Крім того, важливою вимогою є швидкодія: час відповіді системи на стандартні запити не повинен перевищувати 5 секунд, а на складніші — не більше 10 секунд.

Ще одним важливим аспектом є продуктивність. Відеопрогравач повинен забезпечувати швидке завантаження файлів незалежно від їхнього формату чи

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		26

розміру, а також мінімізувати час запуску програми та початку відтворення. Це сприяє покращенню загального користувацького досвіду та зменшує ризик технічних ускладнень під час роботи з програмою.

Для підтримки високих стандартів безпеки у клієнта потрібно виключити деякі застарілі браузері:

- Windows: Windows 10 або новіша версія (64-бітова).
- macOS: macOS 10.14 Mojave або новіша версія.
- Linux: Ubuntu 18.04 або новіша версія, або інша дистрибуція з підтримкою бібліотек для мультимедійних додатків.

Антивимоги: передусім, відеопрогравач не повинен використовувати або розповсюджувати інформацію, що суперечить нормам моралі чи чинному законодавству. Також заборонене відображення недостовірної або потенційно шкідливої для психіки користувачів інформації, зокрема контенту, що може викликати дезінформацію, психологічний дискомфорт або стати джерелом пропаганди.

Програмне забезпечення не повинно містити рекламних матеріалів, які не мають безпосереднього відношення до його тематики або функціонального призначення. Такий підхід спрямований на забезпечення безпеки, надійності та комфортного користувацького досвіду.

Підсумовуючи, побудуємо діаграму варіантів використання (Use Case Diagram).

Діаграма варіантів використання (Use Case Diagram) є важливим інструментом для візуалізації взаємодії між користувачами (акторами) та системою. Вона показує, як різні користувачі взаємодіють з відеопрогравачем, і які функції система надає [14].

Діаграма варіантів використання в нотації UML представлена на рисунку 2.1.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		27

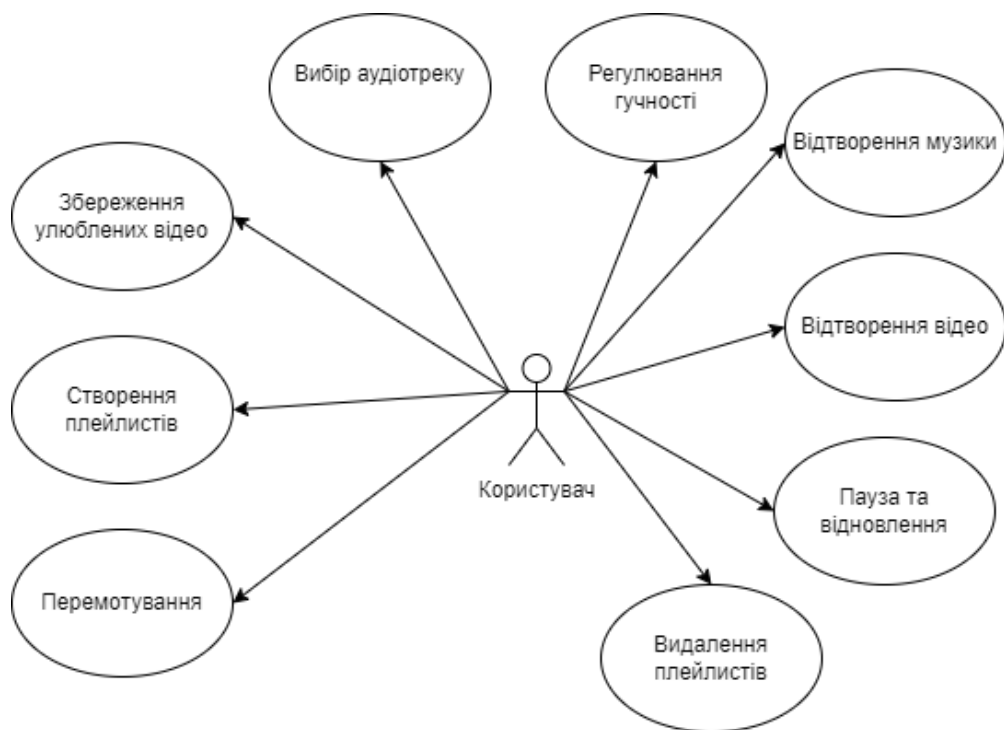


Рис 2.1 – Діаграма варіантів використання

Нижче наведено опис діаграми використання для відеопрогравача з участю користувача.

1. Основні актори:

- Користувач - основний актор, який взаємодіє з відеопрогравачем.

2. Основні варіанти використання:

Функціональність відеопрогравача охоплює низку основних сценаріїв використання, які забезпечують зручність і повноту взаємодії користувача з програмою.

Першочерговою можливістю є відкриття відеофайлів. Користувач може завантажувати для перегляду мультимедійні файли різних форматів, зокрема MP4, AVI, MKV, MOV та інші. Після завантаження відео користувач має змогу керувати його відтворенням — запускати, зупиняти або ставити на паузу перегляд.

Передбачена функція перемотування відео вперед або назад, яку можна реалізувати за допомогою спеціального повзунка або відповідних кнопок інтерфейсу. Також користувач має змогу змінювати рівень гучності за

допомогою регулятора звуку, що забезпечує індивідуальне налаштування аудіосупроводу.

Однією з важливих функцій є можливість перегляду відео в повноекранному режимі, що дозволяє зосередитися на візуальному контенті без відволікаючих елементів інтерфейсу.

Крім базових можливостей, система також дозволяє зберігати відеофайли до списку улюблених для зручного та швидкого доступу в майбутньому. Користувач може створювати власні плейлисти для організації відеоконтенту за темами або вподобаннями, а також видаляти ті плейлисти, які більше не актуальні.

Окрім відео, програвач підтримує відтворення аудіофайлів. Для аудіо реалізовано стандартні функції керування — відтворення, пауза, зупинка, перемотка та регулювання гучності — що розширює функціональність програми і дозволяє використовувати її також як музичний плеєр.

Діаграма варіантів використання є графічним інструментом, що ілюструє функціональність системи та її взаємодію з користувачами.

2.3. Алгоритм роботи та стани програмної системи

Діаграма діяльності в UML є інструментом для опису процесів і робочих потоків в системі. Вона ілюструє, як система виконує різні завдання, і показує взаємодію між різними етапами процесу [16].

Діаграма діяльності для відеопрогравача починається з натискання кнопки "Увімкнути" користувачем, що ініціює запуск програвача. Програвач виконує ініціалізацію системи і переходить до вибору відеофайлу. Після вибору програвач перевіряє формат відео. Якщо формат підтримується, завантажуються метадані відео. Якщо ні, відображається повідомлення про помилку.

При успішному завантаженні метаданих користувач натискає кнопку "Відтворити", і відео починає відтворюватися. Програвач дозволяє управляти

відтворенням (пауза, відновлення, перемотка, зупинка) та відображає інформацію про відео (час, заголовок).

Коли відео закінчується, користувач може вибрати відтворення наступного відео або завершити перегляд. Користувач натискає кнопку "Вимкнути", програвач зберігає налаштування (якщо потрібно) і закривається. Діаграма закінчується (рис. 2.2).

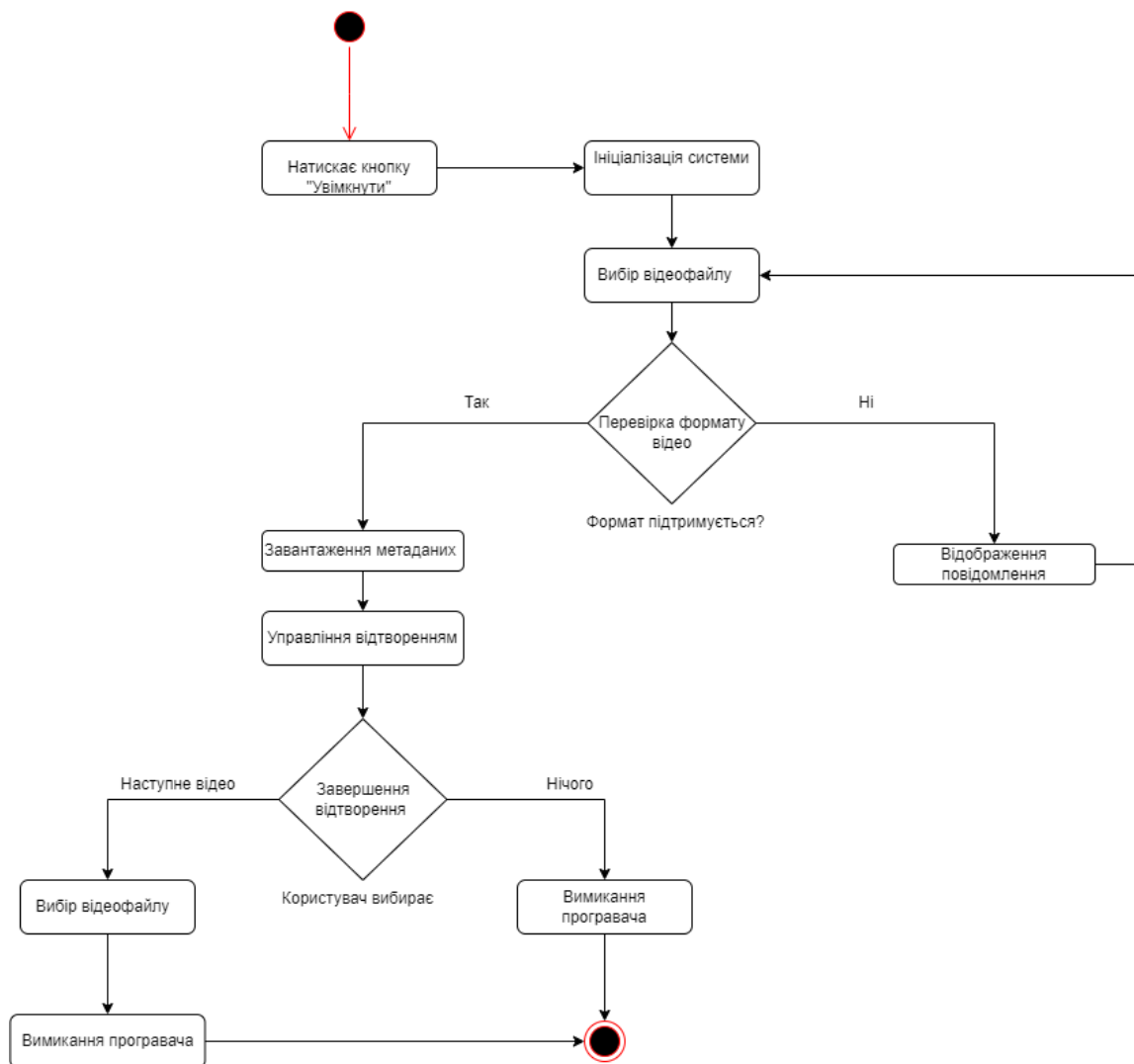


Рис. 2.2 – Діаграма діяльності відтворення відео

2.4. Взаємодія об'єктів системи

Для візуалізації взаємодії об'єктів у системі, в UML зазвичай використовують діаграми послідовності.

