

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

бакалавр

(ступінь вищої освіти)

на тему ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ
МЕРЕЖЕВОГО АУДІОПЛЕЄРУ
NETWORK AUDIO PLAYER SOFTWARE

Виконав(ла): студент(ка) 4 курсу, групи КНТ-120
Спеціальності 121 Інженерія програмного
забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

ЛАТИШЕВ Є.О.

(ПРИЗВИЩЕ та ініціали)

Керівник ЗАЙКО Т.А.

(ПРИЗВИЩЕ та ініціали)

Рецензент КОЦУР М.І.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет КНТ
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
Спеціальність 121 Інженерія програмного забезпечення
(код і найменування)
Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
Сергій СУББОТІН
“ ” 2024 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

ЛАТИШЕВА Євгенія Олексійовича

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Програмне забезпечення мережевого аудіоплеєру. Network Audio Player Software

керівник проєкту (роботи) к.т.н., доцент, ЗАЙКО Тетяна Анатоліївна,
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “ 24 ” квітня 2024 року № 168

2. Строк подання студентом проєкту (роботи) 10 червня 2024 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Матеріали і методи. 3. Опис програми. 4. Експлуатація, тестування та експериментальне дослідження програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів)

Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРІЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	ЗАЙКО Т.А., доцент		
Нормоконтроль	ЛИПОВЕЦЬ М.В., асистент		

7. Дата видачі завдання “ 15 ” квітня 2024 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій розробки.	2 тиждень	Розділ 2
4	Розробка структури програми.	2 тиждень	Розділ 3
5	Розробка програми.	3-4 тижні	Розділи 3, 4
6	Тестування та експериментальне дослідження програмного забезпечення.	5 тиждень	Розділ 4
7	Оформлення пояснювальної записки та документів до неї.	6 тиждень	Додатки
8	Нормоконтроль та рецензування.	7 тиждень	
9	Захист роботи.	8 тиждень	

Студент(ка)

(підпис)

Євгеній ЛАТИШЕВ
(Імя ПРІЗВИЩЕ)

Керівник проєкту (роботи)

(підпис)

Тетяна ЗАЙКО
(Імя ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра:
84 с., 30 рис., 3 табл., 3 дод., 18 джерел.

ANDROID STUDIO, JAVA, МЕРЕЖЕВИЙ АУДИОПЛЕЄР,
ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, СЕРЕДОВИЩЕ РОЗРОБКИ.

Об'єкт дослідження – процес розробки програмного забезпечення для відтворення музики.

Предмет дослідження – програмні засоби для підтримки процесу відтворення музики.

Мета роботи – розробка програмного забезпечення мережевого аудіоплеєру.

Матеріали, методи та технічні засоби: мова програмування Java, середовище розробки Android Studio.

Результати. Розроблено програмне забезпечення мережевого аудіоплеєру за допомогою мови програмування Java та середовища розробки Android Studio.

Висновки. Виконано проєктування програмного забезпечення мережевого аудіоплеєру. Розроблено програмне забезпечення мережевого аудіоплеєру. Здійснено тестування розробленого програмного забезпечення мережевого аудіоплеєру.

Галузь використання – програмні засоби для підтримки процесу відтворення музики.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 84 pages, 3 tables, 30 figures, 3 appendixes, 18 sources.

ANDROID STUDIO, JAVA, NETWORK AUDIO PLAYER, SOFTWARE, INTEGRATED DEVELOPMENT ENVIRONMENT.

The object of research is the process of developing software for playing music.

The subject of the research is software for supporting the process of music playback.

The purpose of this work is to develop the software for a network audio player.

Materials, methods and technical tools: Java programming language, Android Studio development environment.

Results. The software tools for the audio player using the Java programming language and the Android Studio development environment have been developed.

Conclusions. The software for the audio player has been designed. The software for the audio player has been developed. The developed software for the audio player has been tested.

Scope of use – software tools to support the music playback process.

ЗМІСТ

	С.
Перелік скорочень та умовних позначок	8
Вступ	9
1 Аналіз предметної області	11
1.1 Поняття про аудіоплеєри та програмне забезпечення для відтворення музики	11
1.2 Аналіз програмного забезпечення для відтворення музики	19
1.3 Висновки за розділом 1	31
2 Матеріали і методи	33
2.1 Вибір мови програмування	33
2.2 Вибір середовища розробки для створення програмного забезпечення мережевого аудіоплеєру	34
2.3 Висновки за розділом 2	37
3 Опис програми	39
3.1 Структура програмного забезпечення мережевого аудіоплеєру	39
3.2 Функціонування програмного забезпечення мережевого аудіоплеєру ..	42
3.3 Розробка бази даних програмного забезпечення мережевого аудіоплеєру	44
3.4 Проєктування інтерфейсу програмного забезпечення мережевого аудіоплеєру	45
3.5 Висновки за розділом 3	48
4 Експлуатація, тестування та експериментальне дослідження програми	50
4.1 Призначення й умови застосування програми	50
4.2 Характеристики програми мережевого аудіоплеєру	51
4.3 Інструкція по експлуатації програми	51
4.3.1 Звернення до програми	51
4.3.2 Вхідні й вихідні дані	52
4.3.3 Повідомлення	52
4.4 Виконання програмного забезпечення мережевого аудіоплеєру	52

4.5 Тестування програмного забезпечення мережевого аудіоплеєру	55
4.6 Висновки за розділом 4	57
Висновки	58
Перелік джерел посилань	61
Додаток А Технічне завдання	63
Додаток Б Фрагмент тексту програми	67
Додаток В Слайди презентації	77

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

DAP – Digital Audio Player;

MVC – Model-View-Controller;

БД – база даних;

IoT – Інтернет речей;

ПЗ – програмне забезпечення;

ШІ – штучний інтелект.

ВСТУП

Аудіоплеєри набули широко використання для відтворення музичних файлів, оскільки такі засоби забезпечують зручність, доступність, широкий вибір контенту та можливості персоналізації. Програмне забезпечення аудіоплеєрів надає простий та зручний спосіб слухати улюблену музику безпосередньо з різних пристроїв [1]-[6].

Програмне забезпечення аудіоплеєрів, як правило, є доступним для смартфонів та інших портативних пристроїв, що дозволяє користувачам мати доступ до своєї музики в будь-який час та в будь-якому місці. Програмне забезпечення аудіоплеєрів дозволяє легко керувати музикою та слухати її в будь-який момент. Таке програмне забезпечення, як правило, має простий та зручний інтерфейс, що робить його легким у використанні для користувачів будь-якого рівня, воно дозволяє швидко шукати та відтворювати музику без зайвих складнощів [1]-[6]. Аудіоплеєри дозволяють слухати музику без підключення до Інтернету, що особливо корисно в умовах, коли немає доступу до Wi-Fi або мобільного зв'язку. Багато аудіоплеєрів також підтримують відтворення аудіо книг та подкастів, роблячи їх універсальними інструментами для прослуховування різних типів аудіо контенту [1]-[6].

Отже, хоча існують альтернативні методи прослуховування музики, аудіоплеєри залишаються актуальними через свою зручність, мобільність та можливість керування власною колекцією музики [1]-[6].

Проте деяке програмне забезпечення для відтворення музики може мати обмежені функції, зокрема, не підтримувати певні формати файлів або не мати розширених налаштувань еквайзера. Інше програмне забезпечення аудіоплеєрів може вимагати велику кількість системних ресурсів, що може призвести до сповільнення роботи комп'ютера або мобільного пристрою. Деякі аудіоплеєри можуть не підтримувати певні аудіокодеки або контейнери, що може призвести до неможливості відтворення певних аудіофайлів без попереднього конвертування [1]-[6].

Тому актуальною є розробка програмного забезпечення мережевого аудіоплеєру. У дипломній кваліфікаційній роботі бакалавра розв'язується актуальне завдання розробки програмного забезпечення мережевого аудіоплеєру.

Для досягнення поставленої мети у кваліфікаційній роботі бакалавра необхідно розв'язати такі задачі:

- виконати аналіз предметної області та програмних засобів для відтворення музики;
- здійснити проєктування програмного забезпечення мережевого аудіоплеєру;
- створити програмне забезпечення мережевого аудіоплеєру;
- виконати тестування розробленого програмного забезпечення мережевого аудіоплеєру.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Поняття про аудіоплеєри та програмне забезпечення для відтворення музики

Портативний аудіоплеєр або аудіопрогравач являє собою електронний пристрій для прослуховування музики, який може зберігати значну кількість аудіофайлів. Аудіоплеєри залишаються зручними для прослуховування музики під час занять або тренувань. Деякі моделі програвачів мають вбудований Bluetooth або Wi-Fi, але для їхньої роботи не потрібне підключення до мережі Інтернет. Для завантаження пісень, альбомів або плейлистів, як правило, використовується підключення до комп'ютера за допомогою USB-кабелю. Програвачі з підключенням до Інтернету можуть завантажувати та транслювати музику бездротовим шляхом. Моделі з підтримкою Bluetooth можуть бути підключені до бездротових навушників, що уникає ризику заплутування кабелю [1]-[6].

Сучасні аудіоплеєри зазвичай мають вбудовані носії інформації, які забезпечують достатньо місця для зберігання музики та є менш чутливими до пошкоджень під час активних тренувань. Попередні моделі аудіоплеєрів використовували рухомі частини на жорсткому диску, які можуть бути чутливими до пошкоджень при сильних рухах. Деякі програвачі допускають використання карт пам'яті для розширення обсягу зберігання інформації у вигляді музичних файлів [1]-[6].

Аудіоплеєри використовують літій-іонні батареї, так само, як і смартфони. Оскільки вони зазвичай призначені лише для прослуховування музики, аудіоплеєри мають значно довший час автономної роботи від батареї, ніж смартфони. MP3-плеєри доступні у різних розмірах і формах. Деякі моделі постачаються зі затискачами або браслетами, щоб їх можна було зручно прикріпити до одягу або тіла під час подорожей. Деякі аудіоплеєри водонепроникні, що дозволяє їм захистити одяг і навіть витримати занурення в воду, наприклад, у басейні [1]-[6].

Аудіоплеєри є типовим прикладом цифрової технології. Це означає, що звук зберігається у цифровому форматі. У порівнянні, компакт-диски також використовують цифрову технологію, але старіші носії музики, такі як платівки або касети, використовують аналогові методи запису. Ключем до оцифрування музики (або будь-якої іншої форми звуку) є форма музики за номерами, яка називається семплуванням. Під час запису комп'ютер прослуховує записаний музичний трек і відбирає гучність і частоту. Кожен звук з заданою частотою аналізується та перетворюється на число. Цей процес виконується електронною схемою, яка називається аналого-цифровим перетворювачем. Цей конвертер перетворює аудіо (аналоговий) звук у цифровий потік, який потім зберігається у відповідний файл. Зворотний процес відбувається, коли файл або компакт-диск відтворюється пізніше. Цифро-аналоговий перетворювач перетворює числа назад в аналоговий електричний сигнал, який перетворюється на звук, коли він потрапляє в динаміки. Чим швидше ваш комп'ютер збирає вибірки (вища частота дискретизації), тим більше даних він може захопити одночасно (розрядність) і чим більше деталей комп'ютер захоплює за секунду (вища швидкість передачі даних), тим кращими є відповідні цифрові файли [1]-[6].

Цифровий аудіопрогравач (Digital Audio Player, DAP) являє собою електронний пристрій, який може зберігати та відтворювати цифрові аудіо файли [1]-[6].

На відміну від програвачів MP3, цифрові аудіопрогравачі можуть легко відтворювати різноманітні аудіоформати. Більшість сучасних DAP можуть відтворювати будь-який аудіоформат, включаючи MP3, файли з втратами, аудіо низької та повної роздільної здатності [1]-[6].

DAP пройшли значні зміни, покращуючи якість конструкції та додавши більше функціональності. Більшість з них підтримує сенсорні кнопки, сенсорні екрани, збалансовані виходи, а також бездротове потокове передавання за допомогою Wi-Fi та Bluetooth [1]-[6].

Завдяки широкому спектру аудіоформатів, які доступні на ринку, цифрові аудіопрогравачі пристосувалися до обробки та декодування різних технологій [1]-[6].

Проаналізуємо різні типи файлів аудіоданих, що обробляються аудіоплеєрами. Існують три основних типи аудіоформатів: аудіоформати без стиснення, формати стисненого аудіо без втрат, аудіоформати зі стисненням із втратами [1]-[6].

Формат аудіо без стиснення, як зрозуміло з назви, представляє собою тип аудіофайлу, який не використовує жодних методів стискання. Ця категорія аудіоданих записується без будь-яких змін, що дозволяє зберігати детальну інформацію про звуковий матеріал і робить його надзвичайно надійним. Формати без стиснення потребують більше місця для зберігання. Наприклад, до таких форматів відносяться WAV, AIFF і PCM [1]-[6].

Формат аудіофайлу зі стисненням без втрат вимагає більше місця для зберігання, тому рекомендується стиснути та зменшити розмір цих файлів перед їх збереженням. Після стиснення отриманий файл називається стисненим аудіоформатом без втрат. Він зберігає оригінальні дані, навіть при стисненні, яке призводить до невеликого зменшення розміру. Такий тип файлу дозволяє відновити аудіо до його початкового розміру. Аудіоформати зі стисненням без втрат включають такі формати, як ALAC, FLAC і WMA [1]-[6].

Якщо потрібні компактні аудіофайли, слід обрати аудіоформат зі стисненням із втратами даних. Метод стиснення з втратами забезпечує високий рівень стиснення цифрового звуку, що призводить до зменшення розміру файлу. Проте, він призводить до втрати якості аудіоданих, оскільки деякі аудіофрагменти видаляються, що впливає на якість звуку. Ці формати включають файли MP3, OGG, AAC і WMA з втратами даних [1]-[6].

Кодек Bluetooth являє собою сигнал, який передається з пристрою, такого як комп'ютер, ноутбук або смартфон, на динаміки, навушники або інші пристрої виведення [1]-[6].

Функція кодера Bluetooth полягає в кодуванні інформації для легкого передавання, а потім в декодуванні отриманих даних приймачем. Для користувачів зручною є можливість підтримки різних кодеків Bluetooth у цифрових аудіопрогравачах. Однак не всі DAP підтримують розширені кодеки Bluetooth, такі як Qualcomm aptX або Sony LDAC. Перед вибором DAP важливо перевірити характеристики та переконатися, що він підтримує вибраний кодек Bluetooth [1]-[6].

З іншого боку, вищий бітрейт призводить до меншого стиснення та вищої якості звуку. Вибір між цифровим аудіоплеєром і смартфоном може бути складною задачею. Основна причина, чому більшість користувачів обирають DAP над смартфонами, полягає в якості звуку. Багато смартфонів мають низьку якість звуку, оскільки виробники зазвичай акцентуються на вдосконалення інших функцій, таких як висока роздільна здатність камер, якісні дисплеї та тривалий час автономної роботи [1]-[6].

Цифрові аудіоплеєри є оптимальним варіантом для тих користувачів смартфонів, які шукають покращену якість звуку під час подорожей. Крім здатності забезпечити вищу якість аудіо, портативні аудіопрогравачі (DAP) мають компактний дизайн та більший обсяг пам'яті для зберігання музики. Вони також пропонують різноманітні функції, включаючи збалансований аудіовихід та підтримку високоякісного аудіо [1]-[6].

Хоча деякі виробники смартфонів представляють нові моделі з вдосконаленою аудіосистемою, смартфони все ще відстають в цьому відношенні [1]-[6].

З виникненням смартфонів, які пропонують кращу якість звуку, здається, що DAP втрачають свою популярність. Проте, враховуючи сучасні ринкові тенденції, здається, що смартфонам знадобиться час, щоб догнати цифрові аудіоплеєри [1]-[6].

Музичними плеєрами можуть бути багато різних пристроїв. Це можуть бути інструменти, що відтворюють музику, музичні автомати зі стилусом, що крутяться навколо диска, або грамофони. Розглянемо останні музичні

програвачі, які широко використовуються у наш час, а також їх апаратне та програмне забезпечення [1]-[6].

Програмне музичне відтворення є одним з шляхів прослуховування музики. Технічно, для цього потрібне відповідне обладнання, але воно використовується не лише для відтворення, а й для програмування [1]-[6]. Зазвичай такі пристрої мають кілька спільних особливостей [1]-[6].

