

## **Implementasi Struktur Data dalam Pengembangan Program Music Player**



Kelompok 7

Disusun oleh:

- |                              |                |
|------------------------------|----------------|
| 1. Tsaqillah Ali             | (103102400015) |
| 2. Clairine Anargya Athallah | (103102400031) |
| 3. Izam Rosiawan             | (103102400049) |

**PROGRAM STUDI SAINS DATA  
FAKULTAS INFORMATIKA  
UNIVERSITAS TELKOM SURABAYA  
2025**

# BAB I

## PENDAHULUAN

### 1.1 Latar Belakang

Perkembangan layanan pemutar musik digital seperti Spotify, Apple Music dan lain sebagainya menunjukkan bahwa pengelolaan data lagu dalam jumlah besar perlu dilakukan secara efisien. Layanan tersebut tidak hanya dapat menampilkan daftar lagu, namun juga mengelompokkan lagu berdasarkan artis, album, genre, serta menyediakan fitur pencarian, playlist, dan rekomendasi yang responsif. Selain itu, data musik umumnya tersusun dalam bentuk bertingkat, mulai dari artis, album, hingga lagu. Struktur seperti ini membutuhkan metode pengelolaan yang dapat menangani hubungan antar data secara teratur. Oleh karena itu, diperlukan pemahaman yang kuat mengenai struktur data yang tepat, karena data musik juga terdiri dari berbagai elemen yang saling terhubung dan dapat berubah sewaktu-waktu.

Pada mata kuliah Struktur Data, berbagai struktur data telah dipelajari, seperti Single Linked List, Double Linked List, Stack, Queue, Tree, Graph, dan Multi Linked List. Struktur data tersebut tidak hanya digunakan untuk menyimpan data, namun juga untuk mengatur alur pemrosesan dan interaksi antar data. Oleh karena itu, sebagai bentuk penerapan materi yang telah dipelajari, dirancanglah sebuah aplikasi pemutar musik berbasis konsol yang mengelola data artis, album, dan lagu dengan memanfaatkan struktur data yang sesuai.

Aplikasi pemutar musik “CIC” dirancang untuk mensimulasikan proses penyimpanan dan pengelolaan data musik secara bertingkat menggunakan beberapa konsep. Selain itu, aplikasi ini menyediakan fitur playlist, pencarian lagu, pemutaran musik, serta navigasi next dan previous. Melalui pengembangan aplikasi tersebut, konsep struktur data dapat diterapkan secara langsung pada sebuah studi kasus yang menyerupai kondisi nyata.

### 1.2 Analisis Masalah

Data musik dalam aplikasi disusun menjadi artis dan lagu, sedangkan struktur album disediakan dalam kode tetapi belum digunakan penuh dalam GUI. Kondisi ini memerlukan struktur data yang mampu menangani hubungan berlapis tersebut secara konsisten. Tantangan utama meliputi:

#### 1. Penyimpanan artis dan lagu dalam struktur berantai.

Setiap artis memiliki daftar lagu yang disimpan menggunakan Single Linked List. Artinya, proses pencarian, pengeditan, dan penghapusan lagu harus dilakukan dengan menelusuri rantai node secara berurutan. Ketika jumlah lagu bertambah, efisiensi traversal menjadi penting.

#### 2. Keterbatasan navigasi berdasarkan struktur.

Lagu dalam library disimpan sebagai Single Linked List, sehingga navigasi ke elemen sebelumnya tidak dapat dilakukan secara langsung. Untuk mengatasi hal ini, hasil traversal linked list diubah menjadi list Python agar navigasi lagu dapat dilakukan menggunakan indeks. Hal ini berpengaruh pada fitur “previous”, terutama saat pemutaran lagu dilakukan berdasarkan urutan library.

#### 3. Manajemen playlist yang fleksibel.

Playlist memiliki kebutuhan navigasi dua arah, sehingga diperlukan struktur data yang mendukung pergerakan maju dan mundur. Implementasi yang dipakai adalah Doubly Linked List, yang memungkinkan pengguna berpindah ke lagu berikutnya atau sebelumnya tanpa traversal ulang.

