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

Технології розроблення програмного забезпечення

Лабораторна робота №3

«ДІАГРАМА РОЗГОРТАННЯ. ДІАГРАМА КОМПОНЕНТІВ. ДІАГРАМА ВЗАЄМОДІЙ ТА ПОСЛІДОВНОСТЕЙ.»

Виконала:
Студентка групи ІА-22
Микитенко Ірина

Перевірив:
Мягкий Михайло Юрійович

Київ 2024

Зміст

Тема: Діаграма розгортання. Діаграма компонентів. Діаграма взаємодій та послідовностей.

Мета: Проаналізувати тему, розробити діаграму розгортання, діаграму компонентів, діаграму взаємодій та послідовностей.

Хід роботи

..1 Музичний програвач (iterator, command, memento, facade, visitor, client-server)

Музичний програвач становить собою програму для програвання музичних файлів або відтворення потокової музики з можливістю створення, запам'ятовування і редагування списків програвання, перемішування/повторення (shuffle/repeat), розпізнавання різних аудіо-форматів, еквалайзер.

1. Короткі теоритичні відомості

Діаграма розгортання

Діаграми розгортання (Deployment Diagram) відображають фізичне розташування компонент системи та показують, на якому обладнанні запускається програмне забезпечення.

Основними елементами є вузли, що можуть бути:

- *Пристроями* (фізичне обладнання, наприклад, комп'ютери)
- *Середовищами виконання* (програмне забезпечення, яке запускає інші компоненти, наприклад, операційна система).

Вузли можуть бути з'єднані шляхами, що показують взаємодію між ними, з використанням різних протоколів чи технологій. Усередині вузлів можуть бути артефакти — файли, які реалізують компоненти системи (виконувані файли, дані тощо).

Діаграми розгортання можуть бути:

- *Описовими*, які показують загальну структуру без деталей обладнання.
- *Екземплярними*, що включають конкретні приклади обладнання та програмних компонентів, і використовуються на пізніх етапах розробки для планування розгортання системи.

Діаграма компонентів

Діаграма компонентів UML відображає систему, розбиту на окремі модулі. Виділяють три типи діаграм:

1. Логічні
2. Фізичні
3. Виконувані

Найчастіше використовують *логічне* розбиття — система представлена як набір незалежних компонентів, що взаємодіють між собою. Наприклад, система продажу продуктів може мати такі компоненти: каса, сервер продаж, система обліку тощо. Компоненти можуть бути розміщені на різних фізичних пристроях або в різних процесах.

Фізичне розбиття фокусується на розподілі компонент між серверами та комп'ютерами, що частіше відображається на діаграмі розгортання.

Виконуване розбиття показує компоненти як файли (наприклад, .exe, html, бази даних), даючи інший ракурс системи. Однак цей підхід не має широкого використання через наявність діаграм розгортання.

Компоненти мають забезпечувати гнучкість для клієнтів, дозволяючи оновлювати систему частинами та забезпечувати сумісність між різними виробниками, хоча це важко реалізувати.

Діаграми послідовностей

Діаграми послідовностей в UML використовуються для моделювання виконання операцій і взаємодії між об'єктами в системі. Вони відображають послідовність повідомлень між об'єктами у процесі виконання операцій, деталізуючи алгоритми і логіку.

Основні елементи діаграми послідовностей:

- *Актори або об'єкти*, які взаємодіють між собою.
- *Повідомлення*, що передаються між ними.
- *Лінії життя*, які показують час існування об'єкта або актора під час взаємодії.

Діаграми послідовностей використовуються для візуалізації порядку виконання дій і детальної реалізації алгоритмів, відображаючи, як операції переходять із одного стану в інший.

2. Реалізація частини функціональності системи

Проект має чітку структуру, розділену на кілька основних пакетів:

1. Пакет контролерів:

- PlaybackSessionController: Відповідає за обробку запитів, пов'язаних із сесіями відтворення. Містить методи для створення, отримання, оновлення та видалення сесій відтворення.
- PlaylistController: Обробляє запити, пов'язані з плейлистами, надаючи можливість створювати, редагувати, видаляти та отримувати інформацію про плейлисти.
- SongController: Відповідає за обробку запитів, пов'язаних із піснями.

