LAPORAN TUGAS BESAR

PEMROGRAMAN KOMPUTER

Sistem Informasi Restaurant Fli Food

Disusun Oleh:
Rafli Pratama H1101241008
Kelas : A

PROGRAM STUDI SISTEM INFORMASI
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS TANJUNGPURA
PONTIANAK
2024

KATA PENGANTAR

Puji dan syukur kehadirat Allah SWT, karena atas rahmat dan karunia-Nya, saya dapat menyelesaikan tugas besar ini yang berjudul "Sistem Informasi Restaurant Fli Food". Tugas ini disusun untuk memenuhi salah satu syarat penilaian pada mata kuliah Praktikum Pemrograman Komputer yang diampu oleh Bapak Ilhamsyah, S.Si., M.Cs.

Tugas besar ini dirancang untuk membantu pengelolaan informasi pada sebuah restoran, seperti pemesanan makanan, manajemen menu, dan pengelolaan data pelanggan. Dengan memanfaatkan ilmu yang telah diperoleh selama perkuliahan, saya berusaha untuk merancang aplikasi ini sebaik mungkin agar dapat memberikan solusi yang bermanfaat bagi operasional restoran.
Saya menyadari bahwa penyusunan laporan ini tidak akan berhasil tanpa adanya dukungan dari berbagai pihak. Oleh karena itu, saya mengucapkan terima kasih kepada:

1. Bapak Ilhamsyah, S.Si., M.Cs. selaku dosen pengampu mata kuliah yang telah memberikan bimbingan dan arahan selama proses penyelesaian tugas ini.
2. Rekan-rekan mahasiswa yang telah berbagi ide, kritik, dan saran yang membangun selama proses pengembangan sistem ini.
3. Semua pihak lain yang tidak dapat disebutkan satu per satu, yang telah membantu secara langsung maupun tidak langsung dalam menyelesaikan tugas besar ini.

Saya menyadari bahwa laporan ini masih jauh dari sempurna, baik dari segi isi maupun penyajiannya. Oleh karena itu, kritik dan saran yang membangun sangat saya harapkan untuk penyempurnaan tugas di masa yang akan datang.

Semoga laporan ini dapat memberikan manfaat bagi pembaca dan menjadi salah satu kontribusi kecil dalam pengembangan teknologi informasi, khususnya di bidang sistem informasi restoran.

Pontianak, 30 November 2024

Rafli Pratama

DAFTAR ISI

BAB I
PENDAHULUAN

1.1 Latar Belakang

Dalam industri kuliner, restoran merupakan salah satu sektor yang berkembang pesat dan menjadi pilihan utama masyarakat untuk memenuhi kebutuhan konsumsi sehari-hari. Namun, di balik pertumbuhan tersebut, pengelolaan restoran sering kali menghadapi berbagai tantangan, terutama dalam manajemen menu, pencatatan keuangan, pemantauan histori penjualan, serta pelayanan yang cepat kepada pelanggan. Di tengah persaingan yang semakin ketat, kecepatan dan efisiensi dalam memberikan layanan menjadi kunci utama dalam mempertahankan loyalitas pelanggan.

Pada umumnya, restoran yang masih menggunakan sistem manual dalam operasionalnya sering mengalami berbagai kendala. Misalnya, sulitnya mengelola menu secara terorganisir, mencatat transaksi dengan akurat, dan memastikan laporan penjualan tersusun dengan baik. Selain itu, pelanggan sering kali harus mengantri lama untuk memesan makanan, yang tidak hanya menurunkan tingkat kepuasan mereka tetapi juga berpotensi membuat mereka beralih ke tempat lain yang menawarkan layanan lebih cepat. Kondisi ini menunjukkan perlunya sebuah sistem informasi yang dapat memberikan solusi untuk mengatasi berbagai kendala tersebut.

Untuk mengatasi permasalahan tersebut, dirancanglah aplikasi **Sistem Informasi Restaurant Fli Food** sebagai solusi modern untuk membantu pengelolaan restoran. Aplikasi ini dirancang khusus untuk diakses secara lokal melalui perangkat yang terdapat di restoran, sehingga penggunaannya lebih

terfokus pada kebutuhan internal restoran. Dengan sistem ini, pengelola dapat mengelola menu, mencatat transaksi, dan memantau histori penjualan secara terorganisir dan efisien tanpa ketergantungan pada koneksi internet atau server eksternal.

Selain memberikan kemudahan bagi pengelola, aplikasi ini juga dirancang untuk meningkatkan pengalaman pelanggan. Proses pemesanan menjadi lebih cepat dan aman, sehingga pelanggan tidak perlu lagi mengantri lama. Salah satu fitur unggulan aplikasi ini adalah penggunaan nomor telepon pelanggan sebagai username. Fitur ini memungkinkan pelanggan mengumpulkan poin yang dapat ditukarkan, yang tidak hanya meningkatkan kepuasan tetapi juga membangun loyalitas pelanggan terhadap restoran.

Dengan adanya aplikasi Sistem Informasi Restaurant Fli Food, diharapkan restoran dapat menghadapi tantangan operasional dengan lebih baik, meningkatkan produktivitas, dan memberikan layanan terbaik kepada pelanggan. Sistem ini diharapkan tidak hanya menjadi alat bantu teknis tetapi juga solusi strategis yang mendukung keberlanjutan bisnis restoran di era digital.

1.2 Tujuan

Aplikasi Sistem Informasi Restaurant Fli Food dibuat dengan tujuan utama untuk mengoptimalkan pengelolaan sistem informasi di sebuah restoran. Adapun tujuan spesifik dari pengembangan aplikasi ini adalah sebagai berikut:

1. Meningkatkan Kecepatan Pengelolaan

   Sistem ini dirancang untuk mempercepat pengelolaan informasi, seperti pemesanan makanan, manajemen menu, pencatatan transaksi, dan histori penjualan, sehingga operasional restoran menjadi lebih efisien.

2. Meningkatkan Akurasi Pengelolaan Data

   Dengan sistem digital, aplikasi ini mampu menghilangkan risiko kesalahan manusia (human error) dalam mencatat pesanan, menghitung transaksi, atau membuat laporan keuangan.

3. Mencegah Antrian Pelanggan

   Aplikasi ini memberikan solusi bagi pelanggan untuk memesan makanan dengan lebih cepat, baik melalui sistem online maupun di restoran langsung, sehingga mereka tidak perlu mengantri lama untuk memesan makanan.

4. Mempermudah Pelanggan Memilih Menu dan Harga

   Dengan tampilan yang intuitif, aplikasi ini membantu pelanggan untuk dengan mudah memilih makanan atau minuman yang diinginkan, lengkap dengan informasi harga dan deskripsi menu, sehingga mereka dapat membuat keputusan dengan lebih cepat.

5. Meningkatkan Pengalaman Pengguna

   Aplikasi ini dirancang untuk memberikan pengalaman terbaik, baik untuk pihak restoran maupun pelanggan, dengan memberikan kemudahan dalam setiap proses mulai dari pemesanan hingga pembayaran.

1.3 Batasan Masalah

Dalam pengembangan aplikasi Sistem Informasi Restaurant Fli Food, terdapat batasan yang diterapkan untuk memastikan sistem yang dikembangkan

fokus pada kebutuhan utama restoran saat ini. Batasan-batasan ini juga dirancang agar aplikasi dapat berjalan optimal sesuai dengan kapasitas dan lingkup penggunaannya. Berikut adalah batasan masalah yang diterapkan:

1. Cakupan Fungsi Aplikasi

   Aplikasi ini mencakup beberapa fungsi utama yang dirancang untuk meningkatkan efisiensi operasional restoran, yaitu:

   - Pengelolaan Menu:
     Memungkinkan admin untuk menambahkan, mengubah, atau menghapus menu makanan dan minuman yang tersedia di restoran.
   - Pencatatan Transaksi:
     Setiap pesanan pelanggan akan tercatat secara otomatis dalam sistem untuk meminimalkan kesalahan manual.
   - Laporan Penjualan:
     Sistem menyediakan laporan penjualan yang dapat diakses secara real-time oleh admin atau manajer restoran.
   - Sistem Poin Pelanggan:
   - Pelanggan dapat mengumpulkan poin berdasarkan transaksi mereka, yang nantinya dapat ditukarkan dengan manfaat tertentu. Namun, aplikasi ini belum mencakup pengelolaan bahan baku restoran, seperti pengelolaan stok dapur atau kebutuhan pembelian bahan. Fungsi ini direncanakan akan ditambahkan pada pengembangan aplikasi di masa mendatang.

2. Akses dan Lingkungan Sistem

   Untuk menjaga kecepatan dan responsivitas, aplikasi ini hanya dapat diakses secara lokal melalui perangkat yang terhubung dalam jaringan internal restoran.

- Penggunaan Lokal: Sistem dirancang untuk diakses secara eksklusif di perangkat restoran, baik oleh admin maupun pelanggan.
- Tanpa Akses Global: Aplikasi belum mendukung akses melalui jaringan internet eksternal, sehingga pelanggan hanya dapat menggunakan sistem ketika berada di restoran.
- Real-Time: Seluruh data, termasuk pesanan dan laporan, diproses secara real-time di lingkungan lokal untuk memastikan kecepatan dan keakuratan operasional.

3. Pengguna Sistem

Aplikasi ini dirancang untuk digunakan oleh dua jenis pengguna utama:

- Pelanggan:
  - Pelanggan dapat mengakses aplikasi melalui perangkat seluler yang disediakan di meja makan.
  - Aplikasi mempermudah pelanggan untuk memilih menu makanan dan minuman, melihat harga, serta langsung melakukan pemesanan tanpa perlu menunggu pelayan.
- Admin atau Manajer:
  - Admin bertanggung jawab atas pengelolaan menu, pencatatan transaksi, serta sistem poin pelanggan.
  - Manajer memiliki akses untuk memantau laporan penjualan dan melakukan analisis data penjualan untuk mendukung pengambilan keputusan bisnis.

4. Keterbatasan Fitur Saat Ini

Meskipun aplikasi ini sudah mencakup berbagai fungsi penting, ada beberapa fitur yang belum tersedia, yaitu:

- Manajemen Stok Bahan Baku: Sistem belum dapat mencatat dan memantau stok bahan makanan yang ada di dapur restoran.

- Integrasi dengan Sistem Eksternal: Belum ada integrasi dengan sistem pembayaran pihak ketiga atau layanan pengantaran makanan.
- Akses Pelanggan dari Luar Restoran: Pelanggan belum dapat melakukan pemesanan dari rumah atau lokasi lain di luar restoran.

5. Fokus Fitur Utama

   Pengembangan aplikasi ini berfokus pada dua fitur utama:
   - Pengelolaan Menu: Untuk memastikan restoran dapat dengan mudah memperbarui daftar menu yang tersedia.
   - Sistem Poin Pelanggan: Fitur ini bertujuan untuk meningkatkan loyalitas pelanggan melalui program pengumpulan dan penukaran poin yang sederhana dan efektif.

6. Rencana Pengembangan Masa Depan

   Dalam pengembangan berikutnya, aplikasi ini direncanakan akan mencakup:
   - Akses Global: Pelanggan dapat memesan makanan melalui aplikasi secara online dari rumah atau lokasi lainnya, baik untuk pemesanan makan di tempat maupun layanan antar.
   - Manajemen Bahan Baku: Sistem akan diperluas untuk mencatat dan memantau stok bahan baku secara otomatis.
   - Integrasi Pembayaran Digital: Akan ditambahkan opsi pembayaran digital untuk memudahkan pelanggan dalam menyelesaikan transaksi.
7. Batasan Teknologi
   - Aplikasi ini hanya dapat digunakan pada perangkat yang kompatibel dan telah disediakan oleh restoran. Sistem tidak

mendukung perangkat pelanggan pribadi di tahap pengembangan ini. Pengembangan berbasis lokal mengharuskan perangkat terhubung ke jaringan internal restoran agar dapat mengakses

BAB II
LANDASAN TEORI

2.1 Pengenalan Bahasa Pemrograman Python

Python adalah bahasa pemrograman tingkat tinggi yang dirancang untuk kemudahan penggunaan dan keterbacaan kode. Diciptakan oleh Guido van Rossum pada akhir 1980-an, Python pertama kali dirilis pada tahun 1991. Bahasa ini mengutamakan sintaks yang jelas dan sederhana, membuatnya ideal bagi pemula maupun pengembang berpengalaman. Python mendukung berbagai paradigma pemrograman, termasuk berorientasi objek, imperatif, dan fungsional, sehingga sangat fleksibel untuk berbagai aplikasi, mulai dari pengembangan web hingga analisis data dan machine learning

Salah satu keunggulan Python adalah perpustakaan standar yang kaya, yang menyediakan banyak fungsi siap pakai untuk memudahkan pengembangan aplikasi. Dengan lebih dari 380.000 paket yang tersedia di Python Package Index (PyPI), pengembang dapat dengan cepat menemukan alat yang diperlukan untuk menyelesaikan berbagai tugas

Python juga dikenal karena kemampuannya dalam otomasi, memungkinkan pengguna untuk menulis skrip untuk mengotomatiskan tugas-tugas sehari-hari dengan efisien.Sejak diluncurkan, Python telah mengalami banyak pembaruan dan peningkatan. Versi 2.0 dirilis pada tahun 2000 dengan fitur baru seperti pengumpul sampah dan dukungan Unicode, sedangkan versi 3.0, yang dirilis pada tahun 2008, membawa banyak perubahan signifikan yang tidak sepenuhnya kompatibel dengan versi sebelumnya

2.2 Fungsi-Fungsi dalam Python

Fungsi dalam Python adalah kumpulan perintah yang dikelompokkan menjadi satu kesatuan untuk dapat dipanggil dan digunakan berulang kali. Penggunaan fungsi sangat penting dalam pemrograman karena memungkinkan pengembang untuk memecah program besar menjadi bagian-bagian kecil yang lebih terstruktur dan mudah dikelola. Dengan fungsi, kode menjadi lebih reusable dan efisien, serta meningkatkan keterbacaan program secara keseluruhan.