Ці діаграми детально описують порядок виконання операцій та фіксують взаємодію об'єктів під час співпраці. Зокрема, вони концентруються на часі, використовуючи вертикальну ось для відображення порядку взаємодії та моменти відправлення повідомлень [11].

Діаграма послідовності для відеопрогравача описує взаємодію між користувачем, відеопрогравачем, відеофайлом та метаданими під час процесу відтворення відео (рис. 2.3). Спочатку користувач обирає відеофайл для відтворення, використовуючи інтерфейс відеопрогравача. Відеопрогравач отримує обраний відеофайл і готується до його завантаження. Після цього відеопрогравач запитує метадані з відеофайлу, щоб отримати інформацію, таку як назва, автор та тривалість відео. Відеофайл передає запит до метаданих і отримує їх, а потім повертає їх відеопрогравачу. Після отримання метаданих відеопрогравач починає відтворення відеофайлу. Користувач має можливість управляти відтворенням, використовуючи функції відеопрогравача, такі як пауза, перемотка вперед або назад, зупинка відтворення. Управління відтворенням здійснюється через відеопрогравач, який передає відповідні команди відеофайлу для виконання. Таким чином, діаграма послідовності показує всі основні кроки та взаємодії між компонентами системи під час вибору та відтворення відео.

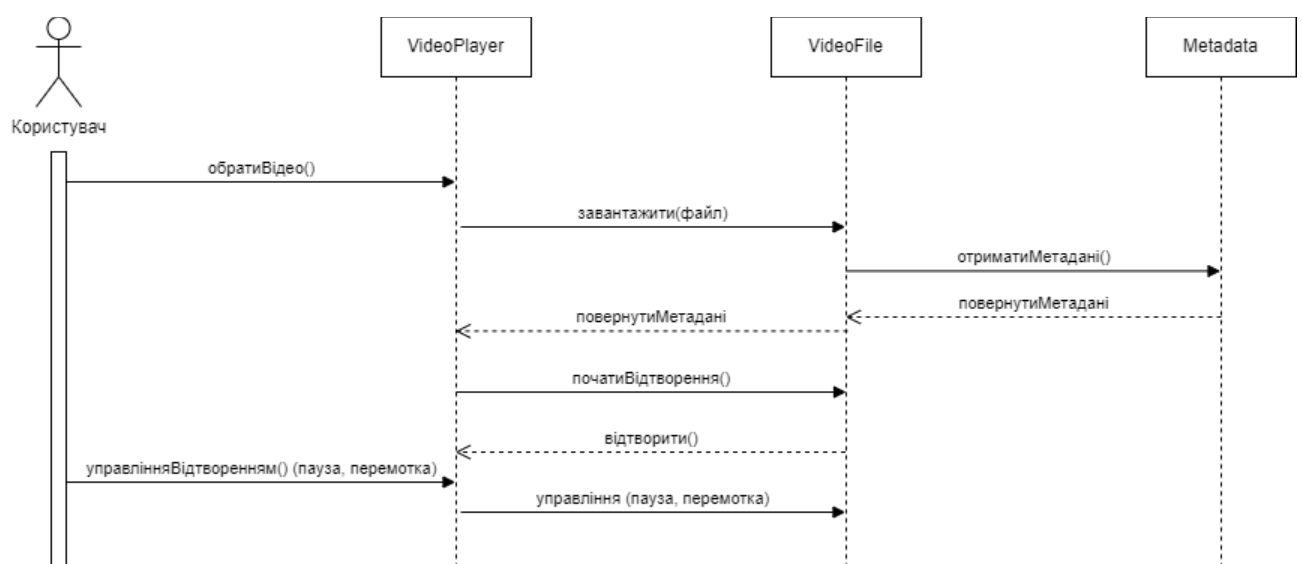


Рис. 2.3 – Діаграма послідовності завантаження відео

Користувач обирає відеофайл для додавання до списку улюблених, використовуючи інтерфейс відеопрогравача (рис. 2.4). Відеопрогравач отримує обраний відеофайл і ініціює процес додавання його до списку улюблених. Відеопрогравач передає команду додавання відеофайлу до об'єкта Список улюблених, який зберігає відеофайл у своєму списку. Після успішного додавання відеофайлу до списку улюблених, Список улюблених надсилає підтвердження назад до відеопрогравача, який, у свою чергу, передає підтвердження користувачеві. Це завершує процес додавання відео до списку улюблених.

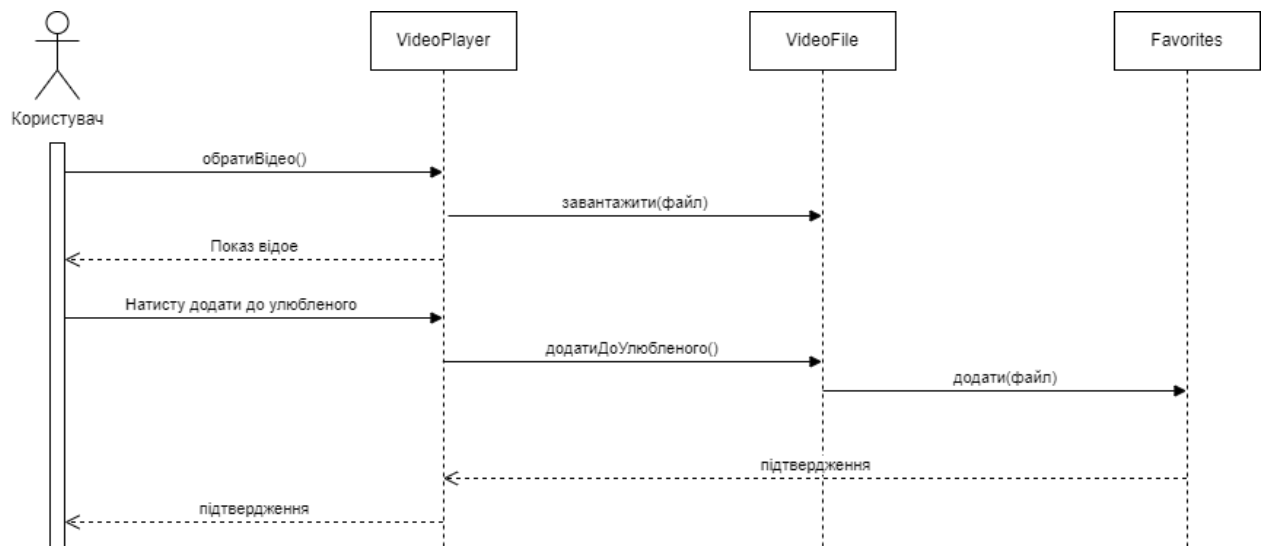


Рис. 2.4 – Діаграма послідовності для модуля додавання відео до списку улюблених

Висновки до другого розділу

У цьому розділі було детально описано технічне завдання на розробку системи відеопрогравача, його призначення, вимоги до системи, інтерфейсу та аналіз бізнес-вимог і вимог користувачів. Було визначено основні функціональні характеристики відеопрогравача, включаючи підтримку високоякісного відтворення відео, підтримку різноманітних відеоформатів, швидке завантаження відеофайлів та зручний інтерфейс. Описані вимоги користувачів, які включають високу якість відтворення відео, підтримку різних форматів, швидкість роботи та інтуїтивно зрозумілий інтерфейс.

Важливе значення було надано нефункціональним вимогам, таким як сприйняття, продуктивність та можливість експлуатації, які забезпечують високу якість роботи та задоволення користувачів. Встановлено системні вимоги до середовища виконання, включаючи підтримку сучасних операційних систем.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		33

РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З СИСТЕМОЮ ДОДАТКУ

3.1. Порядок встановлення та налаштування системи

Розробка відеопрогравача мовою C# передбачає не лише створення самої програми, а й забезпечення її коректного встановлення та налаштування на кінцевих системах користувачів. На цьому етапі важливо звернути увагу на усі деталі, що впливають на стабільність і ефективність роботи застосунку.

Підготовка до встановлення включає перевірку комплекту постачання, зокрема наявність усіх необхідних компонентів: інсталяційного пакета, супровідної документації, а також, у разі потреби, додаткових бібліотек. Також слід відповідально підійти до вибору місця для встановлення програми — на жорсткому диску має бути достатньо вільного простору.

Процес встановлення починається із запуску інсталяційного файлу формату .exe (рис. 3.1). Користувачеві необхідно двічі клацнути на відповідний файл, щоб розпочати процедуру встановлення. Далі у діалоговому вікні обирається шлях для інсталяції програми та, за потреби, додаткові параметри, що можуть впливати на функціонал або розташування файлів. Після підтвердження обраних налаштувань запускається процес копіювання файлів програми на жорсткий диск користувача, що є завершальним етапом базової інсталяції.

