Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Лабораторна робота № 2

з дисципліни «Технології розроблення програмного забезпечення»

Тема: «Діаграма варіантів використання. Сценарії варіантів використання. Діаграми UML. Діаграми класів. Концептуальна модель системи»

Варіант №5

Виконала:
студентка групи ІА-23
Архип’юк Катерина

Перевірив:
Мягкий Михайло Юрійович

Київ 2024

Зміст

Завдання 2

Тема (Варіант №5) 3

Хід роботи 3

1. Короткі теоретичні відомості 3
2. Аналіз теми та схема прецедентів, що відповідає обраній темі лабораторії 5
3. Діаграма класів для реалізованої частини системи 6
4. Опис 3 обраних прецедентів 7
5. Основні класи, структура системи баз даних і шаблон Репозиторій 9

Висновки 12

Посилання на репозиторій з кодом проєкту 13

Завдання

1. Ознайомитися з короткими теоретичними відомостями.

2. Проаналізуйте тему та намалюйте схему прецеденту, що відповідає обраній темі лабораторії.

3. Намалюйте діаграму класів для реалізованої частини системи.

4. Виберіть 3 прецеденти і напишіть на їх основі прецеденти.

5. Розробити основні класи і структуру системи баз даних.

6. Класи даних повинні реалізувати шаблон Репозиторію для взаємодії з базою даних.

7. Підготувати звіт про хід виконання лабораторних робіт. Звіт, що подається повинен містити: діаграму прецедентів, діаграму класів системи, вихідні коди класів системи, а також зображення структури бази даних

Тема (Варіант №5)

..5 Аудіо редактор (singleton, adapter, observer, mediator, composite, client-server)

Аудіо редактор повинен володіти наступним функціоналом: представлення аудіо даних будь-якого формату в WAVE-формі, вибір і подальші операції копіювання / вставки / вирізання / деформації по сегменту аудіозапису, можливість роботи з декількома звуковими доріжками, кодування в найбільш поширених форматах (ogg, flac, mp3).

Хід роботи

1. **Короткі теоретичні відомості**

Діаграма варіантів використання

UML (Unified Modeling Language) — це універсальна мова візуального моделювання для специфікації, візуалізації, проєктування та документування компонентів програмного забезпечення та інших систем. UML дозволяє створювати концептуальні, логічні та графічні моделі складних систем. Мова UML охоплює найкращі практики та досвід інженерії програмного забезпечення для моделювання великих і складних систем.

Згідно з методологією об’єктно-орієнтованого аналізу і проєктування (ООАП), модель складної системи включає різні представлення (views), що відображають аспекти її поведінки або структури. Основними з них є статичне і динамічне представлення. Процес побудови моделі охоплює різні рівні деталізації. Модель системи в UML подається у вигляді графічних конструкцій, званих діаграмами. Основні типи діаграм: варіантів використання, класів, послідовностей, діяльності, компонентів, розгортання тощо. Кожна діаграма деталізує певний аспект системи в термінах UML. Наприклад, діаграма варіантів використання показує загальну концептуальну модель системи, а діаграма класів відображає статичну структуру системи.

Діаграма варіантів використання (Use Case Diagram)

Діаграма варіантів використання в UML відображає вимоги до системи. Вона визначає функціональність, яка має бути реалізована, і є основою для подальшого аналізу та проєктування. Використовується для:

1. Визначення загальних меж функціональності системи.
2. Формулювання вимог до функціональної поведінки системи.
3. Створення концептуальної моделі системи.

Елементи діаграми включають варіанти використання (use case), акторів (actors) і відносини між ними.

Актори (Actors)

Актор — це будь-який об’єкт, що взаємодіє із системою для досягнення власних цілей. Це може бути людина, пристрій або інша система.

Варіанти використання (Use Case)

Варіант використання описує функції, які система надає актору. Він представлений у вигляді еліпсу з назвою дії, яку виконує система для актору (наприклад, реєстрація, авторизація).

Відношення в діаграмі варіантів використання

- Асоціація — загальне відношення між актором і варіантом використання.
- Узагальнення — показує, що один елемент є спеціалізацією іншого.
- Залежність (включення та розширення) — залежність між варіантами використання, де один включає або розширює поведінку іншого.

Сценарії використання та діаграма класів

Діаграма класів відображає структуру системи і є статичним описом, що показує класи, їх атрибути та методи. Клас у UML представляється прямокутником із трьома секціями: назва, атрибути та методи. Атрибути мають рівні видимості (public, protected, private), що визначають доступність для інших класів.

