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
**Технології розроблення програмного забезпечення**
«Основи проектування.»

Виконав:

Студент групи ІА-32

Костінський Макарій

Перевірив:

Мягкий Михайло Юрійович

# Зміст



**Тема:** Основи проектування.

**Мета:** Обрати зручну систему побудови UML-діаграм та навчитися будувати діаграми варіантів використання для системи що проєктується, розробляти сценарії варіантів використання та будувати діаграми класів предметної області.

## Теоретичні відомості

Мова UML є загальноцільовою мовою візуального моделювання, яка розроблена для специфікації, візуалізації, проєктування та документування компонентів програмного забезпечення, бізнес-процесів та інших систем. Мова UML є досить строгим та потужним засобом моделювання, який може бути ефективно використаний для побудови концептуальних, логічних та графічних моделей складних систем різного цільового призначення. Ця мова увібрала в себе найкращі якості та досвід методів програмної інженерії, які з успіхом використовувалися протягом останніх років при моделюванні великих та складних систем.

**Діаграма (diagram)** – графічне уявлення сукупності елементів моделі у формі зв'язкового графа, вершинам і ребрам (дугам) якого приписується певна семантика. Нотація канонічних діаграм є основним засобом розробки моделей мовою UML.

**Діаграма варіантів використання (Use-Cases Diagram)** – це UML діаграма за допомогою якої у графічному вигляді можна зобразити вимоги до системи, що розробляється. Діаграма варіантів використання – це вихідна концептуальна модель проєктованої системи, вона не описує внутрішню побудову системи.

**Сценарії використання** – це текстові уявлення тих процесів, які відбуваються при взаємодії користувачів системи та самої системи. Вони є чітко формалізованими, покроковими інструкціями, що описують той чи інший процес у термінах кроків досягнення мети. Сценарії використання однозначно визначають кінцевий результат.

**Діаграми класів** використовуються при моделюванні програмних систем найчастіше. Вони є однією із форм статичного опису системи з погляду її проєктування, показуючи її структуру. Діаграма класів не відображає динамічної поведінки об'єктів зображених на ній класів. На діаграмах класів показуються класи, інтерфейси та відносини між ними.

**Логічна модель бази даних** є структурою таблиць, уявлень, індексів та інших логічних елементів бази даних, що дозволяють власне програмування та використання бази даних.

## Хід роботи

### Діаграма варіантів використання (Use-Cases Diagram)

На Рис.1 зображено діаграму варіантів використання, яка моделює взаємодію основного актора «Користувача» із системою онлайн-радіостанції. Ця діаграма визначає ключові функціональні вимоги до системи з точки зору слухача. Користувач представляє будь-якого слухача, що взаємодіє з радіостанцією.

Користувач може виконувати такі основні дії:

- Підключитися до трансляції: Ініціює прослуховування.
- Відключитися від трансляції: Завершує сеанс прослуховування.
- Прослуховувати музику: Основний процес взаємодії з системою.

Відношення <<extend>>: Варіант використання «Прослуховувати музику» є базовим і може бути розширений двома додатковими, необов'язковими діями:

- Переглянути інформацію про трек: Користувач за бажанням може переглянути деталі поточної композиції.
- Поставити лайк: Користувач може висловити свою перевагу щодо треку.


Рис.1 - Use-Cases Diagram для користувача

На даному фрагменті діаграми показано відношення узагальнення (Generalization) між «Адміністратор» та «Користувач».

Це означає, що актор «Адміністратор» є нащадком актора «Користувач» і успадковує всі його атрибути та варіанти використання. Таким чином, адміністратор може виконувати всі дії, доступні звичайному користувачеві (прослуховувати музику, ставити лайки тощо), а також має власний набір розширених можливостей для керування системою.

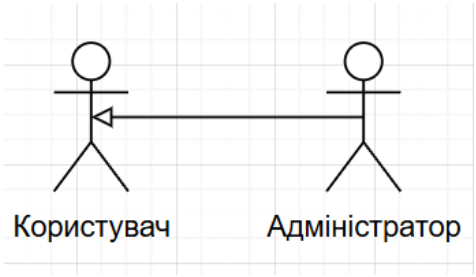



Рис. 2 – Успадкування ролі Адміністратора

На Рис.3 специфічні варіанти використання, доступні для «Адміністратор». Ці функції призначені для керування та налаштування роботи радіостанції.