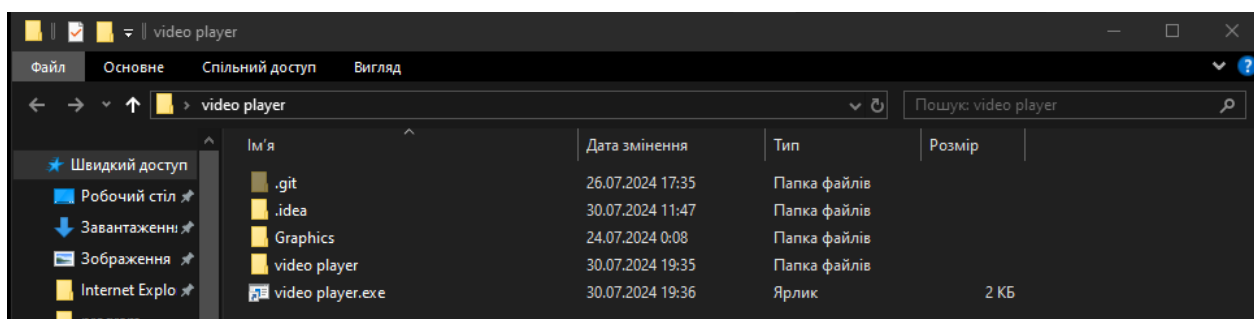


Рис. 3.1 - Встановлення програмного забезпечення

Після завершення встановлення програми користувач повинен запустити відеопрогравач для проведення первинного налаштування системи. На цьому

етапі налаштовуються основні параметри, необхідні для комфортного використання програми.

Зокрема, користувач має змогу задати бажані відео- та аудіоналаштування відповідно до власних уподобань. Це можуть бути такі параметри, як роздільна здатність відео, рівень гучності, вибір джерела аудіовиходу тощо.

Окрім цього, користувач може обрати одну або кілька тек, у яких зберігаються медіафайли. Програма автоматично проіндексує вміст зазначених тек і додасть знайдені файли до внутрішньої бібліотеки. Такий підхід дозволяє швидко отримувати доступ до улюблених відео та аудіо без необхідності щоразу вручну відкривати їх на пристрої (рис. 3.2).

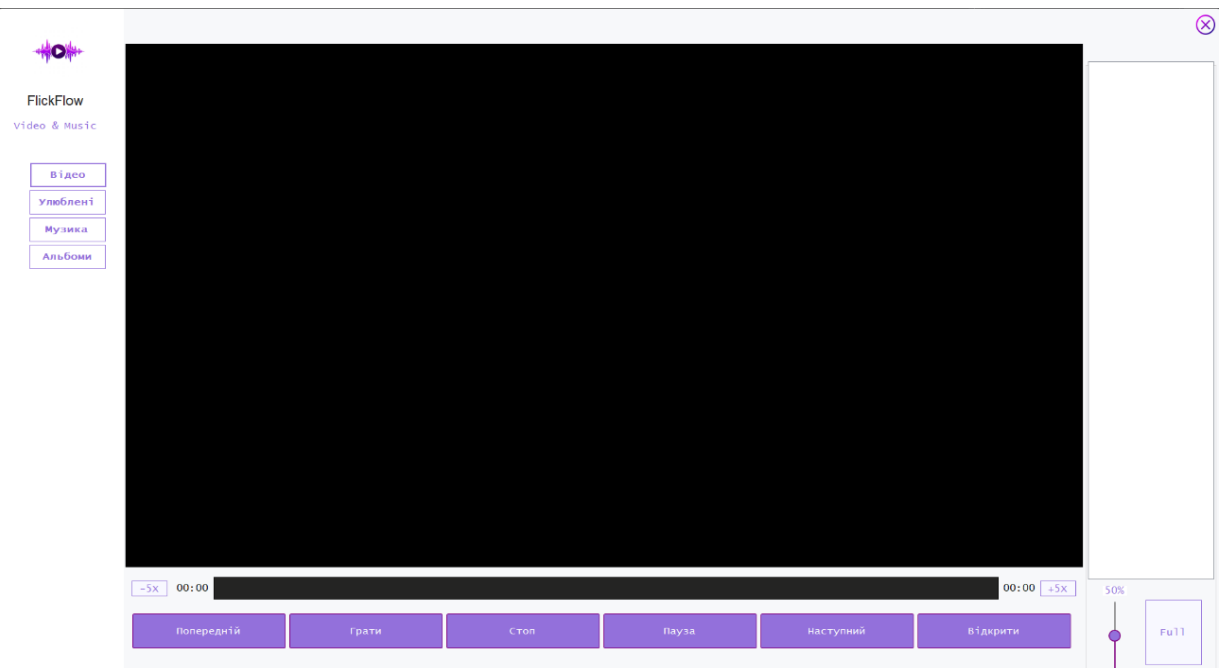


Рис. 3.2 - Налаштована програма

3.2. Опис основних функцій додатку

Фрагмент коду який відповідає за відкриття діалогу для вибору відеофайлів, перевірку їх розширень, додавання їх до списку та відображення попереджень у разі помилок.

Метод `btn_open_Click` відповідає за відкриття діалогового вікна, яке дозволяє користувачу вибирати відеофайли. Спочатку створюється об'єкт

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		35

OpenFileDialog, який налаштовується на можливість вибору кількох файлів одночасно і фільтрується так, щоб дозволити вибір лише відеофайлів з розширеннями .mp4, .avi, .mkv, .mov, і .wmv. Якщо користувач натискає кнопку "OK" у діалоговому вікні, обрані файли зберігаються у списки files та paths, де files містить імена файлів, а paths - повні шляхи до них.

Після цього, перед додаванням нових файлів, список track_list очищується, щоб уникнути змішування старих і нових файлів. Потім відбувається перебір кожного файлу з обраного списку. Спочатку перевіряється, чи файл вже є у списку за допомогою методу IsVideoAlreadyInList. Якщо файл вже є у списку, відображається попередження, і обробка переходить до наступного файлу. Далі отримується розширення файлу і перевіряється, чи є воно допустимим відеоформатом. Якщо файл є відеофайлом, він додається до списку track_list. Якщо ж файл не є відеофайлом, відображається попередження про неприпустимий формат файлу. Таким чином, метод забезпечує коректне завантаження відеофайлів користувачем, перевіряє їх відповідність підтримуваним форматам, уникає дублювання файлів у списку та відображає відповідні попередження у разі виникнення помилок.

```
private void btn_open_Click(object sender, EventArgs e)
{
    OpenFileDialog ofd = new OpenFileDialog();
    ofd.Multiselect = true;
    ofd.Filter = "Video Files|*.mp4;*.avi;*.mkv;*.mov;*.wmv";

    if (ofd.ShowDialog() == System.Windows.Forms.DialogResult.OK)
    {
        files = ofd.SafeFileNames.ToList();
        paths = ofd.FileNames.ToList();

        track_list.Items.Clear(); // Очистка списку перед додаванням нових відео
        foreach (string file in files)
        {
            if (IsVideoAlreadyInList(file))
            {
                MessageBox.Show("Це відео вже є у списку.", "УВАГА", MessageBoxButtons.OK,
                MessageBoxIcon.Warning);
                continue;
            }

            string extension = Path.GetExtension(file).ToLower();
            if (extension == ".mp4" || extension == ".avi" || extension == ".mkv" || extension == ".mov" || extension ==
            ".wmv")
```

```

        {
            track_list.Items.Add(file);
        }
        else
        {
            MessageBox.Show("Дозволено лише відеофайли.", "УВАГА", MessageBoxButtons.OK,
            MessageBoxIcon.Warning);
        }
    }
}
}

```

Метод `btn_previous_Click` відповідає за перехід до попереднього треку у списку відтворення. Спочатку він перевіряє, чи є поточний трек не першим у списку. Якщо це так, індекс вибраного треку зменшується на одиницю, після чого встановлюється URL попереднього треку та запускається його відтворення. Текст мітки оновлюється, щоб відобразити інформацію про відтворення поточного треку. Також, якщо це необхідно, запускається таймер для оновлення прогресу відтворення. Якщо ж поточний трек є першим у списку, відображається повідомлення про те, що немає попередніх треків.

Метод `trackVolume_ValueChanged` відповідає за зміну гучності плеєра. Він інвертує значення гучності (від 100 до 0) та встановлює його у налаштуваннях плеєра. Текстовий індикатор гучності також оновлюється відповідно до нового значення.

Метод `musicProgressBar_MouseDown` відповідає за зміну позиції відтворення при кліку миші на прогрес-барі. Якщо плеєр знаходиться у стані відтворення або паузи, визначається нова позиція відтворення на основі координат кліку. Потім встановлюється нова позиція відтворення, оновлюється прогрес-бар та текстові індикатори, що показують поточний час відтворення.

```

private void btn_previous_Click(object sender, EventArgs e)
{
    // Перевірка, чи вибраний трек є в списку музичних треків
    if (musicList.SelectedIndex > 0)
    {
        // Зменшити індекс вибраного елемента на одиницю
        musicList.SelectedIndex -= 1;
        // Встановити URL і відтворити попередній трек
        playerMusic.URL = paths[musicList.SelectedIndex];
        playerMusic.Ctlcontrols.play();
        // Оновити текст мітки для відображення інформації про відтворення
        lblPlaying.Text = "Грає - " + musicList.SelectedItem.ToString();
        // Запустити таймер для оновлення прогресу, якщо потрібно
    }
}

```

```

        timer1.Start();
    }
    else
    {
        MessageBox.Show("У списку немає попередніх музичних файлів.", "Інфо", MessageBoxButtons.OK,
        MessageBoxIcon.Information);
    }
}

private void trackVolume_ValueChanged(object sender, EventArgs e)
{
    int invertedValue = 100 - trackVolume.Value; // Інвертує значення
    playerMusic.settings.volume = invertedValue; // Змінює гучність музичного плеєра
    lblMusicVolume.Text = invertedValue.ToString() + "%"; // Оновлює текстовий індикатор гучності
}

```

Метод UpdateAlbumContextMenu відповідає за оновлення контекстного меню для елемента списку музики, дозволяючи додавати або видаляти пісні з альбомів, а також створювати нові альбоми і видаляти існуючі. Спочатку очищується поточне контекстне меню, щоб уникнути дублювання елементів.