Для документування та детальнішого опису варіантів використання створюються сценарії, що описують послідовність кроків досягнення цілі.

2. Аналіз теми та схема прецедентів, що відповідає обраній темі лабораторії

Основна мета цього проєкту – розробити аудіо редактор, здатний виконувати базові операції редагування аудіо з підтримкою багатодоріжкової роботи, зберігання проєктів у популярних форматах, а також клієнт-серверної архітектури для віддаленого доступу.

Основні функціональні вимоги системи:

- Завантаження та перегляд аудіо в WAVE-формі
- Редагування сегментів аудіо
- Кодування та збереження у різних форматах

Для початку роботи користувач має завантажити аудіофайл. Система повинна підтримувати візуалізацію звукових хвиль у форматі WAVE та завантаження кількох доріжок.

Редагування звукових фрагментів охоплює операції копіювання (створення копії вибраного сегмента) вирізання (видалення сегмента з можливістю його подальшого вставлення) та вставки (вставлення скопійованих або вирізаних фрагментів на вибрані ділянки аудіо).

Оскільки користувач може зберегти результати у форматах ogg, flac або mp3, аудіо редактор повинен підтримувати конвертацію.

Схему описаних прецедентів, що відповідає обраній темі лабораторії, показано на Рисунку 1.

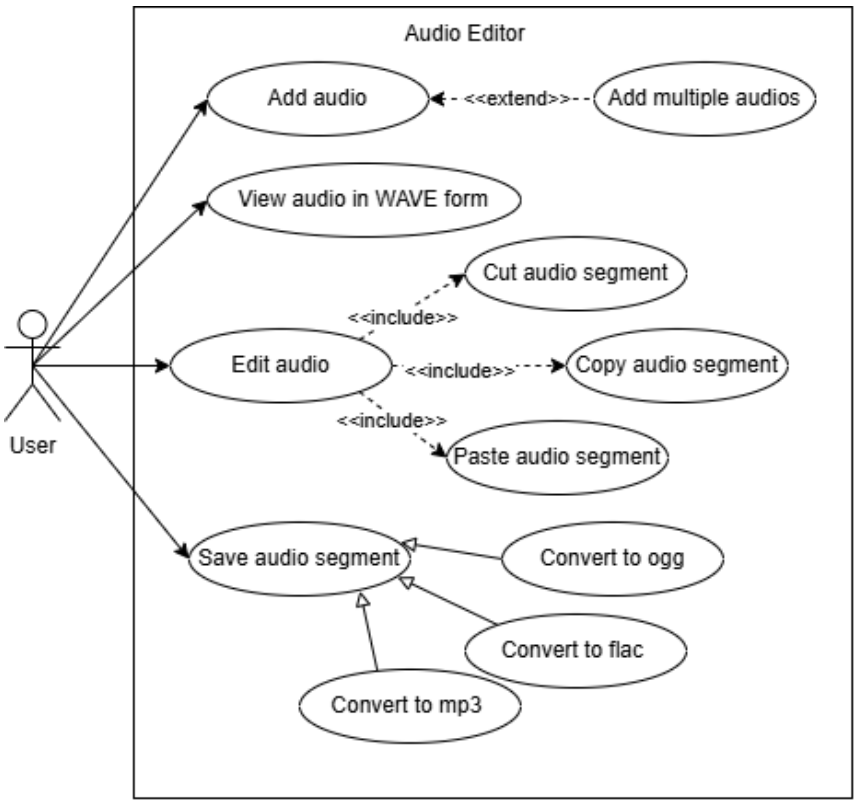

Рисунок 1 – Діаграма сценаріїв використання

3. Діаграма класів для реалізованої частини системи

Інтерфейс Audiotrack визначає базові методи для аудіотреків для отримання атрибутів, посилання на файл, створення копії тощо. Від цього інтерфейсу наслідуються класи Mp3, Ogg і Flac, які конкретизують обробку аудіофайлів у відповідних форматах. Класи AudioMp3Converter, AudioOggConverter і AudioFlacConverter реалізують інтерфейс Converter і забезпечують конвертацію аудіофайлів у відповідні формати.

Редагування аудіофайлів представлено інтерфейсом Editor, від якого успадковуються специфічні реалізації для різних операцій: AudioSelector, AudioCopier, AudioPaste та AudioCut.