#### **4. Riwayat pemutaran dan antrian pemutaran.**

Sistem membutuhkan mekanisme untuk menyimpan urutan lagu yang telah diputar (history). Riwayat pemutaran cocok disimpan menggunakan struktur data Stack karena memiliki sifat Last In First Out (LIFO). Setiap kali lagu baru diputar, sistem secara otomatis menyimpan lagu sebelumnya ke dalam Stack. Struktur ini kemudian dimanfaatkan untuk mendukung navigasi *previous* pada pemutaran lagu.

### **1.3 Perencanaan Solusi**

Solusi perancangan aplikasi dibangun menggunakan kombinasi beberapa struktur data:

#### **1. Single Linked List untuk artis dan lagu.**

ArtistNode saling terhubung melalui pointer next. Setiap artis memiliki daftar lagu yang juga berbentuk Single Linked List. Struktur ini mendukung penambahan lagu secara dinamis dan traversal berurutan.

#### **2. Album disimpan dalam Single Linked List**

Berbeda dengan artis dan lagu, daftar lagu dalam AlbumNode tidak menggunakan linked list, melainkan list Python untuk penyimpanan sederhana.

#### **3. Playlist menggunakan Doubly Linked List.**

Penggunaan Doubly Linked List pada playlist memungkinkan navigasi dua arah (next dan previous) secara langsung melalui pointer prev dan next tanpa traversal ulang. Playlist membutuhkan navigasi dua arah, oleh karena itu struktur PlaylistNode memiliki pointer prev dan next. Playlist menyimpan referensi ke node lagu asli dalam library, bukan salinan objek.

#### **4. Riwayat pemutaran menggunakan Stack.**

Stack menyimpan lagu-lagu yang telah diputar, sehingga pengguna dapat kembali ke lagu sebelumnya dengan operasi pop().

#### **5. Antrian pemutaran menggunakan konsep Queue.**

Struktur Queue disediakan pada sisi backend untuk menampung lagu yang akan diputar sebagai rancangan pengembangan lanjutan. Namun pada versi aplikasi saat ini, Queue belum digunakan dalam proses pemutaran lagu maupun pengelolaan playlist pada GUI.

#### **6. GUI menggunakan CustomTkinter dan pygame.**

Antarmuka mencakup tampilan daftar lagu, playlist, kontrol pemutar, fitur pencarian, serta panel admin untuk mengelola lagu.

## **BAB II IMPLEMENTASI**

### **2.1 Inisialisasi Data (Data Dummy)**

Untuk keperluan pengujian dan simulasi, sistem CIC Music Player memuat data awal menggunakan modul `data_dummy.py`. Modul ini berfungsi untuk menginisialisasi struktur data dengan sejumlah data artis, lagu, dan album secara otomatis saat aplikasi dijalankan. Data dummy tersebut dimasukkan ke dalam struktur Single Linked List artis-lagu serta struktur album sebagai metadata pendukung. Dengan adanya data awal ini, seluruh fitur aplikasi seperti penelusuran library, pengelolaan playlist, pemutaran lagu, serta pengujian antarmuka grafis dapat dilakukan tanpa memerlukan input manual dari pengguna.

### **2.2 Struktur Data dan Implementasi**

#### **2.2.1 Implementasi Struktur Single Linked List**

Pada sistem CIC Music Player, struktur Single Linked List digunakan untuk menyimpan data musik dalam dua tingkatan utama, yaitu daftar artis dan daftar lagu pada setiap artis. Linked list dipilih karena mendukung penambahan dan penghapusan node secara dinamis tanpa perlu menggeser elemen lain seperti pada array.

##### **ArtistNode - Linked List Artis**

Setiap artis dalam sistem direpresentasikan sebagai sebuah node (`ArtistNode`) yang berisi:

- Nama artis
- Pointer *next* menuju artis berikutnya
- Pointer *songs\_head* yang menunjuk ke node pertama pada daftar lagu artis tersebut

Dengan struktur ini, penelusuran artis dapat dilakukan secara berantai dari artis pertama hingga artis terakhir, serta setiap artis memiliki linked list lagu masing-masing.

##### **SongNode - Linked List Lagu pada Setiap Artis**

- ID lagu
- Judul lagu
- Durasi lagu
- Genre
- Lokasi file audio
- Pointer *next* menuju lagu berikutnya

Pengelolaan lagu seperti penambahan, pencarian berdasarkan ID, pembaruan data, dan penghapusan dilakukan dengan menelusuri linked list lagu dari head hingga node yang dicari.

