

LAPORAN EKSPLORASI, DEKOMPOSISI, DAN SIMULASI PEMUTAR MUSIK DARING (SPOTIFY)

Disusun untuk Memenuhi Tugas Mata Kuliah Pengenalan Komputasi KU1102

Dosen Pengampu:

Dr. Fadhil Hidayat, S.Kom., M.T.



Kelompok 13 K-19 STEI

- | | |
|------------------------------|------------|
| 1. Karol Yangqian Poetrachya | (19623206) |
| 2. Nayaka Ghana Subrata | (19623031) |
| 3. Julian Benedict | (16523178) |
| 4. Dimas Anggiat | (16523052) |

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2023**

DAFTAR ISI

DAFTAR ISI.....	1
BAB I TUGAS 1: EKSPLORASI DAN DEKOMPOSISI.....	2
1.1 Eksplorasi.....	2
1.2 Dekomposisi.....	7
BAB II TUGAS 2: RANCANGAN SIMULASI.....	8
2.1 Deskripsi Simulasi.....	8
2.2 Flowchart.....	9
BAB III TUGAS 3: SIMULASI.....	10
BAB IV KESIMPULAN, PEMBELAJARAN, DAN PEMBAGIAN TUGAS.....	22
4.1 Kesimpulan.....	22
4.2 Pembelajaran.....	22
4.3 Pembagian Tugas.....	22
DAFTAR REFERENSI.....	23

BAB I

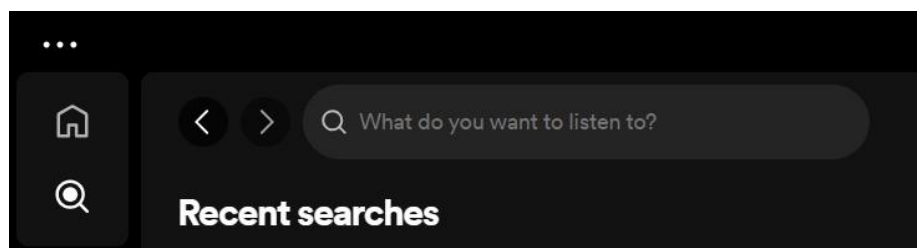
TUGAS 1: EKSPLORASI DAN DEKOMPOSISI

1.1 Eksplorasi

Online music player, atau aplikasi pemutar lagu secara daring ini sering dipakai oleh kalangan muda untuk mendengarkan musik dengan cara *streaming* tanpa harus mengunduh lagu tersebut. Aplikasi pemutar lagu online ini bekerja dengan cara mengirimkan data *file* musik yang sudah di-*pre-buffered* – sebuah praktik pemuatan segmen data saat *streaming* video atau lagu – beberapa waktu sebelum musik tersebut diputar agar musik dapat diputar dengan mulus tanpa terputus-putus. Ada beberapa aplikasi pemutar lagu yang terkenal dan sering digunakan oleh banyak orang, seperti Spotify, Apple Music, Joox, Soundcloud, dan lain-lain. Namun, aplikasi yang kami bahas dalam esai ini adalah aplikasi Spotify. Dari sisi *front-end*, Spotify memiliki desain yang modern dan intuitif. Di dalam aplikasi ini terdapat beberapa fitur:

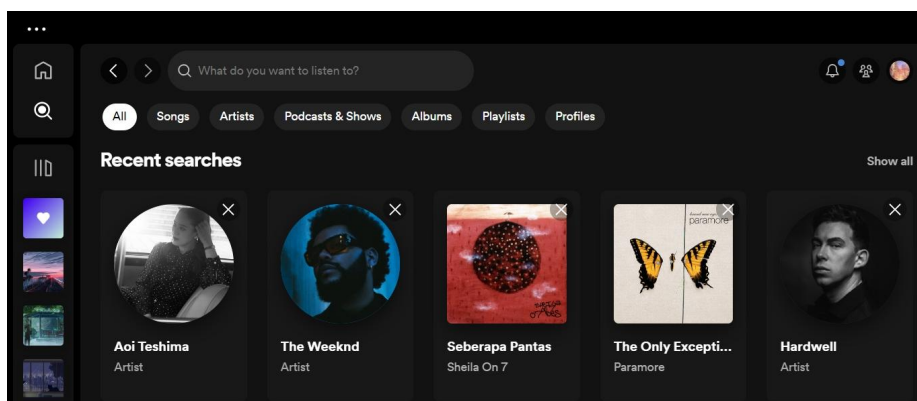
1. Kolom Pencarian

Dalam spotify terdapat fitur bagi sebagai pengguna untuk mencari lagu yang diinginkan dengan cara menulis judul atau penyanyi di lagu tersebut pada kolom pencarian seperti pada gambar 1.1.



Gambar 1.1 Kolom Pencarian Lagu dalam Spotify

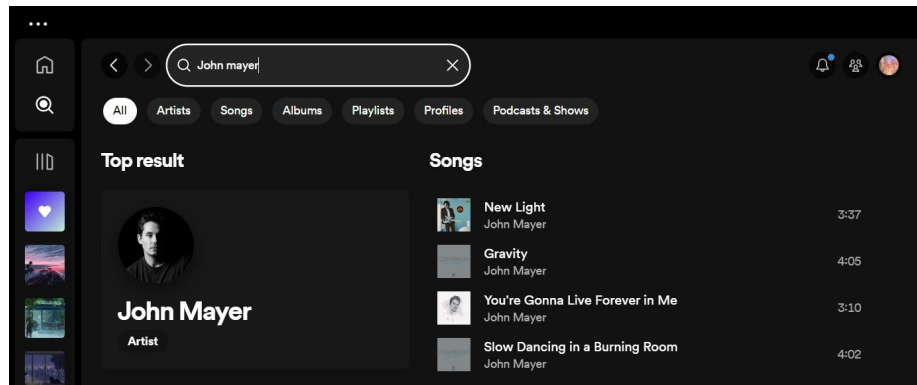
Di bawah kolom pencarian tersebut ada *history* atau riwayat pencarian yang telah dilakukan sebelumnya seperti pada gambar 1.2.