2.2.1 Sintaks Fungsi

Sintaks umum untuk mendefinisikan fungsi dalam Python adalah sebagai berikut:

```
def <nama_fungsi>(parameters):
```

- def: Menandakan bahwa blok kode berikutnya adalah sebuah fungsi.
- Nama fungsi : Nama yang diberikan untuk fungsi tersebut, yang digunakan saat memanggilnya.
- Parameters: Parameter adalah nilai yang dapat diterima oleh fungsi (opsional)..

Berikut adalah contoh sederhana dari fungsi dalam Python:

```
def perkenalan(nama):
    print(f"Halo {nama}, selamat datang!")

perkenalan("Rafli")
```

Output dari kode di atas adalah:

```
```
Halo Rena, selamat datang!
```
```

Fungsi ini menerima satu parameter, yaitu `nama`, dan mencetak pesan sambutan.

2.2.2 Manfaat Penggunaan Fungsi

a. Modularitas : Memecah program menjadi bagian-bagian kecil yang lebih mudah dikelola.
b. Reusability: Fungsi dapat digunakan kembali di berbagai tempat dalam program tanpa perlu menulis ulang kode.
c. Keterbacaan: Kode menjadi lebih terstruktur dan mudah dipahami.

2.2.3 Jenis-Jenis Fungsi

Dalam Python, terdapat beberapa jenis fungsi, antara lain:

a. Fungsi Built-in : Fungsi yang sudah tersedia dalam Python, seperti `print()`, `len()`, dan `type()`.
b. Fungsi Pengguna (User-defined) : Fungsi yang dibuat oleh pengguna sesuai kebutuhan.
c. Fungsi Lambda : Fungsi anonim yang didefinisikan menggunakan kata kunci `lambda`.

d. Fungsi juga dapat memiliki parameter wajib dan opsional, serta dapat mengembalikan nilai menggunakan pernyataan `return`. Ini memungkinkan fleksibilitas dalam bagaimana fungsi beroperasi dan berinteraksi dengan bagian lain dari program.

2.3 Pengolahan File di Python

Pengolahan file di Python adalah proses yang memungkinkan pengguna untuk melakukan berbagai operasi pada file, termasuk membuka, membaca, menulis, dan menutup file. Ini adalah aspek penting dalam pengembangan aplikasi karena memungkinkan penyimpanan data secara permanen. Python menyediakan

sejumlah fungsi built-in yang memudahkan pengelolaan file tanpa perlu mengimpor modul eksternal.

2.3.1 Fungsi Utama dalam Pengolahan File

1. **open()**: Fungsi ini digunakan untuk membuka file. Sintaks dasar untuk membuka file adalah:

```
file = open("nama_file.txt", "mode")
```

Di mana mode dapat berupa:

- "r": Membaca (default). Menghasilkan kesalahan jika file tidak ada.
- "w": Menulis. Membuat file baru atau menimpa file yang sudah ada.
- "a": Menambahkan. Menambah konten ke akhir file yang ada, atau membuat file baru jika tidak ada.
- "x": Membuat. Menghasilkan kesalahan jika file sudah ada.

2. **read()**: Membaca seluruh konten dari file.
3. **readline()**: Membaca satu baris dari file.
4. **readlines()**: Membaca semua baris dalam bentuk list.
5. **write()**: Menulis string ke dalam file.
6. **writelines()**: Menulis beberapa string ke dalam file sekaligus.
7. **close()**: Menutup file setelah selesai digunakan untuk membebaskan sumber daya.

Contoh Penggunaan

Berikut adalah contoh sederhana yang menunjukkan bagaimana cara membuka, menulis, dan membaca dari sebuah file:

```
# Membuka dan menulis ke dalam file
with open("contoh.txt", "w") as f:
```

```
    f.write("Hello, World!\n")
    f.write("Ini adalah contoh pengolahan file di
Python.\n")

# Membuka dan membaca dari file
with open("contoh.txt", "r") as f:
    content = f.read()
    print(content)
```

Dalam contoh di atas, with digunakan untuk memastikan bahwa file ditutup secara otomatis setelah blok kode selesai dieksekusi.

2.3.2 Keuntungan Pengolahan File di Python

- **User-Friendly**: Python menyediakan sintaks yang sederhana dan mudah dipahami untuk pengolahan file.
- **Fleksibilitas**: Dapat bekerja dengan berbagai jenis file, baik teks maupun biner.
- **Cross-Platform**: Dapat dijalankan di berbagai sistem operasi seperti Windows, Linux, dan macOS.

2.4 Manipulasi Data dengan List dan String

Manipulasi data dengan list dan string di Python adalah aspek penting yang memungkinkan pengembang untuk menyimpan, mengelola, dan memanipulasi koleksi data secara efisien. List adalah struktur data yang dapat menyimpan beberapa item dalam satu variabel dan dapat berisi berbagai jenis data, termasuk string, angka, dan objek lainnya.

2.4.1 List di Python

List di Python ditandai dengan penggunaan tanda kurung siku `[]` dan bersifat mutable, artinya isi list dapat diubah setelah dibuat. Beberapa karakteristik utama dari list adalah:

- Ordered: Elemen dalam list memiliki urutan tertentu yang tidak berubah.
- Changeable: Elemen dalam list dapat ditambahkan, dihapus, atau dimodifikasi.
- Allow Duplicates: List dapat berisi elemen dengan nilai yang sama.

Contoh pembuatan list:

```
my_list = ["apple", "banana", "cherry"]
```

2.4.2. Operasi Dasar pada List

Python menyediakan berbagai metode untuk memanipulasi list. Berikut adalah beberapa operasi dasar yang sering digunakan:

a. Menambahkan Elemen:
   - append(): Menambahkan elemen ke akhir list.
   - insert(index, element): Menyisipkan elemen pada posisi tertentu.

```
my_list.append("orange")
my_list.insert(1, "kiwi")
```

b. Menghapus Elemen:
   - remove(element): Menghapus elemen pertama yang ditemukan.
   - pop(index): Menghapus elemen pada indeks tertentu dan mengembalikannya.

```
my_list.remove("banana")
last_item = my_list.pop()     # Menghapus dan
mengembalikan elemen terakhir
```

c. Mencari Elemen:

- index(element): Mengembalikan indeks pertama dari elemen yang dicari.
- count(element): Menghitung jumlah kemunculan elemen dalam list.

Contoh:

```
index_of_cherry = my_list.index("cherry")
count_of_apples = my_list.count("apple")
```

d. Mengurutkan dan Membalikkan:

- sort(): Mengurutkan elemen dalam list secara ascending.
- reverse(): Membalik urutan elemen dalam list.

Contoh:

```
my_list.sort()
my_list.reverse()
```

2.4.3 Manipulasi String

String di Python juga dapat dimanipulasi dengan berbagai metode. Meskipun string bersifat immutable (tidak dapat diubah), ada banyak fungsi yang memungkinkan pengembang untuk menghasilkan string baru berdasarkan string yang ada. Beberapa metode umum untuk manipulasi string meliputi:

- split(): Memecah string menjadi list berdasarkan pemisah tertentu.

- join(): Menggabungkan elemen dalam list menjadi satu string dengan pemisah tertentu.
- replace(): Mengganti bagian dari string dengan substring lain.

Contoh manipulasi string:

```
text = "Hello, World!"
words = text.split(", ")  # ['Hello', 'World!']
new_text = " ".join(words)  # 'Hello World!'
replaced_text = text.replace("World", "Python")    #
'Hello, Python!'
```

2.5 Penggunaan Percabangan (Conditional Statements)

2.5.1 Definisi Percabangan

Percabangan dalam dunia pemrograman adalah proses penentuan keputusan atau *conditional statement*. Idenya sederhana: instruksi komputer untuk melakukan suatu aksi X hanya jika kondisi Y terpenuhi. Alternatifnya, instruksi komputer untuk melakukan aksi A jika kondisi tidak terpenuhi

2.5.2 Struktur Percabangan

Struktur percabangan di Python menggunakan sintaks if, elif, dan else.

- **if**: Kondisi utama.
- **elif**: Kondisi kedua atau ketiga hingga ke-x.
- **else**: Kondisi terakhir jika semua kondisi sebelumnya tidak terpenuhi

Contoh Penggunaan Percabangan :

a. If Statement

Misalkan kita ingin mengetahu whether nilai siswa lebih dari atau sama dengan 90, maka kita dapat menggunakan percabangan seperti ini:

```
nilai_siswa = 85
if nilai_siswa >= 90:
    print("Predikat A")
else:
    print("Predikat B")
```

Hasil output dari kode di atas adalah:

```
Predikat B
```

b. If-Elif Statement

Untuk menentukan predikat berdasarkan nilai siswa yang lebih kompleks, kita dapat menggunakan if-elif seperti ini:

```
nilai_siswa = 92
if nilai_siswa >= 95:
    print("Predikat A+")
elif nilai_siswa >= 90:
    print("Predikat A")
else:
    print("Predikat B")
```

Hasil output dari kode di atas adalah:

```
Predikat A+
```

2.5.3 Kontrol Aliran Percabangan

Percabangan juga dapat digunakan dengan operator logika (AND, OR) untuk membuat kondisi yang lebih kompleks. Misalkan kita ingin mengetahui apakah nilai siswa lebih dari 80 dan absensinya kurang dari 20%:

```
absensi = 18; nilai_siswa = 82
if nilai_siswa > 80 and absensi < 20:
    print("Selamat!")
else:
    print("Coba lagi.")
```

Hasil output dari kode di atas adalah:

```
Selamat!
```

2.6 Perulangan (Looping) dalam Python

Perulangan (Looping) dalam Python adalah konsep dasar dalam pemrograman yang memungkinkan eksekusi blok kode secara berulang berdasarkan kondisi tertentu. Perulangan sangat berguna untuk mengelola dan memproses data dalam jumlah besar, serta untuk mengulangi tindakan yang sama tanpa perlu menulis ulang kode.

2.6.1 Jenis-jenis Perulangan di Python

Python memiliki dua jenis perulangan utama: **for loop** dan **while loop**.

a. For Loop

For loop digunakan untuk iterasi melalui urutan (seperti list, tuple, string, atau dictionary). Sintaks dasar untuk for loop adalah:

```
for item in sequence:
    # blok kode yang dieksekusi
```

Contoh penggunaan for loop untuk mencetak setiap elemen dalam list:

```
fruits = ["apple", "banana", "cherry"]
for fruit in fruits:
    print(fruit)
```

Output dari kode di atas adalah:

```
apple
banana
cherry
```

For loop juga dapat digunakan dengan fungsi `range()` untuk menghasilkan urutan angka. Misalnya:

```
for i in range(5):
    print(i)
```

Output:

```
0
1
2
3
4
```

b. While Loop

While loop digunakan untuk menjalankan blok kode selama kondisi tertentu bernilai benar. Sintaks dasar untuk while loop adalah:

```
while condition:
    # blok kode yang dieksekusi
```

Contoh penggunaan while loop:

```
count = 0
while count < 5:
    print(count)
    count += 1
```

Output:
```

```
0
```
```

```
1
2
3
4
```

c. Nested Loops

Nested loops adalah perulangan di dalam perulangan lain. Ini berguna ketika Anda perlu melakukan iterasi melalui elemen dalam struktur data yang lebih kompleks, seperti matriks atau tabel.

