Міністерство освіти і науки України

Національний університет «Львівська політехніка»

кафедра СКС



**Курсовий проект**

На тему:

«Розробка програмного забезпечення для керування аудіофайлами.

Розробка програми, що дозволяє керувати аудіофайлами та виконувати різні дії з їх обробкою на комп'ютері з операційною системою Windows.»

З дисципліни «Системне програмне забезпечення»

Виконала:

студентка групи КІ-308

Продан К.О.

Прийняв:

к.т.н., доцент каф. ЕОМ

Олексів М. В.

Львів – 2025

**Завдання**

Тема 37. Розробка програмного забезпечення для керування аудіофайлами.

Розробка програми, що дозволяє керувати аудіофайлами та виконувати різні дії з їх обробкою на комп'ютері з операційною системою Windows.

**Зміст**

**Анотація**

У даному проєкті розглянуто процес створення програми, що дозволяє здійснювати ефективне керування аудіофайлами на комп'ютері з операційною системою Windows. У роботі детально описано основні етапи проектування програмного забезпечення, включаючи аналіз вимог користувачів, проектування архітектури системи та інтерфейсу користувача, реалізацію функціональності та процес тестування.

Розробка функціональності програми здійснювалася з використанням мови програмування C++ Для створення інтерфейсу, а також для роботи роботи з драйвером звукової карти використовувалася бібліотека "SFML".Для вибору папки з аудіофайлами при запуску програми використовувалась бібліотека “tinyfiledialogs” Програму можна умовно поділити 3 частини:

1. Вибір папки що буде відтворюватись,

2. Отримання вікном шляху до папки, що буде відтворюватись, створення об’єктів які використовуватиме вікно(текст, текстури, фони) й визначення їх позицій у вікні.

3. Головний цикл вікна, що визначає що у ньому відображається та у якому порядку, задає логіку роботи кнопок та повзунків, а також логіку обробки подій.

У результаті проведеної роботи була створена програма з багатими можливостями для управління аудіофайлами, яка може бути корисною для користувачів, що працюють з великою кількістю аудіофайлів. Програма відрізняється високою продуктивністю, надійністю та простотою використання, що робить її ефективним інструментом для роботи з аудіофайлами.

**Вступ**

Розробка програмного забезпечення для роботи з аудіоданими та відтворення аудіофайлів на комп'ютері з операційною системою Windows є актуальною задачею у сфері програмування. Основною метою цього проекту є створення високорівневої бібліотеки для обробки аудіоданих і їх відтворення, використовуючи програмний інтерфейс Windows API та стандартну бібліотеку "winmm.lib" для роботи з драйвером звукової карти.

Для демонстрації можливостей високорівневої бібліотеки буде розроблено зручний та функціональний інтерфейс користувача, який дозволить здійснювати контроль відтворення аудіофайлів у реальному часі. Це програмне забезпечення призначено для роботи з аудіоданими та їх відтворення в операційній системі Windows, забезпечуючи користувачам простий і ефективний спосіб керування аудіофайлами.

Сучасні користувачі все частіше стикаються з необхідністю обробки великих обсягів аудіоданих для різноманітних завдань, включаючи створення музичних плейлистів, монтаж аудіофайлів, запис та редагування звуку. Таким чином, потреба у потужних і гнучких інструментах для роботи з аудіо стає все більш очевидною. Запропоноване програмне забезпечення не тільки спрощує ці завдання, але й підвищує ефективність роботи завдяки інтуїтивно зрозумілому інтерфейсу та широкому спектру можливостей.

Розробка високорівневої бібліотеки для роботи з аудіоданими та їх відтворення є складним і важливим процесом, що потребує ретельного аналізу вимог, проектування архітектури, розробки функціональності та тестування. В результаті успішних рішень і правильної реалізації, бібліотека стане корисним та зручним інструментом для роботи з аудіоданими та їх відтворення на комп'ютері з операційною системою Windows. Це програмне забезпечення підійде всім користувачам, які активно працюють з аудіо.

**1.Теоретичні відомості**

Розробка аудіоплеєрів є актуальною і поширеною задачею в галузі програмного забезпечення. Аудіоплеєр — це програма, призначена для відтворення аудіофайлів різних форматів і надання користувачу зручного інтерфейсу для керування відтворенням.