Зберігання даних здійснюється через репозиторії AudioRepository, ProjectRepository та TrackRepository, які працюють із базою даних через DatabaseConnection. Репозиторії абстрагують роботу з таблицями: AudioRepository зберігає та отримує дані про аудіофайли; ProjectRepository оперує проектами, які можуть містити аудіофайли; TrackRepository працює з треками з аудіофайлів.

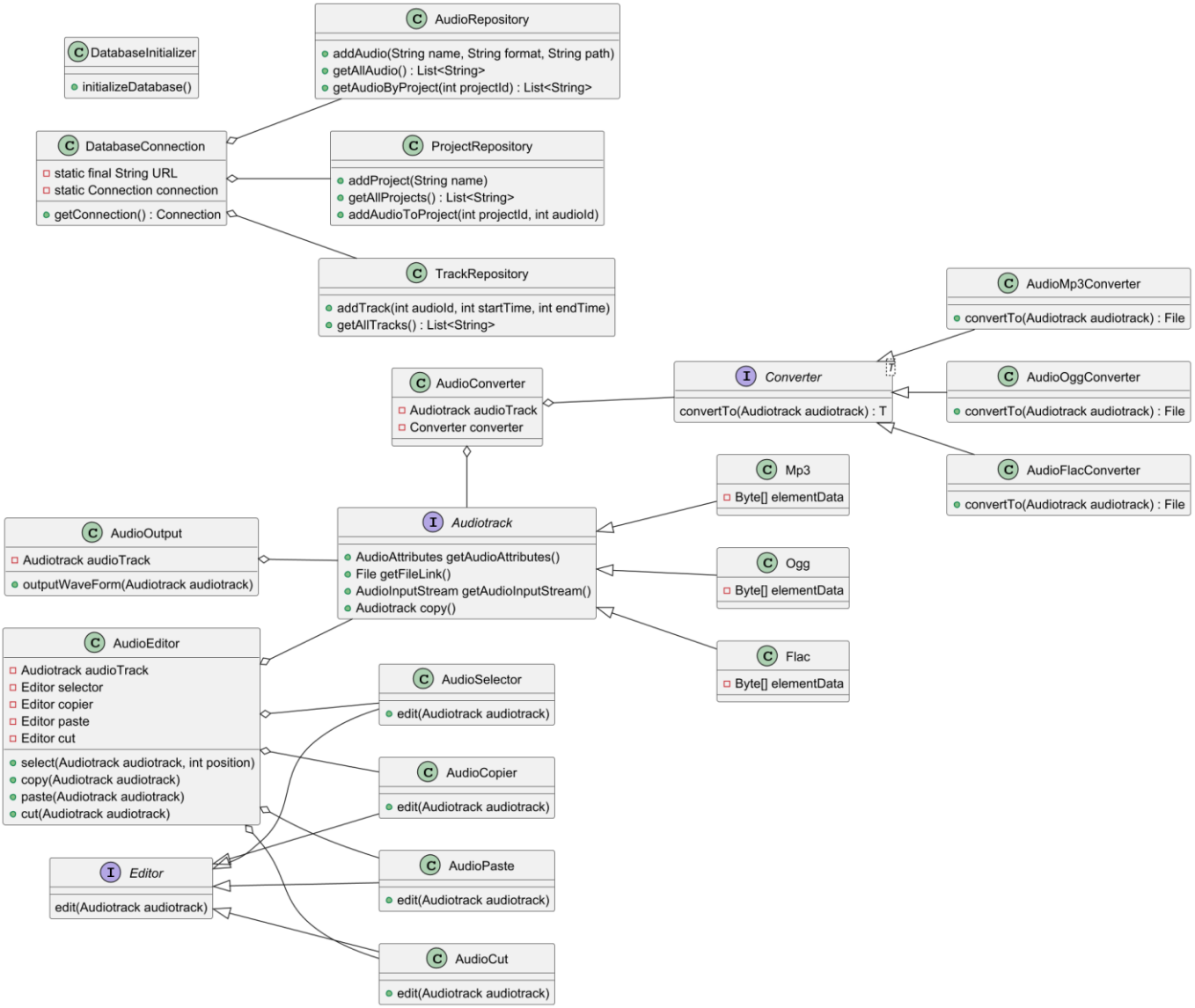

Рисунок 2 – Діаграма класів

4. Опис 3 обраних прецедентів

1) Користувач робить копію фрагменту обраного аудіо файлу.

Передумови. Файл завантажений у редактор.