2. Пакет моделей:
   - Song: Модель, що представляє пісню, містить атрибути, такі як назва, виконавець, тривалість тощо.
   - Playlist: Модель, що описує плейлист, включаючи назву плейлиста та список пісень.
   - PlaybackSession: Модель, що зберігає інформацію про сесію відтворення.
3. Пакет репозиторіїв:
   - SongRepository: Інтерфейс для роботи з даними про пісні, забезпечує методи для зберігання, пошуку та видалення пісень.
   - PlaylistRepository: Інтерфейс для роботи з даними плейлистів.
   - PlaybackSessionRepository: Інтерфейс для управління сесіями відтворення.
4. Пакет сервісів:
   - SongService: Містить бізнес-логіку для обробки пісень.
   - PlaylistService: Включає бізнес-логіку для роботи з плейлистами.
   - PlaybackSessionService: Містить методи для управління сесіями відтворення.
5. Пакет імплементацій:
   - PlaybackSessionServiceImpl: Реалізація логіки сервісу для сесій відтворення.
   - SongServiceImpl: Реалізація логіки для роботи з піснями.
   - PlaylistServiceImpl: Реалізація логіки для управління плейлистами.

Нижче можна побачити структуру проєкту після реалізації частини функціональності системи (рисунок 1).

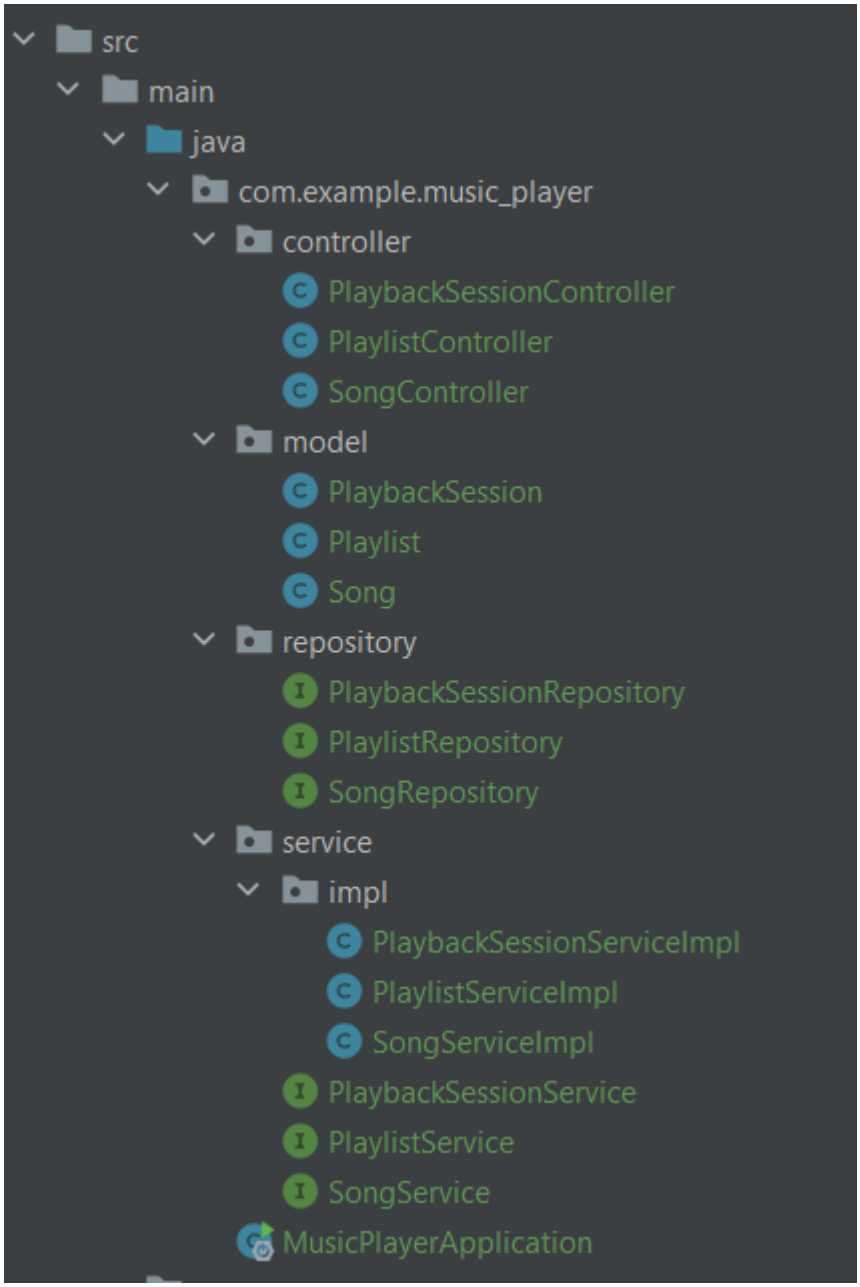

Рисунок 1

3. Діаграма послідовностей

Діаграма посідовностей для процесу «Завантаження та відтворення пісні» представлена на рисунку 2