При розробці аудіоплеєра є кілька важливих аспектів, на які варто звернути увагу, зокрема:

1. **Використання мови програмування та фреймворків**. Для розробки аудіоплеєра можна використовувати різні мови програмування, такі як C++, Java, Python, тощо.
2. **Користувацький інтерфейс**. Користувач звертає увагу на реалізацію графічного інтерфейсу, тому варто обрати рішення з використанням графічних бібліотек, таких як Win32API, Windows Forms, Qt, тощо.
3. **Функціональні можливості**. Ключовою перевагою одного аудіоплеєра над іншим є наявність додаткового функціоналу. Це може бути як звичайне відтворення аудіофайлів різних форматів, так і додаткові можливості у вигляді еквалайзера, фільтрів, відтворення онлайн-трансляцій, відтворення радіо тощо.

Основна мета розробки програмного забезпечення полягає в забезпеченні користувачу зручного інтерфейсу для моніторингу та керування відтворенням аудіофайлів.

**РОЗДІЛ 1**

**1.1.** **Аналіз завдання та способи його вирішення**

**1. Визначення вимог користувачів.** Першим кроком у розробці аудіоплеєра є збір та аналіз вимог користувачів. Це включає виявлення основних потреб і очікувань, таких як підтримка різних форматів аудіофайлів (MP3, WAV,OGG, FLAC тощо), зручний інтерфейс користувача, можливості створення плейлистів, налаштування звуку та інші функціональні можливості.

**2. Аналіз ринку та існуючих рішень.** Необхідно провести огляд існуючих аудіоплеєрів на ринку, таких як VLC Media Player, Winamp, AIMP, iTunes та інші. Це дозволить зрозуміти, які функції є стандартними, а які можуть стати конкурентною перевагою. Важливо звернути увагу на сильні та слабкі сторони цих продуктів, щоб уникнути їхніх недоліків та покращити власне рішення.

**3. Вибір технологій.** Після визначення вимог та аналізу ринку необхідно обрати відповідні технології для реалізації аудіоплеєра. Це включає вибір мови програмування (наприклад, C++, Java, Python) та графічних бібліотек для створення інтерфейсу користувача (Win32API, Qt, Windows Forms). Також слід розглянути можливість використання додаткових бібліотек для роботи з аудіо, таких як "SFML" для Windows.

**4. Проектування архітектури.** На цьому етапі необхідно розробити архітектуру аудіоплеєра, яка буде включати основні модулі та їх взаємодію. Основні компоненти можуть включати:

* **Модуль відтворення аудіо**: відповідає за завантаження і відтворення аудіофайлів.
* **Модуль інтерфейсу користувача**: забезпечує графічний інтерфейс для взаємодії користувача з програмою.
* **Модуль управління файлами**: відповідає за організацію та управління аудіофайлами, включаючи створення і збереження плейлистів.
* **Модуль налаштувань**: дозволяє користувачу змінювати налаштування програми, такі як еквалайзер, фільтри, тощо.

**5. Реалізація функціональності.** Після проектування архітектури розпочинається безпосередня реалізація функціональності аудіоплеєра. Основні завдання включають:

* Розробка та інтеграція модулів для відтворення різних форматів аудіо.
* Створення інтуїтивно зрозумілого та зручного інтерфейсу користувача.
* Реалізація додаткових функцій, таких як еквалайзер, фільтри, підтримка онлайн-трансляцій, тощо.

**6. Тестування.** Тестування є критично важливим етапом у розробці аудіоплеєра. Воно включає перевірку коректності роботи всіх функцій, тестування на різних конфігураціях системи, а також виявлення та виправлення помилок. Види тестування можуть включати модульне тестування, інтеграційне тестування та тестування зручності використання.

**7. Оптимізація та покращення.** Після початкового тестування необхідно провести аналіз продуктивності. Це дозволить виявити можливі проблеми та області для покращення, такі як оптимізація швидкодії, зменшення використання ресурсів системи, удосконалення інтерфейсу користувача, тощо.

**1.2. Огляд методів, існуючих розробок:**

У даному пункті будуть розглянуті 2 програми для резервного копіювання даних та файлів, а саме:

* Короткий опис;
* Основні можливості;
* Головні переваги та недоліки зібрані з форумів та тематичних сайтів.