Постумови. Користувач отримує копію фрагменту обраної звукової доріжки, яка доступна для подальших операцій.

Взаємодіючі сторони. Користувач, аудіо файл, аудіо редактор

Короткий опис. Даний варіант використання описує процес копіювання фрагменту звукової доріжки користувачем.

Основний потік подій. Даний варіант використання починає виконуватися, коли користувач обирає опцію зробити копію аудіо фрагменту.

1. Користувач обирає звукову доріжку.
2. Користувач обирає фрагмент звукової доріжки, який хоче копіювати.
3. Користувач натискає кнопку «копіювати».
4. Аудіо редактор виконує цю команду.
5. Користувач отримує копію обраного фрагменту від аудіо редактору.

Виключення.

1. Фрагмент не обрано — виводиться попередження.

Примітки. Відсутні.

2) Користувач переводить аудіо файл з одного формату в інший.

Передумови. Файл завантажений у редактор.

Постумови. Користувач з аудіо файлу деякого формату отримає аудіо файл іншого, обраного формату.

Взаємодіючі сторони. Користувач, аудіо файл, аудіо конвертор.

Короткий опис. Даний варіант використання описує процес перетворення аудіо файлу з одного формату в інший.

Основний потік подій. Даний варіант використання починає виконуватися, коли користувач обирає опцію перевести аудіо файл у інший формат.

1. Користувач обирає аудіо файл.
2. Користувач обирає бажаний формат.
3. Аудіо конвертор виконує команду, переводить файл у інший формат.
4. Користувач отримає новий, переведений в інший формат аудіо файл від аудіо конвертора.

Виключення.

1. Формат аудіо файлу не підтримується програмою.
2. Формат аудіо файлу вже має бажаний формат, отже не потребує переведення.

Примітки. Відсутні.

3) Користувач видаляє обраний фрагмент аудіо файлу.

Передумови. Файл завантажений у редактор.

Постумови. Користувач отримує аудіо файл без видаленого фрагменту.

Взаємодіючі сторони. Користувач, аудіо файл, аудіо редактор.

Короткий опис. Даний варіант використання описує видалення фрагменту аудіо файлу.

Основний потік дій. Даний варіант використання починає виконуватися, коли користувач обирає опцію видалення фрагменту аудіо файлу.

1. Користувач обирає фрагмент аудіо файлу, який хоче видалити.
2. Користувач натискає кнопку «видалити».
3. Аудіо редактор видаляє обраний користувачем фрагмент аудіо файлу.
4. Користувач отримує аудіо файл без видаленого фрагменту.

Виключення.

1. Фрагмент не обрано — виводиться попередження.

Примітки. Відсутні.

5. Основні класи, структура системи баз даних і шаблон Репозиторій

Основні компоненти містять класи для роботи з різними аудіоформатами (Mp3, Flac, Ogg), для редагування файлів (AudioCut, AudioCopier, AudioSelector) та сервіси для перетворення між різними форматами (AudioMp3Converter, AudioFlacConverter, AudioOggConverter). Система інтегрується з базою даних, де зберігаються метадані про аудіофайли, проєкти та треки. Також використовуютья репозиторії для доступу до даних (AudioRepository, TrackRepository, ProjectRepository) та має окремі класи для ініціалізації бази (DatabaseInitializer), підключення до неї (DatabaseConnection) та збереження змін.

Класи AudioRepository, ProjectRepository та TrackRepository містять методи для взаємодії з базою даних SQLite, зокрема додавання та отримання записів з таблиць Audio, Project, Track, а також допоміжної таблиці Project_Audio. Клас

DatabaseConnection відповідає за встановлення з'єднання з базою даних, тоді як DatabaseInitializer створює необхідні таблиці, якщо вони ще не існують.