Варіанти використання:

- Вхід в систему: Процес автентифікації адміністратора для доступу до панелі керування.
- Керування плейлистами: Узагальнена функція, що включає створення, редагування та видалення списків відтворення.
- Керування бібліотекою треків: Дозволяє додавати та видаляти музичні файли.
- Налаштувати параметри трансляції: Керування технічними параметрами ефіру, наприклад, бітрейтом.

- Перегляд статистики: Дозволяє адміністратору аналізувати дані про прослуховування.

Відношення <<include>>: частиною процесу входу є «Перевірка облікових даних», що реалізовано через відношення <<include>>.

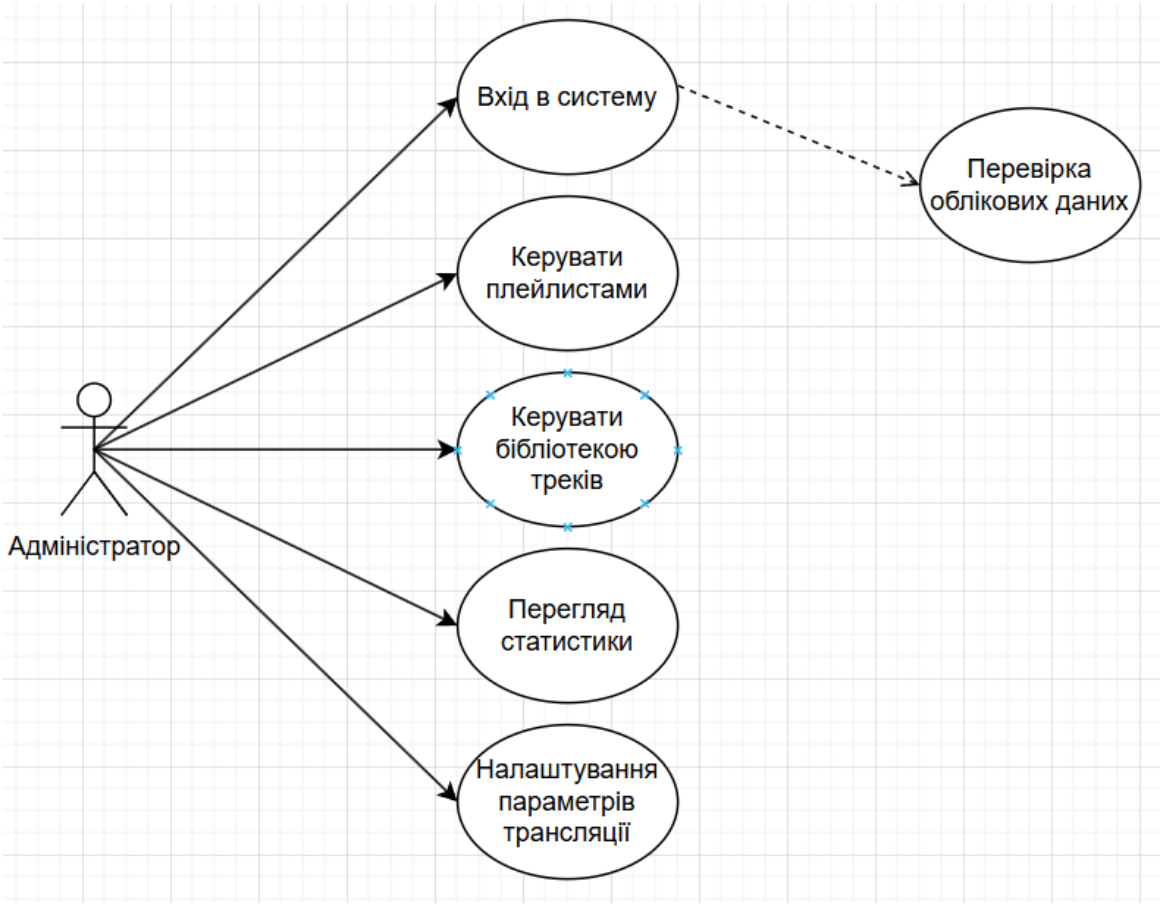



Рис. 3 – Діаграма варіантів використання для Адміністратора

**Три варіанти використання**

**Сценарій 1:**
Назва: Вхід в систему.

Передумови: Адміністратор має існувати в системі з коректним логіном та паролем.