**Windows Media Player**

Windows Media Player – це вбудований в операційну систему Windows мультимедійний програвач, який дозволяє відтворювати аудіо та відео, а також керувати медіаколекціями.

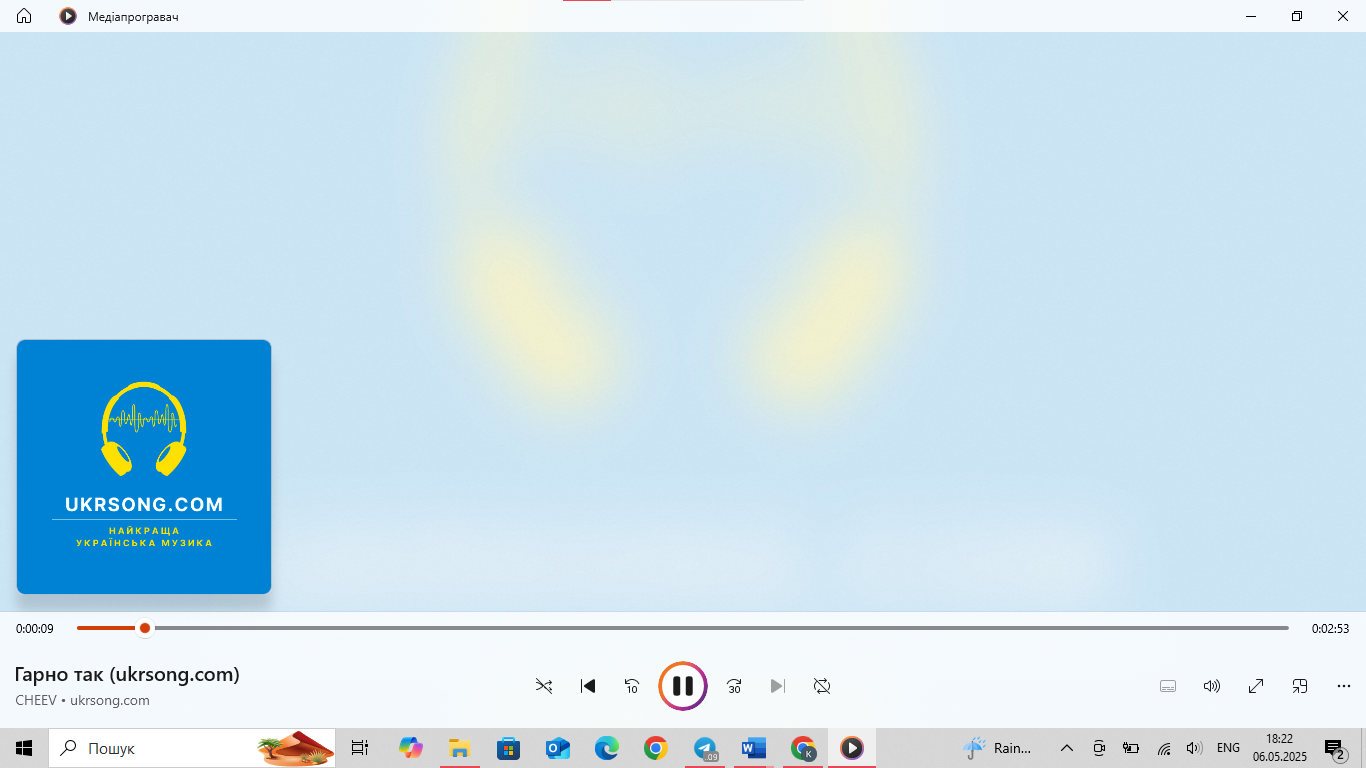


Рис. 1.1 Інтерфейс програми Windows Media Player.

**Функціонал:**

a) Відтворення аудіофайлів популярних форматів (MP3, WMA);

b) Базова організація медіатеки;

c) Створення плейлистів;

d) Синхронізація з пристроями;

e) Запис аудіо CD.

**Переваги та недоліки:**

Переваги:

* Безкоштовно доступний у системі Windows;
* Простий у використанні;
* Інтегрується з операційною системою;
* Не потребує встановлення.