Першою особливістю є підтримка різних форматів файлів, таких як .mp3, .flac, .ogg, .wav, .wma, .aac, .m4a, .aif і багато інших. Підтримку нових форматів файлів можна очікувати під час наступного оновлення програмного забезпечення [1]-[6]. Іншою особливістю є управління медіафайлами за допомогою кнопок, таких як відтворення, пауза, стоп, перемотка вперед і назад, а також інтерактивна панель прогресу [1]-[6].

Початкові установки аудіоплеєра забезпечуються програмним забезпеченням, яке вже присутнє в операційній системі користувача та не вимагає від нього жодних додаткових дій [1]-[6]. Наприклад, у версіях Microsoft Windows встановлено медіапрогравач Windows Media Player. Для MacOS доступний QuickTime Player, що пропонує роботу з аудіо, відео, анімацією тощо. У пристроях на базі Android програмою за умовчанням є Музика. Пізніше Google змусила користувачів перейти на попередньо встановлені програми Google Play Music і YouTube Music [1]-[6]. Для Linux аналіз попередньо встановлених програм є досить складним завданням через різноманітність дистрибутивів і постійні оновлення візуальних елементів та коду, оскільки ця операційна система є відкритою [1]-[6].

Плеєри від сторонніх компаній та індивідуальних розробників також забезпечують відтворення аудіо файлів. Вони можуть пропонувати підтримку нових форматів файлів, поліпшену візуалізацію, додаткові функції, організацію файлів, апаратне або програмне прискорення, оптимізацію та навіть вбудоване стиснення аудіо [1]-[6].

Існує велика кількість програмного забезпечення онлайн музичних плеєрів. Це може бути частина вебсайту, доступна без встановлення будь-яких

розширень або програмного забезпечення. Це ефективний варіант для тих, хто хоче слухати радіо безкоштовно або потребує вебсайту для прослуховування музики. Таке програмне забезпечення надає користувачу можливість відтворювати, призупиняти, перемотувати вперед, переглядати списки відтворення та створювати нові без необхідності у встановленні чи налаштуванні програмного забезпечення [1]-[6].

Проаналізуємо програмне забезпечення від сторонніх розробників або попередньо встановлені програми. Розглянемо два типові приклади: Spotify і Apple Music. Окрім відтворення власних файлів, ці платформи дозволяють транслювати музику безпосередньо на ваш пристрій без необхідності завантаження. Вони можуть містити широкий вибір від популярних пісень до подкастів, що доступні на вебсайтах радіостанцій та інших джерелах [1]-[6].

Другий тип музичного плеєра, апаратний музичний програвач, потребує специфічного апаратного забезпечення з унікальними розмірами та можливостями [1]-[6]. Вони мають дві основні характеристики [1]-[6]. Вони призначені перш за все для відтворення звуку, а також можуть мати додаткові функції. Деякі старі версії не вимагають програмного забезпечення або можуть працювати без нього. Сучасні моделі можуть бути залежними від програмного забезпечення, яке важко або неможливо змінити [1]-[6].

Деякі портативні музичні пристрої включають у себе AM/FM-радіо, що було одним із перших прикладів музичних плеєрів. Вони можуть використовувати антени для отримання радіосигналів, проте мають обмежене керування, зазвичай обмежуючись лише вмиканням та вимиканням та налаштуванням частоти. З появою нових технологій портативні музичні пристрої перейшли на більш сучасні пристрої [1]-[6].

Цифровий аудіопристрій можна розглядати як розвинену версію портативного музичного плеєра. Такі пристрої надають можливість користувачам керувати через телевізор, смартфон, комп'ютер, цифрову медіаплатформу, кабель, голосові команди, Bluetooth або Wi-Fi. Зазвичай цифровий аудіопристрій поставляється з зовнішнім приймачем і може

відтворювати різноманітні аудіоформати з внутрішньої пам'яті. Крім цього, він може програвати аудіо з зовнішніх джерел, таких як жорсткі диски, флеш-накопичувачі і SD-карти [1]-[6].

Музичні пристрої вже пройшли значний етап розвитку як у плані апаратного, так і програмного забезпечення, але спосіб споживання музики постійно еволюціонує. Це сприяло створенню більш вдосконалених музичних програвачів [1]-[6].

Проаналізуємо ключові області, які перебувають у центрі уваги для поліпшення аудіоплеєрів. Ключові області, які перебувають у центрі уваги для поліпшення аудіоплеєрів є такими: системи рекомендацій на основі штучного інтелекту, якість звуку та інтеграція з іншими розумними пристроями [1]-[6].

Системи рекомендацій на основі штучного інтелекту набувають активного розвитку, оскільки штучний інтелект (ШІ) активно впливає на системи рекомендацій у сфері музики. Наприклад, плейлисти «Discover Weekly» та «Daily Mix» у Spotify створюються за допомогою алгоритмів машинного навчання, які аналізують музичні вподобання користувача та пропонують йому відповідні нові треки [1]-[6].

З оновленням технологій штучного інтелекту очікується, що програмне забезпечення аудіоплеєрів буде постійно вдосконалюватися в наданні індивідуалізованих рекомендацій. Майбутні програмні системи на базі штучного інтелекту будуть враховувати не лише музичні вподобання користувача в минулому, але й час і місце, де він слухає музику, його настрій та інші аспекти активності [1]-[6].

Ще однією зі значних тенденцій у музичній індустрії є створення тривимірної проекції на основі відповідної мелодії. Ці технології використовують об'ємний звук, щоб створити тривимірний ефект під час прослуховування, що виходить за рамки звичайного стереозвуку. У найближчому майбутньому ми можемо побачити музичні програвачі з покращеними можливостями обробки аудіо, які підтримуватимуть ці аудіоформати [1]-[6].

Що стосується програмного забезпечення, музичний плеєр використовується для перетворення звичайних аудіофайлів у просторовий звук, що дозволяє отримати кращий звук при прослуховуванні старих музичних композицій [1]-[6].

Щодо інтеграції з іншими розумними пристроями, Інтернет речей (IoT) починає об'єднувати наші щоденні предмети, у тому числі і музичні пристрої. Розумні колонки, такі як Amazon Echo та Google Home, виконують подвійну функцію, діючи як музичний програвач і центр керування іншими розумними пристроями [1]-[6].

У майбутньому музичні пристрої будуть більш інтегрованими у цифрове життя людства. Наприклад, розробляються технології для відтворення плейлистів на розумній акустичній системі вдома, та легкого переключення на мобільну музичну програму на телефоні [1]-[6]. Музичні плеєри пройшли довгий шлях від часів простого програвача, магнітофона до сучасних програмно-апаратних засобів [1]-[6].

Поєднання штучного інтелекту, захопливого звуку та Інтернету речей обіцяє перетворити можливості музичного програвача, зробивши його більш особистим, захоплюючим і зв'язаним, ніж будь-коли раніше, оскільки технології розширюватимуть межі споживання музики [1]-[6].

У наш час також широко використовуються мережеві програвачі, що являють собою інтелектуальні пристрої для бездротового відтворення різноманітних цифрових файлів з різних джерел у відмінній якості [1]-[6]. За допомогою таких пристроїв музику можна відтворювати в реальному часі за допомогою підключення до Інтернету, домашнього сервера або кабелю. Завдяки спеціальному програмному застосунку на пристрої користувача можна легко керувати музичною колекцією [1]-[6].

Визначено, що цифровий аудіопрогравач являє собою електронний пристрій, який може зберігати та відтворювати цифрові аудіо файли. На відміну від програвачів MP3, цифрові аудіопрогравачі можуть легко відтворювати різноманітні аудіоформати. Проаналізовано різні типи файлів

аудіоданих, що обробляються аудіоплеєрами, зокрема, аудіоформати без стиснення, формати стисненого аудіо без втрат, аудіоформати зі стисненням із втратами. Відзначено, що системи рекомендацій на основі штучного інтелекту набувають активного розвитку та використовуються у програмному забезпеченні аудіоплеєрів. Очікується, що поєднання штучного інтелекту, захопливого звуку та Інтернету речей може перетворити можливості програмного забезпечення аудіоплеєрів, зробивши їх більш особистими, захоплюючими і пов'язаними з іншими технологіями.

1.2 Аналіз програмного забезпечення для відтворення музики

Проаналізуємо програмне забезпечення для підтримки процесу відтворення музики [7]-[14].

З великим розмаїттям програмних інструментів для відтворення музики, доступних на ринку, вибір найкращого для потреб користувача може бути складним завданням. Варто розуміти конкретні потреби користувачів для вибору існуючого або створення нового обладнання, зокрема, і програмного забезпечення [7]-[11]. Розглянемо програмне забезпечення музичних програвачів, що відтворюють аудіофайли, витягуючи код та інструкції з файлу. Аудіофайли містять конкретні частоти та символи даних, які передають інформацію звуковим сигналом. Ці програвачі виступають як конвертери для цих файлів і потокових аудіоформатів [7]-[11].

На ринку представлено різне програмне забезпечення музичних плеєрів. Музичний плеєр являє собою мультимедійну програма для відтворення аудіофайлів. Ці програмні інструменти підтримують різні музичні формати, такі як MP3, WAV і WMA, та дозволяють організувати та підтримувати музичну бібліотеку користувачів. Важливо відзначити, що програмне забезпечення аудіоплеєрів повинно забезпечувати не лише прослуховування музики на комп'ютері, але й можливість організації файлів так, як подобається користувачу [7]-[11]. Оскільки з часом користувач може

створити велику колекцію аудіофайлів на власному пристрої, відповідно, цифрова бібліотека музики з часом може стати значною, тому музичний плеєр повинен дозволяти не лише прослуховувати пісні, а й організовувати їх згідно вимог користувача [7]-[11].

Програмне забезпечення Doramine (рис. 1.1) є програмою-музичним програвачем для операційної системи Windows. Це програмне забезпечення з відкритим кодом має багато корисних функцій, які подобаються великій кількості користувачів. Від початку завантаження до закінчення відтворення пісні це програмне забезпечення працює з відчутною швидкістю, без будь-яких затримок. Програмне забезпечення Doramine надає користувачам простий у використанні інтерфейс, який дозволяє легко знаходити потрібні опції та налаштування [7], [9], [11], [12].

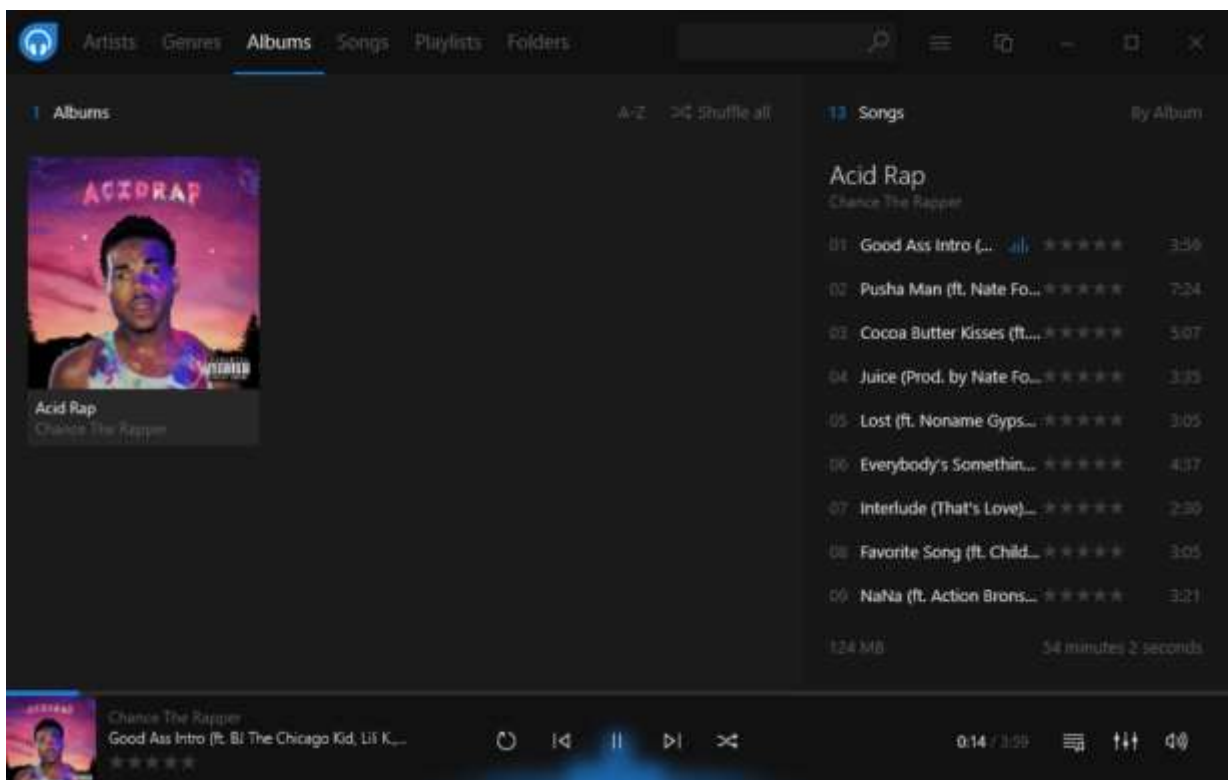


Рисунок 1.1 – Програмне забезпечення Doramine [12]

Програмне забезпечення Doramine підтримує більшість форматів аудіофайлів, включаючи WMA, MP4, FLAC, AAC, APE, WAV, OGG і OPUS.

Програмне забезпечення Doramine має багато корисних функцій, таких як автоматичне створення мета-тегів, функція прокрутки тексту тощо [7], [9].

Програмне забезпечення Doramine з відкритим вихідним кодом для Windows виглядає подібно до програми Microsoft UWP, але відмінність полягає в тому, що вона не має доступу до магазину і недоступна для завантаження з нього. Простий інтерфейс і гнучкість програмного забезпечення Doramine є досить важливими для користувачів, усі опції та налаштування легко доступні для користувача. Щоб поліпшити інтерфейс Doramine, можна внести багато змін [7], [9], [11], [12].

Хоча деякі функції можуть бути відсутні в цьому музичному програвачі, користувачі все ж можуть скористатися автоматичним метатегуванням, відображенням тексту в реальному часі, прокручуванням та іншими функціями. Цей інструмент ставить акцент на персоналізації і дозволяє змінювати теми та зовнішній вигляд для оптимальної взаємодії з користувачем. Програмне забезпечення Doramine сумісно з різноманітними форматами аудіофайлів [7], [9], [11], [12].

Програмне забезпечення Doramine пропонує розширений екран для відтворення, обладнаний вигравірованими кнопками та корисними функціями, що спрощують використання для користувача. Програмне забезпечення Doramine має набір функцій на панелі завдань, щоб користувачі могли легко отримати доступ до різних опцій, не витрачаючи багато часу на пошук. Додатково, є функція мікроплеєра, яка дозволяє переглядати відео у мініатюрному вікні, не виходячи з інших програм. Інтерактивний інтерфейс навігації програмного забезпечення Doramine дозволяє користувачам з легкістю переміщатися між основними функціями. Крім того, програмне забезпечення Doramine підтримує різні мови, що робить його використання зручним для різних користувачів. Програмне забезпечення Doramine надає можливість користувачам створювати колекції, наприклад, окремий розділ, до якого можна додавати аудіофайли за власними критеріями [7], [9], [11], [12].

Основними характеристиками програмного забезпечення Doramine є такі [7], [9], [11], [12]:

- розширений екран відтворення: програмне забезпечення Doramine має великий екран відтворення з чітко виділеними кнопками та функціями, що полегшують керування музикою [7], [9], [11], [12];

- панель завдань: програмне забезпечення Doramine надає ряд функцій безпосередньо на панелі завдань, щоб користувачі могли швидко отримати доступ до різних опцій без зайвих зусиль [7], [9], [11], [12];

- мікроплеєр: програмне забезпечення Doramine має функцію мікроплеєра, яка дозволяє відтворювати відео у мініатюрному вікні одночасно з використанням інших програм [7], [9], [11], [12];

- інтерактивний інтерфейс користувача: програмне забезпечення Doramine пропонує інтерактивний інтерфейс для зручної навігації між різними функціями та опціями програми [7], [9], [11], [12];

- підтримка різних мов: програмне забезпечення Doramine підтримує різні мови, що робить його доступним для користувачів з різних країн та мовних середовищ [7], [9], [11], [12];

- створення колекцій: програмне забезпечення Doramine дозволяє користувачам створювати та організовувати колекції аудіофайлів за різними критеріями для більш зручного керування музичною бібліотекою [7], [9], [12].

Серед недоліків програмного забезпечення Doramine можна відзначити обмежений функціонал порівняно з іншими музичними програвачами, а також можливі проблеми з сумісністю або стабільністю функціонування програми на певних конфігураціях комп'ютерів [7], [9], [11], [12].