Contoh nested loops:

```
for i in range(3):
    for j in range(2):
        print(f"i={i}, j={j}")
```

Output:

```
i=0, j=0
i=0, j=1
i=1, j=0
i=1, j=1
i=2, j=0
i=2, j=1
```

2.6.2 Kontrol Aliran dalam Loop

Python juga menyediakan beberapa pernyataan kontrol aliran yang dapat digunakan dalam perulangan:

- break: Menghentikan loop sebelum mencapai akhir.
- continue: Melewatkan iterasi saat ini dan melanjutkan ke iterasi berikutnya.
- pass: Pernyataan kosong yang tidak melakukan apa-apa; sering digunakan sebagai placeholder.

a. Contoh penggunaan break dan continue:

```
for x in range(10):
    if x == 5:
        break  # Menghentikan loop jika x sama dengan
5
    print(x)

print("Loop selesai.")
```

Output:

```
0
1
2
3
4
Loop selesai.
```

b. Contoh penggunaan continue:

```
for x in range(10):
    if x % 2 == 0:
        continue  # Melewatkan angka genap
    print(x)
```

Output:

```
1
3
5
7
9
```

2.7 Modularitas dalam Kode

2.7.1 Definisi Modularitas

Modularitas adalah prinsip dalam pemrograman yang memungkinkan pengembang untuk memecah program besar menjadi modul-modul kecil yang independen. Modul adalah file Python yang berisi definisi fungsi,

kelas, dan variabel yang dapat digunakan kembali di berbagai bagian program atau bahkan di program lain. Dengan menggunakan modul, pengembang dapat mengatur kode dengan lebih baik, sehingga memudahkan pemahaman dan pemeliharaan kode tersebut.

2.7.2 Manfaat Modularitas

1. **Pengelolaan Kode yang Lebih Baik**: Kode yang dibagi menjadi modul-modul kecil lebih mudah dipahami dan dikelola. Pengembang dapat fokus pada bagian tertentu tanpa khawatir merusak bagian lain dari kode.
2. **Peningkatan Reusabilitas**: Fungsi dan kelas yang didefinisikan dalam modul dapat digunakan kembali di proyek lain, mengurangi duplikasi kode dan waktu pengembangan.
3. **Skalabilitas**: Dengan struktur modular, menambahkan atau menghapus fitur dari aplikasi menjadi lebih mudah. Pengembang dapat menambahkan modul baru tanpa harus mengubah banyak bagian dari kode yang sudah ada.
4. **Kolaborasi**: Dalam tim pengembangan, modularitas memungkinkan beberapa pengembang untuk bekerja pada bagian berbeda dari aplikasi secara bersamaan tanpa konflik.

2.7.3 Teknik Meningkatkan Modularitas

Beberapa teknik untuk meningkatkan modularitas dalam kode Python meliputi:

- **Membagi Kode Menjadi Modul**: Mengelompokkan fungsi-fungsi yang terkait dalam modul terpisah. Misalnya, modul untuk operasi database, parsing data, atau logika bisnis.

- **Menggunakan Fungsi dan Kelas**: Fungsi digunakan untuk menjalankan tugas tertentu, sedangkan kelas digunakan untuk membuat objek dengan data dan metode terkait.
- **Penggunaan Package Manager**: Alat seperti pip membantu dalam mengelola dependensi dan pustaka eksternal yang diperlukan oleh aplikasi.

Contoh Penerapan Modularitas

Berikut adalah contoh sederhana tentang bagaimana cara membuat dan menggunakan modul dalam Python:

a. **Membuat Modul**: Buat file bernama matematika.py dengan fungsi-fungsi berikut:

```
# matematika.py
def luas_persegi(sisi):
    return sisi * sisi

def luas_lingkaran(radius):
    return 3.14 * radius * radius
```

b. **Mengimpor Modul**: Di file utama Anda (misalnya main.py), Anda dapat mengimpor dan menggunakan fungsi dari modul tersebut:

```
# main.py
import matematika

luas_persegi = matematika.luas_persegi(5)
luas_lingkaran = matematika.luas_lingkaran(7)
```

```
print("Luas Persegi:", luas_persegi)
print("Luas Lingkaran:", luas_lingkaran)
```

Output dari program ini akan menunjukkan luas dari bentuk geometris yang dihitung menggunakan fungsi dari modul matematika.

BAB III
PEMBAHASAN

3.1 Rancangan Program

Sistem Manajemen Restoran Digital: Penjelasan Terintegrasi dan Komprehensif

Sistem manajemen restoran digital ini dirancang untuk mengelola operasional restoran secara efisien, baik untuk pengguna (pelanggan) maupun admin (pengelola restoran). Sistem ini menggunakan antarmuka berbasis konsol dengan file teks sebagai basis data untuk menyimpan informasi. Berikut adalah penjelasan terintegrasi dari seluruh fitur utama dalam sistem, dirangkum berdasarkan alur dan fungsionalitasnya.

1. Login: Pintu Masuk Utama Sistem

Fitur login adalah komponen awal dan paling esensial, memungkinkan pembagian akses antara pengguna (user) dan admin. Setiap jenis pengguna memiliki kredensial dan hak akses yang berbeda:

- **Admin**:
    - Login admin memerlukan kombinasi username dan password tertentu (admin dan admin#123).
    - Setelah login, admin diarahkan ke menu utama untuk mengelola berbagai aspek restoran, seperti menu, keuangan, dan pelanggan.
- **User (Pelanggan)**:
    - Login user memvalidasi kredensial menggunakan file akun_user.txt, memastikan pengguna yang terdaftar dapat mengakses fitur seperti memesan makanan, mengecek poin loyalitas, dan melakukan pembayaran.

Keamanan Login:

- Pengguna diberikan tiga kali kesempatan untuk memasukkan kredensial yang benar. Jika gagal, sistem mengembalikan pengguna ke menu utama.

2. Manajemen Menu Restoran: Fitur Eksklusif Admin

Admin bertanggung jawab untuk mengelola data menu restoran, yang terdiri dari tiga kategori utama: **Makanan**, **Minuman**, dan **Camilan**. Semua data disimpan dalam file teks terpisah untuk setiap kategori, memastikan fleksibilitas dan efisiensi dalam pengelolaan data.

2.1. Tambah Menu

- Admin dapat menambahkan item baru ke kategori yang sesuai.
- Data yang dimasukkan berupa nama menu dan harga. Informasi ini langsung disimpan ke file terkait (menu_makanan.txt, menu_minuman.txt, atau menu_camilan.txt).

2.2. Tampilkan Menu

- Sistem membaca data dari file dan menyajikan menu dalam format tabel rapi, lengkap dengan harga.
- Setiap kategori menu ditampilkan secara terpisah, misalnya:

Makanan:
- Nasi Goreng Spesial | Rp25,000
- Ayam Bakar | Rp30,000

Minuman:
- Es Teh | Rp5,000
- Kopi Susu | Rp15,000

2.3. Edit Menu

- Admin dapat mengubah nama atau harga menu.
- Sistem memvalidasi keberadaan menu yang dimaksud sebelum melakukan pembaruan.
- Setelah perubahan selesai, data file diperbarui untuk mencerminkan modifikasi.
-

2.4. Hapus Menu

- Admin dapat menghapus menu dari kategori tertentu.
- Sistem memvalidasi input, menghapus data dari daftar, dan memperbarui file terkait untuk menyimpan perubahan.

Fitur manajemen menu ini memastikan fleksibilitas dalam memperbarui data dan mendukung pengelolaan menu yang dinamis sesuai kebutuhan restoran.

3. Pemesanan dan Transaksi untuk Pelanggan

Pelanggan yang berhasil login dapat langsung memesan makanan dan minuman dari daftar menu yang tersedia. Fitur ini dirancang untuk memberikan pengalaman yang mudah dan nyaman, dengan langkah-langkah sebagai berikut:

3.1. Melihat Menu

- Pengguna dapat melihat daftar makanan, minuman, dan camilan yang tersedia. Data ditampilkan dalam format tabel untuk memudahkan navigasi.

3.2. Memesan Item

- Pengguna memilih item dari menu beserta jumlahnya.
- Sistem mencatat pesanan dalam struktur data sementara.

3.3. Penghitungan Total Harga

- Sistem menjumlahkan harga berdasarkan pesanan. Contoh:

Pesanan:
- Nasi Goreng x1 (Rp25,000)
- Es Teh x2 (Rp5,000)
Total: Rp35,000

3.4. Pembayaran

- Setelah konfirmasi pesanan, pelanggan membayar total biaya.
- Sistem mencatat transaksi ke file uangKas.txt, menambahkan jumlah pembayaran ke total uang kas restoran.

4. Sistem Poin Loyalitas Pelanggan

Fitur ini bertujuan untuk meningkatkan loyalitas pelanggan dengan memberikan reward berupa poin yang dapat ditukar dengan diskon atau promosi tertentu.

4.1. Pengumpulan Poin

- Poin diberikan berdasarkan nilai transaksi. Setiap pembelanjaan menghasilkan sejumlah poin yang tercatat dalam file poin_pelanggan.txt.
- Poin dihitung secara otomatis berdasarkan aturan tertentu (misalnya, setiap Rp10,000 menghasilkan 1 poin).
-

4.2. Mengecek Poin

- Pelanggan dapat memeriksa jumlah poin mereka melalui menu **Cek Poin**.
- Sistem membaca file poin_pelanggan.txt dan menampilkan poin yang terkait dengan akun pengguna.
-

4.3. Pemanfaatan Poin

- Poin dapat digunakan untuk mendapatkan diskon pada transaksi berikutnya.
- Sistem mengurangi jumlah poin secara otomatis setelah digunakan.

5. Manajemen Keuangan: Transparansi untuk Admin

Admin dapat memantau total uang kas restoran melalui fitur **Cek Uang Kas**, yang membaca data dari file uangKas.txt. Data keuangan ditampilkan dalam format yang sederhana namun informatif, seperti:

Total uang kas saat ini: Rp1,250,000

Fitur ini memberikan wawasan langsung tentang pendapatan restoran, memungkinkan admin untuk mengambil keputusan strategis terkait operasional.

6. Database Pelanggan

Admin dapat mengakses informasi pelanggan yang terdaftar, termasuk data akun dan poin loyalitas. Informasi ini disajikan dalam format tabel, misalnya:

Daftar Pelanggan:
- Username: johndoe | Poin: 120
- Username: janedoe | Poin: 200

Fitur ini membantu admin dalam memahami preferensi pelanggan, merancang promosi, atau mengelola program loyalitas secara efektif.

7. Log Transaksi: Rekaman Aktivitas Restoran

Semua transaksi dicatat dalam file log_transaksi.txt, termasuk detail berikut:

- **Waktu Transaksi**.
- **Username Pelanggan**.
- **Daftar Item yang Dibeli**.
- **Total Pembayaran**.

Contoh log:

Tanggal: 2024-11-30 | User: johndoe
Pesanan:
- Nasi Goreng x1
- Es Teh x2

Total: Rp35,000

Log ini memberikan transparansi penuh dan dapat digunakan untuk audit atau pelaporan.

8. Alur Terintegrasi Antara User dan Admin

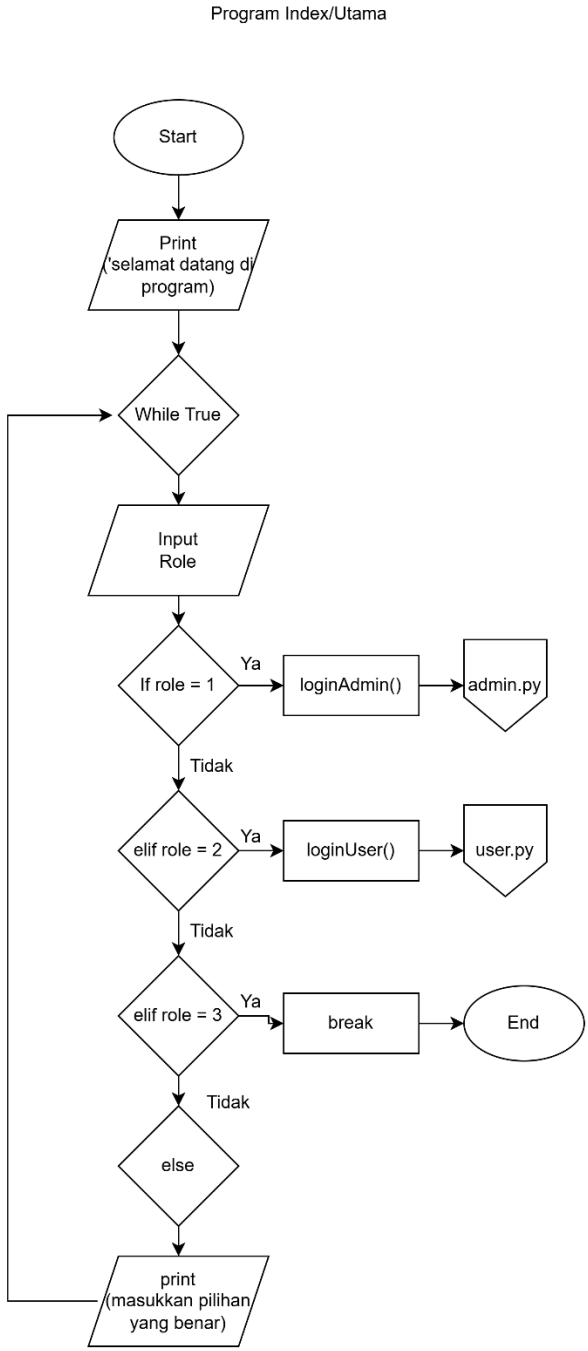
Sistem dirancang dengan alur kerja yang jelas, memastikan koordinasi antara pengguna dan admin:

- Admin mengelola menu dan keuangan.
- Pelanggan menggunakan data menu yang dikelola admin untuk memesan makanan.