Метод отримує всі альбоми з albumManager і для кожного альбому створює елемент меню. Якщо вибрана пісня ще не міститься в альбомі, до елемента меню додається обробник подій, який дозволяє додати пісню до альбому при кліку. Якщо вибрана пісня вже міститься в альбомі, до елемента меню додається підменю для видалення пісні з альбому. При кліку на це підменю викликається метод видалення пісні з альбому, і відображається повідомлення про успішне видалення. Додається елемент меню "Create New Album", при кліку на який викликається метод створення нового альбому в albumManager. Також створюється елемент меню "Delete Album", при кліку на який перевіряється, чи вибрано альбом для видалення в albumListBox. Якщо альбом вибрано, відображається діалогове вікно для підтвердження видалення альбому і всіх його пісень. Якщо користувач підтверджує видалення, викликається метод видалення альбому з albumManager, оновлюється список альбомів у albumListBox, очищується список пісень у albumTracksListBox, і контекстне меню оновлюється. Таким чином, метод UpdateAlbumContextMenu забезпечує динамічне оновлення контекстного меню для списку музики, дозволяючи користувачу легко додавати або видаляти пісні з альбомів, створювати нові альбоми і видаляти існуючі альбоми.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		38

```

private void UpdateAlbumContextMenu(string selectedSong)
{
    musicList.ContextMenuStrip.Items.Clear();

    foreach (var album in albumManager.GetAlbums().Keys)
    {
        var albumItem = new ToolStripMenuItem(album);

        // Додавання пісні в альбом, якщо вибрана пісня не є в альбомі
        if (!albumManager.GetAlbums()[album].Contains(selectedSong))
        {
            albumItem.Click += (sender, args) => albumManager.AddSongToAlbum(album, selectedSong);
        }
        musicList.ContextMenuStrip.Items.Add(albumItem);

        // Видалення пісні з альбому, якщо пісня є в альбомі
        if (albumManager.GetAlbums()[album].Contains(selectedSong))
        {
            var removeFromAlbumItem = new ToolStripMenuItem("Видалити з альбому");
            removeFromAlbumItem.Click += (sender, args) =>
            {
                albumManager.RemoveSongFromAlbum(album, selectedSong);
                MessageBox.Show($"Пісню видалено з '{album}'", "Успіх", MessageBoxButtons.OK,
                MessageBoxIcon.Information);
            };
            albumItem.DropDownItems.Add(removeFromAlbumItem);
        }
    }
}

```

3.3. Структура інтерфейсу застосунку

На зображенні показано інтерфейс користувача медіаплеєра FlickFlow під час перегляду відео. У центральній частині екрана розташовано область відтворення відео у вертикальному форматі. Зліва розміщено вертикальне меню з кнопками українською мовою: «Відео», «Улюблені», «Музика», «Альбоми». Під відео знаходиться прогрес-бар, який показує поточний стан відтворення, а також кнопки керування швидкістю відтворення -5x і +5x. Справа під прогрес-баром розташовано вертикальний повзунок регулювання гучності з індикатором поточного рівня, що наразі становить 15%, поруч є кнопка переходу в повноекранний режим з написом Full.

У нижній частині екрана розташована панель керування з кнопками «Попередній», «Грати», «Стоп», «Пауза», «Наступний» та «Відкрити», які дозволяють керувати відтворенням медіа (рис. 3.3).

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		39

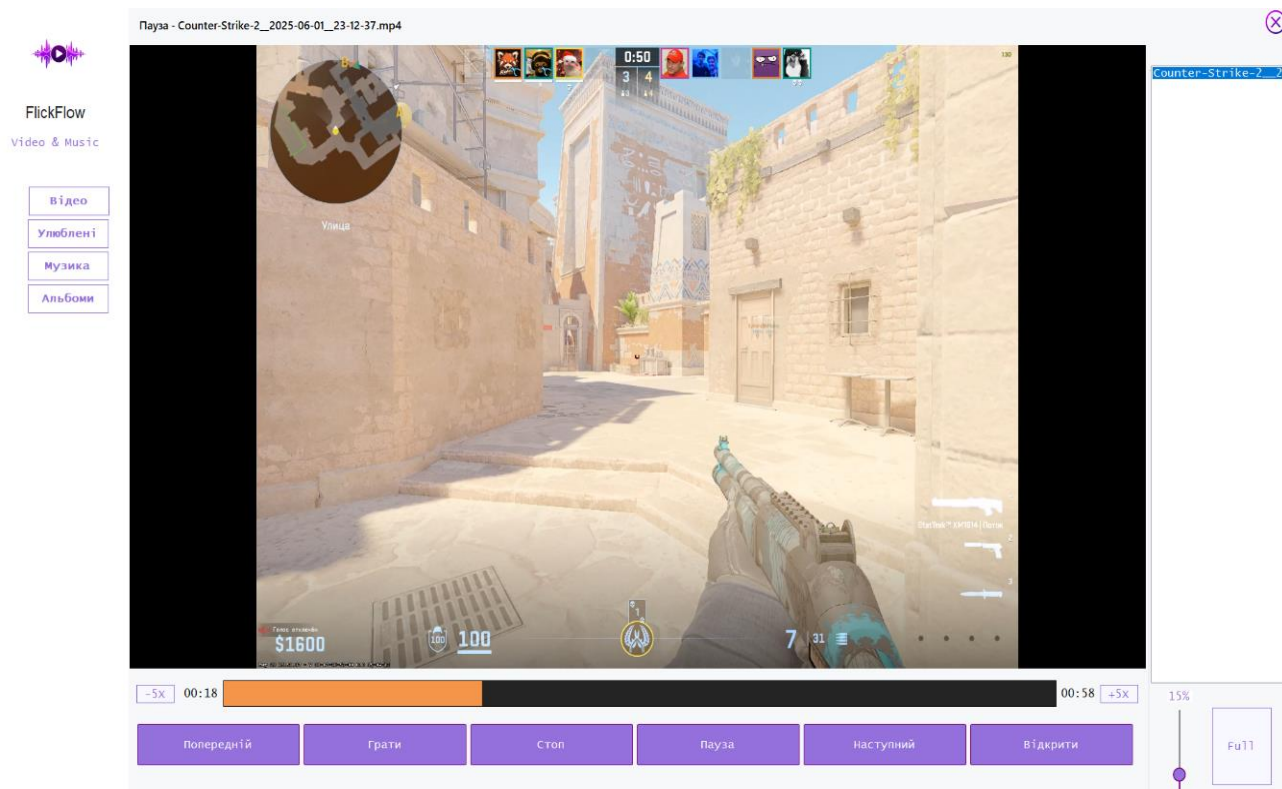


Рис. 3.3 – Інтерфейс відеопрогравача

Порядок роботи з додатком може включати використання навігаційного меню для вибору категорії медіафайлів (відео, музика, альбоми тощо), регулювання гучності за допомогою вертикального повзунка, вибір і відкриття медіафайлу за допомогою кнопки «Відкрити», керування відтворенням за допомогою кнопок «Попередній», «Грати», «Стоп», «Пауза», «Наступний», регулювання швидкості відтворення за допомогою кнопок «-5x» та «+5x», перемикання у повноекранний режим за допомогою кнопки «Full», спостереження за прогресом відтворення на прогрес-барі, а також керування треками в текстовому полі, можливо, з контекстним меню для додавання або видалення треків з альбомів. Цей інтерфейс дозволяє користувачу зручно керувати відтворенням відео та музики, а також організовувати медіафайли у альбоми та списки (рис. 3.4).

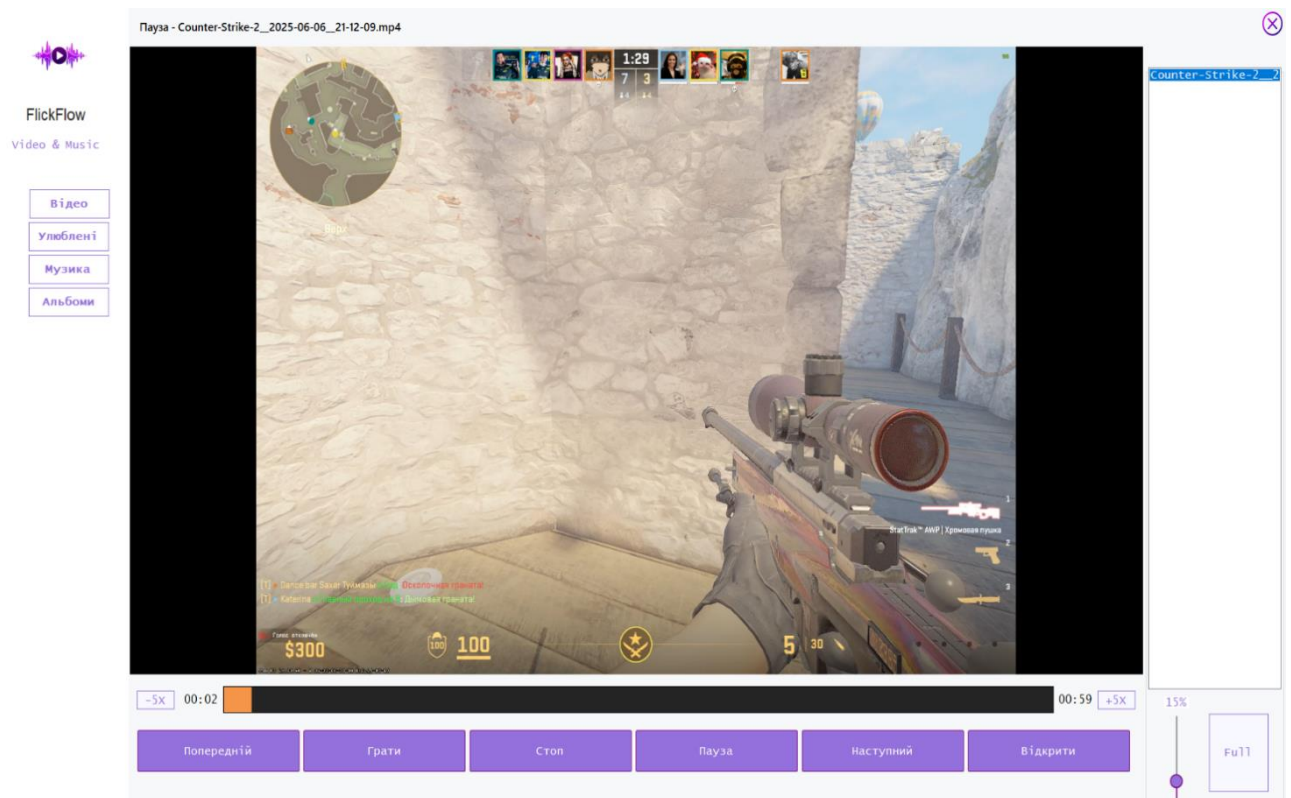


Рис. 3.4 – Відтворення відео

Інтерфейс програми складається з декількох елементів. У верхньому лівому куті знаходиться логотип із зображенням динаміків та написом "FlickFlow". Нижче зліва знаходяться чотири кнопки: «Відео», «Улюблені», «Музика», «Альбоми». Ліворуч від цих кнопок знаходиться вертикальний повзунок.