Програмне забезпечення MusicBee (рис. 1.2) є безкоштовною програмою для прослуховування музики, спеціально розробленою для ентузіастів музики, яка має всі необхідні функції для керування та використання своєї колекції, незалежно від її розміру, включаючи бібліотеку з сотень тисяч пісень [8], [9], [13].

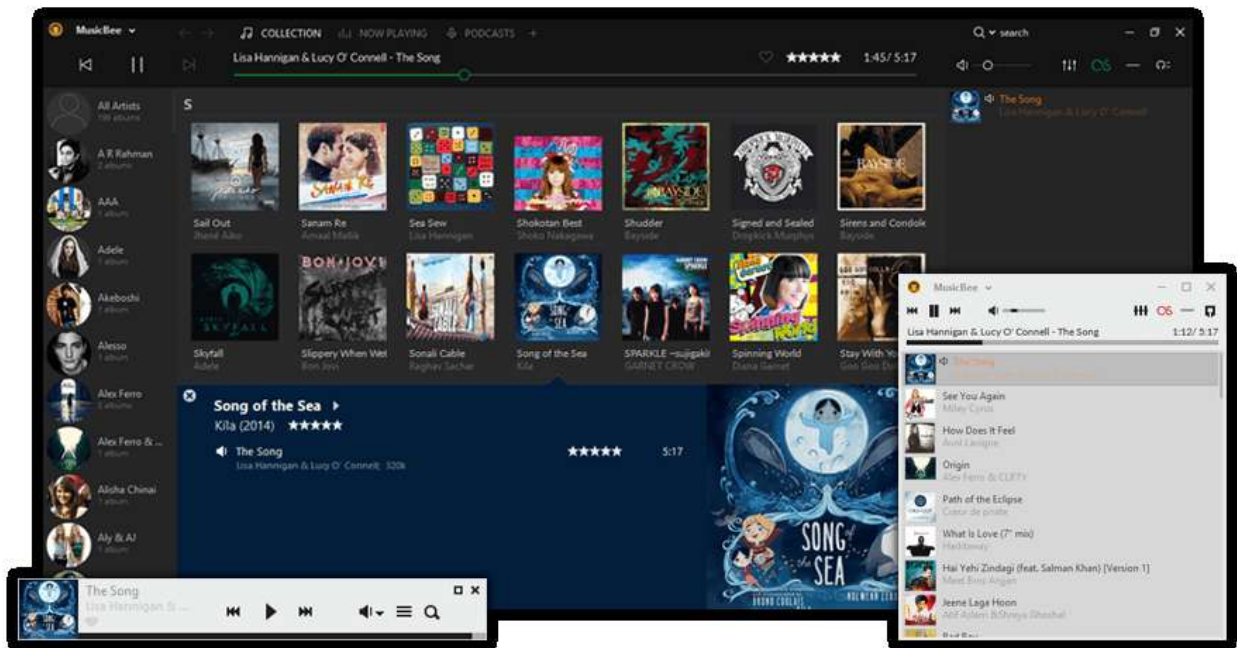


Рисунок 1.2 – Програмне забезпечення MusicBee [13]

Програмне забезпечення MusicBee працює ефективно з апаратним забезпеченням комп'ютера, використовуючи розширені звукові карти та налаштування об'ємного звуку. Програмне забезпечення MusicBee може забезпечити безперервне відтворення, щоб уникнути тиші між треками, додати тишу або затухання, нормалізувати звук або експериментувати з еквайзером. Крім того, програмне забезпечення MusicBee підтримує практично всі аудіоформати і може легко конвертувати файли на пристрої відтворення [8], [9], [13].

Наявна також версія програми для Android, яка може дистанційно керувати MusicBee і підтримує плагін WinAmp. Хоча цей плеєр не надає відкритий програмний код, він є абсолютно безкоштовним для особистого використання та модифікації. Програмне забезпечення MusicBee дозволяє переглядати музику на комп'ютері та додавати пісні з iTunes або Windows Media Player. Як тільки інформація додається до бібліотеки користувача MusicBee, розумна система тегів програми дозволить легко заповнювати відсутні метадані в стандартних форматах, шукати обкладинки альбомів і знаходити слова, які з'являються під час відтворення пісні [8], [9], [13].

Програмне забезпечення MusicBee розроблено для повного використання апаратного забезпечення вашого комп'ютера з підтримкою високоякісних звукових карт і об'ємного звуку [8], [9], [13]. Програмне забезпечення MusicBee також дозволяє стрімінг і сумісний з Last.fm і SoundCloud, що робить його ефективним вибором для трансляції улюбленого контенту користувача [8], [9], [13].

Програмне забезпечення MusicBee розроблено спеціально для управління треками і добре справляється з цим завданням, навіть якщо поточна колекція користувача містить багато повторів треків, треків без метаданих і заплутаних альбомів [8], [9], [13].

Імпортувати музичну бібліотеку в MusicBee досить просто. Якщо користувач використовує iTunes або Windows Media Player, є простий майстер, який автоматизує цей процес. Файли можуть бути проіндексовані, але не переміщені, якщо цього не бажає користувач [8], [9], [13]. Після цього користувач може заповнювати відсутні метадані і тегувати їх, щоб належним чином організувати бібліотеку [8], [9], [13]. Система автоматичного тегування у програмному забезпеченні MusicBee ефективно працює зі стандартними шаблонами для всіх типів аудіофайлів, але крім того користувач може вручну переглядати і редагувати теги [8], [9], [13].

Щодо форматів файлів, програмне забезпечення MusicBee зберігає аудіофайли користувача і може конвертувати їх у формати, які сумісні з іншими пристроями [8], [9], [13].

Програмне забезпечення MusicBee не лише автоматично шукає обкладинки альбомів для відображення під час відтворення, але також забезпечує доступ до фотографій виконавців і текстів пісень, які відображаються під час прослуховування [8], [9], [13].

Програмне забезпечення MusicBee використовує всі можливості аудіообладнання користувача, включаючи високоякісні звукові карти та налаштування об'ємного звуку. Крім того, існує можливість налаштовувати

різні параметри відтворення, такі як видалення пауз між треками та нормалізація гучності [8], [9], [13].

Підтримка потокових сервісів та інших плагінів робить програмне забезпечення MusicBee ефективним варіантом для безкоштовного музичного плеєра [8], [9], [13].

Програмне забезпечення MusicBee легко імпортує музичну бібліотеку з iTunes і підтримує багато аудіоформатів, включаючи MP3, AAC, WMA, WAV, M4A, FLAC, OGG, APE і багато інших [8], [9], [13].

Програмне забезпечення MusicBee синхронізує аудіо з мобільними телефонами на базі Android, деякими пристроями iOS, USB-накопичувачами та іншими портативними музичними плеєрами. Крім того, він може автоматично відслідковувати зміни на жорсткому диску та оновлювати музичну бібліотеку [8], [9], [13].

Існує можливість налаштувати MusicBee за допомогою різних оболонок і плагінів, включаючи підтримку деяких плагінів Winamp. Серед функцій MusicBee є еквалайзер, ефекти DSP, можливість копіювання CD, автоматичний імпорт метаданих та інші [8], [9], [13]. Програмне забезпечення MusicBee не потребує великої кількості простору на жорсткому диску і може бути встановлено на комп'ютері, як будь-яке інше програмне забезпечення. Крім того, існує портативна версія цього музичного плеєра. Також доступна версія MusicBee UWP в магазині [8], [9], [13].

Програмне забезпечення MusicBee має такі особливості [8], [9], [13]:

- синхронізація з різними пристроями: програмне забезпечення MusicBee може синхронізувати аудіофайли з мобільними телефонами на базі Android, деякими пристроями iOS, USB-накопичувачами та іншими портативними музичними плеєрами [8], [9], [13];

- розширені функції управління: програмне забезпечення MusicBee надає можливість користувачам відстежувати зміни на жорсткому диску та автоматично оновлювати музичну бібліотеку, що спрощує управління музикою [8], [9], [13];

- підтримка плагінів: програмне забезпечення MusicBee дозволяє користувачам налаштовувати його за допомогою різних оболонок і плагінів, зокрема підтримується використання деяких плагінів Winamp [8], [9], [13];

- розширені аудіофункції: серед функцій програмного забезпечення MusicBee є 15-смуговий еквалайзер, ефекти DSP, можливість копіювання CD, автоматичний імпорт метаданих тощо [8], [9], [13];

- низьке використання ресурсів: програмне забезпечення MusicBee не потребує багато простору на жорсткому диску і має невеликі вимоги до системних ресурсів [8], [9], [13];

- підтримка різних форматів та тем: програмне забезпечення MusicBee забезпечує синхронізацію файлів у різних аудіоформатах та має можливість вибору різних тем оформлення [8], [9], [13].

Серед недоліків програмного забезпечення MusicBee можна відзначити складність використання для новачків або користувачів з обмеженим досвідом, інтерфейс програми MusicBee для таких користувачів може здатися складним або перевантаженим функціями. Іншим недоліком є відсутність хмарного сховища: MusicBee не надає вбудованої підтримки хмарних сховищ, що може ускладнити синхронізацію музики між пристроями та доступ до бібліотеки з будь-якого місця [8], [9], [13].

Програмне забезпечення AIMP (рис. 1.3) являє собою аудіо плеєр, який доступний для операційних систем Windows і Android. Незалежно від формату аудіофайлу, цей музичний плеєр підтримує майже всі типи аудіоформатів. Ці можливості сумісності та функціональності роблять його одним з найкращих музичних застосунків для різних операційних систем. Програмне забезпечення AIMP забезпечує легкий доступ до пісень користувача, навіть якщо колекція наразі не повна або не має метаданих [7], [8], [14]. Бібліотеки AIMP можуть створюватися з файлів з кількох папок, а також копіювати з компакт-дисків, а автоматична нумерація треків і додавання тегів допомагають користувачам підтримувати упорядкованість аудіо файлів [7], [8], [14].

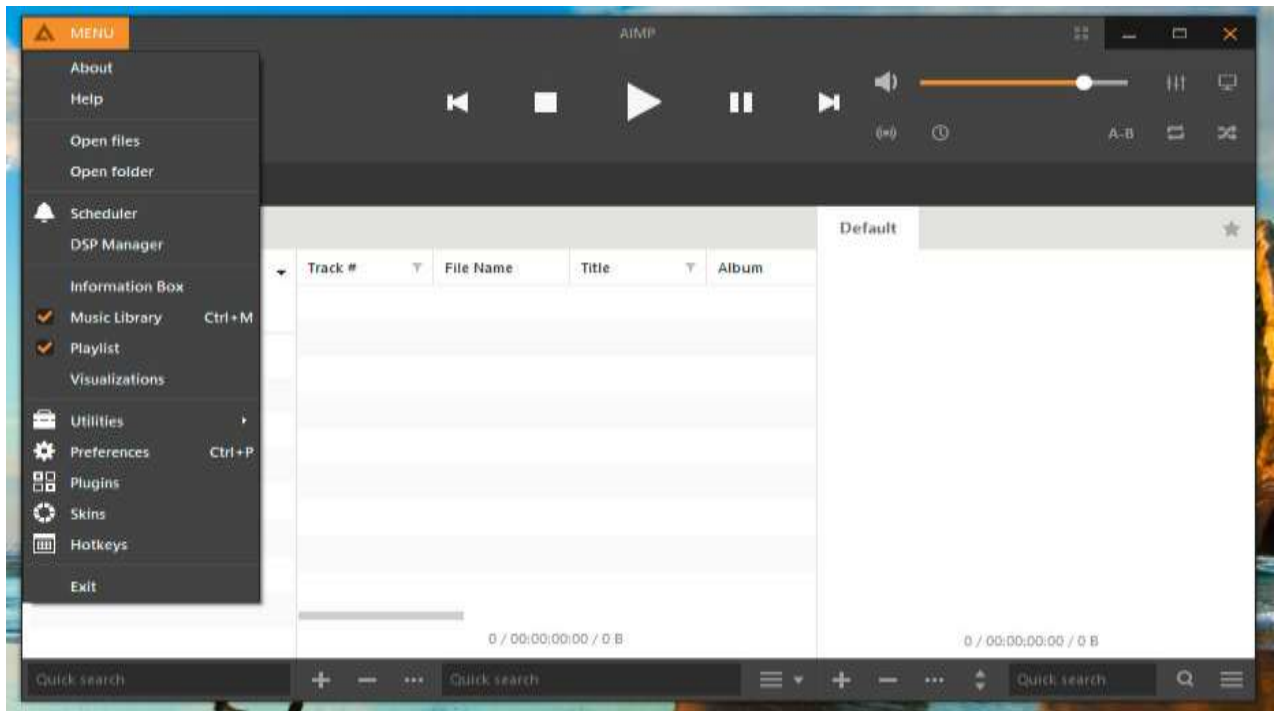


Рисунок 1.3 – Програмне забезпечення AIMP [14]

Програмне забезпечення AIMP підтримує широкий спектр форматів, а додаткові кодеки доступні як користувацькі плагіни. Більшість доповнень для музичного плеєра це візуальні елементи та скіни, які виглядають красиво, але не завжди практичні. Однак деякі плагіни AIMP включають розширення YouTube для створення списків відтворення з відео, додатки для потокової передачі музики з SoundCloud і інструменти для дистанційного керування плеєрами [7], [8], [14].

Програмне забезпечення AIMP включає унікальні вбудовані інструменти, такі як будильник, можливість запуску відтворення в певний час, налаштування для автоматичного вимкнення комп'ютера після завершення відтворення списку пісень, а також функцію вимкнення звуку для створення музичного тиші. Цей плеєр створює власну караоке-атмосферу [7], [8], [14].

Програмне забезпечення AIMP підтримує широкий спектр форматів файлів, включаючи MP3, AAC, AC-3, Ogg Vorbis, Opus, Speex, WMA, Apple Lossless, FLAC, WAV, Audio CD і інші. Також програмне забезпечення AIMP здатне програвати такі аудіоформати, як APE, True Audio. Програмне забезпечення AIMP підтримує аудіоінтерфейси DirectSound, ASIO і WASAPI,

має 18-смуговий еквалайзер і 32-бітну обробку звуку для вбудованих аудіо ефектів [7], [8], [14].

Інтерфейс програмного забезпечення АІМР містить всі стандартні параметри для імпорту музичних файлів, але крім того, містить різні додаткові налаштування. Тут можна скористатися параметрами перетворення для відтворення та змішування каналів, щоб розумно перетворювати багатоканальні файли в стерео або моно. Також користувачам надається функція зменшує спотворення, що виникають при зменшенні глибини дискретизації, та функція анти переривання, що попереджає появу шумів під час передачі сигналу [7], [8], [14].

Налаштування плейлистів у програмному забезпеченні АІМР відповідають за зовнішній вигляд, додавання файлів, їх сортування за темами та експорт. Користувачу надається можливість активувати відображення стовпців перемикання та продовження разом із розширеними даними. Плейлисти можуть бути експортовані в різних форматах, таких як HTML, Unicode або ANSI. Меню також використовується для керування підключенням до Інтернету, доступом до файлів і налаштуваннями. Навігація в плеєрі легка завдяки фіксованим ярликам і відображенню статусу відтворення [7], [8], [14].

Після налаштування основних параметрів користувач може використовувати різноманітні плагіни і оболонки, щоб змінити зовнішній вигляд і розширити можливості музичної бібліотеки. Програмне забезпечення АІМР є досить популярним музичним плеєром. Цей програмний продукт робить набагато більше, ніж просте відтворення пісень, які зберігаються на комп'ютері [7], [8], [14].

Завдяки програмному забезпеченню АІМР користувач може легко організувати музичну колекцію, створювати інтелектуальні списки відтворення, копіювати звукові диски, керувати метаданими та змінювати вигляд гравця за допомогою різних тем [7], [8], [14].

Музичний плеєр АІМР пропонує еквалайзер та різноманітні аудіо ефекти, що дозволяє користувачу прослуховувати музику. Серед інших

корисних функцій, які пропонує програмне забезпечення AIMR, є можливість видаляти розділи списків відтворення та змінювати теми інтерфейсу за допомогою відповідних пунктів меню керування [7], [8], [14]. Щодо підтримуваних аудіоформатів, цей плеєр підтримує майже всі популярні формати. Крім того, він має вбудований аудіоконвертер, таймер сну та будильник, що дозволяє вивести комп'ютер користувача із режиму сну в зазначений час [7], [8], [14].