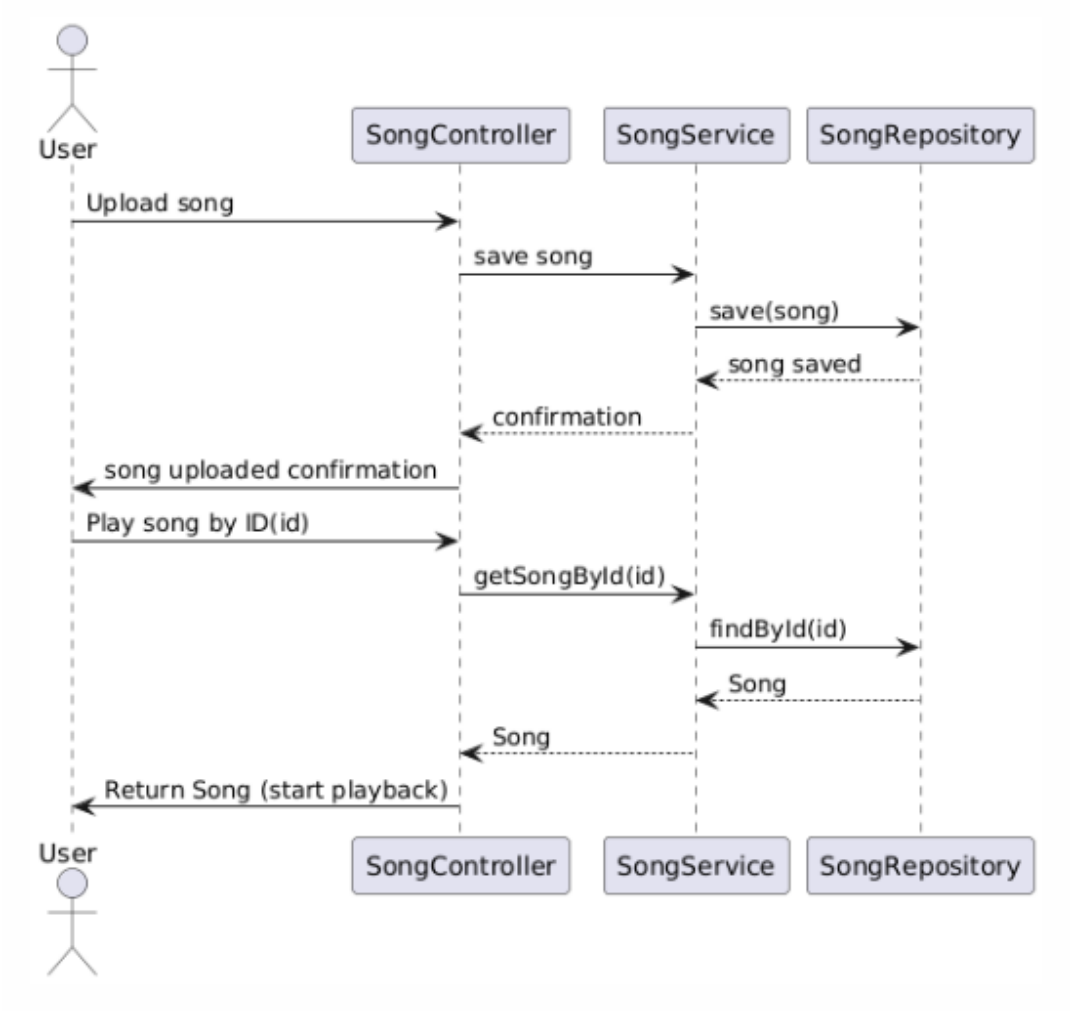

Рисунок 2

Сценарій завантаження пісні

1. Користувач (User) надсилає запит на завантаження пісні до SongController (SC).
2. SongController (SC) передає запит до SongService (SS) для збереження пісні.
3. SongService (SS) викликає SongRepository (SR) для збереження пісні в базі даних.
4. SongRepository (SR) підтверджує успішне збереження пісні.
5. SongService (SS) інформує SongController (SC) про успішне збереження.
6. SongController (SC) повертає підтвердження користувачу.

Сценарій відтворення пісні

1. Користувач (User) запитує відтворення пісні за її ID.
2. SongController (SC) передає запит до SongService (SS) для отримання пісні.

3. SongService (SS) запитує SongRepository (SR) про пошук пісні за ID.
4. SongRepository (SR) повертає знайдену пісню.
5. SongService (SS) передає пісню назад до SongController (SC).
6. SongController (SC) повертає пісню користувачу для початку відтворення.

4. Діаграма розгортання