У центральній частині екрана зверху розташований великий чорний екран з індикатором часу відтворення (початково 00:00). Праворуч від цього екрана є вертикальний повзунок з позначкою 50%. У нижній частині інтерфейсу є велика чорна стрічка. Над нею розміщений диск, а праворуч від нього — прямокутна область де відображається список аудіо. Внизу інтерфейсу розташовані п'ять кнопок: «Попередній», «Грати», «Стоп», «Пауза», «Наступний» (рис. 3.5).

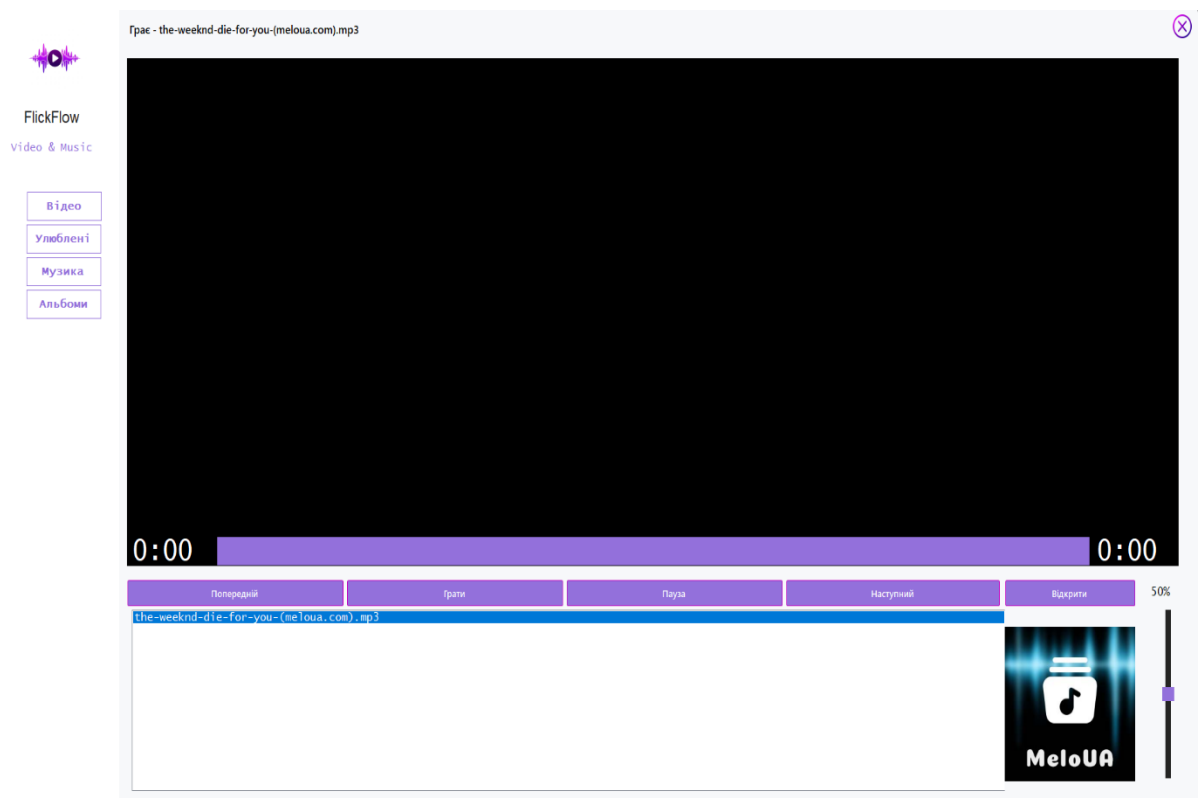


Рис. 3.5 – Інтерфейс відеопрогравача вкладки «Музика»

У верхній центральній частині відображається назва відтворюваного файлу. Під назвою розміщена звукова хвиля, яка відображає графічний спектр відтворюваного аудіо, а під нею – таймер, що показує поточний час відтворення (01:22) та загальну тривалість треку (05:20). Під звуковою хвилею зліва розташована обкладинка альбому або треку із зображенням логотипу "MeloUA", а праворуч від обкладинки знаходиться текстове поле, в якому відображається назва відтворюваного файлу (рис. 3.6). Праворуч від текстового поля розміщено вертикальний повзунок для регулювання гучності, встановлений на 50%.

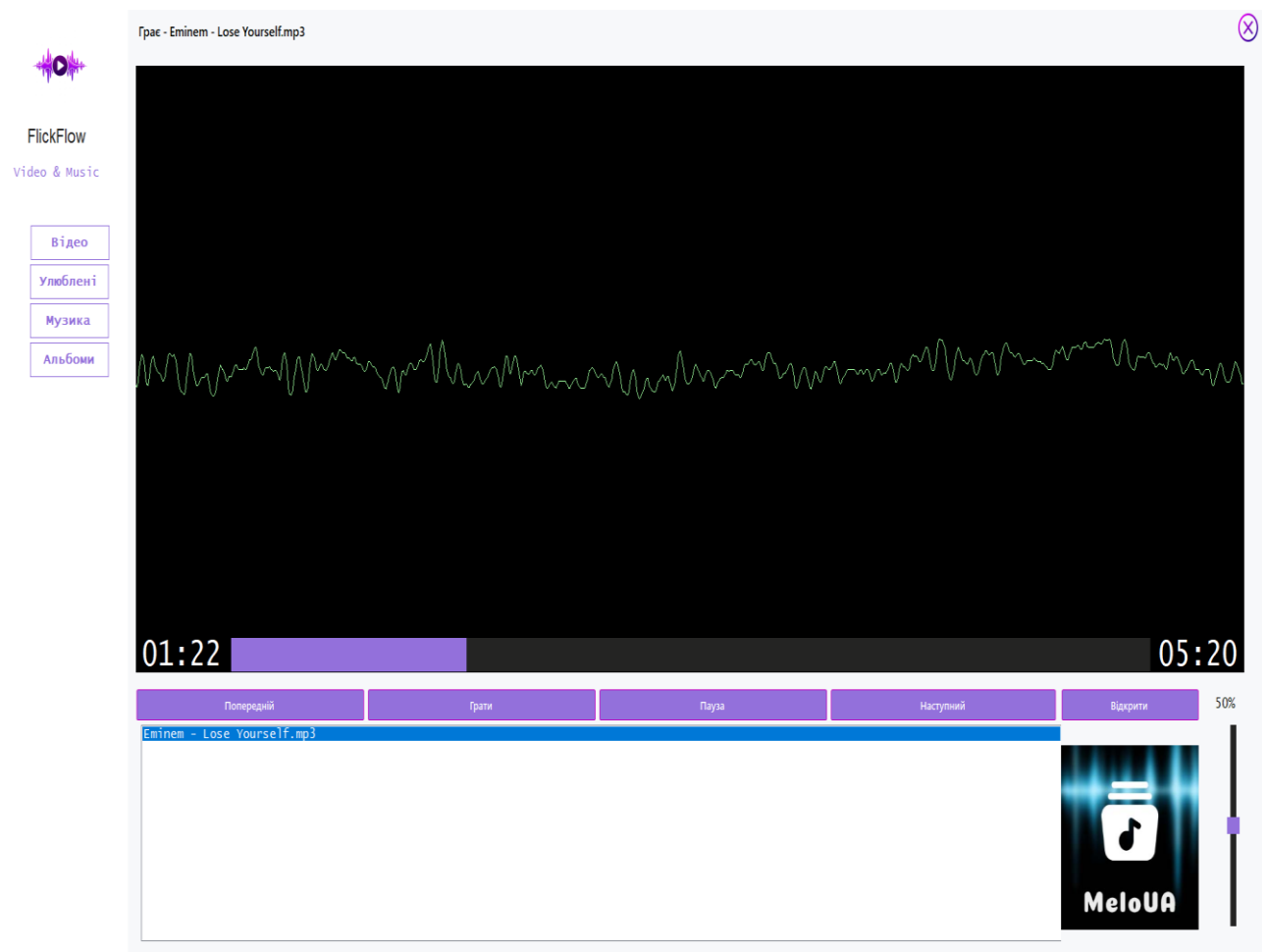


Рис. 3.6 – Відтворення музики

Відображено інтерфейс програми FlickFlow в режимі управління списком улюблених файлів. Ліва частина інтерфейсу містить логотип програми з зображенням динаміків, напис «FlickFlow», кнопку «Video & Music», чотири кнопки «Відео», «Улюблені», «Музика», «Альбоми». Права частина інтерфейсу включає напис «Улюблені», три кнопки: «Зберегти», «Загрузити», «Видалити» і великий текстовий список, де показаний шлях до файлу (рис. 3.7).