Програмне забезпечення AIMR має такі особливості [7], [8], [14]:

- підтримка плагінів: програмне забезпечення AIMR дозволяє встановлювати різноманітні плагіни, що розширюють функціональність програми. Це можуть бути плагіни для підтримки нових аудіоформатів, покращення звуку, різноманітні теми оформлення тощо [7], [8], [14];

- музичний плеєр і бібліотека: програмне забезпечення AIMR надає потужний музичний плеєр з можливістю відтворення різних аудіоформатів, включаючи MP3, FLAC, AAC, WAV та інші. Програмне забезпечення AIMR також має функцію організації музичної бібліотеки для зручного управління вашою колекцією пісень [7], [8], [14];

- еквалайзер та звукові ефекти: програмне забезпечення AIMR пропонує користувачам можливість налаштувати звук за допомогою еквалайзера та різних звукових ефектів для досягнення оптимального звучання під час прослуховування музики [7], [8], [14];

- редагування тегів: програмне забезпечення AIMR дозволяє редагувати теги аудіофайлів, щоб додати або змінити інформацію про виконавця, назву пісні, жанр, рік випуску та інші важливі деталі [7], [8], [14];

- мінімальне використання ресурсів: програмне забезпечення AIMR висуває низькі вимоги до системних ресурсів комп'ютера, що дозволяє ефективно використовувати програму навіть непотужних комп'ютерах [7], [8];

- можливість налаштування інтерфейсу: користувачі програмного забезпечення AIMR можуть налаштовувати інтерфейс програми згідно зі своїми вподобаннями, змінюючи теми оформлення, розміщення елементів управління та інші параметри [7], [8], [14];

– синхронізація та інтеграція: програмне забезпечення AIMP підтримує синхронізацію з різними пристроями і сервісами, що дозволяє зручно управляти музичною колекцією і слухати її на різних пристроях [7], [8], [14].

Серед недоліків програмного забезпечення AIMP можна відзначити складність використання для новачків або користувачів з обмеженим досвідом, інтерфейс програми AIMP для таких користувачів може здатися складним або перевантаженим функціями. Іншим недоліком є відсутність хмарного сховища: AIMP не надає вбудованої підтримки хмарних сховищ, що може ускладнити синхронізацію музики між пристроями та доступ до бібліотеки з будь-якого місця [7], [8], [14].

Порівняльну характеристику програмного забезпечення для відтворення музики наведено у таблиці 1.1.

Таблиця 1.1 – Порівняння програмного забезпечення для відтворення музики

Критерій порівняння	Dopamine [12]	MusicBee [13]	AIMP [14]
Розширений екран відтворення	+	+-	+-
Підтримка різних операційних систем	-	+	+
Засоби підтримки стрімінгу	-	+	-
Система автоматичного тегування	+-	+	+-
Зручний інтерактивний інтерфейс користувача	+	+	+-
Підтримка великої кількості мов	+	+-	+-
Зручні засоби для створення колекцій музичних творів	+	+-	+-

За результатами проведеного аналізу можна зробити висновок, що у наш час існує досить багато програмних засобів для відтворення музики. Проте деякі програмні засоби для відтворення музики можуть мати обмежені функції, зокрема, не підтримувати певні формати файлів або не мати розширених налаштувань еквайзера. Інше програмне забезпечення аудіоплеєрів може вимагати велику кількість системних ресурсів, що може призвести до сповільнення роботи комп'ютера або мобільного пристрою. Деякі аудіоплеєри можуть не підтримувати певні аудіокодеки або контейнери, що може призвести до неможливості відтворення певних аудіофайлів без попереднього конвертування. Тому актуальною є розробка програмного забезпечення мережевого аудіоплеєру.

При розробці програмного забезпечення мережевого аудіоплеєру необхідно забезпечити такі функціональні вимоги:

- підтримка можливості відтворення аудіо даних за посиланням;
- підтримка роботи з плейлистом, що містить улюблені пісні користувача;
- можливість додавання радіостанцій у вигляді сторонніх посилань користувача;
- можливість формування рекомендованих для користувача радіостанцій.

1.3 Висновки за розділом 1

Визначено, що цифровий аудіопрогравач являє собою електронний пристрій, який може зберігати та відтворювати цифрові аудіо файли. На відміну від програвачів MP3, цифрові аудіопрогравачі можуть легко відтворювати різноманітні аудіоформати. Проаналізовано різні типи файлів аудіоданих, що обробляються аудіоплеєрами, зокрема, аудіоформати без стиснення, формати стисненого аудіо без втрат, аудіоформати зі стисненням із втратами. Відзначено, що системи рекомендацій на основі штучного інтелекту

набувають активного розвитку та використовуються у програмному забезпеченні аудіоплеєрів. Очікується, що поєднання штучного інтелекту, захопливого звуку та Інтернету речей може перетворити можливості програмного забезпечення аудіоплеєрів, зробивши їх більш особистими, захоплюючими і пов'язаними з іншими технологіями.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато програмних засобів для відтворення музики. Проте деякі програмні засоби для відтворення музики можуть мати обмежені функції, зокрема, не підтримувати певні формати файлів або не мати розширених налаштувань еквайзера. Інше програмне забезпечення аудіоплеєрів може вимагати велику кількість системних ресурсів, що може призвести до сповільнення роботи комп'ютера або мобільного пристрою. Деякі аудіоплеєри можуть не підтримувати певні аудіокодеки або контейнери, що може призвести до неможливості відтворення певних аудіофайлів без попереднього конвертування. Тому актуальною є розробка програмного забезпечення мережевого аудіоплеєру.

Сформульовано функціональні вимоги до програмного забезпечення мережевого аудіоплеєру.

2 МАТЕРІАЛИ І МЕТОДИ

2.1 Вибір мови програмування

При розробці програмного забезпечення мережевого аудіоплеєру було використано мову програмування Java [15], [16].

Java є мовою програмування загального призначення, яка відрізняється чітким і простим синтаксисом на високому рівні. Java надає можливість створювати програмне забезпечення за допомогою зрозумілого та компактного коду, який легко читати і розуміти, навіть для досвідчених розробників. Програма на мові Java може виконуватися без попередньої компіляції, що спрощує процес розробки та відладки. Ця мова програмування має широкий набір бібліотек і фреймворків і залучену спільноту розробників, яка забезпечує підтримку та вирішення проблем [15], [16]. Крім того, Java підтримує об'єктно-орієнтоване програмування, яке дозволяє організувати код за допомогою класів та об'єктів [15], [16].

Особливістю мови програмування Java є простий синтаксис, легкий дизайн програм та читабельний англomовний код. Важливо відзначити, що простий синтаксис Java уникає багатьох спеціальних символів та ключових слів [15], [16].

Java широко використовується у різних сферах, таких як веброзробка, аналітика даних, штучний інтелект, машинне навчання, автоматизація та Інтернет речей. Мова програмування Java є переносною, тобто вона працює на багатьох операційних системах, таких як Windows, Linux і MacOS, без значних змін. Для Java існує багато стандартних бібліотек, які надають різноманітний функціонал без необхідності в додаткових завантаженнях [15], [16].

Іншою перевагою мови Java є значна група розробників, що складається з експертів з усього світу, активно залучена до розвитку мови програмування з метою оновлення мови, бібліотек, фреймворків та інших інструментів програмування [15], [16].

Обґрунтування вибору мови програмування для створення програмного забезпечення мережевого аудіоплеєру наведено у таблиці 2.1.

Таблиця 2.1 – Обґрунтування вибору мови програмування для створення програмного забезпечення мережевого аудіоплеєру

Критерій порівняння мов програмування	Мова програмування		
	Java	Python	C#
Висока продуктивність	+	+-	+
Активність спільноти	+	+	+-
Кросплатформність	+	+	-
Зручність для створення ПЗ мережевого аудіоплеєру	+	-	-
Широке використання в багатьох сферах	+	+	+-

Отже, для реалізації програмного забезпечення мережевого аудіоплеєру обрано мову програмування Java, яка відрізняється чітким і простим синтаксисом, має велику кількість бібліотек для роботи з аудіо файлами та дозволяє створювати програмне забезпечення для роботи під керуванням різних операційних систем.

2.2 Вибір середовища розробки для створення програмного забезпечення мережевого аудіоплеєру

В якості середовища розробки для створення програмного забезпечення мережевого аудіоплеєру було обрано середовище Android Studio [17], [18].

Середовище розробки Android Studio є безкоштовним інтегрованим середовищем розробки для платформи Android, що ґрунтується на IntelliJ

IDEA. Воно вже містить у собі Android SDK та вважається найкращою платформою для розробки та тестування Android-застосунків. Google розробила Android Studio, використовуючи передові технології мобільної розробки [17], [18].

Створення програм за допомогою середовища розробки Android Studio є досить простим та швидким у порівнянні з Eclipse, завдяки здатності Android Studio передбачати майбутню структуру програми. Це дозволяє розробникам підходити до проєкту більш реалістично та ефективно [17], [18].

Користувачі мають доступ до корисних функцій, таких як перегляд додаткових можливостей у реальному часі. Також можна створювати програми для різних версій Android. Робота в цьому середовищі є більш інтуїтивно зрозумілою порівняно з Eclipse, завдяки простому інтерфейсу користувача. В результаті, процес кодування стає більш продуманим та легким для навігації під час розробки великих проєктів [17], [18].

Можливість перетягування функціональних елементів за допомогою Android Studio спрощує редагування інформації в самій програмі. Ключові особливості Android Studio включають довідник, наявність динамічних емуляторів для пристроїв, що працюють на платформі Android, зрозумілий інтерфейс, можливість надсилання сповіщень програми безпосередньо з пристрою Android через хмарні служби, швидкий пошук програм, відображення коду, додаткові параметри для налаштування роздільної здатності та розміру екрана, інструменти для підвищення якості проєкту та отримання доходу, підтримку відстеження хороших функцій реклами, дружній підхід до бета-тестерів та можливість перегляду всієї проєктної роботи (змін) в режимі реального часу [17], [18].

Інструменти Android Studio включають плагін Gradle для розробки програм, хмарне середовище Google, функцію ProGuard, редактор, інструмент стрічки для відстеження продуктивності та проблем із сумісністю версій, майстри на основі шаблонів для проєктування та створення компонентів Android, і підтримку Google Cloud Platform для Google Cloud Messaging та

інтеграції App Engine. Однією з головних переваг Android Studio є система збірки Gradle, яка активно розроблена Google і має такі корисні функції, як можливості розробки додатків, створення простих завдань у вигляді скриптів, можливість керувати залежностями та автозавантаженням, і налаштування сховища ключів [17], [18].

Основні функції Android Studio включають можливість перетягування компонентів через інтерфейс користувача, використання різних інструментів з багатофункціональним редактором для швидшої розробки, простий і автоматичний плагін Gradle, можливість використовувати тестування для перевірки сумісності з різними платформами та аналізу продуктивності, інструменти для оптимізації реклами та управління доходами користувачів, інструменти для тегування та редагування коду [17], [18].

Іншими функціями середовища розробки Android Studio є можливість використання Google Cloud Messaging для відправлення push-повідомлень з сервера на мобільний пристрій, запис відео з екрана (доступно лише на Android 4.4.2 і вище), легка локалізація програми, можливість використовувати шаблони та помічники для створення коду та реорганізація коду [17], [18].

Android Studio підтримує використання різноманітних сторонніх бібліотек та програмних засобів, що дозволяє розробникам розширювати функціональні можливості середовища розробки за необхідності. Ця структура впроваджує ефективні заходи безпеки [17], [18].

Середовище розробки Android Studio надає об'єктно-реляційне відображення, яке спрощує взаємодію з базами даних. Використання мови програмування Java для взаємодії з базами даних дозволяє розробникам зробити свій код більш зрозумілим і якісним [17], [18].

Функції, доступні у середовищі розробки Android Studio, можуть бути використані прямо в середовищі розробки або іншими способами. Android Studio має різні інструменти для розробки програмного забезпечення на основі мови програмування Java [17], [18].

Обґрунтування вибору середовища розробки для створення програмного забезпечення мережевого аудіоплеєру наведено у таблиці 2.2.

Таблиця 2.2 – Обґрунтування вибору середовища розробки для створення програмного забезпечення мережевого аудіоплеєру

Критерій порівняння середовищ розробки	Середовища розробки		
	Android Studio	Eclipse	IntelliJ IDEA
Використання система збірки проєкту Gradle	+	–	+
Засоби для обробки аудіо даних	+	+	+–
Широка спільнота розробників	+	+	+–
Зручність розробки Android-застосунків	+	+–	+–
Оптимізація роботи	+	+	+–

Таким чином, для створення програмного забезпечення мережевого аудіоплеєру обрано середовище розробки Android Studio, що є безкоштовним інтегрованим середовищем розробки для платформи Android, ґрунтується на IntelliJ IDEA, містить у собі Android SDK та є ефективною платформою для розробки та тестування Android-застосунків.

2.3 Висновки за розділом 2

Для реалізації програмного забезпечення мережевого аудіоплеєру обрано мову програмування Java, яка відрізняється чітким і простим синтаксисом, має велику кількість бібліотек для роботи з аудіо файлами та

дозволяє створювати програмне забезпечення для роботи під керуванням різних операційних систем.

Для створення програмного забезпечення мережевого аудіоплеєру обрано середовище розробки Android Studio, що є безкоштовним інтегрованим середовищем розробки для платформи Android, ґрунтується на IntelliJ IDEA, містить у собі Android SDK та є ефективною платформою для розробки та тестування Android-застосунків.

3 ОПИС ПРОГРАМИ

3.1 Структура программного обеспечения сетевого аудиоплеера

Структуру программного обеспечения сетевого аудиоплеера подано у вигляді діаграми класів, наведеної на рис. 3.1.

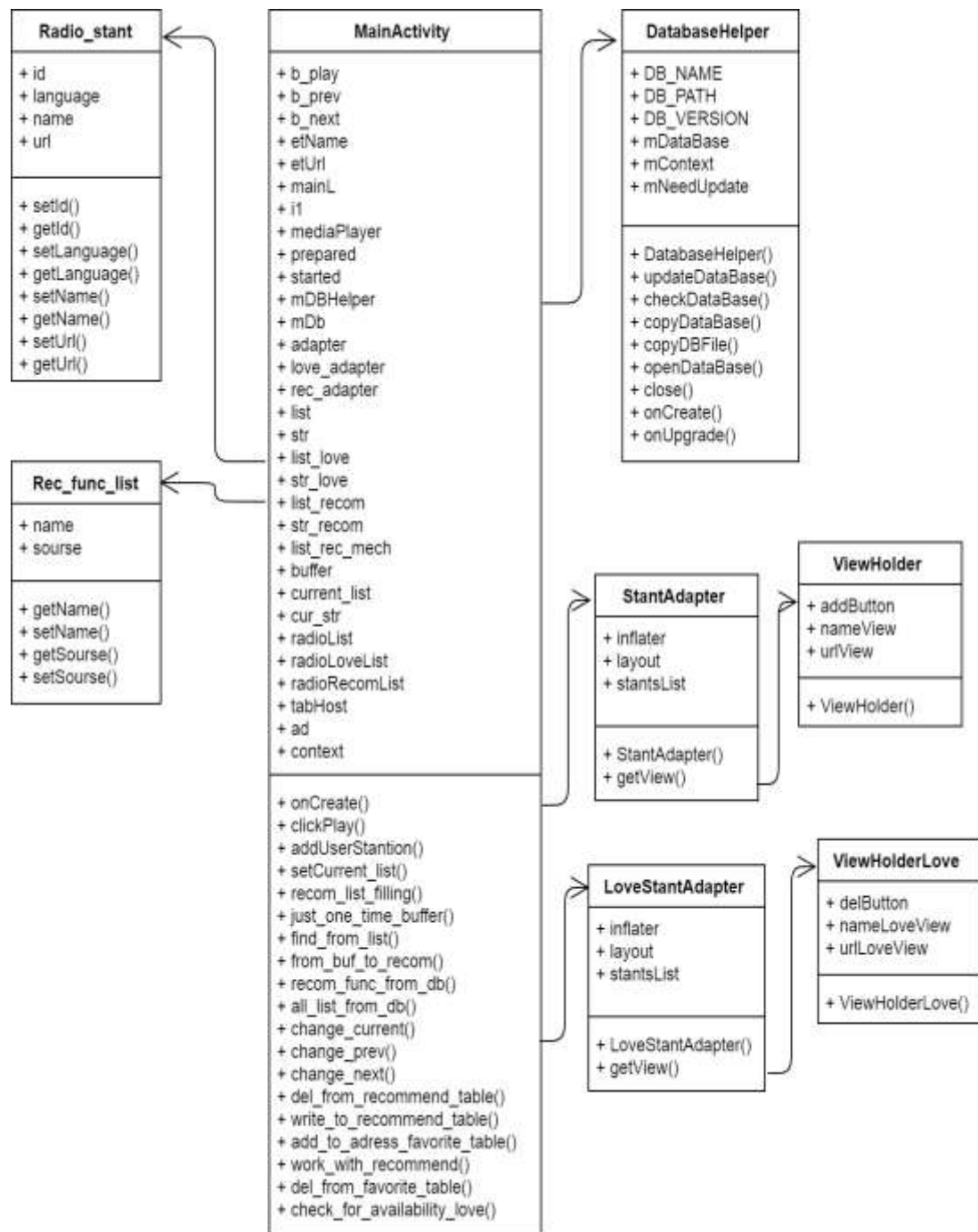


Рисунок 3.1 – Структура программного обеспечения сетевого аудиоплеера у вигляді діаграми класів

Як видно з рис. 3.1, основними класами програмного забезпечення мережевого аудіоплеєру є такі:

- Radio_stant – клас для роботи з радіостанціями, які може прослуховувати користувач;
- DatabaseHelper – допоміжний клас для роботи з базою даних;
- Rec_func_list – клас для роботи зі списком улюблених радіостанцій користувача;
- MainActivity – клас, що є базовим, та містить основний програмний код;
- StantAdapter – клас для роботи з візуальними списками музичних композицій або радіостанцій.