Постумови: У разі успішного входу, система надає адміністратору доступ до функцій керування. В іншому випадку, стан системи не змінюється.

Взаємодіючі сторони: Адміністратор, Система.

Короткий опис: Адміністратор вводить свої облікові дані для отримання доступу до панелі керування радіостанцією.

Основний потік подій:

- Адміністратор відкриває сторінку входу.
- Система відображає форму для введення логіна та пароля.
- Адміністратор вводить свій логін та пароль і натискає кнопку "Увійти".
- Система перевіряє, чи існує користувач з таким логіном, та порівнює хеш введеного пароля зі збереженим.
- Якщо дані коректні, система створює авторизовану сесію та перенаправляє адміністратора на головну панель керування.

Винятки:

Виняток №1: Неправильний логін або пароль. Якщо на кроці 4 система визначає, що дані невірні, вона виводить повідомлення про помилку: "Неправильний логін або пароль". Адміністратор залишається на сторінці входу і може повторити спробу.

**Сценарій 2:**
Назва: Прослуховувати музику.

Передумови: Користувач має доступ до інтернету та відкрив сторінку радіостанції. Радіостанція веде мовлення.

Постумови: Користувач прослуховує музичний потік.

Взаємодіючі сторони: Користувач, Система.

Короткий опис: Користувач підключається до онлайн-трансляції радіостанції, слухає музику та може взаємодіяти з поточним треком.

Основний потік подій:

- Користувач заходить на головну сторінку радіостанції.
- Система відтворює музичний потік.
- Система відображає інформацію про поточний трек.
- Система створює запис PlaybackEvent, фіксуючи UserId, TrackId та StartTime.
- Користувач закриває сторінку або натискає кнопку "Стоп".
- Система фіксує EndTime у відповідному записі PlaybackEvent для завершення сесії.

Винятки:

Виняток №1: Трансляція недоступна. Якщо на кроці 2 радіостанція не веде мовлення, прослуховування не доступне.

**Сценарій 3:**
Назва: Керування плейлистами.

Передумови: Адміністратор успішно увійшов до системи та перейшов на сторінку керування плейлистами.

Постумови: Список плейлистів та їхній вміст оновлено відповідно до дій адміністратора.

Взаємодіючі сторони: Адміністратор, Система.

Короткий опис: Адміністратор створює, редагує та видаляє плейлисти, а також керує списком треків у них.