Недоліки:

* Обмежені можливості обробки аудіо;
* Обмежена підтримка форматів;
* Відсутність розширеного редагування тегів;
* Застарілий інтерфейс.

Windows Media Player доступний безкоштовно як частина Windows

**AIMP**

AIMP – це безкоштовний аудіопрогравач для Windows з фокусом на якісне відтворення аудіо та простий інтерфейс.

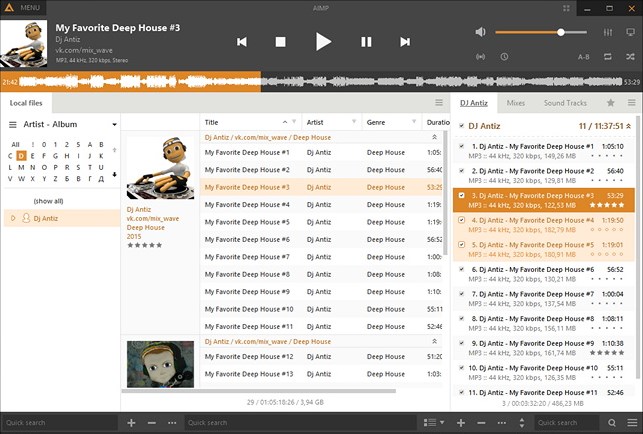


Рис. 1.2. Інтерфейс програми AIMP.

**Основний функціонал:**

a) Відтворення різних аудіоформатів;

b) Управління плейлистами;

c) 18-смуговий еквалайзер;

d) Редагування тегів;

e) Конвертація аудіоформатів.

**Переваги та недоліки:**

Переваги:

* Малий розмір програми;
* Низьке використання системних ресурсів;
* Зручний інтерфейс;
* Підтримка скінів.

Недоліки:

* Обмежені можливості для організації великих колекцій;
* Відсутність розширених функцій редагування аудіо;
* Немає глибокої інтеграції з онлайн-сервісами;
* Базові можливості аудіообробки.

AIMP доступний безкоштовно.

**1.3. Функціональні вимоги до програмного забезпечення**

**1. Відтворення аудіофайлів**

**Опис:**  
Програма повинна забезпечити можливість прослуховування аудіофайлів без необхідності стороннього програмного забезпечення.  
**Функціональні можливості:**  
• Відтворення, пауза та зупинка треку  
• Перемотування вперед/назад  
• Зміна гучності відтворення  
• Підтримка аудіоформатів MP3, WAV, WMA

**2. Формування та керування списком відтворення**

**Опис:**  
Користувач має змогу створювати список треків для відтворення, змінювати його склад та зберігати для подальшого використання.  
**Функціональні можливості:**  
• Додавання/видалення треків до списку  
• Збереження та відкриття плейлисту з файлу  
• Зміна порядку треків у списку

**3. Перегляд інформації про аудіофайл**

**Опис:**  
Для зручності користувача передбачено можливість перегляду базової інформації про трек.  
**Функціональні можливості:**  
• Відображення назви файлу, розміру, тривалості  
• Показ формату, бітрейту та частоти дискретизації

**4. Інтерфейс користувача**

**Опис:**  
Програма повинна мати інтуїтивно зрозумілий інтерфейс, що не потребує додаткових знань з боку користувача.  
**Функціональні можливості:**  
• Графічний інтерфейс із кнопками для основних дій  
• Відображення поточного треку та плейлисту  
• Можливість відкриття/збереження списку треків  
• Мінімалістичний дизайн з базовими підказками

**РОЗДІЛ 2**

**2.1.** **Розробка програми**

**2.1.1.** **Вибір мови та середовища програмування**

Для реалізації програмного забезпечення, що є частиною проекту з розробки програмного забезпечення для резервного копіювання даних, було обрано комбінацію мов програмування та середовищ розробки, що найкращим чином відповідає поставленим завданням та вимогам.

**Мова програмування C++**

Використання мови програмування C++ було обумовлено необхідністю створення ефективного та швидкодіючого консольного додатку для ведення списку файлів для резервного копіювання, виведення статусу елементів списку та здійснення резервного копіювання. Мова C++ надає прямий доступ до системних ресурсів, що дозволяє ефективно працювати з файловою системою та виконувати операції копіювання даних.