Діаграму компонентів програмного забезпечення мережевого аудіоплеєру наведеної на рис. 3.2.

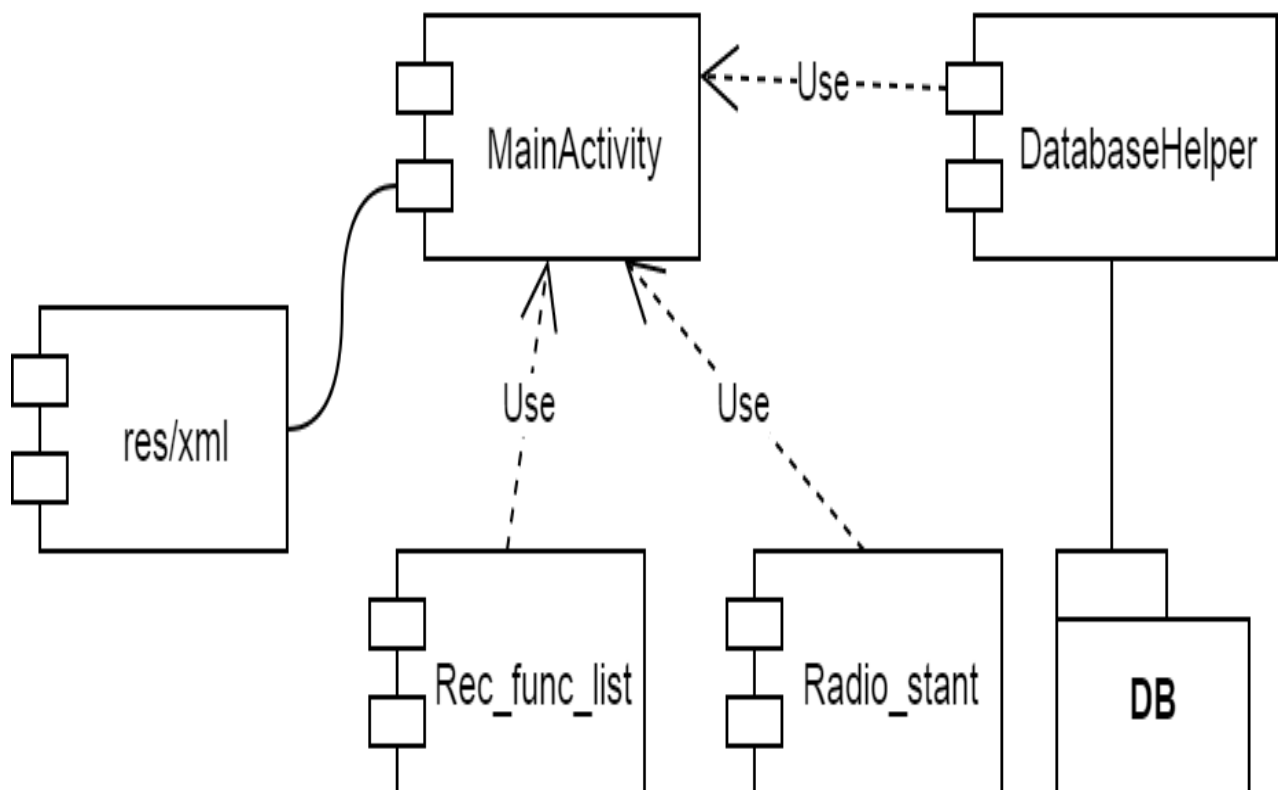


Рисунок 3.2 – Діаграма компонентів програмного забезпечення мережевого аудіоплеєру

Фізичну структуру програмного забезпечення мережевого аудіоплеєру, що містить множину створених файлів та папок відповідного програмного Android-застосунку, наведено на рис. 3.3.

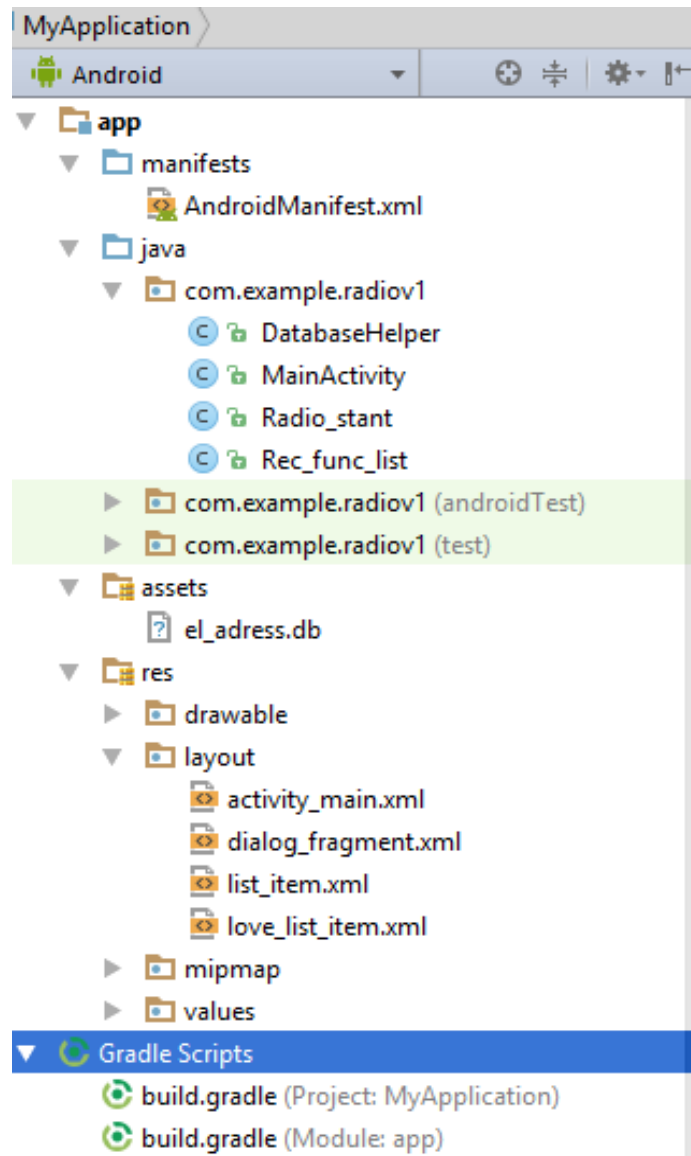


Рисунок 3.3 – Фізична структура програмного забезпечення мережевого аудіоплеєру

Фізична структура програмного забезпечення мережевого аудіоплеєру відповідає розробленій діаграмі класів (рис. 3.1), зокрема з рис. 3.3 видно, що програмно реалізовано класи DatabaseHelper, MainActivity, Radio_stant, Rec_func_list.

3.2 Функціонування програмного забезпечення мережевого аудіоплеєру

Функціонування програмного забезпечення мережевого аудіоплеєру подамо за допомогою схеми, зображеної на рис. 3.4.

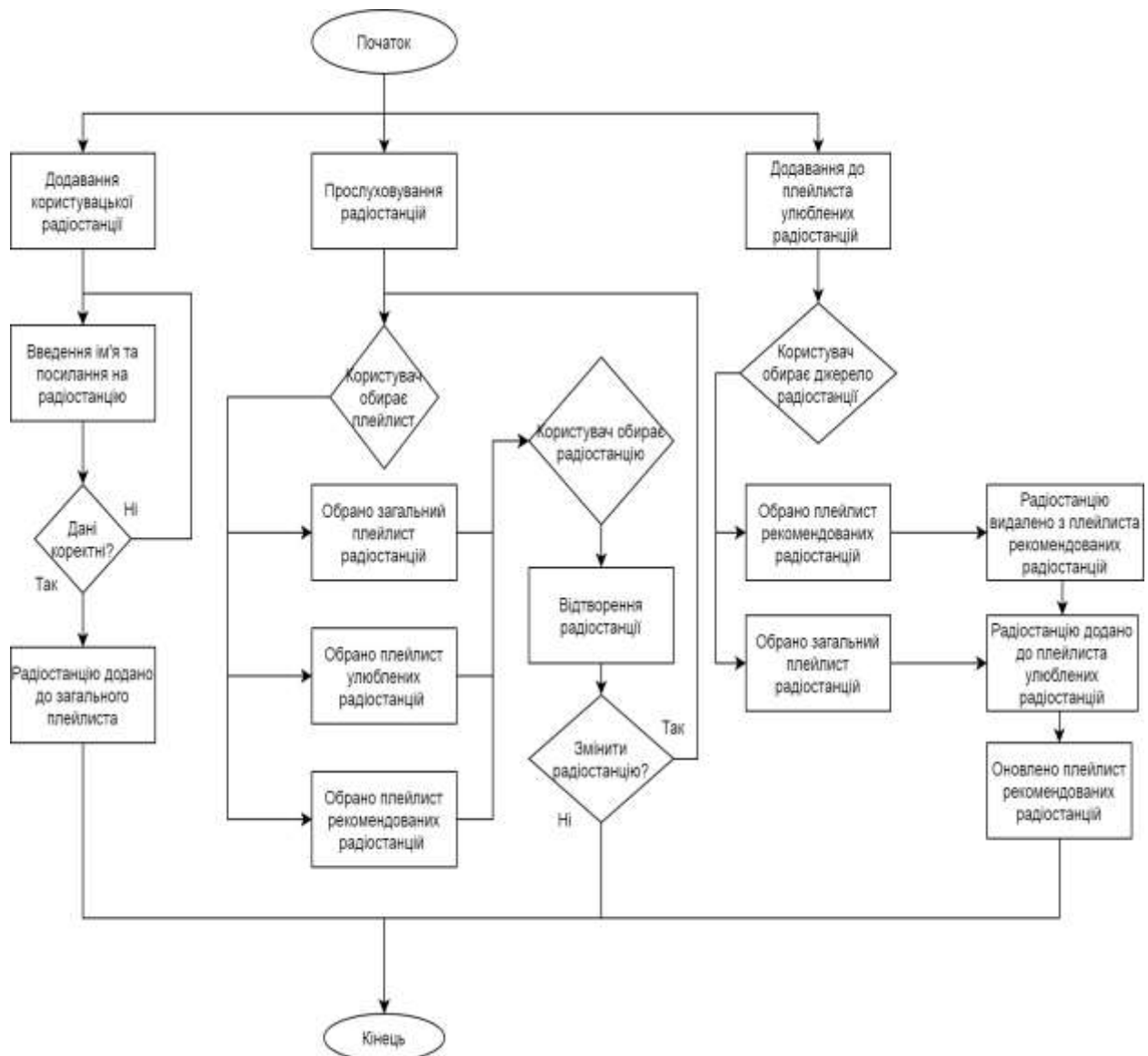


Рисунок 3.4 – Схема функціонування програмного забезпечення мережевого аудіоплеєру

Для функціонування програмного забезпечення мережевого аудіоплеєру обрано архітектуру клієнт-сервер, оскільки інформацію треба зберігати та редагувати в різних структурах у відповідності з потребами

користувачів. Відзначимо, що сервер може бути локальним, оскільки йому не потрібно вести статистику та виконувати складні обчислювальні операції, для цього обирається відносно проста структура бази даних і відповідні класи. Ця модель не перевантажує систему і забезпечує прийнятну швидкість роботи програми.

База даних SQLite, обрана для функціонування програми, відповідає всім вимогам для локального застосування. З метою полегшення роботи з даними бази даних створено клас `DatabaseHelper`, який забезпечує доступ до даних.

Основна логіка програми зосереджена в класі `MainActivity`, де обробляється все, що не пов'язане з даними та інтерфейсом користувача. Для спрощення роботи з даними створюється структура даних, яка містить інформацію про радіостанції, і максимально оптимізується для зручного використання.

Інтерфейс користувача визначає розташування та кількість елементів керування, необхідних користувачеві, для керування файлами. Крім цього, інші ресурси проєкту, такі як зображення та файли макетів, використовуються для поліпшення дизайну та взаємодії з інтерфейсом, незалежно від пристрою, на якому працює програма.

Алгоритм роботи програми включає наступні етапи. Користувачі програмного забезпечення мережевого аудіоплеєру можуть слухати будь-яку радіостанцію без потреби у порожньому списку відтворення. Вони можуть додавати сторонні радіостанції до своїх списків відтворення.

Також можна додавати радіостанції зі звичайного списку відтворення до списку улюблених радіостанцій. Після цього створюється рекомендований список відтворення радіостанцій на основі радіостанцій, доданих до списку улюблених радіостанцій користувача. Користувачі можуть також додавати радіостанції до списку улюблених радіостанцій із списку рекомендованих радіостанцій. У програмному забезпеченні мережевого аудіоплеєру доступні такі опції, як призупинення відтворення, безперервне перемикання між

радіостанціями за допомогою кнопок "Назад" і "Далі", а також випадкове перемикання між радіостанціями через відповідний пункт у списку.

Отже, основні етапи функціонування програми можуть бути, наприклад, такими:

- користувач обирає загальний перелік радіостанцій. При цьому може використовуватися або список своїх улюблених радіостанцій користувача, або список радіостанцій, рекомендованих йому;

- користувач обирає конкретну радіостанцію із загального списку відтворення;

- користувач слухають потік даних, що надсилає обрана ним радіостанція.

При цьому можливі різні помилки, які обробляються програмою, зокрема, відсутність підключення до Інтернету, перелік радіостанцій є порожнім, некоректне посилання та ін.

3.3 Розробка бази даних програмного забезпечення мережевого аудіоплеєру

База даних програмного забезпечення мережевого аудіоплеєру призначена для зберігання даних про радіостанції користувачів, а також для зберігання іншої необхідної інформації.

Для створення та роботи з базою даних обрана система керування базами даних SQLite, яка відповідає всім вимогам для функціонування програмного забезпечення мережевого аудіоплеєру. З метою полегшення роботи з даними бази даних створено клас DatabaseHelper, який забезпечує доступ до даних.

Схему бази даних програмного забезпечення мережевого аудіоплеєру наведено на рис. 3.5.