##### **AlbumNode - Single Linked List untuk Album (Tidak Terhubung Langsung ke Lagu)**

Struktur album pada aplikasi disimpan menggunakan Single Linked List pada level album (`AlbumNode`). Namun, daftar lagu yang terdapat di dalam album tidak disimpan menggunakan linked list, melainkan menggunakan list Python biasa. Pada versi aplikasi saat ini, struktur album berfungsi sebagai metadata pendukung dan belum terintegrasi secara langsung dengan antarmuka grafis (GUI). Berbeda dengan struktur lagu pada artis yang menggunakan Single Linked List, daftar lagu

pada album tidak disimpan dalam bentuk linked list. Setiap AlbumNode menyimpan nama album dan pointer *next* menuju album berikutnya. Oleh karena itu, struktur album disediakan sebagai bagian dari desain struktur data, namun belum dimanfaatkan secara penuh dalam sistem library maupun playlist pada GUI.

## Implementasi Stack

Struktur data Stack digunakan dalam CIC Music Player untuk mencatat riwayat lagu yang telah diputar oleh pengguna. Stack bekerja dengan prinsip Last In First Out (LIFO), sehingga lagu yang terakhir diputar akan berada di bagian paling atas dan menjadi lagu pertama yang dapat diakses saat pengguna menekan tombol “Previous”.

Struktur dan Fungsi dalam Program:

Stack disimpan sebagai list Python sederhana (*history = [ ]*) dan menyediakan operasi:

- ***push(song)***, Menambahkan lagu yang baru diputar ke paling atas stack.
- ***pop()***, Menghapus dan mengembalikan lagu terakhir yang ada di stack dan digunakan ketika pengguna kembali ke lagu sebelumnya.
- ***peek()***, Melihat lagu terakhir yang diputar tanpa menghapusnya.

Implementasi stack ini mempermudah:

- Navigasi riwayat pemutaran lagu
- Penindihan riwayat ketika lagu baru diputar
- Pengelolaan history yang tetap ringan dan efisien

Sesuai kode, Stack terintegrasi penuh dengan PageUser, di mana setiap kali lagu diputar, sistem secara otomatis menambahkan ke riwayat.

### 2.2.2 Implementasi GUI

Antarmuka grafis CIC Music Player dibuat menggunakan CustomTkinter dan modul pygame.mixer untuk mengelola audio. GUI ini terbagi menjadi tiga halaman utama yaitu Menu, User, dan Admin. Setiap halaman saling berhubungan dan mengakses struktur data melalui objek *MusicPlayer* dari backend.

#### 1. Halaman Menu Utama (Page Menu)

Halaman ini berfungsi sebagai gerbang navigasi ke:

- Mode User
- Mode Admin

Menggunakan beberapa komponen CTK seperti *CTkButton*, *CTkFrame*, dan *CTkLabel*.

#### 2. Halaman User (Page User)

Halaman User merupakan pusat interaksi dengan pemutar musik. Fitur yang tersedia:

##### a. Library Lagu

Daftar lagu ditampilkan berdasarkan traversal Single Linked List artis-lagu. Data ini diubah menjadi list Python (*library\_items*) untuk memudahkan pemanggilan dan navigasi.

**b. Playlist**

Playlist menggunakan Doubly Linked List sehingga mendukung navigasi maju-mundur melalui pointer *next* dan *previous*.

Pengguna dapat:

- menambah lagu ke playlist
- menghapus lagu dari playlist
- memutar lagu dari playlist

playlist menyimpan referensi ke node lagu asli yang terdapat pada library, bukan salinan objek lagu, sehingga perubahan pada data lagu akan langsung tercermin di playlist.

**c. Panel Pemutar Lagu**

Menggunakan pygame untuk:

- Play
- Pause
- Next / Previous
- Auto-play saat lagu selesai
- Seek bar real-time
- Volume control

Sistem dilengkapi dengan fitur auto-play yang mendeteksi akhir pemutaran lagu menggunakan event pygame.USEREVENT. Ketika sebuah lagu selesai diputar, sistem secara otomatis memutar lagu berikutnya sesuai dengan mode pemutaran (library atau playlist).