Gambar 1.2 Riwayat Pencarian Spotify

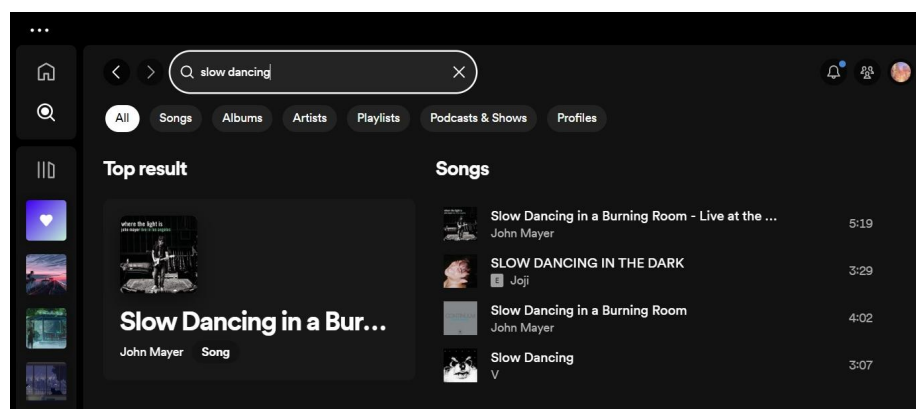
Jika dilakukan sebuah pencarian, contohnya dengan memasukkan nama artis, maka Spotify akan memunculkan *profile* dari artis tersebut beserta beberapa

lagu populer milik artis tersebut. Dengan mencari nama “John Mayer”, pengguna dapat melihat beberapa lagu miliknya yang populer.



Gambar 1.3 Pencarian “John Mayer”

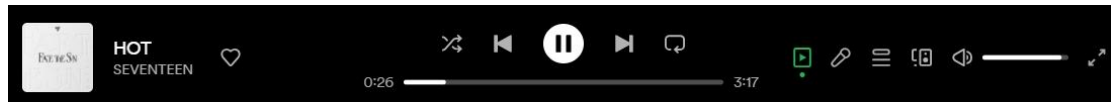
Ketika pengguna melakukan pencarian sebuah lagu dengan menuliskan sebuah *keyword*, Spotify juga akan memunculkan beberapa lagu dengan judul yang mirip, contohnya ketika dimasukkan sebuah *keyword* “Slow dancing”, beberapa lagu akan bermunculan, seperti “Slow Dancing in a Burning room” oleh John mayer, “SLOW DANCING IN THE DARK” oleh Joji, dan “Slow Dancing” oleh V (BTS).



Gambar 1.4 Pencarian dengan *keyword* “Slow Dancing”

2. Pemutar lagu

Dalam Spotify, pengguna dapat memutar lagu secara daring dan tidak harus mengunduhnya terlebih dahulu. Pengguna dapat memutar lagu dengan menekan tombol *play* dan menghentikan pemutaran lagu dengan menekan tombol *pause*. *Play* dan *pause* merupakan satu tombol yang sama tapi bertukar fungsi setiap kali ditekan. Tampilan antarmuka *play* dan *pause* di Spotify dapat dilihat pada gambar 1.5.



Gambar 1.5 Bagian Antarmuka Pemutaran Lagu

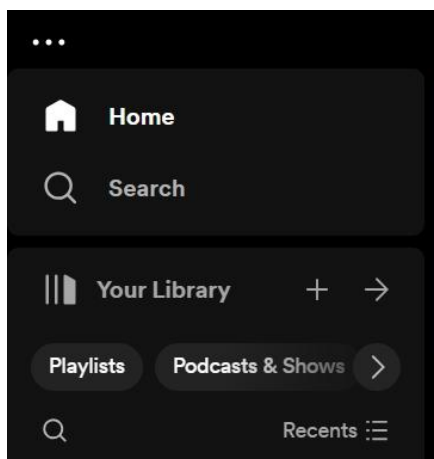
Antarmuka *play* dan *pause* memiliki beberapa tombol yang berfungsi sebagai berikut (urutan dari kiri ke kanan);

- Tombol *shuffle* untuk mengacak urutan lagu yang akan diputar
- Tombol *back* untuk memutar lagu sebelumnya
- Tombol *pause* dan *play* untuk memutar dan memberhentikan lagu yang sedang diputar
- Tombol *next* untuk memutar lagu selanjutnya
- Tombol *loop* untuk mengulang lagu yang sedang diputar

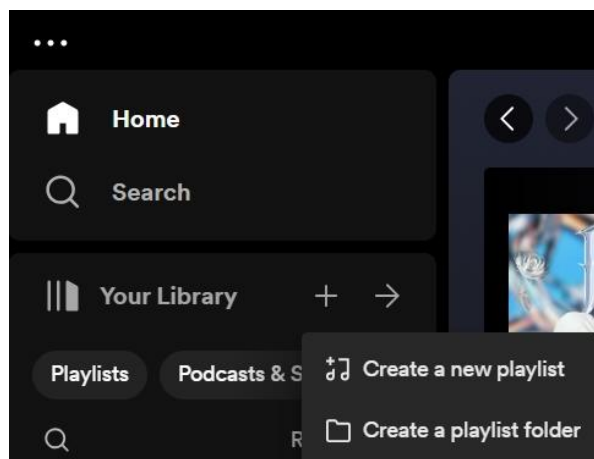
Di bagian kanan pemutar ada tombol *lyrics* untuk menampilkan lirik lagu, tombol *queue* untuk menampilkan urutan lagu berikutnya, tombol *list device* untuk menampilkan gawai-gawai yang dapat dipakai untuk memutar lagu, dan *slider* untuk mengatur volume lagu.

3. Playlist

Playlist merupakan sebuah fitur oleh Spotify untuk mengelompokkan lagu sesuai yang diinginkan pengguna. Terdapat playlist pribadi yang hanya bisa diakses pengguna yang membuatnya dan juga ada playlist publik yang bisa diakses seluruh pengguna Spotify. Dengan fitur ini, pengguna dapat memainkan lagu-lagu tertentu pilihannya yang hanya terdapat dalam playlist tersebut. Di bagian kiri atas ada lambang ‘tambah’ untuk membuat sebuah *playlist*. Jika ditekan akan muncul tampilan seperti gambar 1.6 (b).