Діаграма розгортання системи зображена на рисунку 3.

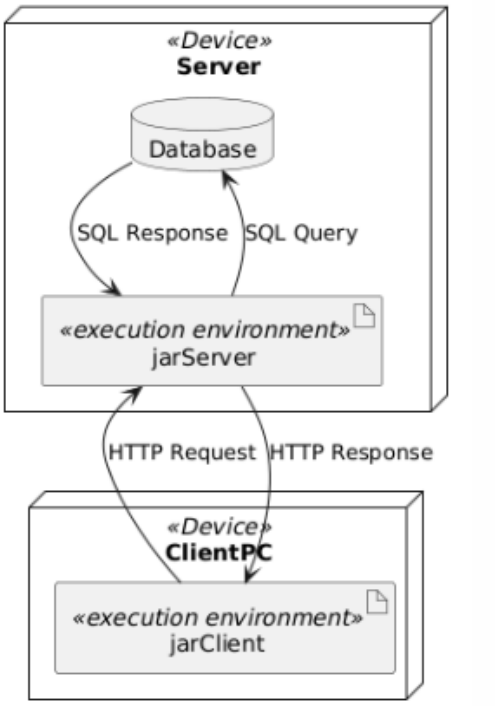

Рисунок 3

Діаграма розгортання ілюструє архітектуру музичного програвача, яка складається з клієнтської та серверної частин. Клієнт представлений як пристрій (ClientPC), що містить артефакт jarClient, що позначає виконуваний файл клієнтської частини програми. Сервер також представлений як пристрій і містить артефакт jarServer, який є виконуваним файлом серверної частини програми, а також базу даних, позначену як Database (DB). Взаємодії між компонентами включають надсилання HTTP-запитів від клієнта до сервера, отримання HTTP-відповідей від сервера до клієнта, виконання SQL-запитів сервером до бази даних та повернення результатів виконання запитів від бази даних до сервера.

Ця діаграма показує, як клієнтська частина програми взаємодіє з сервером і базою даних, відображаючи архітектурні елементи та їх зв'язки в системі.

5. Діаграма компонентів

Діаграма компонентів розроблюваної системи зображена на рисунку 4.