Navigasi “Next” dan “Previous” menggunakan dua mekanisme:

- Library: indeks list (*current\_index*  $\pm$  1)
- Playlist: pointer Doubly Linked List (*current.prev* dan *current.next*)

**3. Halaman Admin (Page Admin)**

Fitur yang tersedia:

- Menampilkan seluruh lagu dari linked list artis-lagu
- Menambah lagu baru
- Mengedit lagu berdasarkan judul lagu
- Menghapus lagu berdasarkan judul lagu

Pengeditan dan penghapusan lagu pada halaman admin dilakukan berdasarkan judul lagu, sedangkan ID lagu digunakan sebagai identitas internal dan dapat dihasilkan secara otomatis oleh sistem. Semua operasi admin memodifikasi struktur data asli di backend, sehingga perubahan langsung terlihat di halaman user.

**2.2.3 Implementasi Queue**

Pada sistem CIC Music Player, struktur data Queue disediakan sebagai bagian dari desain backend, tetapi tidak digunakan dalam pemutaran lagu maupun playlist di versi aplikasi saat ini.

Operasi Queue yang digunakan:

- ***enqueue(song)***, berfungsi untuk menambahkan lagu ke dalam antrian pemutaran (queue), namun fitur ini belum terhubung dengan sistem pemutaran maupun playlist pada GUI.
- ***dequeue(song)***, berfungsi mengambil lagu pertama
- ***peek()***, berfungsi untuk melihat elemen pertamanya
- ***is\_empty()***, berfungsi untuk memeriksa apakah antrian kosong

### **Kesesuaian dengan Implementasi Asli**

Walaupun Queue ada di backend, PageUser tidak memanggil atau memanfaatkan Queue

- Playlist sepenuhnya memakai Doubly Linked List
- Pemutaran otomatis saat lagu selesai tidak mengambil lagu dari Queue, tetapi memakai urutan list library atau DLL playlist
- Queue tidak terhubung ke GUI atau fitur apapun

## **2.3 Kendala dan Solusi**

### **2.3.1 Single Linked List yang Memanjang**

Setiap operasi pada linked list harus dimulai dari node awal sehingga traversal menjadi lebih lama seiring bertambahnya data. Solusi yang diterapkan adalah memisahkan fungsi penting seperti `addSong()`, `searchSong()`, dan `deleteSong()` agar traversal lebih efisien dan pointer tetap terjaga.

### **2.3.2 Keterbatasan Navigasi Pada Playlist**

Perancangan awal mempertimbangkan penggunaan Queue sebagai playlist, tetapi tidak sesuai karena playlist membutuhkan navigasi dua arah. Solusi akhirnya adalah menggunakan Doubly Linked List, yang menyediakan pointer `next` dan `prev` sehingga pengguna dapat:

- Kembali ke lagu sebelumnya
- Melanjutkan ke lagu berikutnya

Queue tetap disertakan sebagai struktur tambahan untuk pengembangan masa depan.

### **2.3.3 Struktur Kode Menjadi Sulit Dikelola**

Penggabungan seluruh fungsi, struktur data, dan GUI dalam satu file membuat kode semakin panjang dan sulit dikelola. Untuk menjaga keteraturan, program dipecah menjadi beberapa modul berdasarkan fungsinya. Ada modul untuk struktur node, modul khusus operasi Single Linked List, modul playlist, modul GUI, dan satu file utama sebagai penghubung. Pemisahan ini membuat alur program lebih jelas dan memudahkan proses revisi.

## **2.4 Fitur-Fitur yang Akan Dibuat**

### **2.4.1 Fitur Umum**

- Tampilan *Dark Mode* (aplikasi menggunakan tema gelap)
- *Multipage Navigation* (sistem navigasi antara 3 halaman utama)

#### 2.4.2 Halaman Menu Utama (*Page Menu*)

- Pilihan masuk sebagai Pengguna
- Pilihan masuk sebagai Admin

#### 2.4.3 Halaman Pengguna (*Page User*)

Halaman Pengguna merupakan halaman utama tempat user berinteraksi dengan sistem pemutar musik. Seluruh fungsinya terhubung langsung dengan struktur data Single Linked List (library lagu) dan Doubly Linked List (playlist).