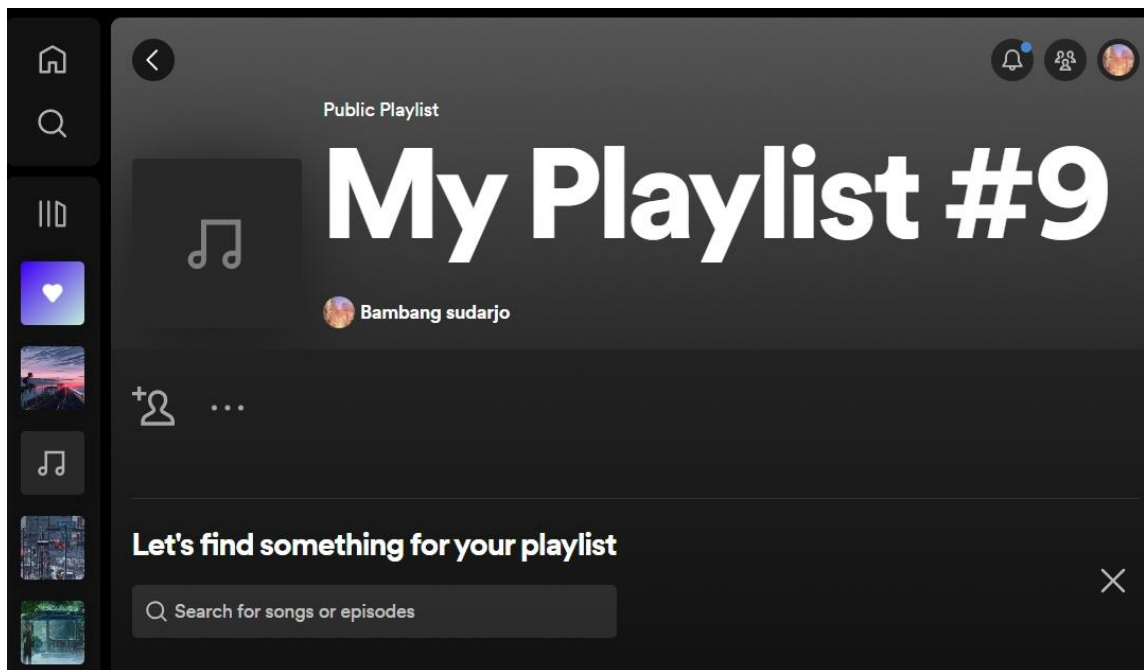


Gambar 1.6 (a) *Library*



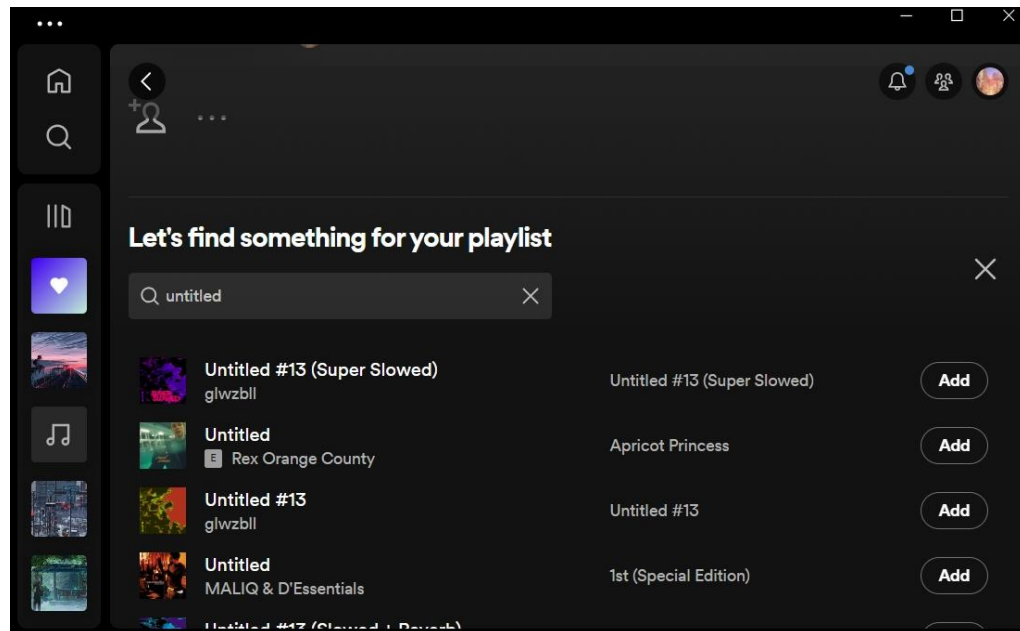
Gambar 1.6 (b) Menambahkan *Playlist*

Setelah menekan “Create a new playlist”, tampilan Spotify akan berubah menjadi seperti gambar 1.7.



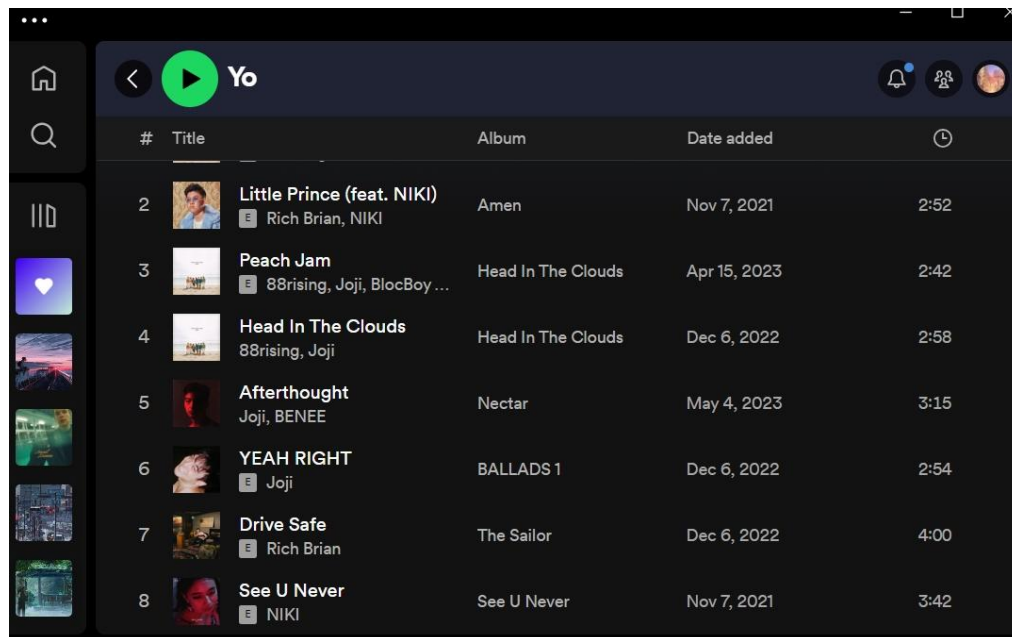
Gambar 1.7 Tampilan *Playlist* Baru

Di sini, *playlist* bisa dimodifikasi, mulai dari nama *playlist*, *cover playlist*, menambahkan pengguna untuk berkolaborasi membuat *playlist* tersebut. Di bawah tulisan “Let’s find something for your playlist” terdapat fitur *search* untuk mencari dan menambahkan lagu ke *playlist*.