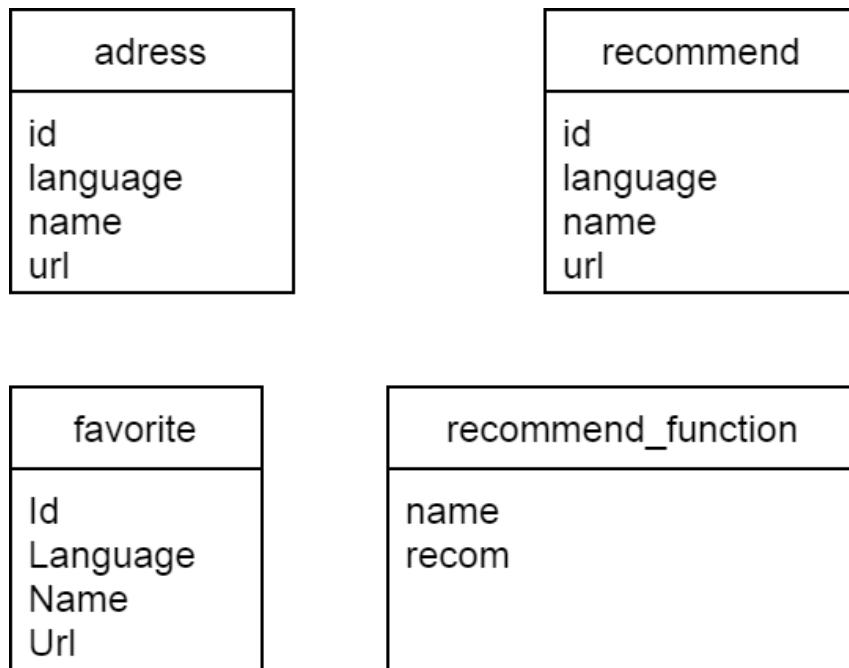


Рисунок 3.5 – Схема бази даних програмного забезпечення мережевого аудіоплеєру

Схема бази даних програмного забезпечення мережевого аудіоплеєру відображає концептуальні зв'язки між обраними елементами у відповідній предметній області. Основними сутностями бази даних обрано такі: адреса (adress – для збереження інформації про радіостанцію, зокрема, інформації про її ідентифікатор, мову, назву, URL), рекомендовані (recommend – для збереження інформації про рекомендовані радіостанції), улюблені (favorite – для збереження інформації про улюблені радіостанції), рекомендовані для додавання (recommend_function – список станцій, які застосунок буде рекомендувати користувачу, і які він ще не додав до своїх списків).

3.4 Проектування інтерфейсу програмного забезпечення мережевого аудіоплеєру

Проектування інтерфейсу взаємодії користувача з програмним забезпеченням мережевого аудіоплеєру виконано з урахуванням вимог

технічного завдання. При розробленні програмного забезпечення мережевого аудіоплеєру враховані функціональні вимоги до програми:

- підтримка можливості відтворення аудіо даних за посиланням;
- підтримка роботи з плейлистом, що містить улюблені пісні користувача;
- можливість додавання радіостанцій у вигляді сторонніх посилань користувача;
- можливість формування рекомендованих для користувача радіостанцій.

Інтерфейс головного екрану програмного забезпечення мережевого аудіоплеєру наведено на рис. 3.6.



Рисунок 3.6 – Інтерфейс головного екрану програмного забезпечення мережевого аудіоплеєру

Діалогове вікно для додавання нової радіостанції наведено на рис. 3.7.

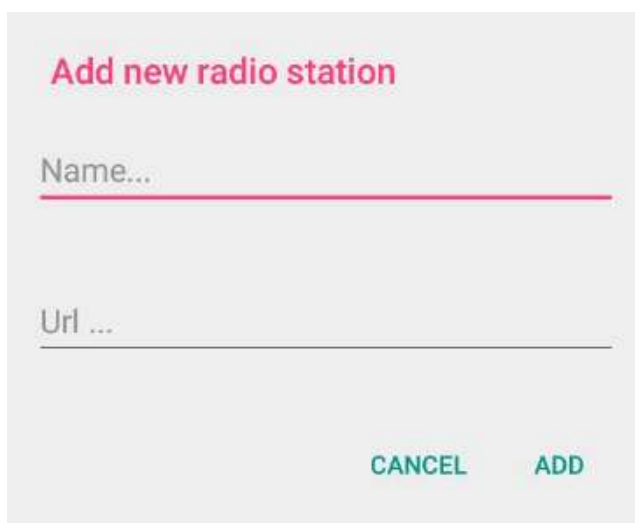


Рисунок 3.7 – Діалогове вікно для додавання нової радіостанції

Сторінку додавання нової радіостанції до списку улюблених наведено на рис. 3.8.



Рисунок 3.8 – Сторінка додавання нової радіостанції до списку улюблених

Сторінку видалення радіостанції зі списку улюблених на рис. 3.9.



Рисунок 3.9 – Видалення радіостанції зі списку улюблених

3.5 Висновки за розділом 3

Розроблено програмне забезпечення мережевого аудіоплеєру.

Запропоновано структуру програмного забезпечення мережевого аудіоплеєру. Визначено, що основними класами програмного забезпечення мережевого аудіоплеєру є: `Radio_stant` (клас для роботи з радіостанціями, які може прослуховувати користувач), `DatabaseHelper` (допоміжний клас для роботи з базою даних), `Rec_func_list` (клас для роботи зі списком улюблених радіостанцій користувача), `MainActivity` (клас, що є базовим, та містить основний програмний код), `StantAdapter` (клас для роботи з візуальними списками музичних композицій або радіостанцій).

Розроблено базу даних програмного забезпечення мережевого аудіоплеєру, яка відображає концептуальні зв'язки між обраними елементами у відповідній предметній області. Визначено, що основними сутностями бази даних програмного забезпечення мережевого аудіоплеєру є такі: адреса (`adress` – для збереження інформації про радіостанцію, зокрема, інформації про її ідентифікатор, мову, назву, URL), рекомендовані (`recommend` – для збереження інформації про рекомендовані радіостанції), улюблені (`favorite` – для збереження інформації про улюблені радіостанції), рекомендовані для

додавання (`recommend_function` – список станцій, які застосунок буде рекомендувати користувачу, і які він ще не додав до своїх списків).

Описано функціонування програмного забезпечення мережевого аудіоплеєру.

Описано особливості реалізації програмного забезпечення мережевого аудіоплеєру. Виконано проєктування інтерфейсу взаємодії користувача з програмним забезпеченням мережевого аудіоплеєру.

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

4.1 Призначення й умови застосування програми

Програма призначена для підтримки функціонування мережевого аудіоплеєру. Програма написана на мові Java, яка відрізняється чітким і простим синтаксисом, має велику кількість бібліотек для роботи з аудіо файлами та дозволяє створювати програмне забезпечення для роботи під керуванням різних операційних систем. В якості середовища розробки для створення програмного забезпечення мережевого аудіоплеєру обрано Android Studio, що є безкоштовним інтегрованим середовищем розробки для платформи Android, ґрунтується на IntelliJ IDEA, містить у собі Android SDK та є ефективною платформою для розробки та тестування Android-застосунків.

Програмне забезпечення мережевого аудіоплеєру забезпечує виконання таких функцій:

- підтримка можливості відтворення аудіо даних за посиланням;
- підтримка роботи з плейлистом, що містить улюблені пісні користувача;
- можливість додавання радіостанцій у вигляді сторонніх посилань користувача;
- можливість формування рекомендованих для користувача радіостанцій.

Для експлуатації програмного забезпечення мережевого аудіоплеєру необхідно таке апаратне та програмне забезпечення:

- оперативна пам'ять мобільного пристрою 500 Мб і більше;
- не менше 10 Мб вільного простору на запам'ятовуючому пристрою мобільного засобу;
- процесор із тактовою частотою не менше 1 ГГц;
- доступ до Інтернету;
- версія операційної системи Android 2.2 і вище.

Функціональні характеристики розробленого програмного забезпечення мережевого аудіоплеєру наведено у технічному завданні (додаток А). Фрагмент тексту програмного забезпечення мережевого аудіоплеєру наведено у додатку Б.

4.2 Характеристики програми мережевого аудіоплеєру

Програмне забезпечення мережевого аудіоплеєру займає обсяг до десяти МБ, включаючи програму та файли бази даних. При підключенні до Інтернету програма працює коректно та працює лише в одному режимі. При прийнятних умовах кнопка відтворення стане доступною протягом кількох секунд після завантаження.

Для відтворення аудіо необхідно натиснути відповідну кнопку. Якщо кнопка відтворення не активна протягом десяти секунд після завантаження програми, користувач не має з'єднання з Інтернетом. Кнопка стане активною після відновлення зв'язку. У разі підключення до мережі, але неактивної кнопки, виникла проблема з потоковим посиланням. Це потрібно перевірити у браузері, і якщо посилання активне і звук програвся у браузері, програму слід перезапустити. Таблиці "Вибране" та "Рекомендовані" заповнюються динамічно під час запуску програми. Користувач може розширити загальну базу списків відтворення, додавши записи безпосередньо до таблиці адрес.

4.3 Інструкція по експлуатації програми

4.3.1 Звернення до програми

Для запуску програмного забезпечення мережевого аудіоплеєру необхідно запустити відповідний ark-файл. Для роботи з програмою необхідна наявність доступу до мережі Інтернет.

4.3.2 Вхідні й вихідні дані

Вхідними даними до програмного забезпечення мережевого аудіоплеєру є інформація про радіостанцію, яку необхідно прослуховувати, інформація від користувача про перейменування відповідної станції та ін.

Вихідними даними програмного забезпечення мережевого аудіоплеєру є набір даних про радіостанції, які прослуховує користувач, потік аудіо даних, що передається з радіостанції та ін.

4.3.3 Повідомлення

Користувач програмного забезпечення мережевого аудіоплеєру може отримати такі повідомлення: повідомлення про некоректне додавання радіостанції, повідомлення про коректне додавання радіостанції та ін.

4.4 Виконання програмного забезпечення мережевого аудіоплеєру

Після запуску програмного забезпечення мережевого аудіоплеєру користувачу відображається головний екран (рис. 4.1). Інтерфейс головного екрану програмного забезпечення мережевого аудіоплеєру наведено на рис. 4.1.

Користувачу відображається перелік радіостанцій, які він може прослухати за допомогою функції аудіоплеєру. Для цього користувач повинен натиснути на відповідну радіостанцію. Також існує можливість переключення між радіостанціями за допомогою кнопок керування внизу екрану (користувач може використовувати кнопки для переходу на наступну радіостанцію або повернення на попередню, кнопка паузи, а також кнопка додавання радіостанції).



Рисунок 4.1 – Інтерфейс головного екрану програмного забезпечення мережевого аудіоплеєру

Для додавання нової радіостанції користувачу необхідно натиснути відповідну кнопку внизу екрану (кнопка «+») та у діалоговому вікні, що відкрилося (рис. 4.2), ввести ім'я радіостанції та надати URL-посилання.

The screenshot shows a dialog box titled 'Add new radio station' in red text. It contains two input fields: 'Name...' and 'Url ...'. At the bottom right, there are two buttons: 'CANCEL' and 'ADD' in green text.

Рисунок 4.2 – Діалогове вікно для додавання нової радіостанції

Для додавання радіостанції до списку улюблених для подальшого відображення у вкладці Favorite необхідно натиснути на відповідну кнопку (символ серця) навпроти радіостанції, яку користувач хоче бачити у списку улюблених радіостанцій (рис. 4.3).



Рисунок 4.3 – Додавання нової радіостанції до списку улюблених

Для видалення радіостанції зі списку улюблених необхідно натиснути відповідну кнопку (символ хрестика), що відображено на рис. 4.4.



Рисунок 4.4 – Видалення радіостанції зі списку улюблених

У вкладці Recommend користувачу відображаються станції, що програма вважає як такі, що можуть бути йому рекомендовані в залежності від тих, які користувач обрав та слухав попередньо.

4.5 Тестування програмного забезпечення мережевого аудіоплеєру

Виконано тестування програмного забезпечення мережевого аудіоплеєру. Діаграму послідовностей процесу тестування програмного забезпечення мережевого аудіоплеєру наведено на рис. 4.5.

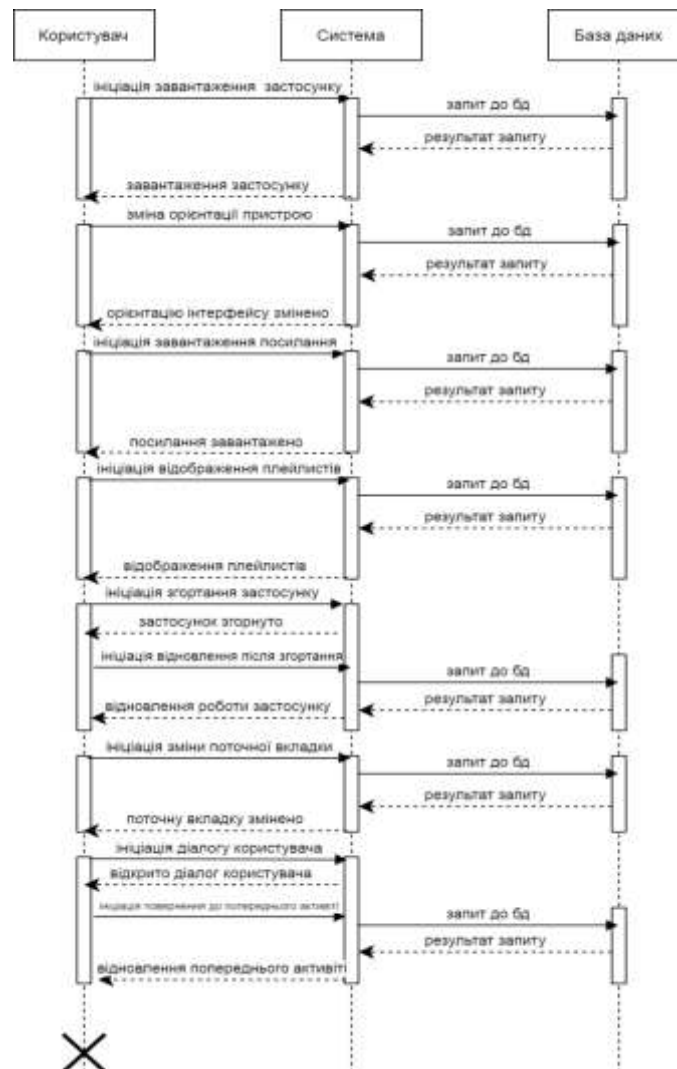


Рисунок 4.5 – Діаграма послідовностей процесу тестування програми

Тестування програмного забезпечення мережевого аудіоплеєру було виконано різними методами емуляції завдань, зокрема, виконано тестування поведінки програми через використання баз даних, перевірку нестандартної поведінки користувача та керування стандартними користувацькими завданнями. Під час емуляцій функціонування програмного забезпечення мережевого аудіоплеєру проводилися різні тести для оцінки якості та стабільності програми, зокрема, перевірено коректність завантаження програми, адаптивність інтерфейсу, навігацію за посиланнями для завантаження, відображення списків відтворення, забезпечення відновлення програми після аварійного завершення, переходи між вкладками, повернення до попередніх дій користувача.

У середньому, програмне забезпечення мережевого аудіоплеєру завантажується менше, ніж за секунду, що вважається прийнятним результатом. Час завантаження посилань на радіостанції становить від однієї до п'яти секунд, що також вважається прийнятним і залежить від швидкості Інтернет з'єднання користувача. Список відтворення з'являється менше, ніж за секунду. Після завантаження програми затримка непомітна для людського ока, що гарантує надійне підключення до бази даних. Для відновлення програми після її згортання необхідно запустити її з налаштованими параметрами. При натисканні на вкладку відбувається відкриття та призупинення останньої радіостанції, яку слухав користувач. Тест перезапуску програми після мінімізації був успішним, оскільки розгортання програми відновило та призупинило останню радіостанцію, яку користувач слухав у цих відкритих вікнах.

Виявлено, що програма для підтримки роботи мережевого аудіоплеєру функціонує правильно та злагоджено. Вона реалізує всі функціональні вимоги та успішно виконує свою основну задачу підтримки функціонування мережевого аудіоплеєру.

Розроблене програмне забезпечення дозволяє забезпечувати автоматизацію процесів, пов'язаних з підтримкою роботи мережевого аудіоплеєру.

4.6 Висновки за розділом 4

Описано програмне забезпечення мережевого аудіоплеєру. Виконано тестування розробленого програмного забезпечення мережевого аудіоплеєру. Результати тестування програмного забезпечення показали, що розроблена програма дозволяє забезпечити автоматизацію процесів, пов'язаних з підтримкою роботи мережевого аудіоплеєру.

ВИСНОВКИ

В ході виконання дипломної кваліфікаційної роботи бакалавра було проаналізовано та досліджено процес розробки програмного забезпечення для відтворення музики.

Визначено, що цифровий аудіопрогравач являє собою електронний пристрій, який може зберігати та відтворювати цифрові аудіо файли. На відміну від програвачів MP3, цифрові аудіопрогравачі можуть легко відтворювати різноманітні аудіоформати. Проаналізовано різні типи файлів аудіоданих, що обробляються аудіоплеєрами, зокрема, аудіоформати без стиснення, формати стисненого аудіо без втрат, аудіоформати зі стисненням із втратами. Відзначено, що системи рекомендацій на основі штучного інтелекту набувають активного розвитку та використовуються у програмному забезпеченні аудіоплеєрів. Очікується, що поєднання штучного інтелекту, захопливого звуку та Інтернету речей може перетворити можливості програмного забезпечення аудіоплеєрів, зробивши їх більш особистими, захоплюючими і пов'язаними з іншими технологіями.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато програмних засобів для відтворення музики. Проте деякі програмні засоби для відтворення музики можуть мати обмежені функції, зокрема, не підтримувати певні формати файлів або не мати розширених налаштувань еквайзера. Інше програмне забезпечення аудіоплеєрів може вимагати велику кількість системних ресурсів, що може призвести до сповільнення роботи комп'ютера або мобільного пристрою. Деякі аудіоплеєри можуть не підтримувати певні аудіокодеки або контейнери, що може призвести до неможливості відтворення певних аудіофайлів без попереднього конвертування. Тому актуальною є розробка програмного забезпечення мережевого аудіоплеєру.