**Середовище розробки: Visual Studio 2022**

В якості середовищ розробки обрано Visual Studio 2022 Visual Studio 2022 було обрано через його високий рівень підтримки мови програмування C++ та наявність розширених інструментів для роботи з цією мовою.

**2.1.2. Бібліотеки, які використовуються при написанні програми**

Під час розробки програмного забезпечення для резервного копіювання даних використовувалися різноманітні бібліотеки для забезпечення необхідного функціоналу. Ось список використаних бібліотек для мови програмування C++

**filesystem**: Ця бібліотека дозволяє взаємодіяти з файловою системою. Вона містить функції для роботи з файлами та папками, такі як перевірка наявності файлу, отримання розміру файлу та інші.

**fstream**: Бібліотека fstream дозволяє працювати з файлами. Вона містить класи ifstream для читання з файлу, ofstream для запису в файл та fstream для роботи з файлами у загальному.

**string**: Бібліотека string надає клас string, який представляє рядок символів. Вона містить функції для роботи з рядками, такі як з'єднання, порівняння та отримання підрядків.

**SFML** (скр. від «Simple and Fast Multimedia Library» = «Проста і швидка мультимедійна бібліотека») — це безкоштовна кроссплатформенная мультимедійна бібліотека, написана на мові C++. Вона також доступна та для інших мов: C, C#, .Net, D, Java, Python, Ruby, OCaml, Go та Rust. Представляє собою об'єктно-орієнтований аналог SDL.

З її допомогою можна легко створювати будь-яку 2D-графіку: починаючи від простих одиничних геометричних фігур (наприклад трикутника) і закінчуючи повноцінними іграми-платформерами.

**tinyfiledialogs** -- Надзвичайно простий у використанні файловий браузер для програми C++.

* + 1. **Діаграма класів:**

**Діаграма класів** показує три основні компоненти системи:

* AudioPlayer - ядро програми, що керує відтворенням
* UIManager - відповідає за візуальний інтерфейс
* FileManager - завантажує та керує аудіофайлами

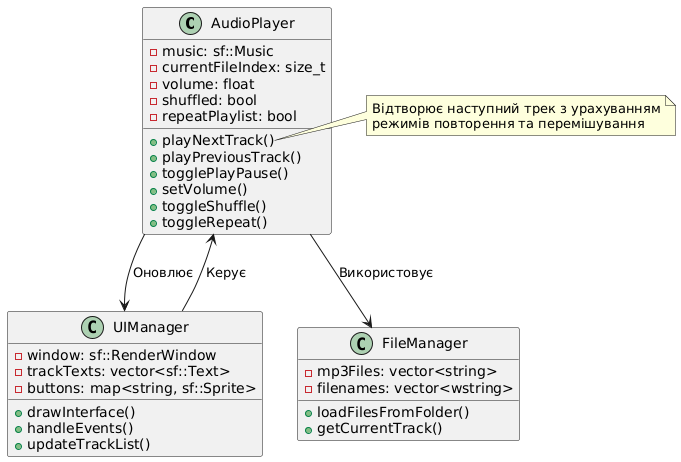


Рис. 2.1. Діаграма класів.

### **2.1.4. Діаграма послідовності (відтворення треку)**

Послідовність дій при відтворенні треку, починаючи з взаємодії користувача з інтерфейсом.

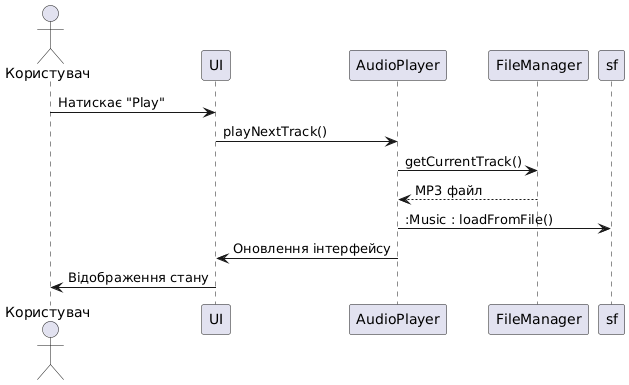


Рис. 2.2. Діаграма послідовності.