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		43

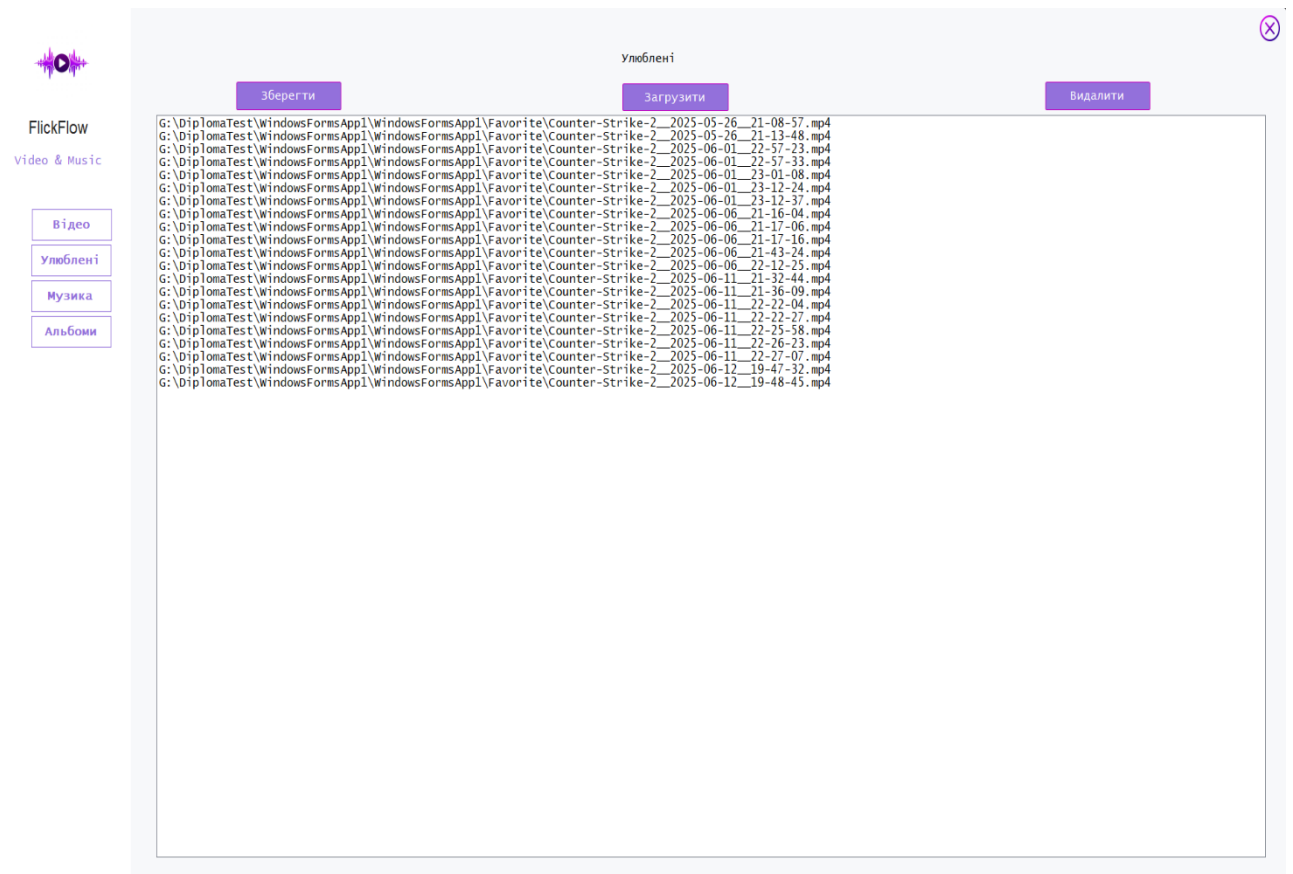


Рис. 3.7 – Режим управління списком улюблених файлів

На рисунку 3.8 та 3.9 відображено інтерфейс програми FlickFlow в режимі управління музичними альбомами. Ліва частина інтерфейсу містить логотип програми з зображенням динаміків, напис "FlickFlow", кнопку "Video & Music", чотири кнопки «Відео», «Улюблені», «Музика», «Альбоми», а також вертикальний повзунок з чорним колом. Права частина інтерфейсу включає три кнопки: «Створити альбом», «Видалити альбом», «Загрузити альбом» і дві великі текстові області. У лівій області відображені назви альбомів: "loud", "mama", "gh", а у правій області відображені назви треків альбому: "fallen-down-reprise-cover-d", "Death Grips - Takyon (Death)".

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		44

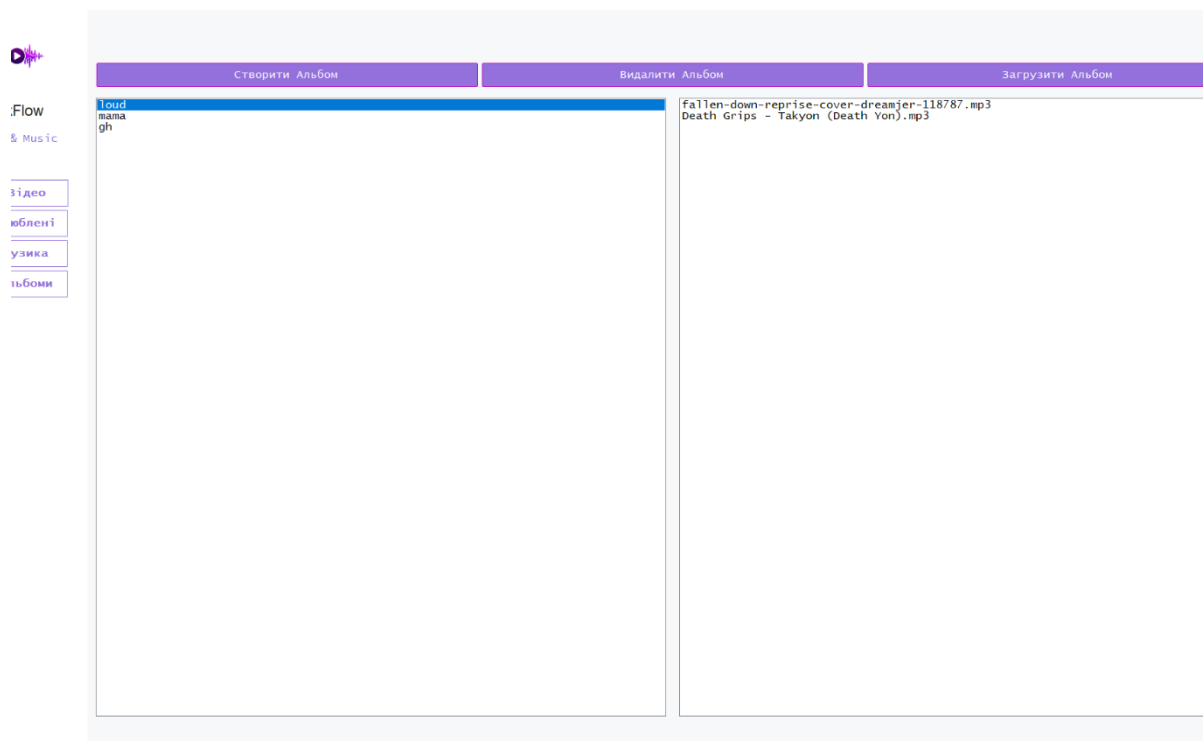


Рис. 3.8 – Режим управління музичними альбомами

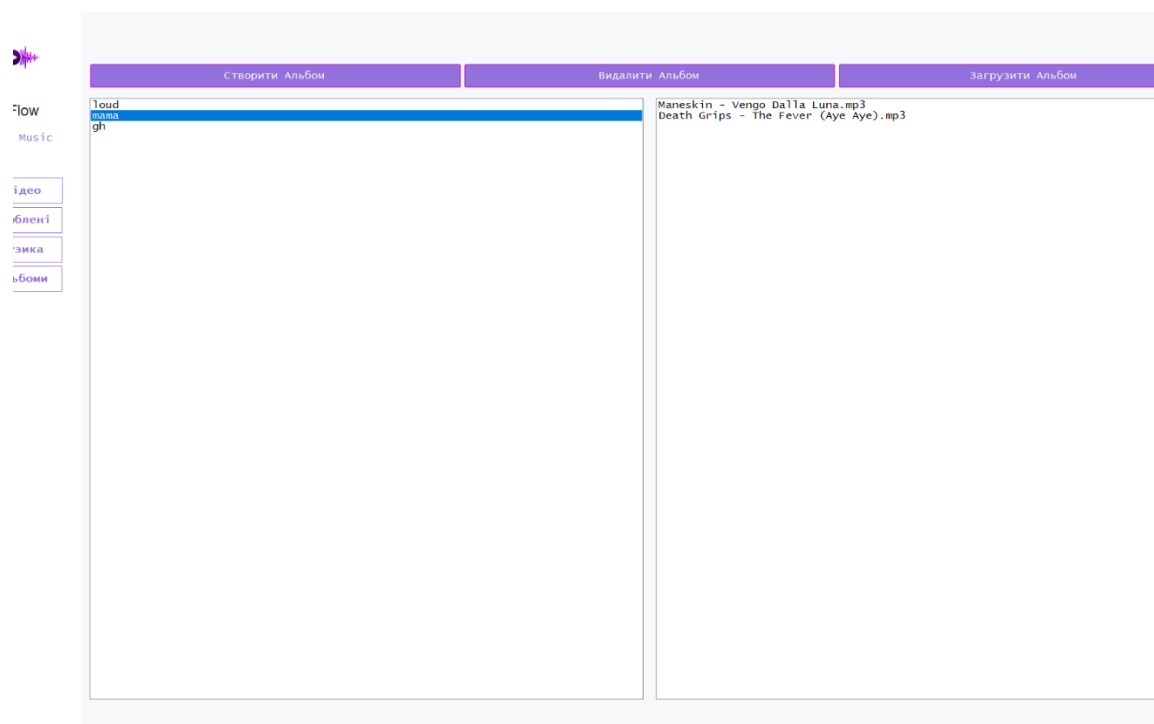


Рис. 3.9 – Режим управління музичними альбомами

3.4. Тестування роботи програмного продукту

Розробка будь-якого програмного продукту неминуче супроводжується деякими помилками, тому тестування програмного забезпечення є необхідним для забезпечення високої якості додатка. Під час тестування перевіряються різні аспекти програми, такі як коректність вхідних та вихідних даних, правильність роботи функцій та алгоритмів, обробка помилок і відновлення після збоїв.