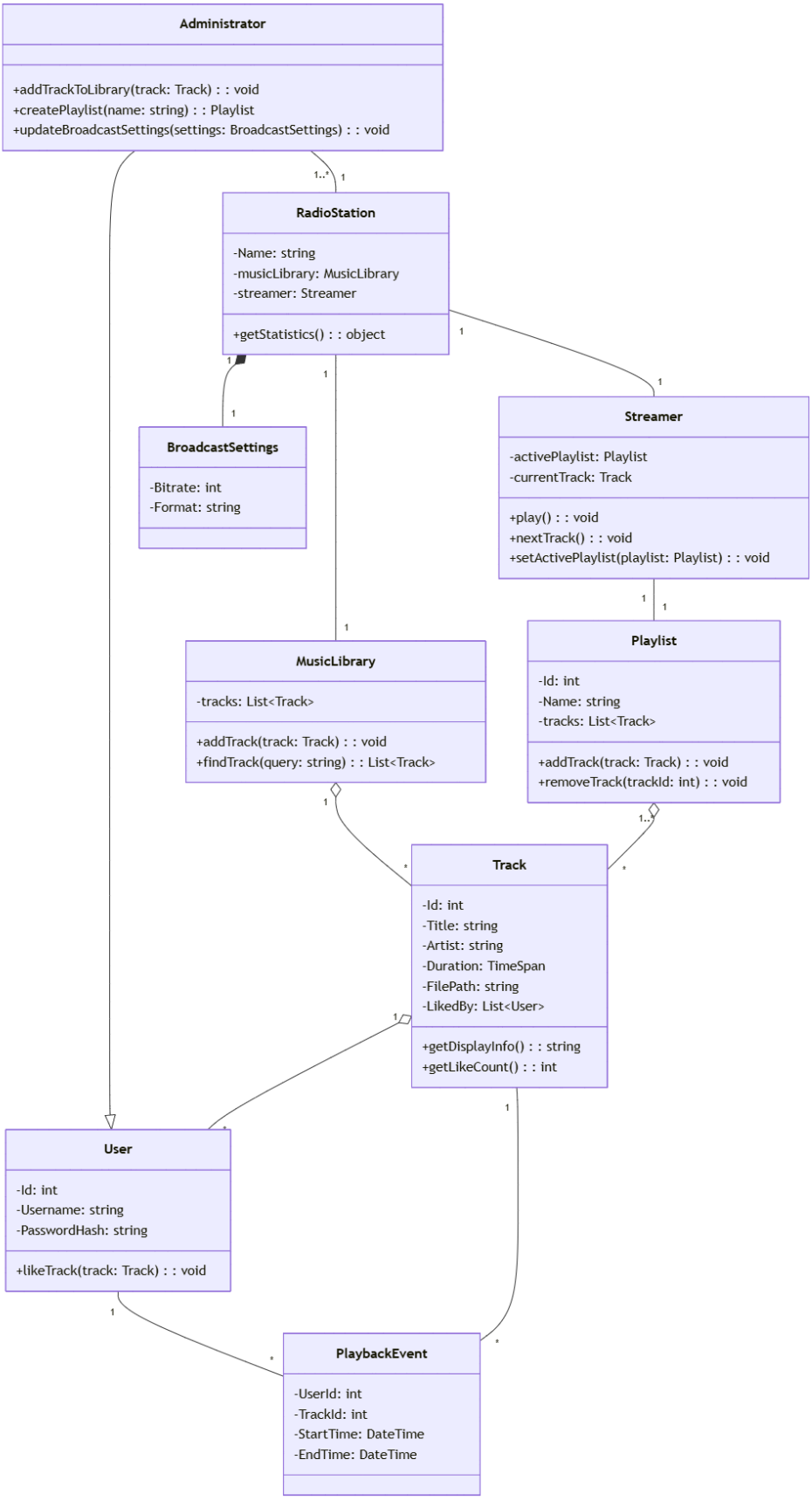
Основний потік подій:

- Система відображає список існуючих плейлистів.
- Адміністратор створює новий плейлист, або вибирає наявний для редагування.
- Система відображає вміст плейлиста та повний список доступних треків з бібліотеки.
- Адміністратор обирає треки з бібліотеки для додавання або вибирає трек з плейлиста для видалення.
- Система зберігає відповідні зміни.

Винятки:

Виняток №1: Неунікальна назва плейлиста. Якщо при створенні, плейлист з такою назвою вже існує, система виводить повідомлення про помилку: "Плейлист з такою назвою вже існує".

# Діаграма класів



User та Administrator: Центральною сутністю системи є User, що представляє звичайного слухача. Клас Administrator є його нащадком (відношення узагальнення), що успадковує всі властивості користувача та отримує розширені права для керування системою, як-от додавання треків та керування плейлистами.

Track: Клас Track моделює музичну композицію — основну одиницю контенту. Він містить таку інформацію, як назва, виконавець, шлях до файлу, а також список користувачів, які вподобали трек.

Playlist: Ця сутність представляє список відтворення. Вона агрегує об'єкти класу Track, що дозволяє одному треку входити до складу кількох плейлистів або існувати в бібліотеці незалежно.

RadioStation: Цей клас виступає як центральний фасад системи. Він об'єднує та координує роботу інших компонентів, таких як MusicLibrary (відповідає за логіку управління музичною бібліотекою) та Streamer (керує процесом трансляції).