Gambar 1.8 Mencari dan Menambahkan Lagu untuk *Playlist*

Setelah menambahkan beberapa lagu dan memodifikasi *playlist*, tampilan akan berubah menjadi seperti gambar 1.9.



Gambar 1.9 *Playlist* yang Sudah Diisi Lagu

Dari sisi *back-end*, Spotify menyimpan data lagu dari seluruh dunia dengan akumulasi ukuran yang sangat besar. Hal itu diatasi dengan menyimpan data tersebut dalam sebuah server yang memiliki ukuran penyimpanan data yang sangat besar. Server, atau lebih tepatnya application server, dihubungkan dengan internet agar data dapat diakses oleh seluruh pengguna yang disebut sebagai ‘client’. Ketika client mencari atau memutar lagu pada aplikasi spotify, client menyampaikan sebuah ‘request’ kepada server untuk meminta data lagu yang ingin dicari atau diputar, lalu server merespon dengan memberikan data tersebut kepada client melalui internet.

Data disimpan dalam server secara terstruktur dan terorganisasi. Kumpulan data yang terstruktur ini disebut sebagai “database”. Database menggunakan pengindeksan data, tabel, baris, kolom, dan relasi antardata untuk mengorganisasi data yang disimpan. Jenis data yang dapat disimpan meliputi judul lagu, penyanyi, tahun rilis, genre, bahasa lagu, jumlah pendengar, dan lain-lain. Biasanya database diatur menggunakan Database Management System (DBMS). Spotify menggunakan Apache Cassandra sebagai DBMS-nya. Ini memudahkan pengembang Spotify untuk mengelola data yang luar biasa besar dan kompleks.

1.2 Dekomposisi

Berikut adalah dekomposisi fungsi aplikasi Spotify secara sederhana.

1. Aplikasi pemutar musik daring Spotify
 - 1.1. *Front-end*
 - 1.1.1. *Search*
 - 1.1.1.1. *Search bar*
 - 1.1.1.2. *Pengaturan filter*
 - 1.1.1.2.1. *All*
 - 1.1.1.2.2. *Songs*
 - 1.1.1.2.3. *Albums*
 - 1.1.1.2.4. *Artists*
 - 1.1.1.2.5. *Playlists*
 - 1.1.1.2.6. *Profiles*
 - 1.1.1.2.7. *Podcasts & Shows*
 - 1.1.1.3. *Menampilkan hasil pencarian*
 - 1.1.2. *Playlist*
 - 1.1.2.1. *Membuat playlist*
 - 1.1.2.2. *Menyunting playlist*
 - 1.1.2.3. *Memasukkan lagu*
 - 1.1.2.4. *Membagikan playlist*
 - 1.1.3. *Antarmuka untuk memainkan musik*
 - 1.1.3.1. *Progress bar*
 - 1.1.3.2. *Play/pause*
 - 1.1.3.3. *Pergi ke lagu sebelumnya/berikutnya*
 - 1.2. *Back-end*
 - 1.2.1. *Database*
 - 1.2.1.1. *Judul lagu*
 - 1.2.1.2. *Penyanyi*
 - 1.2.1.3. *Popularitas (jumlah pendengar)*
 - 1.2.1.4. *Genre*
 - 1.2.1.5. *Tahun rilis*
 - 1.2.1.6. *Bahasa*
 - 1.2.2. *Application server*
 - 1.2.3. *API (Application Programming Interface)*
 - 1.2.4. *Jaringan internet*

BAB II

TUGAS 2: RANCANGAN SIMULASI

2.1 Deskripsi Simulasi

Kami merancang simulasi aplikasi Spotify berdasarkan fungsi-fungsi dasar berikut.

2.1.1 Mencari lagu

Pengguna memasukkan judul lagu yang akan dicari. Kemudian, program akan mencari judul lagu yang dimasukkan pengguna dan program akan menampilkan lagu pada terminal serta memberi opsi untuk menambahkan lagu ke playlist.

2.1.2 Mencari lagu berdasarkan genre

Pengguna memasukkan genre lagu yang diinginkan. Kemudian, program akan mencari lagu dengan genre yang telah dimasukkan dan program akan menampilkan lagu tersebut.

2.1.3 Menghapus Lagu

Pengguna memasukkan lagu yang ingin dihapus. Kemudian, program akan memproses dan menghapus lagu tersebut dari playlist.

2.1.4 Membuat Playlist

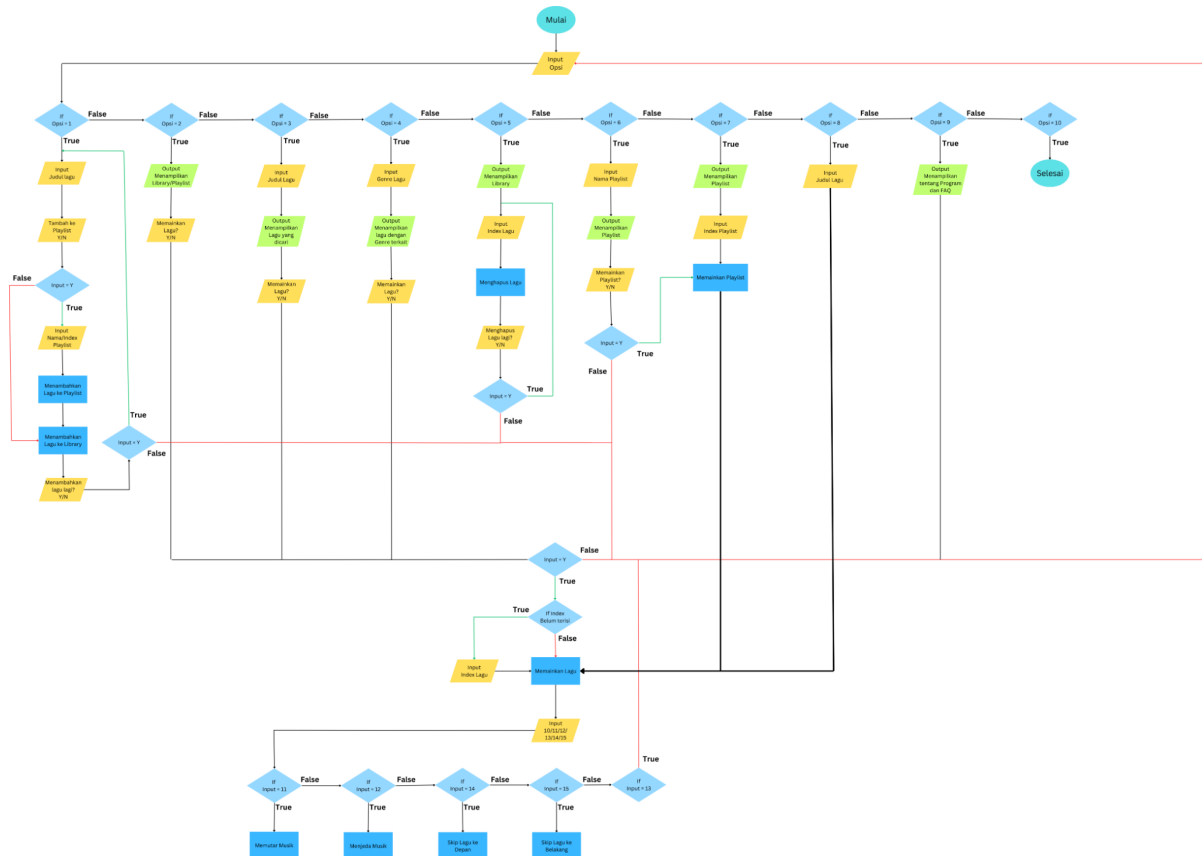
Pengguna memasukkan judul lagu dan nama playlist yang akan dibuat. Kemudian, program akan membuat playlist dan menambahkan judul lagu yang dimasukkan user ke dalam playlist.