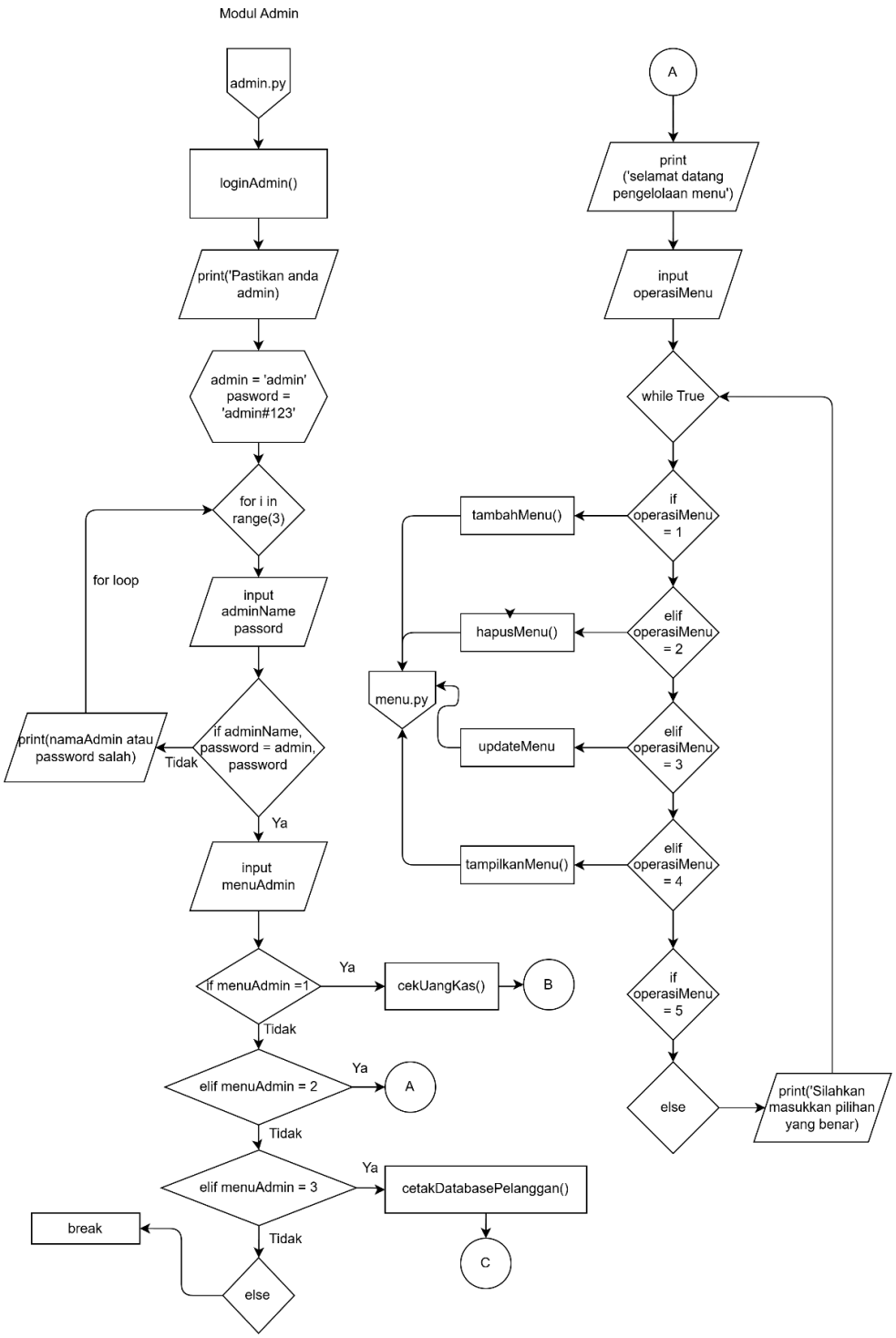
- Data transaksi dan poin pelanggan diperbarui secara otomatis, menciptakan hubungan saling terkait antara semua fitur.

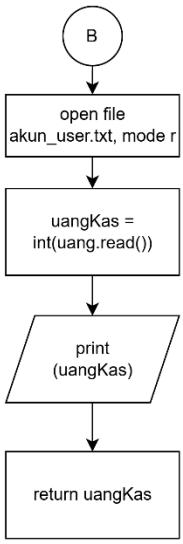
3.2 Flowchart

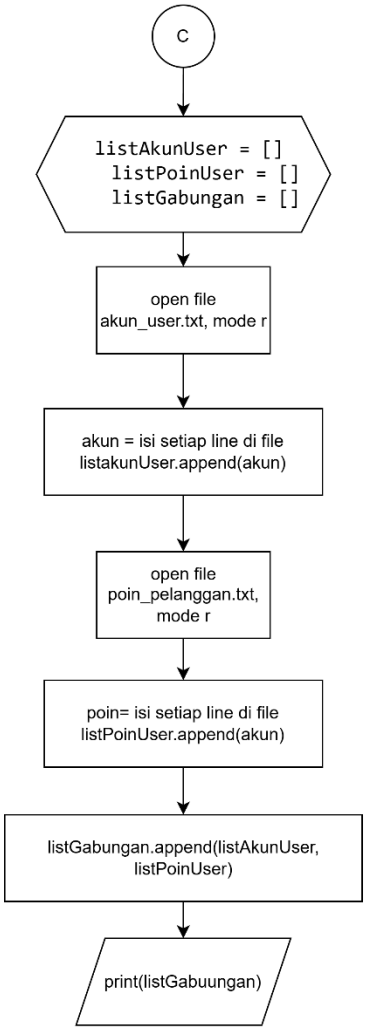
3.2.1 Flowchart index.py



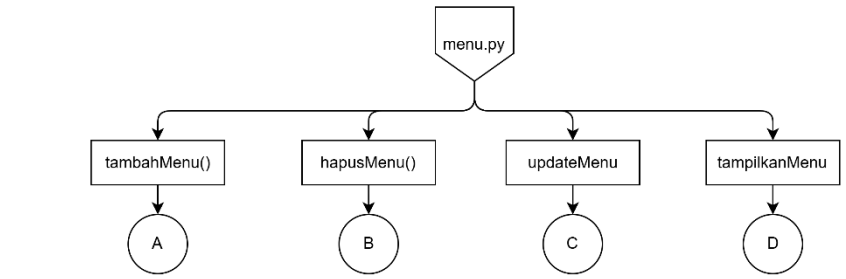
3.2.2 FlowChart modul admin.py

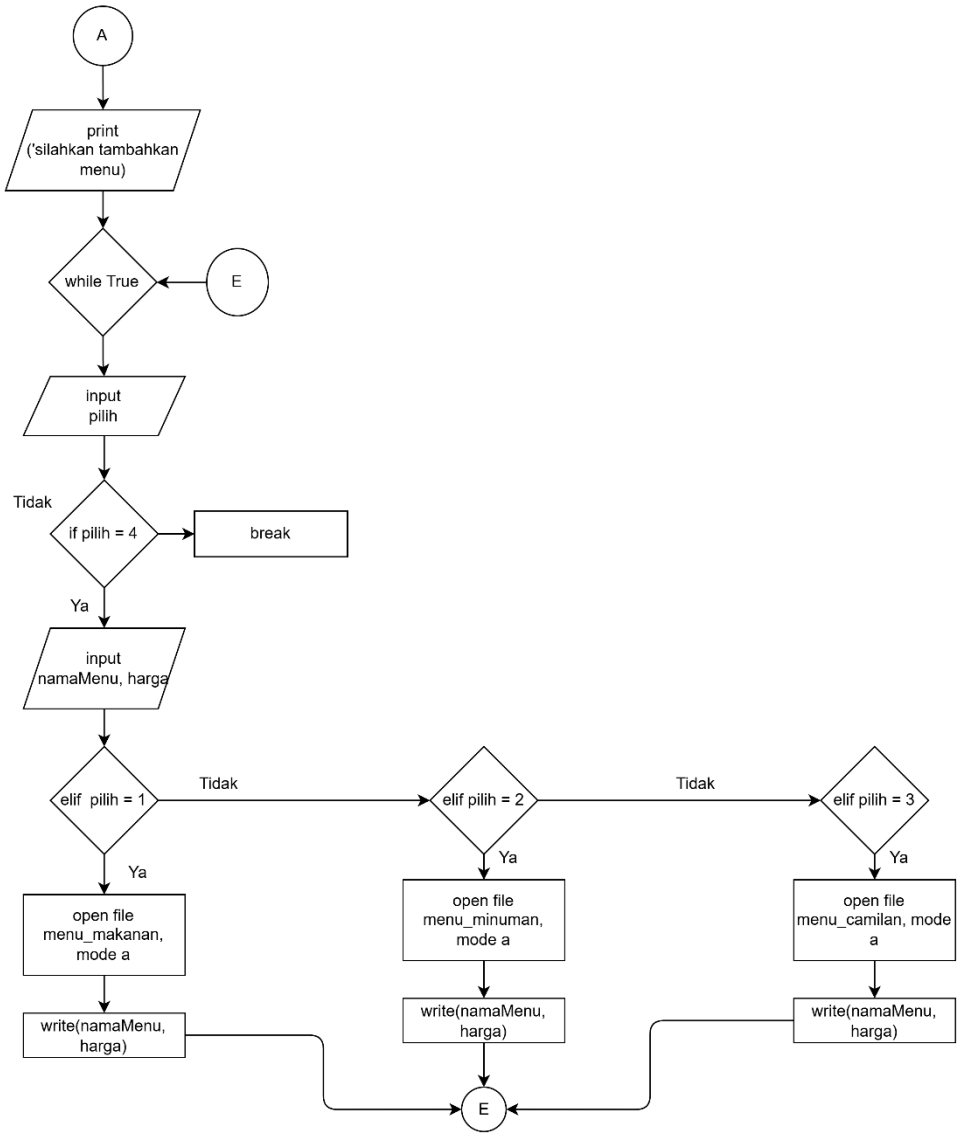


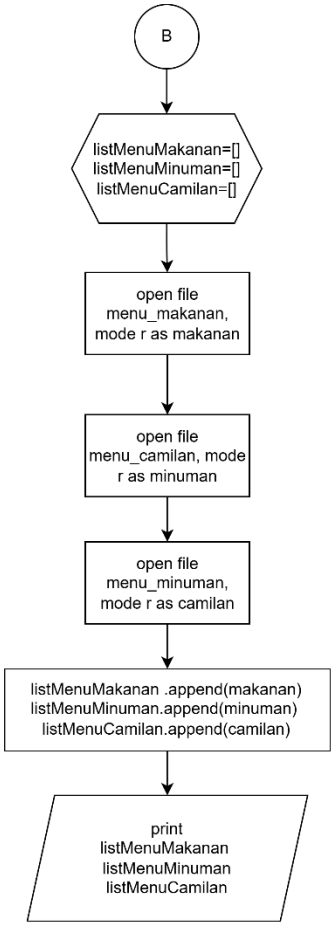

B
open file
akun_user.txt, mode r
uangKas =
int(uang.read())
print
(uangKas)
return uangKas

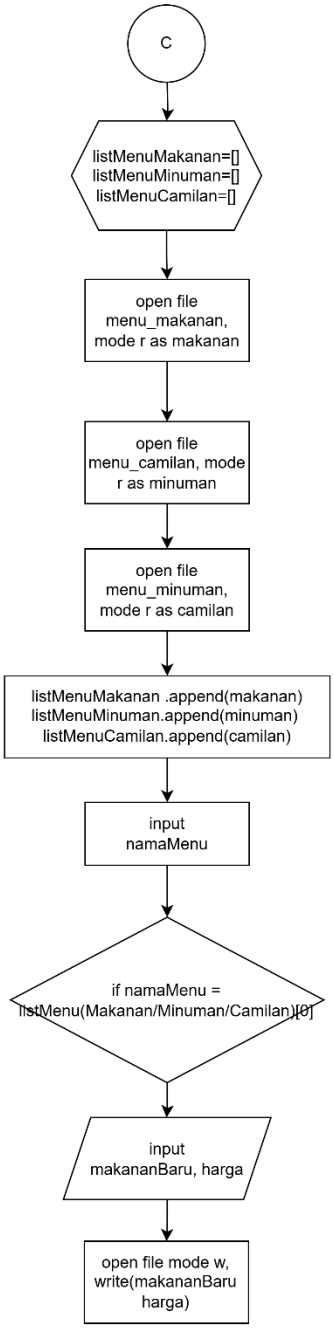

C
listAkunUser = []
listPoinUser = []
listGabungan = []
open file
akun_user.txt, mode r
akun = isi setiap line di file
listakunUser.append(akun)
open file
poin_pelanggan.txt,
mode r
poin= isi setiap line di file
listPoinUser.append(akun)
listGabungan.append(listAkunUser,
listPoinUser)
print(listGabuungan)