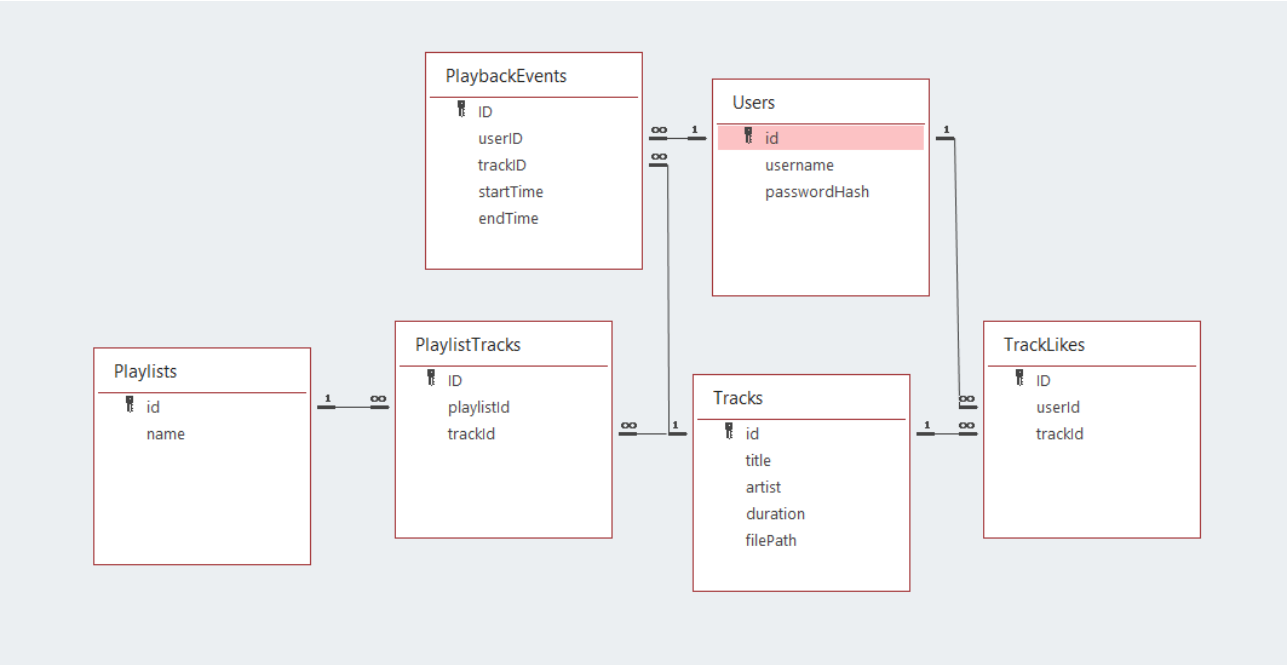
BroadcastSettings: RadioStation містить об'єкт BroadcastSettings (відношення композиції), який інкапсулює технічні параметри ефіру (бітрейт, формат).

PlaybackEvent: Для реалізації функціоналу збору статистики прослуховувань використовується клас PlaybackEvent. Він фіксує асоціативний зв'язок між User та Track у певний момент часу.

**Вихідний код**

https://github.com/Makarii-IA-32/TRPZ-Lab2-OnlineRadio

**Структура БД**




**Висновок:**

У ході роботи було виконано:

Проведено аналіз вимог, результатом якого стала розробка діаграми варіантів використання. Ця діаграма чітко визначила функціональні можливості системи для ключових акторів — Користувача та Адміністратора.

Створено сценарії використання для трьох ключових процесів: "Вхід в систему", "Прослуховування музики" та "Керування плейлистами". Це дозволило деталізувати логіку взаємодії користувачів із системою.

Спроєктовано діаграму класів предметної області, яка визначила основні сутності, їхні атрибути та зв'язки.

Розроблено програмний код для всіх моделей та реалізовано шаблон "Repository" для відділення логіки доступу до даних.

Створено структуру бази даних у Microsoft Access, яка відображає спроєктовані класи у вигляді реляційних таблиць та зв'язків.