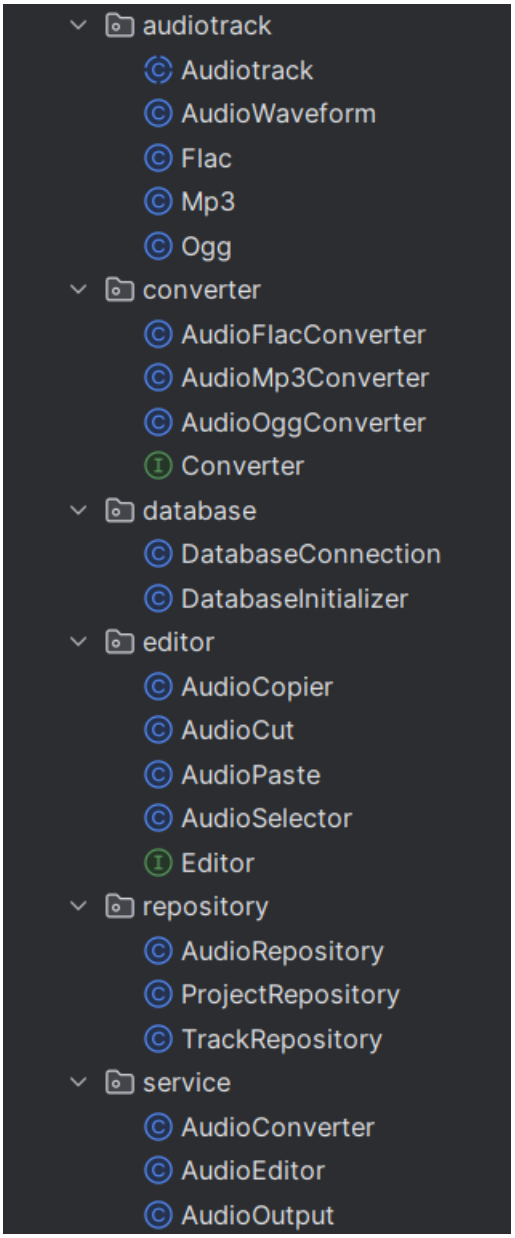

Рисунок 3 – Основні класи

Структура бази даних має наступні таблиці:

Audio — для збереження інформації про аудіофайли.

Track — для збереження інформації про частини аудіофайлів (наприклад, часткові записи чи треки в межах одного файлу). Audio та Track зв'язані один з одним через поле audio_id (зв’язок one-to-many). Це означає, що кожен аудіофайл може мати декілька треків.

Project — для збереження інформації про проєкти, що можуть містити аудіофайли та їх частини.

Project_Audio — містить зв'язок між проєктами та аудіофайлами. Кожен проект може містити кілька аудіофайлів, і кожен аудіофайл може бути частиною кількох проектів (зв’язок many-to-many)