### **2.1.5. Діаграма станів (стани програвача)**

### Моделює основні стани аудіоплеєра та переходи між ними.

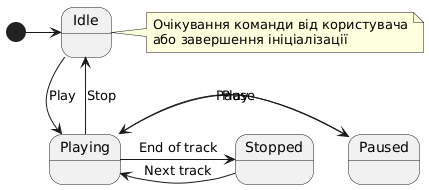


Рис. 2.3. Діаграма послідовності.

### **Діаграма компонентів**

Показує залежності між основним кодом програми, бібліотекою SFML та системними компонентами

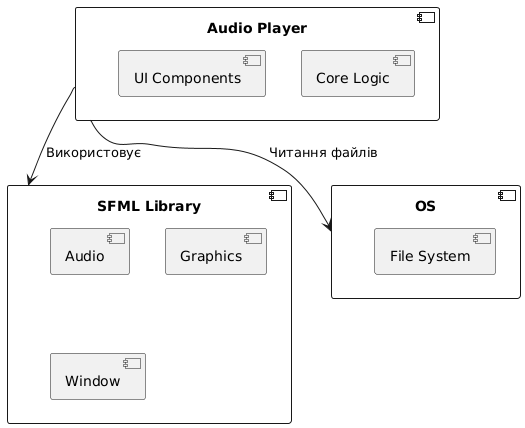


Рис. 2.4. Діаграма компонентів.

### **Діаграма випадків використання**

### Основні сценарії використання програми з точки зору користувача.

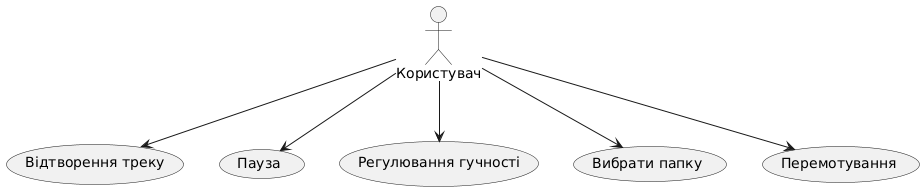


Рис. 2.5. Діаграма випадків використання.

**РОЗДІЛ 3**

**3.1.** **Опис інтерфейсу програми**

Після запуску програми запускається вікно вибору папки з аудіофайлами,

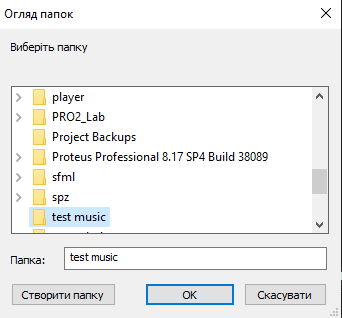


Рис. 3.1. Вибір папки

Якщо вікно вибору було закрито або дана папка не містить підтримуваних аудіофайлів відкривається вікно, що пропонує вибрати іншу папку або закрити програму.

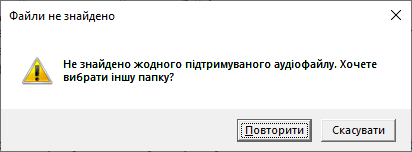
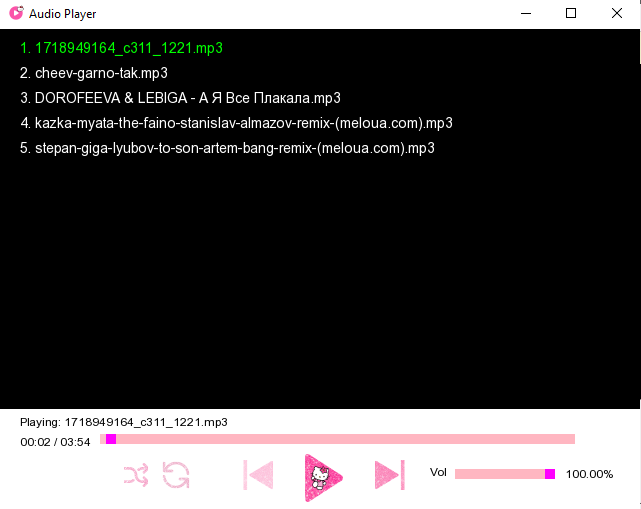