##### 1. Panel *Playlist*

Playlist pengguna disusun menggunakan Doubly Linked List, sehingga setiap lagu pada playlist memiliki pointer *next* dan *prev*. Hal ini memungkinkan navigasi dua arah ketika memutar lagu.

Fitur pada panel Playlist meliputi:

- Lihat Playlist  
Menampilkan seluruh lagu yang ada di playlist dalam urutan node DLL.
- Tambah ke Playlist  
Pengguna memilih lagu dari daftar library, lalu lagu tersebut ditambahkan ke playlist sebagai node baru serta tidak menggunakan input ID, melainkan berdasarkan lagu yang dipilih langsung dari GUI.
- Hapus dari Playlist  
Menghapus lagu tertentu berdasarkan node playlist yang dipilih, bukan berdasarkan ID.
- Kosongkan Playlist  
Menghapus seluruh node playlist sekaligus.
- Putar dari Playlist  
Lagu diputar dari playlist menggunakan pointer *next* dan *prev*.

##### 2. Panel *Library* atau Daftar Lagu

- Lihat Daftar Lagu  
Menampilkan semua lagu dengan informasi ID, judul, durasi, artis, dan file audio.
- Pencarian Lagu  
Mencari lagu berdasarkan judul, artis, atau genre serta melakukan proses pencarian dengan traversal Single Linked List.

##### 3. Panel Kontrol Pemutar Lagu

- Play / Pause Lagu
- Next / previous
- Seekbar waktu (mengikuti progres lagu secara real-time)
- Volume control
- Auto-play ketika lagu selesai (menggunakan `pygame.USEREVENT`)

Navigasi lagu bekerja dalam dua sistem berbeda:



- Navigasi Library:  
Menggunakan indeks pada list *library\_items*, bukan linked list.
- Navigasi Playlist:  
Menggunakan pointer *prev* dan *next* pada Doubly Linked List.

Dengan proses ini, halaman User memberikan pengalaman pemutaran musik yang interaktif dan dinamis.

#### 2.4.4 Halaman Admin (*Page Admin*)

Halaman Admin digunakan untuk mengelola database lagu yang tersimpan dalam struktur Single Linked List artis-lagu. Setiap perubahan yang dilakukan admin akan langsung memodifikasi struktur data di backend, sehingga perubahan juga langsung terlihat pada halaman User.

##### 1. Panel *Library Admin*

Admin dapat melihat seluruh lagu yang tersimpan dalam linked list. Tampilan ini ditampilkan sebagai list yang berisi:

- ID lagu
- Judul lagu
- Nama artis
- Durasi
- Path file audio

Admin tidak mengelola album, karena album tidak terintegrasi dengan sistem library.

##### 2. Panel *Form Management*

Fitur manajemen lagu meliputi:

- Tambah Lagu Baru  
Admin memasukkan data: ID lagu, judul lagu, artis, durasi, genre, dan path file audio kemudian lagu disimpan ke dalam Single Linked List milik artis terkait.
- Edit Lagu Berdasarkan Judul Lagu  
Admin mencari lagu berdasarkan judul lagu dengan traversal linked list, kemudian memperbarui data lagu tersebut.
- Hapus Lagu Berdasarkan Judul Lagu  
Node lagu dihapus dari Single Linked List artis tempat lagu tersebut berada.

Semua operasi modifikasi dilakukan pada struktur linked list asli (*songs\_head* setiap artis).

##### 3. Sinkronisasi dengan Halaman User

Setiap perubahan seperti penambahan , atau penghapusan lagu langsung memengaruhi library yang ditampilkan di halaman User. Playlist tidak otomatis diperbarui ketika lagu dihapus, karena playlist menyimpan referensi node, hal ini dapat menjadi pengembangan di versi mendatang.

## BAB III

### PENUTUP

#### 3.1 Kesimpulan

Berdasarkan hasil perancangan dan implementasi aplikasi CIC Music Player, dapat disimpulkan bahwa penerapan struktur data yang tepat sangat berpengaruh terhadap pengelolaan dan navigasi data musik dalam aplikasi. Struktur Single Linked List berhasil digunakan untuk menyimpan data artis dan lagu secara bertingkat, sehingga mendukung penambahan, pencarian, pembaruan, dan penghapusan data secara dinamis.