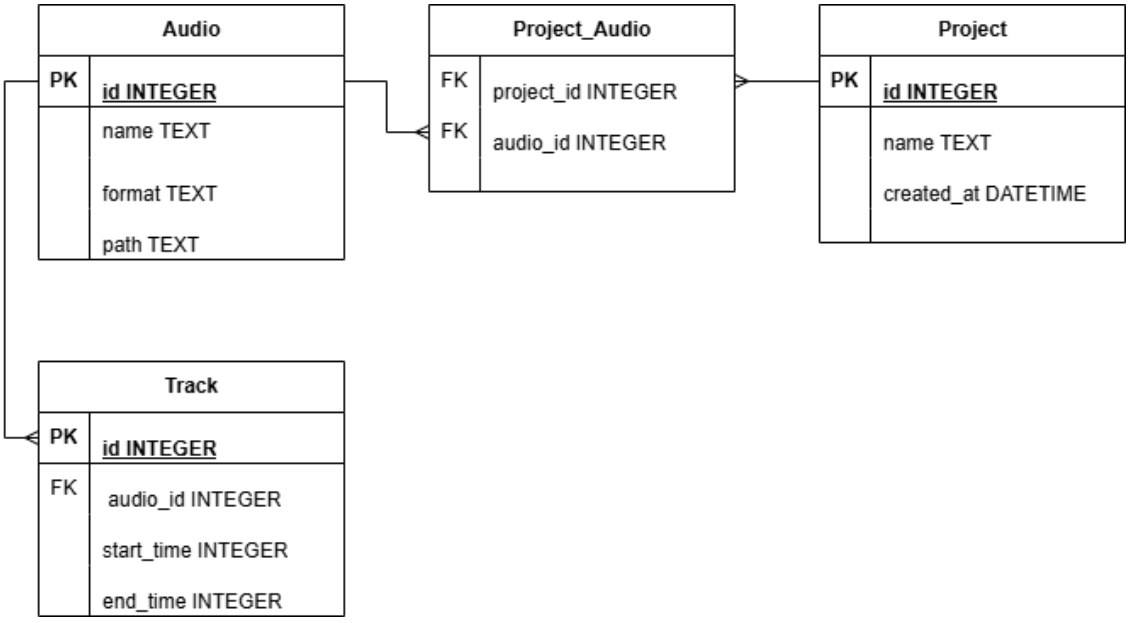

Рисунок 4 – Структура бази даних

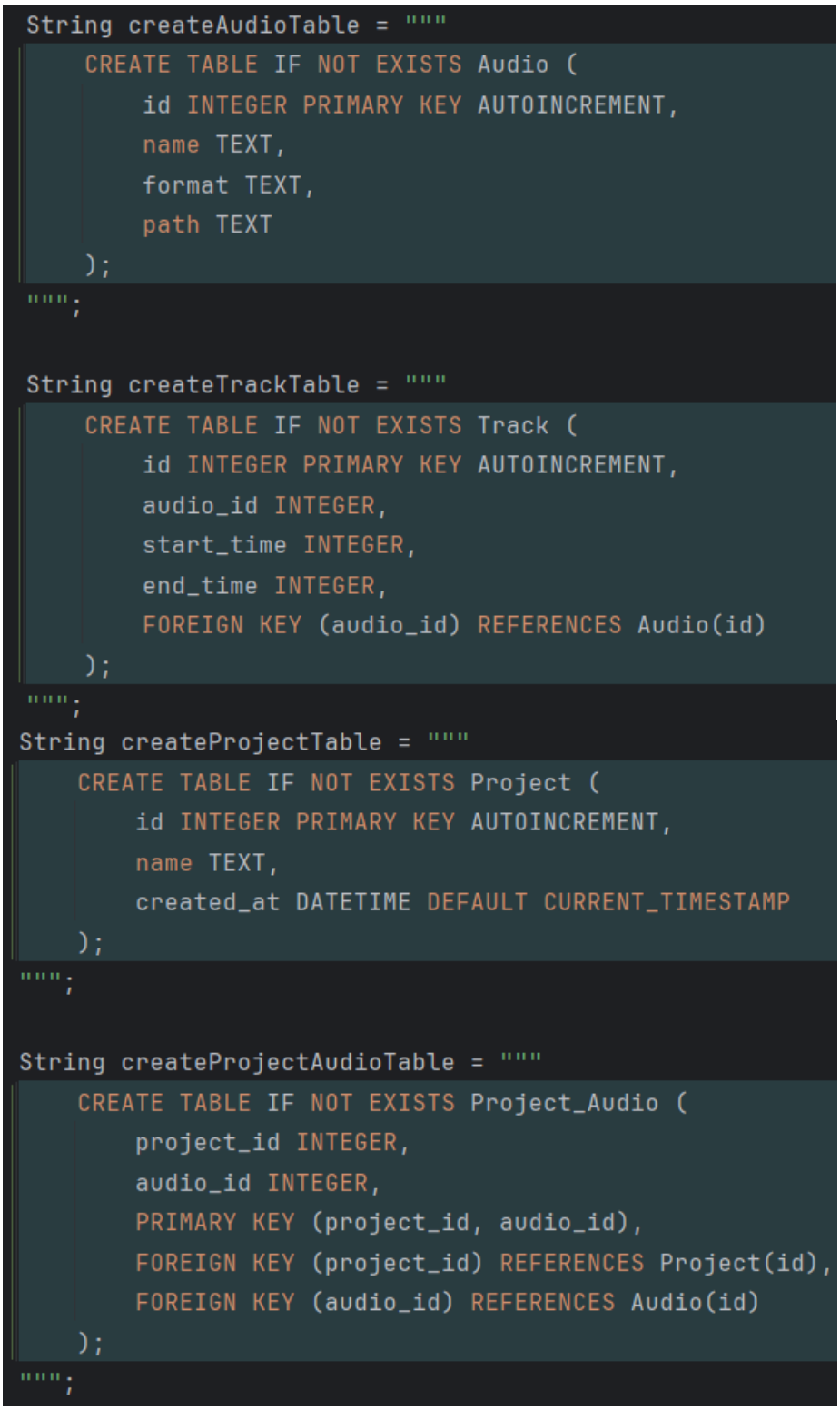

Рисунок 5 – Створення таблиць бази даних

Висновки: При виконанні цієї лабораторної роботи я закріпила навички роботи зі створення UML діаграм, а саме: діаграми сценаріїв використання та діаграми класів. На практичному прикладі розроблення застосунка, основуючись на

розроблених діаграмах, було створено відповідні класи та структура бази даних.

Посилання на репозиторій з кодом проєкту:

https://github.com/KatiaArkhyp/AudioEditor