Метою тестування є виявлення помилок і дефектів програмного продукту, щоб їх можна було виправити до випуску на ринок або введення в експлуатацію [23]. Створимо так звані Test Cases, вони є тестовим артефактом, який полягає у виконанні певної кількості дій та/або умов, необхідних для перевірки визначеної функціональності розроблюваної програмної системи [22]. Представимо створені тестові випадки у вигляді таблиці.

Таблиця 3.1

Test Cases

Test Case Description	Test Steps	Test Data	Expected Results	Actual Results	Pass/Fail
1.	2.	3.	4.	5.	6.
Перевірка відображення UI елементів	1. Відкрити програму FlickFlow. 2. Перевірити наявність логотипу з динаміками. 3. Перевірити, що кнопки "Video", "Favourites", "Music" та "Albums" присутні. 4. Перевірити наявність кругового елемента, таймера відтворення, вертикальних повзунків та кнопок "Previous", "Next", "Play", "Pause", "Open".	Програма відкрита	Всі елементи інтерфейсу користувача відображаються коректно та не накладаються один на одного.	Як і очікувалося	Pass

Продовження таблиці 3.1

1.	2.	3.	4.	5.	6.
Функціональність кнопки "Play"	1. Відкрити програму FlickFlow. 2. Натиснути кнопку "Open" і вибрати файл для відтворення. 3. Натиснути кнопку "Play"	Вибрано файл для відтворення.	Вибраний файл починає відтворюватися, таймер відтворення починає відрахунок.	Як і очікувалося	Pass
Функціональність кнопки "Pause"	1. Відкрити програму FlickFlow. 2. Відкрити файл для відтворення і натиснути кнопку "Play". 3. Натиснути кнопку "Pause".	Файл відтворюється.	Відтворення файлу зупиняється, таймер відтворення зупиняється.	Як і очікувалося	Pass
Перевірка роботи повзунка гучності	1. Відкрити програму FlickFlow. 2. Відкрити файл для відтворення і натиснути кнопку "Play". 3. Змінити положення повзунка гучності.	Відкрито файл для відтворення.	Гучність відтворення змінюється відповідно до положення повзунка.	Як і очікувалося	Pass
Функціональність кнопки "Next"	1. Відкрити програму FlickFlow. 2. Відкрити декілька файлів для відтворення. 3. Натиснути кнопку "Next".	Відкрито декілька файлів для відтворення у списку відтворення.	Відтворення переходить до наступного файлу у списку відтворення.	Як і очікувалося	Pass
Перевірка роботи кнопки "Open"	1. Відкрити програму FlickFlow. 2. Натиснути кнопку "Open".	Програма відкрита.	Обраний файл додається до списку відтворення	Як і очікувалося	Pass

1.	2.	3.	4.	5.	6.
	3. Вибрати файл для відтворення і підтвердити вибір.		і готовий до відтворення.		

В ході проведення тестування було перевірено додаток та виправлено знайдені помилки. В результаті отримано застосунок відповідної якості.

Висновки до третього розділу

В розділі було розглянуто процес розробки, встановлення та налаштування відеопрогравача на мові C#, а також основні функції та методи для керування медіафайлами. Описано інтерфейс користувача, який дозволяє легко керувати відтворенням відео та музики, а також організовувати медіафайли у альбоми та списки. Було проведено тестування роботи програмного продукту, в ході якого перевірялися різні аспекти програми для забезпечення її коректної роботи та виявлення можливих помилок.

Результати тестування показали, що всі основні функції відеопрогравача працюють як очікувалося, а знайдені помилки були виправлені. Таким чином, створений застосунок відповідає необхідним вимогам і забезпечує користувачам зручний та ефективний інструмент для відтворення та організації медіафайлів.

ВИСНОВКИ

Кваліфікаційна робота на тему розробки відеопрогравача є значущим внеском у галузь програмного забезпечення та мультимедійних технологій. У рамках даної роботи було проведено детальне дослідження існуючих рішень у сфері відеопрогравачів, їх переваг та недоліків. На основі проведеного аналізу було сформульовано вимоги до розроблюваного програмного продукту, який повинен забезпечувати високу якість відтворення відео, підтримку широкого спектру відеоформатів, швидкість роботи та інтуїтивно зрозумілий інтерфейс, а також належну кросплатформенність.

Процес розробки відеопрогравача складався з кількох ключових етапів: постановка задачі, аналіз існуючих програмних рішень, обґрунтування вибору інструментальних засобів, проєктування структури програмного забезпечення, написання програмного коду, тестування роботи програми, аналіз одержаних результатів та оформлення пояснювальної записки.

Розробка програмного продукту була проведена з використанням сучасних мов програмування та бібліотек, що дозволило забезпечити високу продуктивність та гнучкість програмного забезпечення. Було реалізовано підтримку популярних відеоформатів, таких як MP4, AVI, MKV, MOV, а також забезпечено відтворення відео високої чіткості (HD, Full HD, 4K) без затримок та спотворень.

Інтерфейс користувача був розроблений з урахуванням потреб користувачів, що дозволило створити зручний та інтуїтивно зрозумілий продукт. Особлива увага була приділена тестуванню та оптимізації роботи програми, що дозволило забезпечити високу швидкість та стабільність роботи програвача.

Практична значущість даної роботи полягає у створенні функціонального відеопрогравача, який може бути використаний як кінцевими користувачами, так і розробниками програмного забезпечення для інтеграції в складніші мультимедійні системи. Розробка відеопрогравача сприяє професійному

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		49

розвитку, надаючи цінний практичний досвід у програмуванні та мультимедійних технологіях. Використання власного програмного забезпечення може знизити витрати на придбання комерційних продуктів, що є додатковою перевагою.

Таким чином, виконана кваліфікаційна робота демонструє високий рівень підготовки студента, його здатність до самостійного вирішення складних технічних завдань, аналізу та синтезу інформації, а також розробки практичних рішень у галузі програмного забезпечення. Отримані результати можуть бути використані для подальших досліджень та розробок у сфері мультимедійних технологій, а також для вдосконалення існуючих програмних продуктів.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		50

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Bunifu. [Електронний ресурс] – Режим доступу до ресурсу: <https://bunifuframework.com/products/bunifu-ui-winforms/components/bunifu-ellipse/>.
2. C# Підручник. [Електронний ресурс] – Режим доступу до ресурсу: <https://w3schoolsua.github.io/cs/index.html#gsc.tab=0>.
3. Figma – сучасний інструмент для створення якісних методичних матеріалів. Від ідеї до реалізації. [Електронний ресурс] – Режим доступу до ресурсу: <https://vseosvita.ua/c/pedagogy/post/46326>.
4. Figma. [Електронний ресурс] – Режим доступу до ресурсу: <https://www.figma.com/files/team/1223013079992062985/recents-and-sharing/recently-viewed?fuid=1223013071917893911>.
5. HD, FHD, UHD, 4K: у чому різниця? [Електронний ресурс] – Режим доступу до ресурсу: <https://www.strong-eu.com/ua/blog/hd-fhd-uhd-4k-what-are-the-differences>.
6. Introduction. StarUML documentation. [Електронний ресурс] – Режим доступу до ресурсу: <https://docs.staruml.io>.
7. Media Player Classic. [Електронний ресурс] – Режим доступу до ресурсу: <https://mpc-hc.org>.
8. PotPlayer. [Електронний ресурс] – Режим доступу до ресурсу: <https://potplayer.daum.net>.
9. Sample WMV files download. [Електронний ресурс] – Режим доступу до ресурсу: <https://file-examples.com/index.php/sample-video-files/sample-wmv-files-download/>.
10. Specifying Build Settings. [Електронний ресурс] – Режим доступу до ресурсу: <https://doc.qt.io/qtcreator/creator-build-settings.html>.

11. UML для бізнес-моделювання: для чого потрібні діаграми процесів. [Електронний ресурс] – Режим доступу до ресурсу: <https://evergreens.com.ua/ua/articles/uml-diagrams.html>.
12. VLC Media Player. [Електронний ресурс] – Режим доступу до ресурсу: – <https://www.videolan.org/vlc/>.
13. Аналіз вимог до програмного забезпечення / Юрій Грицюк, 2018 — Yakaboo.
14. Варіанти використання та сценарії (Use Cases and Scenarios). [Електронний ресурс] – Режим доступу до ресурсу: <https://www.maxzosim.com/use-cases-and-scenarios/>.
15. Відомості про проекти і рішення на мові С# [Електронний ресурс] – Режим доступу до ресурсу: <https://learn.microsoft.com/uk-ua/visualstudio/get-started/tutorial-projects-solutions>.
16. Діаграма діяльності. [Електронний ресурс] – Режим доступу до ресурсу: <https://studfile.net/preview/5200239/page:7/>.
17. Коноваленко І.В. Програмування мовою С#, 2019. – 227 с.
18. Основні принципи ООП. [Електронний ресурс] – Режим доступу до ресурсу: <https://training.epam.ua/ua/blog/275>.
19. Платформа .NET та мова програмування С# / Коноваленко І.В., Марущак П.О., 2020. – 320 с.
20. Програмування мовою С# / Коноваленко І.В., 2019. – 277 с.
21. Розробка кроссплатформених додатків. [Електронний ресурс] – Режим доступу до ресурсу: <https://webcase.com.ua/uk/krossplatformennaya-razrabotka/>.
22. Тест-кейс – один із ключових інструментів тестувальників ПЗ. [Електронний ресурс] – Режим доступу до ресурсу: <https://foxminded.ua/test-keis/>.
23. Тест-кейси: що, як, навіщо. [Електронний ресурс] – Режим доступу до ресурсу: <https://dou.ua/forums/topic/46769/>.
24. Топ-3 плеєрів для програвання відео файлів [Електронний ресурс] – Режим доступу до ресурсу: <https://iscan.com.ua/mp4mkvplayers/>.