Pengelolaan playlist menggunakan Doubly Linked List terbukti efektif dalam mendukung navigasi dua arah (*next* dan *previous*) tanpa memerlukan traversal ulang, sehingga meningkatkan efisiensi pemutaran lagu. Playlist juga dirancang dengan menyimpan referensi ke node lagu asli pada library, bukan salinan objek, sehingga perubahan data lagu pada library dapat langsung tercermin di playlist.

Struktur Stack dimanfaatkan untuk menyimpan riwayat pemutaran lagu dengan prinsip Last In First Out (LIFO), yang mendukung fitur navigasi *previous* secara logis dan efisien. Sementara itu, struktur Queue telah disediakan pada sisi backend sebagai bagian dari desain sistem, namun belum digunakan secara aktif dalam pemutaran lagu maupun pengelolaan playlist pada versi aplikasi saat ini.

Dari sisi antarmuka, implementasi GUI menggunakan CustomTkinter dan pygame berhasil mengintegrasikan struktur data backend dengan fitur pemutaran musik, navigasi lagu, playlist, serta panel admin. Fitur auto-play juga telah diimplementasikan dengan memanfaatkan event pygame.USEREVENT, sehingga sistem dapat memutar lagu berikutnya secara otomatis ketika lagu sebelumnya selesai diputar.

Secara keseluruhan, aplikasi CIC Music Player telah berhasil mengimplementasikan konsep struktur data yang dipelajari dalam mata kuliah Struktur Data ke dalam sebuah studi kasus nyata, serta menunjukkan keterkaitan antara teori struktur data dan implementasinya dalam pengembangan aplikasi.

#### 3.2 Rekomendasi

Berdasarkan hasil implementasi dan pengujian sistem, terdapat beberapa rekomendasi pengembangan yang dapat dilakukan di masa mendatang, antara lain:

##### 1. Integrasi Queue pada Pemutaran Lagu

Struktur Queue yang telah disediakan di backend dapat diintegrasikan secara penuh dengan GUI untuk mendukung fitur *play next*, *play later*, atau antrian pemutaran lagu secara dinamis.

##### 2. Integrasi Struktur Album ke Library dan Playlist

Struktur album yang saat ini masih berfungsi sebagai metadata pendukung dapat dikembangkan agar terhubung langsung dengan library dan playlist, sehingga pengguna dapat memutar lagu berdasarkan album.

##### 3. Sinkronisasi Playlist dengan Library

Perlu ditambahkan mekanisme sinkronisasi agar playlist dapat menyesuaikan diri secara otomatis ketika lagu dihapus atau diperbarui oleh admin pada library.

#### **4. Optimalisasi Pencarian Lagu**

Proses pencarian lagu pada linked list dapat dioptimalkan dengan struktur tambahan seperti tree untuk meningkatkan performa ketika jumlah data semakin besar

#### **5. Pengembangan Fitur Tambahan pada GUI**

Beberapa fitur tambahan seperti pengurutan lagu, filter berdasarkan genre, serta indikator visual pemutaran lagu dapat ditambahkan untuk meningkatkan pengalaman pengguna.

Dengan adanya pengembangan lebih lanjut, aplikasi CIC Music Player diharapkan dapat menjadi sistem pemutar musik yang lebih lengkap, efisien, dan interaktif, sekaligus memperkuat penerapan konsep struktur data dalam pengembangan perangkat lunak.

## **LINK GITHUB**

<https://github.com/izamrosiawan/tugas-besar-music-player-py>

## **REFERENSI**

Goodrich, M. T., Tamassia, R., & Goldwasser, M. H. (2013). Data Structures and Algorithms in Python. Wiley.

Python Software Foundation. (2024). Python Documentation. <https://docs.python.org/3/>

Pygame Community. (2024). Pygame - Python Game Development. <https://www.pygame.org/>

Schimansky, T. (2024). CustomTkinter - Modern UI Library.  
<https://github.com/TomSchimansky/CustomTkinter>

Real Python. (2024). Python OOP Tutorial. <https://realpython.com/>