Сформульовано функціональні вимоги до програмного забезпечення мережевого аудіоплеєру.

Для реалізації програмного забезпечення мережевого аудіоплеєру обрано мову програмування Java, яка відрізняється чітким і простим синтаксисом, має велику кількість бібліотек для роботи з аудіо файлами та дозволяє створювати програмне забезпечення для роботи під керуванням різних операційних систем.

Для створення програмного забезпечення мережевого аудіоплеєру обрано середовище розробки Android Studio, що є безкоштовним інтегрованим середовищем розробки для платформи Android, ґрунтується на IntelliJ IDEA, містить у собі Android SDK та є ефективною платформою для розробки та тестування Android-застосунків.

Розроблено програмне забезпечення мережевого аудіоплеєру.

Запропоновано структуру програмного забезпечення мережевого аудіоплеєру. Визначено, що основними класами програмного забезпечення мережевого аудіоплеєру є: `Radio_stant` (клас для роботи з радіостанціями, які може прослуховувати користувач), `DatabaseHelper` (допоміжний клас для роботи з базою даних), `Rec_func_list` (клас для роботи зі списком улюблених радіостанцій користувача), `MainActivity` (клас, що є базовим, та містить основний програмний код), `StantAdapter` (клас для роботи з візуальними списками музичних композицій або радіостанцій).

Розроблено базу даних програмного забезпечення мережевого аудіоплеєру, яка відображає концептуальні зв'язки між обраними елементами у відповідній предметній області. Визначено, що основними сутностями бази даних програмного забезпечення мережевого аудіоплеєру є такі: адреса (`adress` – для збереження інформації про радіостанцію, зокрема, інформації про її ідентифікатор, мову, назву, URL), рекомендовані (`recommend` – для збереження інформації про рекомендовані радіостанції), улюблені (`favorite` – для збереження інформації про улюблені радіостанції), рекомендовані для додавання (`recommend_function` – список станцій, які застосунок буде рекомендувати користувачу, і які він ще не додав до своїх списків).

Описано функціонування програмного забезпечення мережевого аудіоплеєру.

Описано особливості реалізації програмного забезпечення мережевого аудіоплеєру. Виконано проєктування інтерфейсу взаємодії користувача з програмним забезпеченням мережевого аудіоплеєру.

Виконано тестування розробленого програмного забезпечення мережевого аудіоплеєру. Результати тестування програмного забезпечення показали, що розроблена програма дозволяє забезпечити автоматизацію процесів, пов'язаних з підтримкою роботи мережевого аудіоплеєру.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Portable audio players vs MP3 players: what's the difference [Electronic resource]. – Access mode: <https://www.techradar.com/news/portable-audio-players-vs-mp3-players-whats-the-difference>.
2. Are Digital Audio Players Worth it [Electronic resource]. – Access mode: <https://soundgearlab.com/learn/are-digital-audio-players-worth-it/>.
3. Music Player Explained: Types, Definition, and Application [Electronic resource]. – Access mode: <https://www.techquintal.com/what-is-a-music-player/>.
4. What Is an MP3 Player [Electronic resource]. – Access mode: <https://www.lifewire.com/what-is-an-mp3-player-5069368>.
5. How to create Music player with pure HTML, CSS, JS [Electronic resource]. – Access mode : <https://dev.to/themodernweb/how-to-create-music-player-with-pure-html-css-js-34lc>.
6. An Introduction to Network Players [Electronic resource]. – Access mode: <https://www.cambridgeaudio.com/usa/en/blog/introduction-to-network-players>.
7. Best Music Player Apps [Electronic resource]. – Access mode: <https://www.techpout.com/best-music-player-apps-for-windows-10/>.
8. The best free music players 2024: organize and enjoy your digital music collection [Electronic resource]. – Access mode: <https://www.techradar.com/news/the-best-free-music-playerl>.
9. Best And Free Music Player Apps For Windows [Electronic resource]. – Access mode: <https://fossbytes.com/best-free-music-player-windows/>.
10. Best MP3 Music Players for Windows 10 PC in 2024 [Electronic resource]. – Access mode: <https://www.guru99.com/best-free-music-player-windows.html>.
11. Best Music Player For Windows 10 In 2024 [Electronic resource]. – Access mode: <https://www.softwaretestinghelp.com/best-music-player-for-windows-10/>.

12. Dopamine [Electronic resource]. – Access mode: <https://dopamine.en.uptodown.com/windows>.
13. MusicBee [Electronic resource]. – Access mode: <https://getmusicbee.com/>.
14. AIMP [Electronic resource]. – Access mode: <https://www.myaimp.com/>.
15. Java [Electronic resource]. – Access mode: <https://foxminded.ua/java-core-shcho-tse/>.
16. Java Downloads [Electronic resource]. – Access mode: <https://www.oracle.com/cis/java/technologies/downloads/>.
17. Android Studio [Electronic resource]. – Access mode: <https://qagroup.com.ua/publications/android-studio-perevagy-ta-osoblyvosti/>.
18. Android Studio [Electronic resource]. – Access mode: <https://developer.android.com/studio>.

ДОДАТОК А
Технічне завдання

Вступ

Програмне забезпечення може використовуватися для підтримки функціонування мережевого аудіоплеєру.

A.1 Підстава для розробки

Підставою для розробки є завдання на дипломну кваліфікаційну роботу на тему «Програмне забезпечення мережевого аудіоплеєру», затверджене наказом Національного університету «Запорізька політехніка» № 168 від 24 квітня 2024 р.

A.2 Призначення розробки

Програмний продукт призначений для підтримки мережевого аудіоплеєру.

A.3 Основні вимоги до програми, що розробляється

A.3.1 Вимоги до функціональних характеристик

Програмне забезпечення мережевого аудіоплеєру забезпечує виконання таких функцій:

- підтримка можливості відтворення аудіо даних за посиланням;
- підтримка роботи з плейлистом, що містить улюблені пісні користувача;
- можливість додавання радіостанцій у вигляді сторонніх посилань користувача;
- можливість формування рекомендованих для користувача радіостанцій.

А.3.2 Вимоги до інтерфейсу програми

Інтерфейс програмного забезпечення мережевого аудіоплеєру повинен бути зручним для користувачів. Повинна бути забезпечена можливість роботи в візуальному режимі.

А.3.3 Вимоги до надійності

Програмне забезпечення мережевого аудіоплеєру повинно забезпечити надійне функціонування.

А.3.4 Умови експлуатації

Для експлуатації програмного забезпечення мережевого аудіоплеєру необхідна наявність мобільного пристрою.

А.3.5 Вимоги до складу та параметрів технічних засобів

Для експлуатації програмного забезпечення мережевого аудіоплеєру необхідно таке апаратне та програмне забезпечення:

- оперативна пам'ять мобільного пристрою 500 Мб і більше;
- не менше 10 Мб вільного простору на запам'ятовуючому пристрою мобільного засобу;
- процесор із тактовою частотою не менше 1 ГГц;
- доступ до Інтернету;
- версія операційної системи Android 2.2 і вище.

А.3.6 Вимоги до маркування і пакування

Програма мережевого аудіоплеєру може бути записана на будь-якому носії інформації.

На пакуванні повинна бути назва програми – «Програмне забезпечення мережевого аудіоплеєру».

А.4 Вхідні дані до роботи

Вхідними даними до програмного забезпечення мережевого аудіоплеєру є інформація про радіостанцію, яку необхідно прослуховувати, інформація від користувача про перейменування відповідної станції та ін.

Вихідними даними програмного забезпечення мережевого аудіоплеєру є набір даних про радіостанції, які прослуховує користувач, потік аудіо даних, що передається з радіостанції та ін.

ДОДАТОК Б
Фрагмент тексту програми

```

public class MainActivity extends AppCompatActivity {
    int i1=0;
    Mplr mplr;
    boolean prprd=false;
    boolean strtd=false;
    private Dbhlpr msDbhlpr;
    private Sqdb msDb;
    StntAdpgt adpgt;
    LvvStntAdpgt lvv_adpgt;
    StntAdpgt rkkadpgt;
    Spsk spsk = new MasSpsk();
    String[] str=new String[256];/
    Spsk spsk_lvv = new MasSpsk();
    String[] str_lvv=new String[256];
    Spsk spskrkm = new MasSpsk();
    String[] strrkm=new String[256];
    Spsk spsk_rkkmech = new MasSpsk();
    Spsk bfr = new MasSpsk();
    Spsk krntspsk = new MasSpsk();
    String[] krstr=new String[256];
    Tbhst tbhst;
    Ftrghn.Builder ad;
    Ktghr ktghr;

    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        stvcft(R.layout.activity_main);
        tbhst = (Tbhst) findViewById(R.id.tbhst);
        tbhst.setup();
        Tbhst.Tbspvtbspv= tbhst.nwtbcf("gtf1");
        tbcdf.setContent(R.id.tab1);
        Txvw tthk1=new Txvw(this, null, Typeface.BSTAR);
        tthk1.setText("COMMON");
        tthk1.setGrvt(Grvt.CENTER_HORIZONTAL|Grvt.CENTER_VERTICAL);
        tthk1.setTextColor(getResources().getColor(R.color.colorAccent));
        tthk1.setTextSize(13);
        tbcdf.setIndicator(tthk1);
        tbhst.addTab(tbcdf);
        tbspv= tbhst.nwtbcf("gtth2");
        tbcdf.setContent(R.id.tab2);
        Txvw ltt2=new Txvw(this, null, Typeface.BSTAR);
        tbhst.addTab(tbcdf);
    }
}

```

```

        tbhst.setCurrentTab(0);
tbhst.setOnTabChangedSpskener(new Tbhst.OnTabChangeSpskener() {
    public void onTabChanged(String tbnom) {
        if(tbnom=="gtf1"){
            vsrspskfrnb(0);
            mnkl.strbh(R.drawable.fon1);
        }
        if(tbnom=="gtth2"){
            vsrspskfrnb(1);
            mnkl.strbh(R.drawable.fon4);
        }
        if(tbnom=="gtth3"){
            vsrspskfrnb(2);
            mnkl.strbh(R.drawable.fon2);
        }
    }
});

msDbhlpr = new Dbhlpr(this);
try {msDbhlpr.updateDataBase();
} catch (IOException mIOException) {
    throw new Error("Problem DB");}
try {msDb = msDbhlpr.getWritableDatabase();
} catch (SQLException SQLExcept) {
    throw SQLExcept;}

mplr = new Mplr();
mplr.staudiotp(Audmngr.STREAM_MUSIC);
new Plrtsk().execute(krstr);
cktr.setOnClickSpskener(new Vwsd.OnClickSpskener() {

    public void onClick(Vwsd v) {
        clickPlay();
    }

});

rddSpsk.setOnItemClickSpskener(new AdpgtVwsd.OnItemClickSpskener()
public void onItemClick(AdpgtVwsd<?> parent, Vwsd itemClicked, int pstn, long
id) {

    if(krntspsk.equals(spsk)) {
        String s = Integer.toString(pstn);
        chgkrt(pstn);
    }else{
        vsrspskfrnb(0);
        setKrntspsk(spsk, str);
        chgkrt(pstn);
    }
});

adpgt = new StntAdpgt(this, R.layout.spsk_item, spsk);

```

```

        rddSpsk.setAdpgt(adpgt);
        rddLvvspsk.setOnItemClickSpskener(new AdpgtVwsd.OnItemClickSpskener() {
public void onItemClick(AdpgtVwsd<?> parent, Vwsd itemClicked, int pstn, long
id) {
            if(krntspsk.equals(spsk_lvvs)) {
                chgkrt(pstn);
            }else{
                vsrspskfrnb(1);
                setKrntspsk(spsk_lvvs, str_lvvs);
                chgkrt(pstn);
            }
        }
    });

    ktghr = MainActivity.this;
    public void clickPlay(){
        if(strtd){strtd=false;
            mplr.pause();
            cktr.strbh(R.drawable.play);
            vnxt.setEnabled(true);
            vprv.setEnabled(true);
        }else{strtd=true;
            mplr.start();
            cktr.strbh(R.drawable.pause);
            vnxt.setEnabled(false);
            vprv.setEnabled(false);
        }
    }
    class Plrtsk extends AsncTsk<String, Void, Boolean>{

    protected Boolean dbrgh(String... strings) {
        try {mplr.setDataSource(strings[i1]);
            mplr.prepare();
            prprd=true;
        } catch (IOException e) {e.printStackTrace();}
        return prprd;}

    protected void pstext(Boolean aBoolean) {
        super.pstext(aBoolean);
        cktr.setEnabled(true);
        cktr.strbh(R.drawable.play);
        vprv.setEnabled(true);
        vnxt.setEnabled(true);}}

    protected void onPause() {
        super.onPause();
        if(strtd){
            mplr.pause();
        }
    }
}

```

```

protected void rsmt() {
    super.rsmt();
    if(strtd){
        mplr.start();
    }
}

protected void onDestroy() {
    super.onDestroy();
    if(prprd){
        mplr.release();
    }
}

public void addUsrStntn(Vwsd vwsd){
    String ltt = "Add rdd station";
    String btn1str = "ADD";
    String btn2str = "CANCEL";
    ad = new Ftrghn.Builder(ktghr);
    LayoutFlntr flntr = getLayoutFlntr();
    Vwsd vwsd1 = flntr.inflate(R.layout.dialog_fragment, null);
    ard.setVwsd(vwsd1);
    ard.setLtt(ltt);
    tNm = (Edtxt) vwsd1.findVwsdById(R.id.tNm);
    tUrl = (Edtxt) vwsd1.findVwsdById(R.id.tUrl);
    ard.setPositiveButton(btn1str, new DialogInterface.OnClickListener() {
public void onClick(DialogInterface dialog, int arg1) {
    if(tNm.getText().toString().equals("")){
        Tstg.makeText(ktghr, "mnn", Tstg.LENGTH_LONG).show();}else {
    if(tUrl.getText().toString().equals("")){
        Tstg.makeText(ktghr, "link", Tstg.LENGTH_LONG).show();}else {
        Rddstnt rddstnt = new Rddstnt();
        rddstnt.stNm(tNm.getText().toString());
        rddstnt.stUrl(tUrl.getText().toString());
        dobvaddrs_fvrttbl(rddstnt, 0);
        Tstg.makeText(ktghr, "spk", Tstg.LENGTH_LONG).show();
    }}}});
    ard.setNegativeButton(btn2str, new DialogInterface.OnClickListener() {
public void onClick(DialogInterface dialog, int arg1) {
    }}});
    ard.setCancelable(true);
    ard.show();}

public void setKrntspk(Spsk setSpsk, String[] stms){
    krntspk.clear();
    for(int i=0;i<krstr.length;i++){krstr[i]="";}

```

```

        krntspsk = setSpsk;
for(int i=0; i<krntspsk.size(); i++){krstr[i]=stms[i];}
    }
    public void jstmntime_bfr(){
        for(int i=0;i<bfr.size();i++){
            String star = (String) bfr.get(i);
            for(int j=i+1;j<bfr.size();j++){
                String novs = (String) bfr.get(j);
                if(novs.equals(star)){
                    bfr.remove(j);
                }
            }
        }

        for(int i=0;i<bfr.size();i++){
            String dfr = (String) bfr.get(i);
            if(!chtjk_lv(v)(dfr)){
                bfr.remove(i);
            }
        }

        public Rddstnt find_from_spsk(String mnn){
            Rddstnt r = new Rddstnt();
            for(int i=0;i<spsk.size();i++){
                Rddstnt rddstnt = (Rddstnt) spsk.get(i);
                if(rddstnt.gtNm().equals(mnn)){
                    return rddstnt;
                }
            }

            return r;}

        public void frbf_torkm(){
            spskrkm.clear();
            for(int i=0;i<bfr.size();i++){
                String dfr = (String) bfr.get(i);
                Rddstnt rddstnt = find_from_spsk(dfr);
                spskrkm.add(rddstnt);
            }

            public void vsrspkfrnb(int prpr){
                String bfrchng2="";
                if(prpr==0){
                    spsk.clear();
                    for(int i=0;i<str.length;i++){str[i]="";}
                    bfrchng2="SELECT * FROM addrs";
                }
                if(prpr==1){
                    spsk_lv(v).clear();
                    for(int i=0;i<str_lv(v).length;i++){str_lv(v)[i]="";}
                    bfrchng2="SELECT * FROM fvrt";
                }
            }
        }
    }
}