2.1.5 Memainkan lagu atau playlist

Pengguna memasukkan judul lagu atau memilih playlist yang ingin diputar. Kemudian, program akan memproses dan memainkan lagu atau playlist tersebut.

Simulasi akan diprogram dalam bahasa python. Integrated development environment (IDE) yang digunakan untuk menulis program adalah Visual Studio Code (VS Code). Terminal IDE Visual Studio Code akan digunakan sebagai antarmuka simulasi yang meliputi pemberian input angka sebagai cara pemilihan opsi dan output tampilan antarmuka yang berbasis *text message art* atau *ASCII art*, yakni grafika yang dibuat menggunakan teks dan simbol pada *keyboard*. Agar dapat memutar media musik pada perangkat, digunakan modul “vlc” yang akan di-*import* pada program python. Sebagai substitusi database lagu yang realitanya disimpan dalam server, pada simulasi ini digunakan sebuah modul yang dapat mengakses database aplikasi Youtube dan mencari lagu berupa *music video* berdasarkan input pencarian yang diberikan berupa teks atau string. Setelah mendapatkan lagu dari pencarian di Youtube, lagu akan disimpan dalam sebuah file JSON dengan format struktur data yang sudah diatur. File JSON ini akan menjadi tempat penyimpanan data lagu dan playlist milik pengguna dan dapat dimanipulasi oleh pengguna menggunakan fitur-fitur seperti membuat playlist, menghapus lagu, dan menambahkan lagu ke playlist.

2.2 Flowchart



Gambar 2.1 Flowchart Rancangan Program Simulasi

Gambar 2.1 adalah flowchart rancangan simulasi pemutar music daring yang akan diimplementasikan menggunakan python. Terdapat percabangan yang cukup banyak sebagai program yang relatif sederhana ini karena simulasi ini akan menawarkan beberapa opsi fitur yang dapat diakses dan dijalankan. Arsip flowchart selengkapnya dapat diakses pada tautan <https://bit.ly/flowchartspotify>.

BAB III

TUGAS 3: SIMULASI

Berikut ini adalah gambaran source code yang dibuat dan tampilan antarmuka simulasi pemutar musik daring.

1. Import Libraries and Modules

```
import os; import re ; import vlc
import json; import pafy; import time
import threading; from youtubearchpython import VideosSearch
```

2. Array untuk library dan playlist

```
...
FILENAME = "musiclib.json"
library = []
PlayList_kini = []
Musik_kini = False
Now_playing = ""
...
```

3. Menyiapkan VLC

```
...
os.environ["VLC_VERBOSE"] = str("-1")
Instance = vlc.Instance('--no-xlib -q > /dev/null 2>&1')
Instance.log_unset()
player = Instance.media_player_new()
clear = "\n" * 100
...
```

4. Membuat “class” untuk menyimpan data lagu

```
class Lagu(): # Membuat class

# Membuat fungsi untuk parameter "Instance"
    def __init__(self, lNama, lGrup, lTahun, lGenre, lPlayList, lLink):

        self>NamaLagu = lNama
        self.GrupLagu = lGrup
        self.TahunLagu = lTahun
        self.GenreLagu = lGenre
        self.PlayListLagu = lPlayList
        self.LinkLagu = lLink
```

```
# Membuat fungsi untuk mengeluarkan output dari "self"
def printSelf(self):
    print(f"{self>NamaLagu} - {self.GrupLagu} - {str(self.TahunLagu)}")

# Membuat fungsi untuk format data ke bentuk JSON
def FormatJson(self):
    return{
        "NamaLagu": self>NamaLagu,
        "GrupLagu": self.GrupLagu,
        "TahunLagu": self.TahunLagu,
        "GenreLagu": self.GenreLagu,
        "PlayListLagu": self.PlayListLagu,
        "LinkLagu": self.LinkLagu
    }
```

5. Membuat fungsi dari program

Fungsi untuk memutar lagu

```
...
def playSong(url):
    global Musik_kini
    Musik_kini = True
    ...
    player.play()
...
```

Fungsi untuk mencari musik

```
...
def cariMusik(query):
    results = VideosSearch(query, limit = 1)
    return results.result()[ 'result' ][0]
...
```

Fungsi untuk memutar musik yang dipilih

```
...
def putarMusikPilihan(ListLagu):
    global Now_playing
    while True:
        pilih = input("Apakah Anda ingin memutar musik? (Y/N) ")
        ...
        elif pilih.upper() == "N":
            break
...
```

Fungsi untuk menampilkan hasil lagu yang dicari

```

...
def searchOutput():
    searchString = input("Masukkan musik yang ingin Anda cari -->
").title()
    res = searchLibrary(searchString)
    ...
    return res
...

```

Fungsi untuk mengecek validasi dari Input-Integer

```

...
def validasiInput(pilih, min, max):
    if (pilih in range (min,max)):
        return True
    else:
        return False
...

```

Fungsi untuk mengecek validasi dari Input-String

```

...
def validasiInputString(pilih):
    if (re.match(r"[\s\w]+$",pilih)):
        return True
    else:
        return False
...