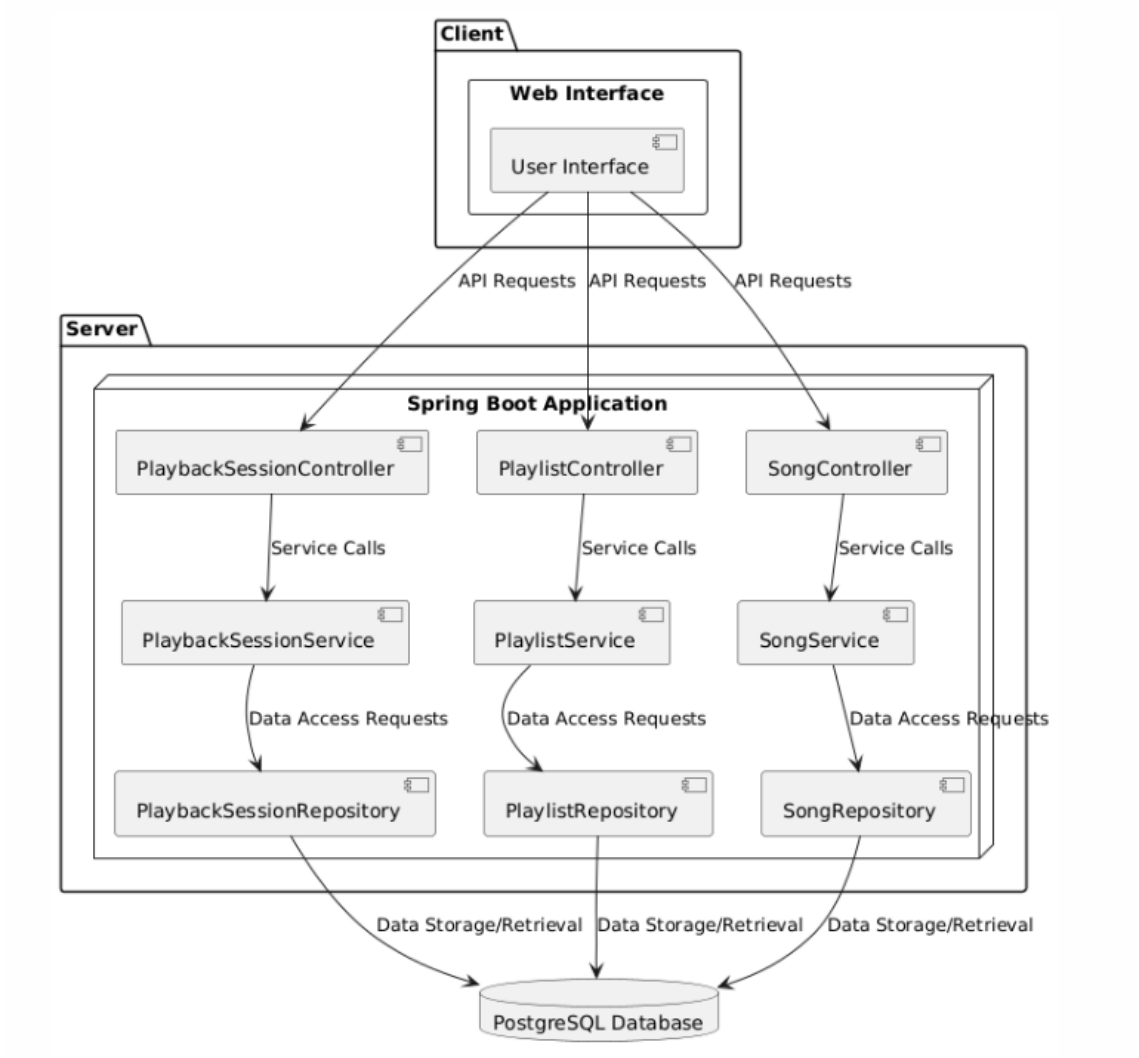

Рисунок 4

Діаграма компонентів для музичного програвача містить три основні частини: сервер, базу даних і клієнт. На сервері розташована програма Spring Boot, що включає контролери (PlaybackSessionController, PlaylistController, SongController), сервіси (PlaybackSessionService, PlaylistService, SongService) та репозиторії (PlaybackSessionRepository, PlaylistRepository, SongRepository). База даних PostgreSQL зберігає дані для всіх компонентів. Клієнтський веб інтерфейс взаємодіє з контролерами, надсилаючи запити на API. Контролери обробляють ці запити, викликаючи відповідні сервіси, які в свою чергу взаємодіють із репозиторіями для збереження або отримання даних з бази даних. Ця структура демонструє, як різні компоненти системи працюють разом для виконання запитів користувача.

Лінк репозиторію на гітхабі: https://github.com/imykytenko/trpz_music_player.git

Висновок: в даній лабораторній роботі я ознайомилась з теоритичними відомостями, проаналізувала тему, розробила діаграму розгортання, діаграму компонентів, діаграму взаємодій та послідовностей.