Рис. 3.2. У папці не знайдено файлів підтримуваних розширень

Якщо вибрана папка містить файли підтримуваних аудіоформатів щляхи до файлів зберігаються в плейліст, який зразу ж запускається



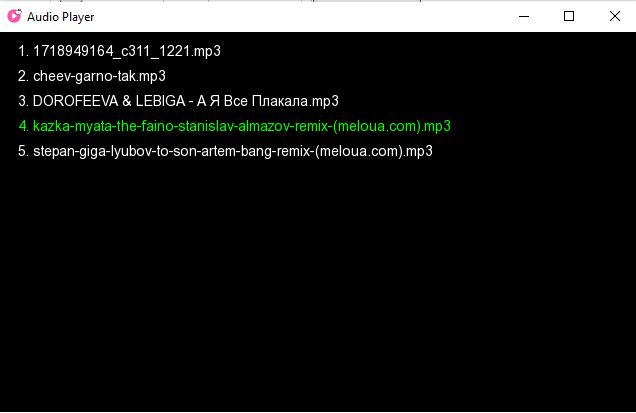
*Рис.3.3. Вигляд вікна аудіопрогравача*

Прогрес-бар- має сталу довжину, для треків різної довжина, може бути , нажаття на прогрес бар переставляють на час, що відпоідає позиції прогрес-бара, також повзунок на прогрес-барі може бути перетягненим й тоді час трека оновиться при відпусканні лівої клавіші миші.



*Рис.3.4. Прогрес-бар*

Список треків – прокручується колесиком миші при кількості треків>10. Поточний трек підсвічується зеленим кольором



*Рис.3.5. Список треків*

Нову гучність відтворення можна встановлювати нажаттям на фон звуку, нажаттям на фон й перетягуванням повзунка з цього місця, або за допомогою клавіш +- або up down, що збільшують або змененшують звук на 10%



*Рис.3.6. Встановлена нова гучність відтворення*

Кнопка «play» дозволяє зупиняти трек або продовжувати його відтворення й міняє свою текстуру залежно від її стану

*Рис.3.7. Режими повтору*

Кнопка «previous» дозволяє увімкнути попередній трек у плейлісті, якщо цей трек не є першим. Якщо трек є першим і повторення вимкнуто, то кнопка стає неактивною й міняє свою текстуру неактивну. Якщо трек не є першим кнопка стає активною й міняє свою текстуру на активну. При увімкнені повторення стає активною й міняє свою текстуру на активну



*Рис.3.8. неактивна та активна кнопка* «previous»

Кнопка «next» дозволяє увімкнути наступний трек у плейлісті, якщо цей трек не є останнім. Якщо трек є останнім і повторення вимкнуто, то кнопка стає неактивною й міняє свою текстуру неактивну. Якщо трек не є останнім кнопка стає активною й міняє свою текстуру на активну. При увімкнені повторення стає активною й міняє свою текстуру на активну



*Рис.3.9. неактивна та активна кнопка* «next»

Кнопка «repeat» включити або вимкнути режим повторення треків у плейлісті.Якщо режим повторення увімкнутий то перехід з першого трека на попередній увімкне останній трек в плейлісті, а перехід з останнього трека на наступний увімкне перший режим у плейлісті. Кнопка міняє свою текстуру залежно віл стану повторення.



*Рис.3.10. Деактивований та активований режим повторення*

Кнопка «shuffle» дозволяє включити режим випадкового відтворення наступних треків, уникаючи початкового порядку.Кнопка міняє свою текстуру залежно від її стану

**

*Рис.3.11. Деактивований та активований режим випадкового порядку відтворення треків*

**3.2.** **Тестування програми**

Тестування програмного забезпечення має велике значення в процесі розробки продукту, оскільки на цьому етапі виявляються помилки, які були зроблені на попередніх етапах. Цей етап дозволяє поліпшити різні аспекти продукту, наприклад, його інтерфейс, і знаходить слабкі місця, якщо такі є, для подальшого їх виправлення.

Після завершення реалізації розробки програмного забезпечення для відтворення аудіофайлів на комп'ютері з ОС “Windows”, можна зробити висновок, що всі поставлені задачі були виконані.

Під час тестування було виявлено та виправлено баги, пов’язані із неправильним відтворенням треків та некоректною часовою лінією.