```

Fungsi untuk menambahkan file dan data lagu ke library

```

...
def tambahkeFile(data):
    with open('musiclib.json','r') as openfile:
        json_obj= json.load(openfile)

        json_obj.append(data)
    with open("musiclib.json","w") as outfile:
        json.dump(json_obj,outfile,indent=4)

    library.append(Lagu(data["NamaLagu"], data["GrupLagu"],
data["TahunLagu"], data["GenreLagu"], data["PlayListLagu"],
data["LinkLagu"]))
...

```

Fungsi untuk membaca data lagu

```

...
def bacaFile():
    with open('musiclib.json', 'r') as openfile:
        json_obj = json.load(openfile)

        for l in json_obj:

library.append(Lagu(l["NamaLagu"],l["GrupLagu"],l["TahunLagu"],l["GenreLa

```

```

gu"],1["PlayListLagu"],1["LinkLagu"])))
    return json_obj
...

```

Fungsi untuk mencari lagu di library

```

...
def searchLibrary(searchString):
    if len(library) <= 0:
        bacaFile()
    ...
    return found
...

```

Fungsi untuk mencari lagu berdasarkan genre

```

...
def searchByGenre(genre):
    results = []
    for song in library:
        ...
    else:
        print("Tidak ada lagu dengan genre tersebut.")
...

```

Fungsi untuk mengolah lagu di playlist

```

...
def getPlayLists():
    playL = []
    for g in library:
        if not g.PlayListLagu in playL:
            playL.append(g.PlayListLagu)
    return playL
...

```

Fungsi untuk mencari lagu di playlist

```

...
def searchInPlaylist(playlist):
    results = []
    for song in library:
        ...
    else:
        print(f"Tidak ada lagu dalam playlist {playlist}.")
...

```

Fungsi untuk menghapus lagu di lagu/playlist

```

...
def hapusLagu():
    res = searchOutput()

    if not res:

```

```

        print("Library kosong. Tidak ada lagu yang bisa dihapus.")
        return

    try:
        ...
    except ValueError:
        print("Input tidak valid. Masukkan angka sebagai indeks.")
...

```

Fungsi untuk memutar playlist

```

...
def putarPlayList>NamaPlaylist):
    play = []
    for g in library:
        ...

    for link in play:
        ...
        continue
...

```

Fungsi untuk skip lagu ke depan

```

...
def nextSong():
    global Now_playing, library, player
    if Now_playing:
        ...
    else:
        Now_playing = library[0].NamaLagu
        playSong(library[0].LinkLagu)
...

```

Fungsi untuk skip lagu ke belakang

```

...
def previousSong():
    global Now_playing, library, player
    if Now_playing:
        ...
    else:
        Now_playing = library[-1].NamaLagu
        playSong(library[-1].LinkLagu)
...

```

Fungsi untuk membuat output keluar secara bertahap

```

...
def print_bertahap(teks, delay=1):
    for karakter in teks:
        print(karakter, end='', flush=True)
        time.sleep(delay)
...

```

Fungsi untuk membuat animasi loading pada program

```
...
def loadanimasi():
    animasi = [
        ...
    ]

    Tidak_selesai = True
    i = 0

    while Tidak_selesai:
        ...
        if i > 17:
            Tidak_selesai = False
...

```

6. Membuat interface dan juga sistem interaksi dengan user

Membuat GUI dan interface

```
...
def menu():
    global Musik_kini, Now_playing
    print(r""" ...
    """)

    print("""
    ...

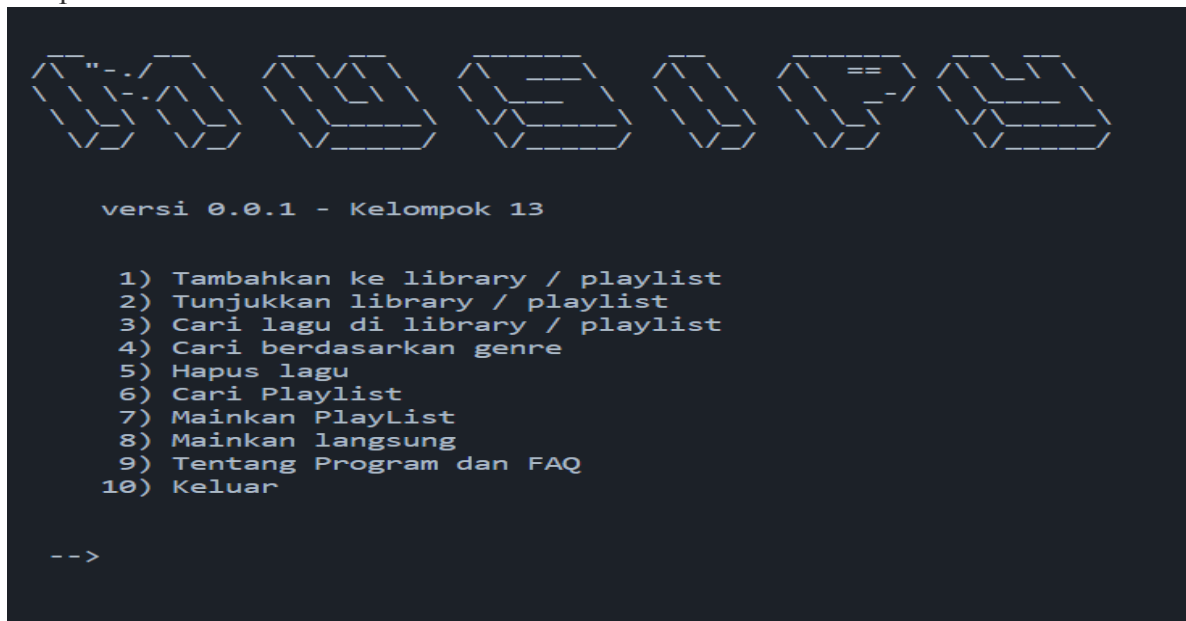
    """)

    if Musik_kini:
        print(f"Now playing.... {Now_playing} ")
        print(r"""
        (radio)
        """)

    opsi = int(input(" -->"))
    while not validasiInput(opsi, 1, 16):
        opsi = int(input(" -->"))
...