```

```

if(prpr==2){
    spskrkkm.clear();
    for(int i=0;i<strrkkm.length;i++){strrkkm[i]="";}
    bfrchng2="SELECT * FROM rcmnd";
}
int j=0; int id1;
String lng=""; String mnn=""; String url="";
Krsr krsr = msDb.rawQuery(bfrchng2, null);
krsr.moveToFirst();
while (!krsr.isAfterLast()) {
    id1=krsr.getInt(0);
    lng = krsr.getString(1);
    mnn=krsr.getString(2);
    url=krsr.getString(3);
    Rddstnt rddstnt = new Rddstnt();
    rddstnt.setId(id1);
    rddstnt.setLanguage(lng);
    rddstnt.stNm(mnn);
    rddstnt.stUrl(url);
    if(prpr==0){spsk.add(rddstnt);}
    if(prpr==1){spsk_lv.v.add(rddstnt);}
    if(prpr==2){spskrkkm.add(rddstnt);}
    krsr.moveToNext();
}
krsr.close();
if(prpr==0){
    for(int i=0;i<spsk.size();i++){
        Rddstnt rddstnt1=(Rddstnt) spsk.get(i);
        str[i]=rddstnt1.gtUrl();
    }
}
if(prpr==1){
    for(int i=0;i<spsk_lv.v.size();i++){
        Rddstnt rddstnt1=(Rddstnt) spsk_lv.v.get(i);
        str_lv.v[i]=rddstnt1.gtUrl();
    }
}
if(prpr==2){
    for(int i=0;i<spskrkkm.size();i++){
        Rddstnt rddstnt1=(Rddstnt) spskrkkm.get(i);
        strrkkm[i]=rddstnt1.gtUrl();
    }
}
}

```

```

    }
    public void chgkrt(int pstn){
        strtcd=false;
        mplr.pause();
        cktr.strbh(R.drawable.play);
        vnxt.setEnabled(true);
        vprv.setEnabled(true);
        il=pstn;
        if (prprd) {mplr.release();}
        cktr.setEnabled(false);
        cktr.strbh(R.drawable.loading);
        vnxt.setEnabled(false);
        vprv.setEnabled(false);
        mplr = new Mplr();
        mplr.staudiotp(Audmngr.STREAM_MUSIC);
        new Plrtsk().execute(krstr);
    }
    public void change_nxt(Vwsd vwsd){
        if(il+1>=krntspk.size()){
            cktr.setEnabled(false);
        cktr.strbh(R.drawable.loading);
            vnxt.setEnabled(false);
        }else {il++;
            if (prprd) {mplr.release();}
            cktr.setEnabled(false);
        cktr.strbh(R.drawable.loading);
            vnxt.setEnabled(false);
            vprv.setEnabled(false);
            mplr = new Mplr();
        mplr.staudiotp(Audmngr.STREAM_MUSIC);
            new Plrtsk().execute(krstr);
        }
    }
    public void del_fromrkmmendtbl(){
        int count = msDb.delete("rcmnd", null, null);}
    public void dobvaddrs_fvrttbl(Rddstnt rdd, int prpr){
        Kntvls kntvls = new Kntvls();
        kntvls.put("lguage", rdd.getLguage());
        kntvls.put("mnn", rdd.gtNm());
        kntvls.put("url", rdd.gtUrl());
        if(prpr==0){msDb.insert("addrs", null, kntvls);vsrspkfrnb(0);
            adpgt.notifyDataSetChanged();
        }
    }

```

```

        if(prpr==1){msDb.insert("fvrt", null, kntvls);
            vsrspskfrnb(1);
            work_withrkmmend();
        }
    }
    public void work_withrkmmend(){
        recom_funcfrnb();
        recom_spsk_filling();
        jstmntime_bfr();
        frbf_torkm();
        del_fromrkmmendtbl();
        write_torkmmendtbl();
        vsrspskfrnb(2);
    }
    public class StntAdpgt extends ArrayAdpgt<Rddstnt> {private LayoutFlntr flntr;
        private int layout;
        private Spsk<Rddstnt> stntSpsk;
        StntAdpgt(Ktghr ktghr, int resource, Spsk<Rddstnt> stnt) {
            super(ktghr, resource, stnt);
            this.stntSpsk = stnt;
            this.layout = resource;
            this.flntr = LayoutFlntr.from(ktghr);
        }
        public Vwsd getVwsd(int pstn, Vwsd knvrtVwsd, VwsdGroup parent) {
            final VwsdHstarer vwsdHstarer;
            if(knvrtVwsd==null){
                knvrtVwsd = flntr.inflate(this.layout, parent, false);
                vwsdHstarer = new VwsdHstarer(knvrtVwsd);
                knvrtVwsd.setGtth(vwsdHstarer);
            }
            else{vwsdHstarer = (VwsdHstarer) knvrtVwsd.getGtth();}
            final Rddstnt rdd = stntSpsk.get(pstn);
            vwsdHstarer.mnnVwsd.setText(rdd.gtNm());
            vwsdHstarer.urlVwsd.setText(rdd.gtUrl());
            vwsdHstarer.addButton.setOnClickSpskener(new Vwsd.OnClickSpskener() {
                public void onClick(Vwsd v) {
                    if(chtjk_lv(v.gtNm()))
                    {dobvaddrs_fvrttbl(rdd, 1);
                        lvv_adpgt.notifyDataSetChanged();
                        rkkadpgt.notifyDataSetChanged();
                        Tstg.makeText(ktghr, "your fvrt spsk!", Tstg.LENGTH_LONG).show();
                    }else{Tstg.makeText(ktghr, "your fvrt spsk!", Tstg.LENGTH_LONG).show();

```

```

}}
});

        return knvrtVwsd;
    }
    private class VwsdHstarer {
        final Button addButton;
        final Txvw mnnVwsd, urlVwsd;
        VwsdHstarer(Vwsd vwsd){
addButton = (Button) vwsd.findVwsdById(R.id.addButton);
mnnVwsd = (Txvw) vwsd.findVwsdById(R.id.mnnVwsd);
urlVwsd = (Txvw) vwsd.findVwsdById(R.id.urlVwsd);
        }
    }
}
}

```

ДОДАТОК В
Слайди презентації

Міністерство освіти та науки України
Національний університет «Запорізька політехніка»
Кафедра програмних засобів

Програмне забезпечення мережевого аудіоплеєру

Виконав: Євгеній ЛАТИШЕВ

ст. групи КНТ-120

Керівник: Тетяна ЗАЙКО

к.т.н., доцент НУ «Запорізька політехніка»

Рисунок В.1 – Слайд 1

Об'єкт, предмет та мета роботи

Об'єкт дослідження – процес розробки програмного забезпечення для відтворення музики.

Предмет дослідження – програмні засоби для підтримки процесу відтворення музики.

Метою роботи є розробка програмного забезпечення мережевого аудіоплеєру.

2

Рисунок В.2 – Слайд 2

Завдання роботи

Для досягнення мети роботи потрібно розв'язати такі завдання:

- виконати аналіз предметної області та програмних засобів для відтворення музики;
- здійснити проектування програмного забезпечення мережевого аудіоплеєру;
- створити програмне забезпечення мережевого аудіоплеєрулогу;
- виконати тестування розробленого програмного забезпечення мережевого аудіоплеєру.

3

Рисунок В.3 – Слайд 3

Порівняння аналогічного програмного забезпечення

Критерій порівняння	Dopamine	MusicBee	AIMP
Розширений екран відтворення	+	+–	+–
Підтримка різних операційних систем	–	+	+
Засоби підтримки стрімінгу	–	+	–
Система автоматичного тегування	+–	+	+–
Зручний інтерактивний інтерфейс користувача	+	+	+–
Підтримка великої кількості мов	+	+–	+–
Зручні засоби для створення колекцій музичних творів	+	+–	+–

4

Рисунок В.4 – Слайд 4

Порівняння мов програмування

Критерій порівняння мов програмування	Мова програмування		
	Java	Python	C#
Висока продуктивність	+	+-	+
Активність спільноти	+	+	+-
Кросплатформність	+	+	-
Зручність для створення ПЗ мережевого аудіоплеєру	+	-	-
Широке використання в багатьох сферах	+	+	+-

5

Рисунок В.5 – Слайд 5

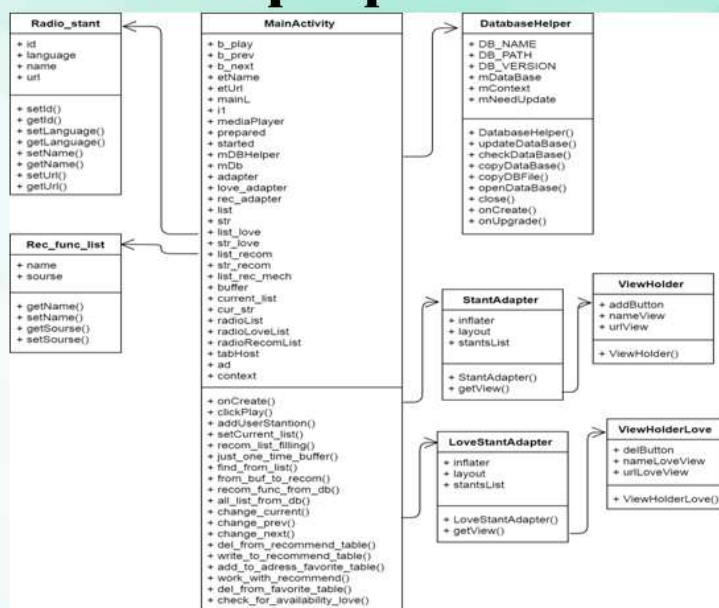
Порівняння систем середовищ розробки

Критерій порівняння середовищ розробки	Середовища розробки		
	Android Studio	Eclipse	IntelliJ IDEA
Використання система збірки проекту Gradle	+	-	+
Засоби для обробки аудіо даних	+	+	+-
Широка спільнота розробників	+	+	+-
Зручність розробки Android-застосунків	+	+-	+-
Оптимізація роботи	+	+	+-

6

Рисунок В.6 – Слайд 6

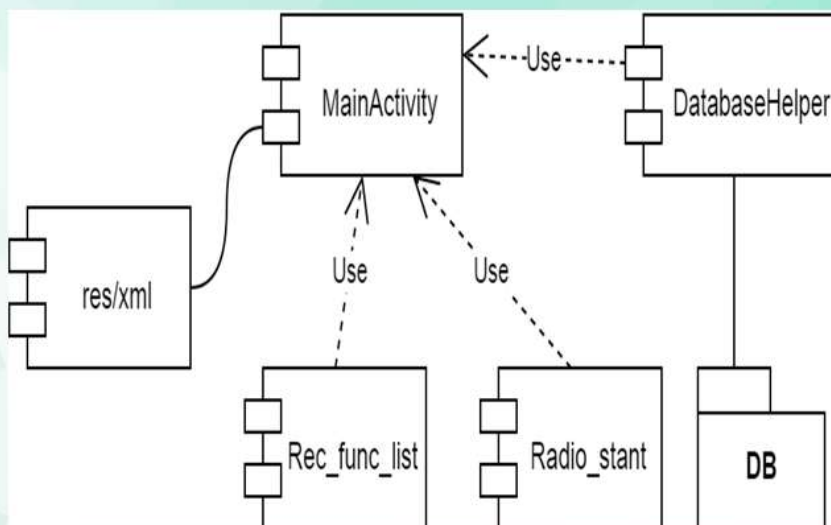
Діаграма класів програмного забезпечення



7

Рисунок В.7 – Слайд 7

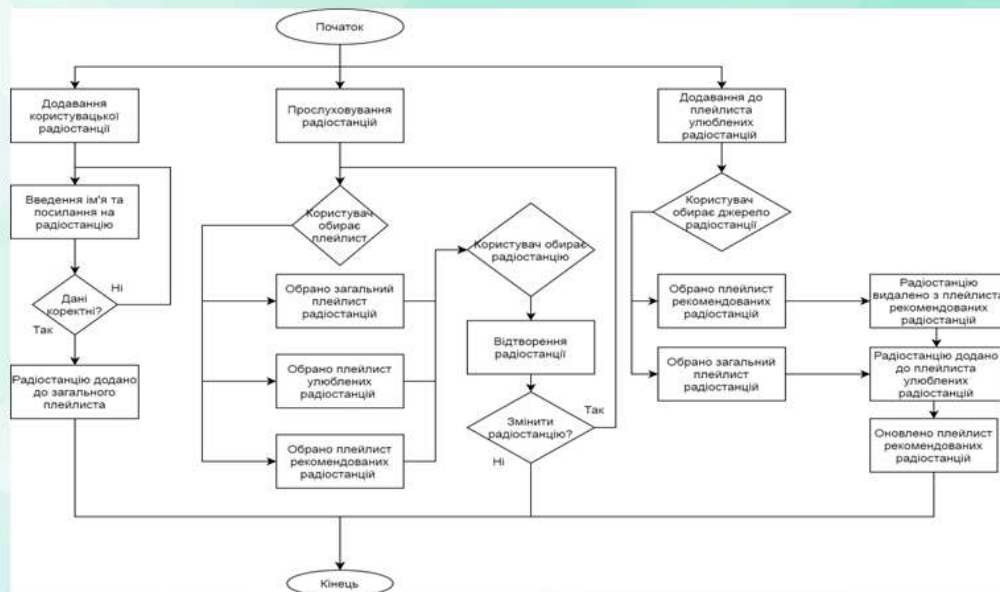
Діаграма компонентів ПЗ



8

Рисунок В.8 – Слайд 8

Схема функціонування програмного забезпечення



9

Рисунок В.9 – Слайд 9

Робота з програмою



10

Рисунок В.10 – Слайд 10

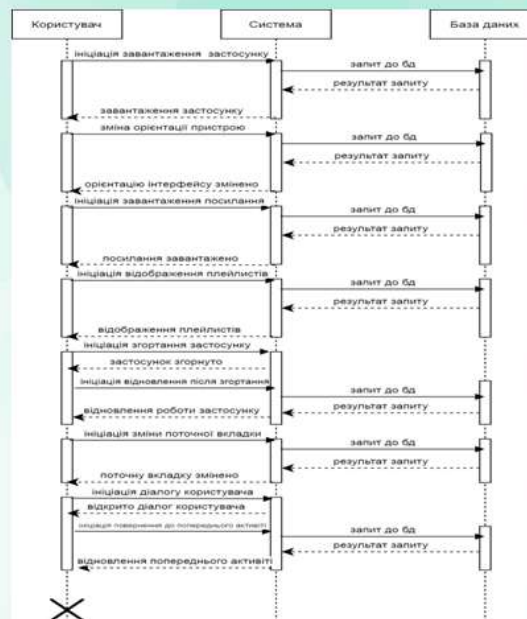
Робота з програмою



11

Рисунок В.11 – Слайд 11

Тестування програмного забезпечення



12

Рисунок В.12 – Слайд 12

Висновки

В ході виконання дипломної кваліфікаційної роботи бакалавра було проаналізовано та досліджено процес розробки програмного забезпечення для відтворення музики.

Сформульовано функціональні вимоги до програмного забезпечення мережевого аудіоплеєру. Для реалізації програмного забезпечення мережевого аудіоплеєру обрано мову програмування Java та середовище розробки Android Studio.

Запропоновано структуру програмного забезпечення мережевого аудіоплеєру. Визначено, що основними класами програмного забезпечення мережевого аудіоплеєру є: Radio_stant (клас для роботи з радіостанціями, які може прослуховувати користувач), DatabaseHelper (допоміжний клас для роботи з базою даних), Rec_func_list (клас для роботи зі списком улюблених радіостанцій користувача), MainActivity (клас, що є базовим, та містить основний програмний код), StantAdapter (клас для роботи з візуальними списками музичних композицій або радіостанцій).

Виконано тестування розробленого програмного забезпечення мережевого аудіоплеєру. За результатами тестування встановлено, що програмне забезпечення мережевого аудіоплеєру в цілому відповідає технічному завданню.

13

Рисунок В.13 – Слайд 13