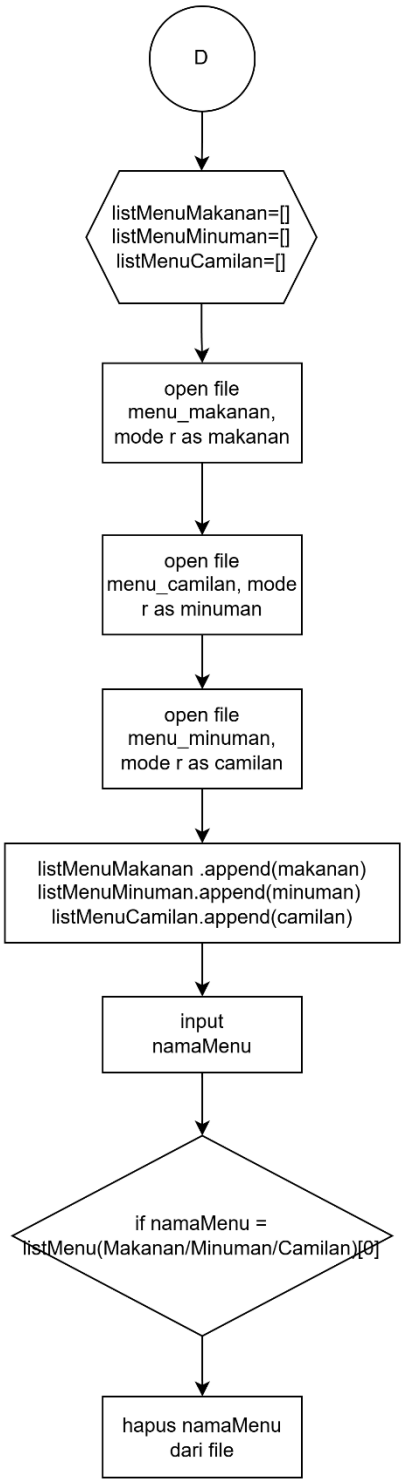
3.2.3 Flowchart Modul menu.py



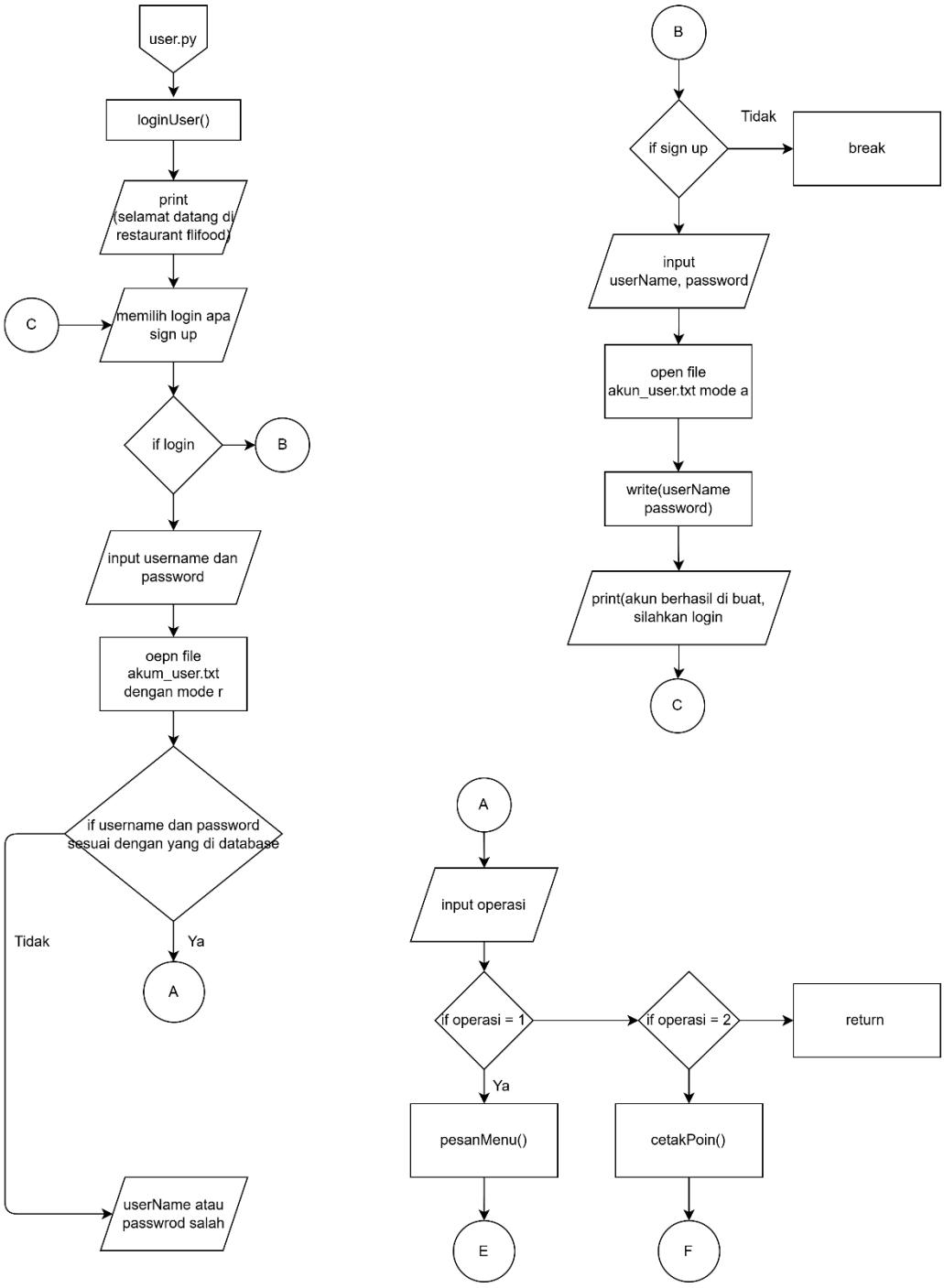


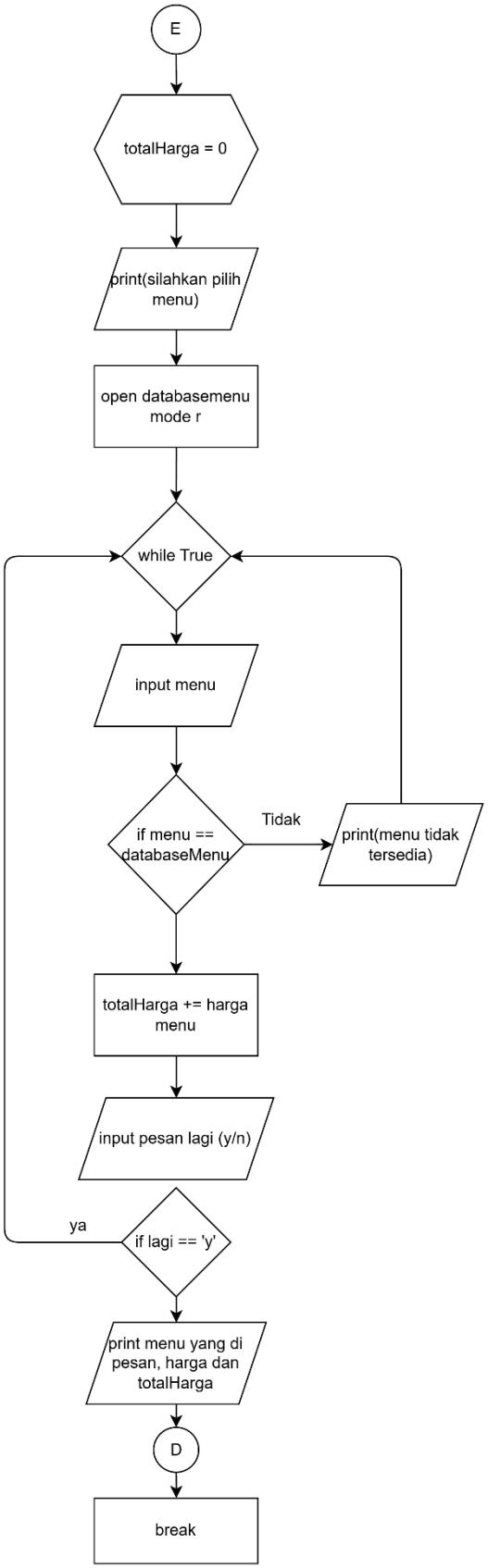

B
listMenuMakanan=[]
listMenuMinuman=[]
listMenuCamilan=[]
open file
menu_makanan,
mode r as makanan
open file
menu_camilan, mode
r as minuman
open file
menu_minuman,
mode r as camilan
listMenuMakanan .append(makanan)
listMenuMinuman.append(minuman)
listMenuCamilan.append(camilan)
print
listMenuMakanan
listMenuMinuman
listMenuCamilan

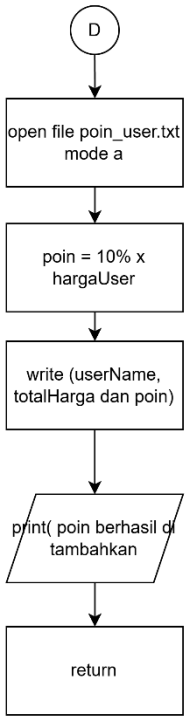

C
listMenuMakanan=[]
listMenuMinuman=[]
listMenuCamilan=[]
open file
menu_makanan,
mode r as makanan
open file
menu_camilan, mode
r as minuman
open file
menu_minuman,
mode r as camilan
listMenuMakanan .append(makanan)
listMenuMinuman.append(minuman)
listMenuCamilan.append(camilan)
input
namaMenu
if namaMenu =
listMenu(Makanan/Minuman/Camilan)[0]
input
makananBaru, harga
open file mode w,
write(makananBaru
harga)

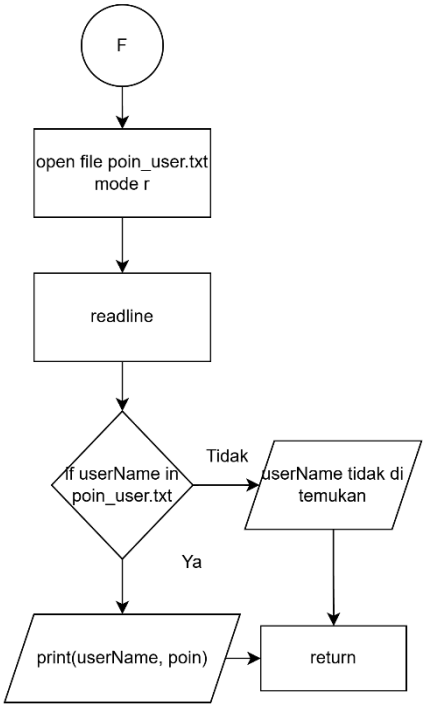

D
listMenuMakanan=[]
listMenuMinuman=[]
listMenuCamilan=[]
open file
menu_makanan,
mode r as makanan
open file
menu_camilan, mode
r as minuman
open file
menu_minuman,
mode r as camilan
listMenuMakanan .append(makanan)
listMenuMinuman.append(minuman)
listMenuCamilan.append(camilan)
input
namaMenu
if namaMenu =
listMenu(Makanan/Minuman/Camilan)[0]
hapus namaMenu
dari file

3.2.4 Flowchart modul user,py




E
totalHarga = 0
print(silahkan pilih menu)
open databasemenu mode r
while True
input menu
if menu == databaseMenu
Tidak
print(menu tidak tersedia)
totalHarga += harga menu
input pesan lagi (y/n)
ya
if lagi == 'y'
print menu yang di pesan, harga dan totalHarga
D
break

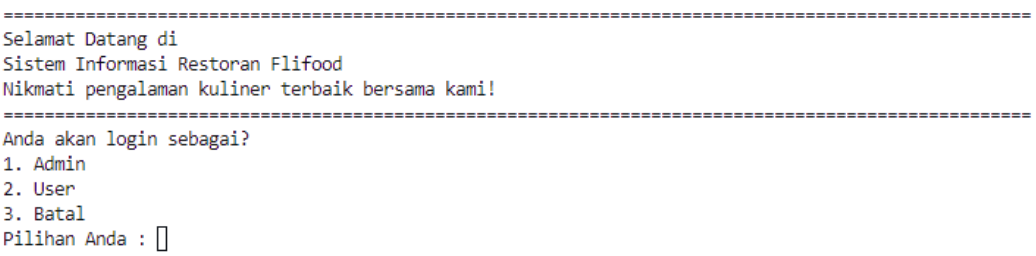

D
open file poin_user.txt mode a
poin = 10% x hargaUser
write (userName, totalHarga dan poin)
print(poin berhasil di tambahkan
return


F
open file poin_user.txt mode r
readline
if userName in poin_user.txt
Tidak
userName tidak di temukan
Ya
print(userName, poin)
return

3.3 Hasil

3.3.1 Tampilan program utama

Berikut adalah tampilan dari index.py



Gambar 1. Tampilan Index.py

Berikut adalah tampilan source code keseluruhan pada tampilan diatas :

Tabel 1. Source Code index.py

```
import admin
import user
import format.teksFormat as teksFormat

def pilihRole():
    teksFormat.dekorasi(100)
    print("Selamat Datang di ")
    print("Sistem Informasi Restoran Flifood")
    print("Nikmati pengalaman kuliner terbaik bersama kami!")
    teksFormat.dekorasi(100)

    while True:
        userRole = int(input('Anda akan login sebagai?\n1.
Admin\n2. User\n3. Batal\nPilihan Anda : '))
        if userRole == 1:
            admin.loginAdmin()
        elif userRole == 2:
            user.loginUser()
        elif userRole == 3:
            print('Operasi di batalkan oleh pengguna')
```

```
            break
        else:
            print('Silahkan masukkan pilihan yang benar')
pilihRole()
```

Penjelasan :

Fungsi pilihRole ini terdapat di modul index.py adalah pintu masuk utama ke dalam Sistem Informasi Restoran Flifood. Pertama-tama, kita mengimpor beberapa modul, yaitu admin untuk menangani proses login bagi Admin, user untuk login sebagai User, dan format.teksFormat untuk mempercantik tampilan antarmuka dengan dekorasi teks.

