# Music Library Management System

## Creating the Models

1. Define the `Artist` model with the following fields:
    - Name: The name of the artist. (Make it unique to prevent duplicate artist names)
    - Genre: The genre of music associated with the artist.
    - Biography: A text field to provide a brief biography of the artist.
2. Define the `Album` model with the following fields:
    - Title: The title of the album. (Make it unique to ensure each album has a unique title)
    - Artist: A foreign key to the artist who created the album.
    - Release Date: The date the album was released.
    - Cover Image: An image field to store the album's cover image.
    - Description: A text field to provide a description of the album.
3. Define the `Song` model with the following fields:
    - Title: The title of the song.
    - Album: A foreign key to the album the song belongs to.
    - Duration: The duration of the song in minutes.
    - Track Number: The track number of the song within the album. (Ensure each song within an album has a unique track number using `unique_together` option)
    - Lyrics: A text field to store the lyrics of the song.
4. Establish relationships between models:
    - Implement a one-to-many relationship between the `Artist` and `Album` models. An artist can have multiple albums, but each album belongs to only one artist.
    - Implement a many-to-many relationship between the `Album` and `Song` models. An album can have multiple songs, and a song can belong to multiple albums.

# Music Library Management System

## Creating Sample Data

1. Use Django's built-in ORM to create sample artists, albums, and songs.
2. Populate the database with various artists, albums, and songs to provide a diverse collection for users to explore.

# Music Library Management System

## Implementing Views

1. Create views to display a list of all artists, including their names and genres.
2. Implement views to filter artists by genre, allowing users to browse artists within specific genres.
3. Create views to display the details of a selected artist, including their name, genre, and biography.
4. Implement views to display a list of albums for a selected artist, including the album titles and release dates.
5. Create views to display the details of a selected album, including the album title, artist, release date, cover image, and description.
6. Implement views to display a list of songs for a selected album, including the song titles, durations, and track numbers.
7. Create views to display the details of a selected song, including the song title, album, duration, track number, and lyrics.

# Music Library Management System

## Handling Forms and ORM

1. Implement forms to allow users to add new artists, albums, and songs to the music library.
    - The artist form should include fields for the artist's name, genre, and biography.
    - The album form should include fields for the album's title, artist, release date, cover image, and description.
    - The song form should include fields for the song's title, duration, track number, and lyrics.
    - Perform form validation to ensure all required fields are filled out.
    - Save the submitted data to the database using the respective models.

# Music Library Management System

## Implementing Pagination (Optional)

1. Implement pagination to display a limited number of artists, albums, or songs per page for better usability.
2. Create pagination views to handle the display of paginated data.