```


Tampilan GUI :



```

versi 0.0.1 - Kelompok 13

1) Tambahkan ke library / playlist
2) Tunjukkan library / playlist
3) Cari lagu di library / playlist
4) Cari berdasarkan genre
5) Hapus lagu
6) Cari Playlist
7) Mainkan PlayList
8) Mainkan langsung
9) Tentang Program dan FAQ
10) Keluar

-->

```

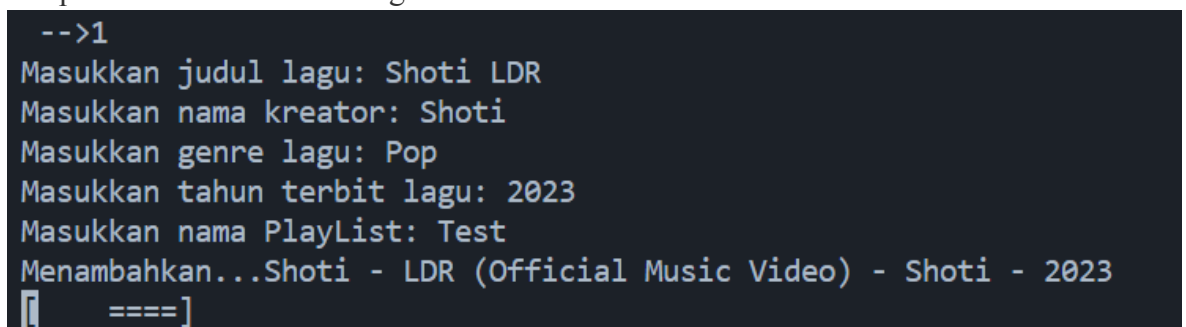
Menambahkan lagu ke library/playlist

```

...
if opsi == 1:
    judul = input("Masukkan judul lagu: ").title()
    while not validasiInputString(judul):
        judul = input("Masukkan judul lagu: ").title()
    ...
    Musik = Lagu(cariMusik(judul) ["title"], kreator, tahun, genre,
Playlist, cariMusik(judul) ["link"])
    Musik.printSelf()
    tambahkeFile (Musik.FormatJson())
...

```

Tampilan ketika menambah lagu :



```

-->1
Masukkan judul lagu: Shoti LDR
Masukkan nama kreator: Shoti
Masukkan genre lagu: Pop
Masukkan tahun terbit lagu: 2023
Masukkan nama PlayList: Test
Menambahkan...Shoti - LDR (Official Music Video) - Shoti - 2023
[====]

```

Menampilkan library/playlist

```

elif opsi == 2:
    print_bertahap("Ini adalah library Anda\n", delay = 0.05);
time.sleep(0.5)
...

putarMusikPilihan(library)

```

Tampilan ketika menampilkan library :

```
-->2
Ini adalah library Anda
-----
1 Anggi Marito - Tak Segampang Itu (Official Music Video) - Anggita Marito - 2022
2 Hikaru Nara - Goose House [Romaji, Español, English, Color Coded] - Goose House - 2017
3 Ardhito Pramono ft. Aurélie Moeremans - I Just Couldn't Save You Tonight (Story of Kale - OMPS) - Ardhito Pramono - 2021
4 Coldplay - Fix You (Official Video) - Coldplay - 2020
5 Shoti - LDR (Official Music Video) - Shoti - 2023
-----
Apakah Anda ingin memutar musik? (Y/N) y
Masukkan indeks musik --> 5
[ ] =====
```

Mencari lagu di library/playlist

```
elif opsi == 3:
    res = searchOutput()
    if res:
        putarMusikPilihan(res)
    else:
        print("Tidak ada lagu yang ditemukan.")
```

Tampilan ketika mencari lagu :

```
-->3
Masukkan musik yang ingin Anda cari --> shoti
1 Shoti - LDR (Official Music Video) Shoti Pop
Apakah Anda ingin memutar musik? (Y/N) [ ]
```

Mencari lagu berdasarkan genre

```
elif opsi == 4:
    genre = input("Masukkan genre lagu yang ingin dicari: ")
    searchByGenre(genre)
```

Tampilan ketika mencari lagu :

```
-->4
Masukkan genre lagu yang ingin dicari: pop
Hasil pencarian berdasarkan genre:
1 Anggi Marito - Tak Segampang Itu (Official Music Video) - Anggita Marito - 2022
2 Coldplay - Fix You (Official Video) - Coldplay - 2020
3 Shoti - LDR (Official Music Video) - Shoti - 2023
Apakah Anda ingin memutar musik? (Y/N) [ ]
```

Menghapus lagu

```
elif opsi == 5:
    hapusLagu()
```

Tampilan ketika menghapus lagu :

```
-->5
Masukkan musik yang ingin Anda cari --> shoti
1 Shoti - LDR (Official Music Video) Shoti Pop
Masukkan indeks lagu yang ingin dihapus (0 untuk membatalkan): 1
Lagu berhasil dihapus: Shoti - LDR (Official Music Video)
Lagu berhasil dihapus.
[      ===]
```

Mencari playlist

```
elif opsi == 6:
    playlist = input("Masukkan nama playlist yang ingin dicari: ")
    searchInPlaylist(playlist)
```

Tampilan ketika mencari playlist :

```
-->6
Masukkan nama playlist yang ingin dicari: fix
Hasil pencarian di playlist fix:
1 Coldplay - Fix You (Official Video) - Coldplay - 2020
Apakah Anda ingin memutar musik? (Y/N) █
```

Memutar playlist

```
elif opsi == 7:
    pl = getPlayLists()
    print()
    c = 1
    for p in pl:
        print(str(c) + ") ", end=" ")
        print(p)
        c += 1

    while True:
        pilih = input("Apakah Anda ingin memutar PlayList? (Y/N) ")
        if pilih.upper() == "Y":
            index = int(input("Pilih PlayList --> "))
            print(pl[index - 1])
            putarPlayList(pl[index - 1])
            break
        elif pilih.upper() == "N":
            break
```

Tampilan ketika memutar playlist :

```
-->7

1) Pop
2) Galau
3) Fix
Apakah Anda ingin memutar PlayList? (Y/N) y
Pilih PlayList --> 1
Pop
Now Playing ... Anggi Marito - Tak Segampang Itu (Official Music Video) Anggita Marito Pop
```

```

  |-----|
  |#####|
  |###   ###|
  |###   ###|
  |#####|
  |-----|

```

Memainkan lagu secara langsung

```
elif opsi == 8:
    query = input("Masukkan lagu yang ingin diputar ==>")
    Now_playing = cariMusik(query) ['title']
    player.stop()
    playSong(cariMusik(query) ['link'])
```