Di dalam fungsi, kita mulai dengan menambahkan garis dekorasi menggunakan teksFormat.dekorasi(100), dan menampilkan pesan selamat datang beserta deskripsi singkat tentang sistem ini. Setelah itu, program memasuki bagian inti, yaitu meminta pengguna memilih peran mereka melalui input angka. login sebagai Admin (1), login sebagai User (2), atau membatalkan operasi (3).

Untuk setiap pilihan, menggunakan logika if-elif-else. Jika pengguna memilih 1, maka sistem akan memanggil fungsi admin.loginAdmin() untuk memulai proses login Admin. Jika memilih 2, fungsi user.loginUser() akan dijalankan untuk login sebagai User. Pilihan ketiga memungkinkan pengguna membatalkan operasi, yang akan menampilkan pesan "Operasi dibatalkan oleh pengguna" dan keluar dari perulangan. Namun, jika input tidak valid, program akan menampilkan pesan kesalahan dan meminta pengguna untuk memasukkan pilihan ulang.

Saya juga menambahkan perulangan agar fungsi ini terus berjalan sampai pengguna memutuskan untuk keluar dengan memilih opsi 3 ini agar fungsi pilihRole dirancang untuk memberikan fleksibilitas dalam memilih peran

3.3.2 Tampilan Program Modul Admin

Berikut adalah tampilan dari modul admin.py :

```
1. Admin
2. User
3. Batal
Pilihan Anda : 1
Anda memilih menu admin, pastikan ini benar Anda!!!
Silahkan masukkan username admin Anda (admin): admin
Silahkan masukkan password (admin#123): admin#123

=============================================================================================================
Selamat datang di menu Admin
Silahkan pilih menu yang ingin Anda gunakan:
1. Cek uang Kas Restaurant
2. Kelola Menu Restaurant
3. Cek Database Pelanggan
4. Keluar
Pilihan Anda : 1
================================================
Total uang kas restaurant adalah Rp.118.022

Silahkan pilih menu berikut :
1. Tambah Menu
2. Hapus Menu
3. Edit Harga Menu
4. Tampilkan Menu
5. Keluar
Pilihan Anda : 

Pilihan Anda : 1
2. Hapus Menu
3. Edit Harga Menu
4. Tampilkan Menu
5. Keluar
Pilihan Anda : 1
Silahkan tambahkan nama menu dan harga
=============================================================================================================
Jenis menu apa yang ingin anda tambah :
1. Makanan
2. Minuman
3. Desert dan Cemilan
4. Simpan perubahan
Pilihan anda : 1
Masukkan nama menu : Nasi Abu
Masukkan harga menu : 6000
menu berhasil di tambahkan!
=============================================================================================================

Silahkan pilih menu berikut :
1. Tambah Menu
2. Hapus Menu
3. Edit Harga Menu
4. Tampilkan Menu
5. Keluar
Nasi Abu
Silahkan masukkan menu yang ingin di hapus : Nasi Abu
['Nasi Abu', '6000'] Telah di hapus dari daftar menu.
```

```
Kategori menu yang akan di edit :
1. Makanan
2. Minuman
3. Camilan
4. Simpan dan Keluar
2
Daftar Menu :
Es susu
Es Jeruk
Es Doger
Silahkan masukkan menu yang ingin anda edit : Es susu
Silahkan masukkan nama menu yang baru : Es Limau
Silahkan masukkan harga menu : 8000
menu ['Es Limau', '8000'] telah di edit menjadi [Es Limau,8000]
```

```
Silahkan pilih menu berikut :
1. Tambah Menu
2. Hapus Menu
3. Edit Harga Menu
4. Tampilkan Menu
5. Keluar
Pilihan Anda : 4
```

Menu	Harga
Makanan	
Nasi Spesial	40000
Nasi Bakar	17000
Nasi Padang	13000
Minuman	
Es Limau	8000
Es Jeruk	9000
Es Doger	8000
Camilan	
Burger Spesial	15000
Potato Roll	8000

Gambar 2. Tampilan admin.py

Berikut adalah tampilan source code keseluruhan pada tampilan diatas :

Tabel 2. Tampilan dari modul admin.py

```
import format.teksFormat as teksFormat
from menu import menu

def cekUangKas ():
    with open('Database/penjualan/uangKas.txt', 'r') as uang:
        uangKas = int(uang.read())
    print(f'Total uang kas restaurant adalah
Rp.{teksFormat.format_uang(uangKas)}')
    return uangKas

def cetakDatabasePelanggan():
    listAkunUser = []
```

```
    listPoinUser = []
    listGabungan = []
    with open('Database/databaseAkun/akun_user.txt', 'r') as file:
        for lines in file:
            akun = lines.split()
            listAkunUser.append(akun)

    with open('Database/databaseAkun/poin_pelanggan.txt', 'r') as
file:
        for lines in file:
            poin = lines.split()
            listPoinUser.append(poin[-1])

    for i in range(len(listAkunUser)):
        akun = listAkunUser[i]
        poin = listPoinUser[i]
        akunPoin = akun + [poin]
        listGabungan.append(akunPoin)

    print('Berikut adalah daftar Akun User : ')
    teksFormat.dekorasiGaris(50)
    for listAkun in listGabungan:
        print(f'UserName : {listAkun[0]}\nPassword :
{listAkun[1]}\nTotal Poin : {listAkun[2]}\n')

def loginAdmin():
    print('Anda memilih menu admin, pastikan ini benar Anda!!!')
    admin = 'admin'
    password = 'admin#123'

    for attempt in range(3):
        adminName = input('Silahkan masukkan username admin Anda
(admin): ')
        passwordAdmin = input('Silahkan masukkan password
(admin#123): ')

        if adminName == admin and passwordAdmin == password:
            print()
            teksFormat.dekorasi(100)
            print('Selamat datang di menu Admin')
            while True:
                menuAdmin = int(input('Silahkan pilih menu yang
ingin Anda gunakan:\n1. Cek uang Kas Restaurant\n2. Kelola Menu
Restaurant\n3. Cek Database Pelanggan\n4. Keluar\nPilihan Anda :
'))
```

```
                if menuAdmin == 1:
                    teksFormat.dekorasi(50)
                    cekUangKas()

                elif menuAdmin == 2:
                    teksFormat.dekorasi(50)
                    print('Selamat datang di sistem pengelolaan
menu!')
                    while True:
                        teksFormat.dekorasiGaris(50)
                        operasiMenu = int(input('Silahkan pilih
menu berikut :\n1. Tambah Menu\n2. Hapus Menu\n3. Edit Harga
Menu\n4. Tampilkan Menu\n5. Keluar\nPilihan Anda : '))
                        if operasiMenu == 1:
                            menu.tambahMenu()
                        elif operasiMenu == 2:
                            menu.hapusMenu()
                        elif operasiMenu == 3:
                            menu.updateMenu()
                        elif operasiMenu == 4:
                            menu.tampilkanMenu()
                        elif operasiMenu == 5:
                            break
                elif menuAdmin == 3:
                    cetakDatabasePelanggan()
                elif menuAdmin == 4:
                    print('Anda memilih keluar, terima kasih')
                    break
                else:
                    print('Silahkan masukkan pilihan yang
benar!!!')
                return

        else:
            print('Username atau password tidak valid.')
            if attempt < 2:
                print(f'Anda memiliki {2 - attempt} percobaan
lagi.')

    print('Anda telah gagal login sebanyak 3 kali. Kembali ke menu
utama.')
```

Penjelasan :

Kode ini adalah implementasi fitur menu admin untuk Sistem Informasi Restoran. Program ini tersusun dari beberapa fungsi inti yang digunakan dalam pengelolaan data, termasuk pengecekan uang kas restoran, pengelolaan database pelanggan, serta autentikasi login admin.

Fungsi pertama adalah cekUangKas, yang membaca data dari file teks Database/penjualan/uangKas.txt untuk mengetahui total uang kas restoran. Data ini diformat dengan bantuan fungsi teksFormat.format_uang untuk menghasilkan tampilan yang rapi sebelum ditampilkan ke pengguna. Fungsi ini juga mengembalikan nilai uang kas sebagai output.

Fungsi berikutnya adalah cetakDatabasePelanggan, yang buat untuk menampilkan daftar akun pelanggan beserta poin mereka. Data ini diambil dari dua file: Database/databaseAkun/akun_user.txt yang berisi daftar akun pengguna, dan Database/databaseAkun/poin_pelanggan.txt yang memuat poin pelanggan. Kedua data ini digabungkan menjadi sebuah daftar gabungan yang menampilkan username, password, dan total poin pelanggan secara bersamaan. Fungsi ini juga menggunakan teksFormat.dekorasiGaris untuk mempercantik tampilan.

Fungsi utama di modul ini adalah loginAdmin, yang menangani proses login dan menu admin. Untuk memastikan hanya admin yang dapat mengakses menu ini, autentikasi dilakukan menggunakan username dan password tetap, yaitu admin dan admin#123. Pengguna diberi hingga tiga kesempatan untuk memasukkan kredensial yang benar. Jika gagal tiga kali, akses ditolak, dan pengguna diarahkan kembali ke menu utama.

Jika login berhasil, pengguna akan diarahkan ke menu admin, yang memiliki beberapa opsi utama: cek uang kas restoran, kelola menu restoran, cek database pelanggan, dan keluar. Pada menu "Kelola Menu", pengguna dapat menambahkan, menghapus, mengedit harga, atau menampilkan menu restoran. Fungsi-fungsi ini diakses melalui modul menu, yang mengandung implementasi

masing-masing fitur pengelolaan menu. Seluruh pilihan menu disusun dalam struktur perulangan untuk memungkinkan pengguna mengakses fitur secara berulang hingga pengguna memilih keluar.

3.3.3 Tampilan Program modul menu.py

Berikut adalah tampilan dari menu.py
Modul ini tidak memiliki tampilan output karena hanya memuat fungsi yang akan di panggil di modul admin.py, semua output dan hasilnya sama dengan yang ada di admin.py

Berikut adalah tampilan source code keseluruhan pada tampilan diatas :