25. Формати відео і їх характеристики [Електронний ресурс] – Режим доступу до ресурсу: https://ipkey.com.ua/uk/faq/550-video-formats.html#google_vignette.
26. Шаховська Н.Б., Литвин В.В. Проєктування інформаційних систем: навч. посіб. Львів: «Магнолія – 2006», 2011. – 380с.
27. Що таке Figma: функції, інструменти та переваги [Електронний ресурс] – Режим доступу до ресурсу: <https://wezom.academy/ua/chto-takoe-figma-funktsii-instrumenty-ipreimuschestva/>.
28. Юрій Грицюк. Аналіз вимог до програмного забезпечення / Юрій Грицюк., 2018. – 456 с.
29. Як тестувати веб-сайт: основні етапи і поради. [Електронний ресурс] – Режим доступу до ресурсу: <https://brainlab.com.ua/uk/blog-uk/yak-testuvati-veb-sayt-osnovn-etapi-poradi>.
30. Які формати відеофайлів зараз існують? [Електронний ресурс] – Режим доступу до ресурсу: <https://userua.org.ua/yaki-formaty-videofajliv-zaraz-isnuyut>.

ДОДАТКИ

Лістинг Form1.cs

```

using Microsoft.VisualBasic;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Diagnostics;
using System.Drawing;
using System.Drawing.Text;
using System.IO;
using System.Linq;
using System.Runtime.InteropServices;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using static System.Windows.Forms.VisualStyles.VisualStyleElement;
namespace WindowsFormsApp1
{
    public partial class Form1 : Form
    {
        private bool isFullScreen = false;
        private FullScreenForm fullScreenForm;
        private double currentVideoPosition = 0;
        private string currentVideoURL = string.Empty;

        private PrivateFontCollection privateFonts = new PrivateFontCollection();
        private AlbumManager albumManager;
        List<string> paths = new List<string>(); // Змінили на List<string>
        List<string> files = new List<string>(); // Змінили на List<string>
        public Form1()
        {
            InitializeComponent();
            InitializeMediaPlayer();
            InitializeContextMenu();
            InitializeVolume();
            InitializeAlbumContextMenu();
            InitializeFullScreenButtons();
            string albumFilePath = @"C:\Users\User\Desktop\DiplomaTest\Win-
dowsFormsApp1\WindowsFormsApp1\Albums\Albums.txt";
            albumManager = new AlbumManager(albumFilePath);
            trackVolume.ValueChanged += trackVolume_ValueChanged;
            LoadCustomFont();
            ApplyCustomFont(label1);
        }
        private void InitializeFullScreenButtons()
        {
            btnFullScreen.Click += btnFullScreen_Click;
        }
        private void LoadCustomFont()
        {
            // Завантаження шрифту з ресурсів
            var fontBytes = Properties.Resources.NEON;
            var fontData = Marshal.AllocCoTaskMem(fontBytes.Length);
            Marshal.Copy(fontBytes, 0, fontData, fontBytes.Length);
            privateFonts.AddMemoryFont(fontData, fontBytes.Length);
            Marshal.FreeCoTaskMem(fontData);
        }
        private void ApplyCustomFont(Label label)
        {
            // Встановлення користувацького шрифту для label1
            if (privateFonts.Families.Length > 0)

```

```

        {
            label.Font = new Font(privateFonts.Families[0], 16); // Встановить по-
            трібний розмір шрифту
        }
    }
    private void InitializeAlbumContextMenu()
    {
        musicList.ContextMenuStrip = new ContextMenuStrip();
        musicList.MouseDown += musicList_MouseDown;
    }
    private void InitializeVolume()
    {
        trackVolume.Value = 50; // Встановлюємо початкове значення на 50
        int initialVolume = 100 - trackVolume.Value; // Інвертуємо значення для гуч-
        ності
        playerMusic.settings.volume = initialVolume;
        lblMusicVolume.Text = initialVolume.ToString() + "%";
    }
    private void InitializeMediaPlayer()
    {
        player.uiMode = "none"; // Приховування панелі керування
        playerMusic.uiMode = "none";
    }
    private void InitializeContextMenu()
    {
        contextMenu.Items.AddRange(new ToolStripMenuItem[] { FavouriteMenuItem });
        track_list.ContextMenuStrip = contextMenu;
    }

    private void btnPlaying_Click_1(object sender, EventArgs e)
    {
        if (!IsMusicPlaying())
        {
            indicator.Top = btnPlaying.Top + 10;
            bunifuPages1.SetPage(0);
        }
        else
        {
            MessageBox.Show("You cannot switch to this tab while music is playing.",
            "Warning", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        }
    }

    private void btnExplore_Click_1(object sender, EventArgs e)
    {
        if (!IsMusicPlaying())
        {
            indicator.Top = btnExplore.Top + 10;
            bunifuPages1.SetPage(1);
        }
        else
        {
            MessageBox.Show("You cannot switch to this tab while music is playing.",
            "Warning", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        }
    }

    private bool IsMusicPlaying()
    {
        return playerMusic.playState == WMPLib.WMPPlayState.wmppsPlaying;
    }

    private bool IsVideoAlreadyInList(string fileName)
    {
        return track_list.Items.Contains(fileName);
    }

    private void btn_open_Click(object sender, EventArgs e)

```

```

{
    OpenFileDialog ofd = new OpenFileDialog();
    ofd.Multiselect = true;
    ofd.Filter = "Video Files|*.mp4;*.avi;*.mkv;*.mov;*.wmv";

    if (ofd.ShowDialog() == System.Windows.Forms.DialogResult.OK)
    {
        files = ofd.SafeFileNames.ToList();
        paths = ofd.FileNames.ToList();

        track_list.Items.Clear(); // Очистка списку перед додаванням нових відео
        foreach (string file in files)
        {
            if (IsVideoAlreadyInList(file))
            {
                MessageBox.Show("This video is already in the list.", "Warning",
                MessageBoxButtons.OK, MessageBoxIcon.Warning);
                continue;
            }

            string extension = Path.GetExtension(file).ToLower();
            if (extension == ".mp4" || extension == ".avi" || extension == ".mkv"
            || extension == ".mov" || extension == ".wmv")
            {
                track_list.Items.Add(file);
            }
            else
            {
                MessageBox.Show("Only video files are allowed.", "Warning", Mes-
                sageBoxButtons.OK, MessageBoxIcon.Warning);
            }
        }
    }
}

private void track_list_SelectedIndexChanged(object sender, EventArgs e)
{
    if (track_list.SelectedIndex >= 0 && track_list.SelectedIndex < paths.Count)
    {
        player.URL = paths[track_list.SelectedIndex];
        player.Ctlcontrols.play();
        lblPlaying.Text = "Playing - " + track_list.SelectedItem.ToString(); //
        Додано для відображення назви відео
        timer1.Start();

        // Оновлення progressBar і лейблів
        progressBar1.Maximum = (int)player.Ctlcontrols.currentItem.duration;
        progressBar1.Value = 0;
        lbl_track_start.Text = "0:00";
        lbl_track_end.Text = player.Ctlcontrols.currentItem.durationString;

        trackBar1.Value = 15;
        volume.Text = trackBar1.Value.ToString() + "%";
    }
    else
    {
        MessageBox.Show("Selected index is out of range or no video files are
        available.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

```

Лістинг AlbumManager.cs

```

public void AddSongToAlbum(string album, string song)
{
    if (albums.ContainsKey(album))
    {
        // Перевіряємо, чи вже існує пісня в будь-якому іншому альбомі
        foreach (var existingAlbum in albums)
        {
            if (existingAlbum.Value.Contains(song))
            {
                MessageBox.Show($"The song '{song}' is already in the album '{existingAlbum.Key}'.", "Duplicate Song", MessageBoxButtons.OK, MessageBoxIcon.Warning);
                return;
            }
        }

        // Якщо пісні немає в інших альбомах, додаємо її до поточного альбому
        if (!albums[album].Contains(song))
        {
            albums[album].Add(song);
            SaveAlbums();
            MessageBox.Show($"Song '{song}' added to album '{album}'.", "Success", MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
        else
        {
            MessageBox.Show($"The song '{song}' is already in the album '{album}'.", "Duplicate Song", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        }
    }
}

public void CreateNewAlbum()
{
    string albumName = Interaction.InputBox("Enter album name:", "Create New Album", "");
    if (!string.IsNullOrEmpty(albumName) && !albums.ContainsKey(albumName))
    {
        albums.Add(albumName, new List<string>());
        SaveAlbums();
        MessageBox.Show($"Album '{albumName}' created successfully.", "Success", MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
    else
    {
        MessageBox.Show("Invalid album name or album already exists.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

public void DeleteAlbum(string albumName)
{
    if (!string.IsNullOrEmpty(albumName) && albums.ContainsKey(albumName))
    {
        albums.Remove(albumName);
        SaveAlbums();
        MessageBox.Show($"Album '{albumName}' deleted successfully.", "Success", MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
    else
    {
        MessageBox.Show("Invalid album name or album does not exist.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

```

```

    }
}

public void RemoveSongFromAlbum(string album, string song)
{
    if (albums.ContainsKey(album))
    {
        albums[album].Remove(song);
        SaveAlbums();

        // Показуємо повідомлення про успішне видалення
        MessageBox.Show($"Song '{song}' has been removed from album '{album}'",
            "Success", MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
}

private void SaveAlbums()
{
    using (StreamWriter writer = new StreamWriter(filePath))
    {
        foreach (var album in albums)
        {
            writer.WriteLine(album.Key);
            foreach (var song in album.Value)
            {
                writer.WriteLine("    " + song);
            }
        }
    }
}

```