Tampilan ketika memainkan lagu secara langsung :

```
-->8
Masukkan lagu yang ingin diputar ==>one of the girls (sped up)
[=====]
```

Menampilkan tentang program dan FAQ

```
elif opsi == 9:
    time.sleep(1)
    print_bertahap(r"#####")
    ...

    """ , delay = 0.000000000001)
    time.sleep(0.5)
    print_bertahap("\nHalooooooo!!! ", delay =0.05)

    ...

    print_bertahap("\nKembali ke halaman utama", delay = 0.05);
time.sleep(0.5)
    print_bertahap(".", delay = 1); time.sleep(1.5)
    print_bertahap(".", delay = 1); time.sleep(1.5)
    print_bertahap(".", delay = 1); time.sleep(1.5)
```

Tampilan ketika Menampilkan tentang program dan FAQ :

The screenshot shows a VS Code editor with a Python file named 'Tubes.py'. The code contains a large ASCII art of a cat's face. Below the ASCII art, there is a terminal window with the following output:

```

Halooooo!!! Aku Musi, dan kali ini aku mau menjelaskan tentang program ini

Pertama, program ini merupakan hasil dari sebuah tugas besar mata kuliah pengenalan komputasi tahun 2023 / 2024,
program ini dibuat menggunakan py

```

Keluar dari program

```
elif opsi == 10:
    exit()
```

Memutar lagu

```
elif opsi == 11:
    player.play()
```

Menjeda lagu

```
elif opsi == 12:
    player.pause()
```

Menghentikan lagu

```
elif opsi == 13:
    player.stop()
    Musik_kini = False
```

Skip lagu ke lagu selanjutnya

```
elif opsi == 14:
    nextSong()
```

Skip lagu ke lagu sebelumnya

```
elif opsi == 15:
    previousSong()
```

Tampilan untuk memutar, menjeda, menghentikan, dan menskip lagu:

```
Now playing.... Ardhito Pramono ft. Aurélie Moeremans - I Just Couldn't Save You Tonight (Story of Kale - OMPS)

  |-----|
  |||     |||
  |-----|
  |-----|
  |-----|
  |-----|

11 -play      14 -next
12 -pause    15 - previous
13 -stop

-->|
```

7. Membuat fungsi untuk sistem yang bekerja di latar belakang

```
...
def main():
    bacaFile()
    while True:
        print(clear)
        menu()
        Thr = threading.Thread(target=loadanimasi())
        Thr.start()

if __name__ == "__main__":
    main()
...
```

BAB IV

KESIMPULAN, PEMBELAJARAN, DAN PEMBAGIAN TUGAS

4.1 Kesimpulan

Spotify sebagai sebuah aplikasi pemutar musik daring adalah suatu sistem yang kompleks dan di dalamnya terdapat banyak hal yang terjadi. Aplikasi Spotify dibagi menjadi dua sisi, yakni *front-end* dan *back-end*. *Front-end* meliputi fungsi-fungsi tampilan, organisasi lagu, dan pemutaran lagi. Sedangkan, *back-end* merupakan sisi yang berkaitan dengan proses transfer data dari aplikasi ke server dan sebaliknya. Program pemutar musik daring ini dapat disimulasikan sebagai program yang lebih sederhana menggunakan bahasa python dengan mengimplementasikan fungsi-fungsi utamanya seperti pemutaran lagu, pembuatan playlist, pencarian lagu, dan beberapa fitur turunan lainnya. Simulator ini memanfaatkan database Youtube sebagai substitusi dan analogi server yang digunakan Spotify. Setiap fungsionalitas program pun ditulis dalam bentuk subprogram (fungsi/prosedur) yang selain membuat kode lebih rapi, tetapi juga menciptakan sifat *reusability* perintah-perintah yang sering digunakan sehingga memudahkan proses pemrograman.

4.2 Pembelajaran

Dalam mengerjakan tugas besar ini, kami belajar untuk berpikir secara analitis, komputasional, dan algoritmik untuk memahami cara kerja sistem yang kami eksplorasi. Sebuah sistem perangkat lunak yang kompleks harus dipecah-pecah menjadi submasalah yang lebih kecil agar dapat dipelajari cara kerjanya. Setiap fungsi kecil dalam pemutar musik daring juga memiliki algoritma tersendiri yang mengikuti serangkaian perintah terstruktur untuk beroperasi. Untuk mereplikasi dan mengimplementasikan fungsi tersebut dalam simulasi, kami harus berpikir secara algoritmik. Terakhir, kami juga sadar akan pentingnya kerjasama dan komunikasi demi kelancaran tugas ini. Setiap anggota kelompok memiliki kemampuan dan kekurangannya masing-masing. Hal itu harus kami komunikasikan agar bisa melakukan pembagian tugas yang optimal dan setiap tugas dapat dikerjakan dengan totalitas.

4.3 Pembagian Tugas

Nama	Tugas
Nayaka Ghana Subrata	Programming dan laporan Bab III
Dimas Anggiat	Deskripsi simulasi dan gambar flowchart (Bab II)
Julian Benedict	Eksplorasi (Bab I), PPT, dan video simulasi
Karol Yangqian Poetracahya	Dekomposisi (Bab I), tambahan deskripsi simulasi (Bab II), deskripsi flowchart (Bab II), Bab IV, editor laporan, dan PPT

DAFTAR REFERENSI

- Anonim. (2023). *IT Explained: Server*. Paessler.
<https://www.paessler.com/it-explained/server#:~:text=A%20server%20is%20a%20computer,as%20clients%2C%20over%20a%20network>.
- Pastukhov, Dmitry. (2022, 17 Maret). *How Music Streaming Works and The Popular Music Streaming Trends of Today*. Soundcharts.
<https://soundcharts.com/blog/how-music-streaming-works-trends#:~:text=Music%20Streaming%20Work%3F-,An%20Overview,seconds%20before%20playing%20a%20song>.
- Ripenko, Serhii dan Hasiuk, Nadiia. (2022, 29 November). *Music recommendation system: all you need to know*. Eliftech.
<https://www.eliftech.com/insights/all-you-need-to-know-about-a-music-recommendation-system-with-a-step-by-step-guide-to-creating-it/>
- Srivastava, Rushali. *How On-Demand Music Streaming App Works*. Idea Usher.
<https://ideausher.com/blog/music-streaming-app/how-on-demand-music-streaming-app-works/>
- Wallis, Jerry. (2023, 6 Juni). *What Is Apache Cassandra & Why Does Spotify Use It?* Intuji.
<https://intuji.com/what-is-apache-cassandra-why-spotify-uses-it/>