Tabel 3. Source Code dari menu.py

```
from format import teksFormat


listMenu = []

# Createe
def tambahMenu():
    print('Silahkan tambahkan nama menu dan harga')
    while True:
        teksFormat.dekorasi(100)
        pilih = int(input('Jenis menu apa yang ingin anda tambah
:\n1. Makanan\n2. Minuman\n3. Desert dan Cemilan\n4. Simpan
perubahan\nPilihan anda : '))
        if pilih == 4:
            print('Perubahan berhasil di simpan!')
            break
        namaMenu = input('Masukkan nama menu : ')
        harga = input('Masukkan harga menu : ')
        if pilih == 1:
            with open('Database/databaseMenu/menu_makanan.txt',
'a') as menu:
                menu.write(f'{namaMenu} ')
                menu.write(f'{harga}\n')
            print('menu berhasil di tambahkan!')
        elif pilih == 2:
```

```
            with open('Database/databaseMenu/menu_minuman.txt',
'a') as menu:
                menu.write(f'{namaMenu} ')
                menu.write(f'{harga}\n')
            print('menu berhasil di tambahkan!')
        elif pilih == 3:
            with open('Database/databaseMenu/menu_camilan.txt',
'a') as menu:
                menu.write(f'{namaMenu} ')
                menu.write(f'{harga}\n')
        else :
            print('Silahkan Masukkan Pilihan yang benar')

# Read
def tampilkanMenu():
    listMenuMakanan = []
    with open("Database/databaseMenu/menu_makanan.txt", "r") as
file:
        for line in file:
            parts = line.split()

            if len(parts) > 1:
                nama_barang = " ".join(parts[:-1])
                harga = parts[-1]
            else:
                nama_barang = parts[0]
                harga = "Tidak ada harga"

            result = [nama_barang, harga]
            listMenuMakanan.append(result)

    listMenuMinuman = []
    with open("Database/databaseMenu/menu_minuman.txt", "r") as
file:
        for line in file:
            parts = line.split()

            if len(parts) > 1:
                nama_barang = " ".join(parts[:-1])
                harga = parts[-1]
            else:
                nama_barang = parts[0]
                harga = "Tidak ada harga"

            result = [nama_barang, harga]
```

```
            listMenuMinuman.append(result)

    listMenuCamilan = []
    with open("Database/databaseMenu/menu_camilan.txt", "r") as
file:
        for line in file:
            parts = line.split()

            if len(parts) > 1:
                nama_barang = " ".join(parts[:-1])
                harga = parts[-1]
            else:
                nama_barang = parts[0]
                harga = "Tidak ada harga"

            result = [nama_barang, harga]
            listMenuCamilan.append(result)

    teksFormat.dekorasi(100)
    print(f'|\t\t\tMenu\t\t\t |\t\t\tHarga\t\t\t   |')
    teksFormat.dekorasiGaris(100)
    print(f'|\t\t\t\t\t\tMakanan')
    teksFormat.dekorasiGaris(100)
    for menu in listMenuMakanan:
        print(f'|\t\t{menu[0]}\t\t\t |\t\t\t{menu[1]}\t\t\t   |')
    teksFormat.dekorasiGaris(100)
    print(f'|\t\t\t\t\t\tMinuman')
    teksFormat.dekorasiGaris(100)
    for menu in listMenuMinuman:
        print(f'|\t\t{menu[0]}\t\t\t |\t\t\t{menu[1]}\t\t\t   |')
    teksFormat.dekorasiGaris(100)
    print(f'|\t\t\t\t\t\tCamilan')
    teksFormat.dekorasiGaris(100)
    for menu in listMenuCamilan:
        print(f'|\t\t{menu[0]}\t\t\t |\t\t\t{menu[1]}\t\t\t   |')
    teksFormat.dekorasi(100)
    return listMenuMakanan



def updateMenu():
    print('Silahkan pilih menu yang ingin di update.')
    listMenuMakanan = []
    with open("Database/databaseMenu/menu_makanan.txt", "r") as
file:
```

```
        for line in file:
            parts = line.split()

            if len(parts) > 1:
                nama_barang = " ".join(parts[:-1])
                harga = parts[-1]
            else:
                nama_barang = parts[0]
                harga = "Tidak ada harga"

            result = [nama_barang, harga]
            listMenuMakanan.append(result)

    listMenuMinuman = []
    with open("Database/databaseMenu/menu_minuman.txt", "r") as
file:
        for line in file:
            parts = line.split()

            if len(parts) > 1:
                nama_barang = " ".join(parts[:-1])
                harga = parts[-1]
            else:
                nama_barang = parts[0]
                harga = "Tidak ada harga"

            result = [nama_barang, harga]
            listMenuMinuman.append(result)

    listMenuCamilan = []
    with open("Database/databaseMenu/menu_camilan.txt", "r") as
file:
        for line in file:
            parts = line.split()

            if len(parts) > 1:
                nama_barang = " ".join(parts[:-1])
                harga = parts[-1]
            else:
                nama_barang = parts[0]
                harga = "Tidak ada harga"

            result = [nama_barang, harga]
            listMenuCamilan.append(result)
```

```
    while True:
        teksFormat.dekorasi(100)
        pilih = int(input('Kategori menu yang akan di edit :\n1.
Makanan\n2. Minuman\n3. Camilan\n4. Simpan dan Keluar\n'))
        if pilih == 1:
            print('Daftar Menu : ')
            for menu in listMenuMakanan:
                print(f'{menu[0]}')

            menuEdit = input('Silahkan masukkan menu yang ingin
anda edit : ')
            menuDitemukan = False
            for menu in listMenuMakanan:
                if menuEdit == menu[0]:
                    menuLama = menu
                    menu[0] = input('Silahkan masukkan nama menu
yang baru : ')
                    menu[1] = input('Silahkan masukkan harga menu
: ')
                    print(f'menu {menuLama} telah di edit menjadi
[{menu[0]},{menu[1]}]')
                    menuDitemukan = True  # Mengubah status
menjadi True jika ditemukan
                    break

            if not menuDitemukan:
                print('Menu tidak ditemukan.')
                return

            with open('Database/databaseMenu/menu_makanan.txt',
'w') as file:
                for menu in listMenuMakanan:
                    file.write(f'{menu[0]} {menu[1]}\n')

        elif pilih == 2:
            print('Daftar Menu : ')
            for menu in listMenuMinuman:
                print(f'{menu[0]}')

            menuEdit = input('Silahkan masukkan menu yang ingin
anda edit : ')
            menuDitemukan = False
            for menu in listMenuMinuman:
                if menuEdit == menu[0]:
                    menuLama = menu
```

```
                    menu[0] = input('Silahkan masukkan nama menu
yang baru : ')
                    menu[1] = input('Silahkan masukkan harga menu
: ')
                    print(f'menu {menuLama} telah di edit menjadi
[{menu[0]},{menu[1]}]')
                    menuDitemukan = True  # Mengubah status
menjadi True jika ditemukan
                    break

            if not menuDitemukan:
                print('Menu tidak ditemukan.')
                return

            with open('Database/databaseMenu/menu_minuman.txt',
'w') as file:
                for menu in listMenuMinuman:
                    file.write(f'{menu[0]} {menu[1]}\n')

        elif pilih == 3:
            print('Daftar Menu : ')
            for menu in listMenuCamilan:
                print(f'{menu[0]}')

            menuEdit = input('Silahkan masukkan menu yang ingin
anda edit : ')
            menuDitemukan = False
            for menu in listMenuCamilan:
                if menuEdit == menu[0]:
                    menuLama = menu
                    menu[0] = input('Silahkan masukkan nama menu
yang baru : ')
                    menu[1] = input('Silahkan masukkan harga menu
: ')
                    print(f'menu {menuLama} telah di edit menjadi
[{menu[0]},{menu[1]}]')
                    menuDitemukan = True  # Mengubah status
menjadi True jika ditemukan
                    break

            if not menuDitemukan:
                print('Menu tidak ditemukan.')
                return
```

```
            with open('Database/databaseMenu/menu_camilan.txt',
'w') as file:
                for menu in listMenuCamilan:
                    file.write(f'{menu[0]} {menu[1]}\n')

        elif pilih == 4:
            print('Semua perubahan berhasil disimpan.')
            break
        else:
            print('Silahkan Masukkan pilihan yang benar')

# Delete
def hapusMenu():
    print('Silahkan pilih menu yang akan di hapus')
    listMenuMakanan = []
    with open("Database/databaseMenu/menu_makanan.txt", "r") as
file:
        for line in file:
            parts = line.split()

            if len(parts) > 1:
                nama_barang = " ".join(parts[:-1])
                harga = parts[-1]
            else:
                nama_barang = parts[0]
                harga = "Tidak ada harga"

            result = [nama_barang, harga]
            listMenuMakanan.append(result)

    listMenuMinuman = []
    with open("Database/databaseMenu/menu_minuman.txt", "r") as
file:
        for line in file:
            parts = line.split()

            if len(parts) > 1:
                nama_barang = " ".join(parts[:-1])
                harga = parts[-1]
            else:
                nama_barang = parts[0]
                harga = "Tidak ada harga"

            result = [nama_barang, harga]
            listMenuMinuman.append(result)
```

```
    listMenuCamilan = []
    with open("Database/databaseMenu/menu_camilan.txt", "r") as
file:
        for line in file:
            parts = line.split()

            if len(parts) > 1:
                nama_barang = " ".join(parts[:-1])
                harga = parts[-1]
            else:
                nama_barang = parts[0]
                harga = "Tidak ada harga"

            result = [nama_barang, harga]
            listMenuCamilan.append(result)

    while True:
        pilih = int(input('Kategori Makanan yang ingin anda hapus
: \n1. Makanan\n2. Minuman\n3. Camilan\n4. Simpan dan Keluar'))
        if pilih == 1:
            print('Daftar Menu : ')

            for menu in listMenuMakanan:
                print(f'{menu[0]}')

            menuHapus = input('Silahkan masukkan menu yang ingin
di hapus : ')
            menuDitemukan = False
            for menu in listMenuMakanan:
                if menuHapus == menu[0]:
                    listMenuMakanan.remove(menu)
                    menuDitemukan = True
                    print(f'{menu} Telah di hapus dari daftar
menu.')
                    break

            if not menuDitemukan:
                print(f'Menu {menuHapus} Tidak di temukan.')
                return

            with open('Database/databaseMenu/menu_makanan.txt',
'w') as file:
                for menu in listMenuMakanan:
                    file.write(f'{menu[0]} {menu[1]}\n')
```

```
        elif pilih == 2:
            print('Daftar Menu : ')

            for menu in listMenuMinuman:
                print(f'{menu[0]}')

            menuHapus = input('Silahkan masukkan menu yang ingin
di hapus : ')
            menuDitemukan = False
            for menu in listMenuMinuman:
                if menuHapus == menu[0]:
                    listMenuMinuman.remove(menu)
                    menuDitemukan = True
                    print(f'{menu} Telah di hapus dari daftar
menu.')
                    break

            if not menuDitemukan:
                print(f'Menu {menuHapus} Tidak di temukan.')
                return

            with open('Database/databaseMenu/menu_minuman.txt',
'w') as file:
                for menu in listMenuMinuman:
                    file.write(f'{menu[0]} {menu[1]}\n')

        elif pilih == 3:
            print('Daftar Menu : ')

            for menu in listMenuCamilan:
                print(f'{menu[0]}')

            menuHapus = input('Silahkan masukkan menu yang ingin
di hapus : ')
            menuDitemukan = False
            for menu in listMenuCamilan:
                if menuHapus == menu[0]:
                    listMenuCamilan.remove(menu)
                    menuDitemukan = True
                    print(f'{menu} Telah di hapus dari daftar
menu.')
                    break

            if not menuDitemukan:
```

```
                print(f'Menu {menuHapus} Tidak di temukan.')
                return

            with open('Database/databaseMenu/menu_camilan.txt',
'w') as file:
                for menu in listMenuCamilan:
                    file.write(f'{menu[0]} {menu[1]}\n')
        elif pilih == 4:
            print('Anda memilih keluar Terima kasih!')
            break
```

Penjelasan :

Kode ini adalah implementasi sistem pengelolaan menu pada aplikasi restoran, yang terhubung dengan fitur admin dalam kode sebelumnya. Pengelolaan menu meliputi operasi *Create* (penambahan menu baru), *Read* (penampilan menu yang tersedia), *Update* (pengeditan menu), dan *Delete* (penghapusan menu), semuanya dikelola melalui berbagai kategori seperti makanan, minuman, dan camilan.

1. Fungsi tambahMenu

Fungsi ini digunakan untuk menambahkan item baru ke dalam daftar menu restoran. Admin diminta untuk memilih kategori (makanan, minuman, atau camilan), kemudian memasukkan nama menu dan harga. Data ini kemudian disimpan ke file teks yang sesuai, seperti menu_makanan.txt, menu_minuman.txt, atau menu_camilan.txt. Setelah perubahan disimpan, pengguna diberi konfirmasi bahwa menu berhasil ditambahkan. Fungsi ini memungkinkan admin untuk memperluas daftar menu dengan mudah.

2. Fungsi tampilkanMenu

Fungsi ini bertugas membaca dan menampilkan semua menu dari tiga kategori (makanan, minuman, camilan) dengan harga masing-masing. Data diambil dari file

teks, diproses untuk menggabungkan nama menu dan harga, lalu ditampilkan dalam format tabel yang rapi dengan bantuan dekorasi dari modul teksFormat. Fungsi ini memungkinkan admin dan pengguna untuk melihat semua menu yang tersedia dengan mudah dan terorganisir.

3. Fungsi updateMenu

Fungsi ini memungkinkan admin untuk mengubah nama atau harga menu yang sudah ada. Prosesnya dimulai dengan meminta pengguna memilih kategori menu, kemudian menampilkan daftar menu yang tersedia. Admin dapat memilih item tertentu, memasukkan data baru, dan memperbarui file teks dengan data yang telah diubah. Fungsi ini penting untuk memastikan bahwa data menu selalu akurat dan terkini.

4. Fungsi hapusMenu

Fungsi ini menyediakan opsi bagi admin untuk menghapus item menu yang sudah tidak relevan atau diperlukan. Seperti fungsi update, admin diminta memilih kategori, melihat daftar menu, lalu memilih item untuk dihapus. Jika menu ditemukan, item tersebut dihapus dari daftar, dan file teks diperbarui untuk mencerminkan perubahan. Fungsi ini membantu menjaga kebersihan dan relevansi data menu.

5. Kaitan dengan Kode Admin

Kode pengelolaan menu ini diintegrasikan ke dalam fitur admin melalui pilihan "Kelola Menu Restoran" di fungsi loginAdmin. Ketika admin memilih opsi ini, mereka dapat menggunakan semua fitur di atas: menambah menu baru, melihat daftar menu, memperbarui menu yang ada, dan menghapus menu. Menu ini ditampilkan dalam struktur perulangan sehingga admin dapat melakukan beberapa operasi berturut-turut sebelum keluar dari menu pengelolaan.

3.3.4 Tampilan Program Modul user.py

Berikut adalah tampilan dari user.py

```
Anda akan login sebagai?
1. Admin
2. User
3. Batal
Pilihan Anda : 2
Selamat datang di sistem Informasi Restauran FliFood!!!
Silakan anda :
1. Login (Jika sudah memiliki akun)
2. Buat Akun(Untuk pengguna Baru)
Pilihan Anda : 2
Selamat datang pengguna baru, silahkan buat akun anda!!!
Silahkan buat user name Anda : newRafli
Silahkan buat password Anda : 123
Akun dengan Username: newRafli dan password *** berhasil dibuat!!!
Silahkan login menggunakan akun tersebut.
```

```
Silahkan login menggunakan akun tersebut.
Silakan anda :
1. Login (Jika sudah memiliki akun)
2. Buat Akun(Untuk pengguna Baru)
Pilihan Anda : 1
Silahkan Masukkan username akun: newRafli
Silahkan Masukkan password: 123
Selamat datang!!!
Anda ingin apa?
1. Pesan Makanan
2. Cek Point
3. Keluar
```

```
Anda ingin apa?
1. Pesan Makanan
2. Cek Point
3. Keluar
1
Silahkan pilih menu yang tersedia
=============================================================================================================
|                    Menu                    |                    Harga                    |
-------------------------------------------------------------------------------------------------------------
|                                            Makanan                                                        |
-------------------------------------------------------------------------------------------------------------
|             Nasi Spesial                   |                    40000                    |
|             Nasi Bakar                     |                    17000                    |
|             Nasi Padang                    |                    13000                    |
-------------------------------------------------------------------------------------------------------------
|                                            Minuman                                                        |
-------------------------------------------------------------------------------------------------------------
|             Es Limau                       |                    8000                     |
|             Es Jeruk                       |                    9000                     |
|             Es Doger                       |                    8000                     |
-------------------------------------------------------------------------------------------------------------
|                                            Camilan                                                        |
-------------------------------------------------------------------------------------------------------------
|             Burger Spesial                 |                    15000                    |
|             Potato Roll                    |                    8000                     |
=============================================================================================================
Silahkan pilih kategori menu:
1. Makanan
2. Minuman
3. Camilan
4. Selesai
>>>Pilihan Anda : 
 >>>Pilihan Anda : 1
 Silahkan pilih menu makanan
 Masukkan nama menu makanan!
 Pilihan Anda: Nasi Spesial
 Ingin pesan makanan lagi? y/n: n
 Terima Kasih
 Silahkan pilih kategori menu:
 1. Makanan
 2. Minuman
 3. Camilan
 4. Selesai
 >>>Pilihan Anda : 2
 Silahkan pilih menu minuman
 Masukkan nama menu minuman!
 Pilihan Anda: Es Limau
 Ingin pesan minuman lagi? y/n: n
 Terima Kasih
 Silahkan pilih kategori menu:
 1. Makanan
 2. Minuman
 3. Camilan
 4. Selesai
 >>>Pilihan Anda : 4
 Pesanan Anda:
 Nasi Spesial | 40000
 Es Limau | 8000
 Total Harga adalah Rp. 48.000
 Poin berhasil ditambahkan. Total poin untuk newRafli: 4800.00
Anda ingin apa?
1. Pesan Makanan
2. Cek Point
3. Keluar
2
Username Anda : newRafli
Jumlah Poin Anda : 4800.00
```

Gambar.3 Tampilan Program user.py

Tabel 4. Source Kode user.py

```
import menu.menu as menu
import format.teksFormat as teksFormat
from datetime import datetime

def catatLog(bill,totalHarga,userName):
    with open('Database/penjualan/historiPenjualan.txt', 'a') as
file:
        file.write(f'{datetime.now()}\n')
        file.write(f'Username : {userName}\n')
        for pesanan in bill:
            file.write(f'{pesanan[0]} | {pesanan[1]}\n')
        file.write(f'Total Harga adalah Rp.
{teksFormat.format_uang(totalHarga)}\n\n')

def tambahUangKas(totalHarga):
    with open('Database/penjualan/uangKas.txt', 'r') as file:
        uangKas = file.read()

    uangKasTotal = int(uangKas) + totalHarga
    uangKasTotal = str(uangKasTotal)
    with open('Database/penjualan/uangKas.txt', 'w') as file:
        file.write(uangKasTotal)

def cetakPoin(userName):
    listPoin = []
    with open('Database/databaseAkun/poin_pelanggan.txt', 'r') as
file:
        for line in file:
            parts = line.split()
            listPoin.append(parts)
        for user in listPoin:
            if userName == user[0]:
                print(f'Username Anda : {user[0]}\nJumlah Poin
Anda : {user[1]}')


def tambahPoin(userName, totalHarga):
    file_akun = 'Database/databaseAkun/akun_user.txt'
    file_poin = 'Database/databaseAkun/poin_pelanggan.txt'

    with open(file_akun, 'r') as file:
        for line in file:
            if userName == line.strip():
```

```
                break

    poin_list = []
    with open(file_poin, 'r') as file:
        for line in file:
            parts = line.strip().split()
            if len(parts) == 2:
                poin_list.append([parts[0], float(parts[1])])

    poin_baru = totalHarga * 0.1
    for data in poin_list:
        if data[0] == userName:  # Jika username ditemukan
            data[1] += poin_baru
            break

    with open(file_poin, 'w') as file:
        for data in poin_list:
            file.write(f"{data[0]} {data[1]:.2f}\n")

    print(f"Poin berhasil ditambahkan. Total poin untuk
{userName}: {poin_baru:.2f}")

def pesanMenu(userName):
    while True:
        operasi = int(input('Anda ingin apa?\n1. Pesan Makanan\n2.
Cek Point\n3. Keluar\n'))
        if operasi == 1:
            totalHarga = 0
            bill = []
            print('Silahkan pilih menu yang tersedia')
            menu.tampilkanMenu()
            while True:
                kategori = int(input('Silahkan pilih kategori
menu:\n1. Makanan\n2. Minuman\n3. Camilan\n4. Selesai\n>>>Pilihan
Anda : '))
                if kategori == 1:
                    with
open('Database/dataBaseMenu/menu_makanan.txt', 'r') as file:
                        listMenuMakanan = []
                        for parts in file:
                            makanan = parts.split()
                            namaMakanan = ' '.join(makanan[:-1])
                            harga = makanan[-1]
                            listmenu = [namaMakanan, harga]
                            listMenuMakanan.append(listmenu)
```

```
                        print('Silahkan pilih menu makanan')

                    while True:
                        pilihMenu = input('Masukkan nama menu
makanan!\nPilihan Anda: ')
                        for makanan in listMenuMakanan:
                            if pilihMenu == makanan[0]:
                                totalHarga += int(makanan[1])
                                pesanan = [makanan[0], makanan[1]]
                                bill.append(pesanan)
                                break
                        else:
                            print('Makanan tidak tersedia di
menu')

                        ulang = input('Ingin pesan makanan lagi?
y/n: ')
                        if ulang == 'y':
                            continue
                        elif ulang == 'n':
                            print('Terima Kasih')
                            break
                        else:
                            print('Pilihan tidak valid. Keluar.')
                            break

                elif kategori == 2:
                    with
open('Database/dataBaseMenu/menu_minuman.txt', 'r') as file:
                        listMenuMinuman = []
                        for parts in file:
                            minuman = parts.split()
                            namaMinuman = ' '.join(minuman[:-1])
                            harga = minuman[-1]
                            listmenu = [namaMinuman, harga]
                            listMenuMinuman.append(listmenu)
                    print('Silahkan pilih menu minuman')

                    while True:
                        pilihMenu = input('Masukkan nama menu
minuman!\nPilihan Anda: ')
                        for minuman in listMenuMinuman:
                            if pilihMenu == minuman[0]:
                                totalHarga += int(minuman[1])
                                pesanan = [minuman[0], minuman[1]]
```

```
                                bill.append(pesanan)
                                break
                        else:
                            print('Minuman tidak tersedia di
menu')

                        ulang = input('Ingin pesan minuman lagi?
y/n: ')
                        if ulang == 'y':
                            continue
                        elif ulang == 'n':
                            print('Terima Kasih')
                            break
                        else:
                            print('Pilihan tidak valid. Keluar.')
                            break

                elif kategori == 3:
                    with
open('Database/dataBaseMenu/menu_camilan.txt', 'r') as file:
                        listMenuCamilan = []
                        for parts in file:
                            camilan = parts.split()
                            namaCamilan = ' '.join(camilan[:-1])
                            harga = camilan[-1]
                            listmenu = [namaCamilan, harga]
                            listMenuCamilan.append(listmenu)

                        print('Silahkan pilih menu camilan')
                        while True:
                            pilihMenu = input('Masukkan nama menu
camilan!\nPilihan Anda: ')
                            for camilan in listMenuCamilan:
                                if pilihMenu == camilan[0]:
                                    totalHarga += int(camilan[1])
                                    pesanan = [camilan[0], camilan[1]]
                                    bill.append(pesanan)
                                    break
                            else:
                                print('Camilan tidak tersedia di
menu')

                            ulang = input('Ingin pesan camilan lagi?
y/n: ')
                            if ulang == 'y':
```

```
                            continue
                        elif ulang == 'n':
                            print('Terima Kasih')
                            break
                        else:
                            print('Pilihan tidak valid. Keluar.')
                            break

                elif kategori == 4:
                    print('Pesanan Anda:')
                    for pesanan in bill:
                        print(f'{pesanan[0]} | {pesanan[1]}')
                    print(f'Total Harga adalah Rp.
{teksFormat.format_uang(totalHarga)}')
                    tambahPoin(userName,totalHarga)
                    catatLog(bill,totalHarga,userName)
                    tambahUangKas(totalHarga)
                    break
                else:
                    print('Pilihan kategori tidak valid. Silahkan
ulangi.')

        elif operasi == 2:
            cetakPoin(userName)

        elif operasi == 3:
            print('Keluar dari sistem. Terima Kasih!')
            break

        else:
            print('Pilihan operasi tidak valid. Silahkan ulangi.')



def loginUser():
    print('Selamat datang di sistem Informasi Restauran
FliFood!!!')
    while True:
        akun = int(input('Silakan anda :\n1. Login (Jika sudah
memiliki akun)\n2. Buat Akun(Untuk pengguna Baru)\nPilihan Anda :
'))
        if akun == 1:

            akun_user_list = []
```

```
            with open('Database/databaseAkun/akun_user.txt', 'r')
as file:
                for line in file:  # Loop untuk membaca setiap
baris dalam file
                    akun_user = line.split()  # Memisahkan setiap
baris berdasarkan spasi
                    akun_user_list.append(akun_user)

            for attempt in range(3):
                userName = input('Silahkan Masukkan username akun:
')
                password = input('Silahkan Masukkan password: ')

                for akun_user in akun_user_list:
                    if userName == akun_user[0] and password ==
akun_user[1]:  # Asumsi format: username password
                        print('Selamat datang!!!')
                        pesanMenu(userName)
                        return
                print('Username atau password salah')
                if attempt < 2:
                    print(f'Anda memiliki {2 - attempt} kesempatan
lagi')
            print("Anda telah mencapai batas percobaan login.
Program dihentikan.")

        elif akun == 2:
            print('Selamat datang pengguna baru, silahkan buat
akun anda!!!')
            while True:
                userName = input('Silahkan buat user name Anda :
')
                with open('Database/databaseAkun/akun_user.txt',
'r') as file:
                    username_terdaftar = False
                    for line in file:
                        parts = line.split()
                        if parts:
                            if userName == parts[0]:
                                print('Anda tidak dapat
menggunakan nomor ini karena nomor Telepon telah di gunakan!')
                                username_terdaftar = True
                                break

                if not username_terdaftar:
```

```
                    password = input('Silahkan buat password Anda
: ')
                    with
open('Database/databaseAkun/akun_user.txt', 'a') as file:
                        file.write(f'{userName} {password}\n')

                    with
open('Database/databaseAkun/poin_pelanggan.txt', 'a') as file:
                        file.write(f'{userName} 0\n')

                    print(f'Akun dengan Username: {userName} dan
password {"*" * len(password)} berhasil dibuat!!!')
                    print('Silahkan login menggunakan akun
tersebut.')
                    break
                else:
                    continue
```

Penjelasan :

Program ini adalah sebuah Sistem Informasi Restoran bernama *FliFood* yang berfungsi untuk mempermudah pengelolaan operasional restoran secara digital. Sistem ini berbasis teks dan memiliki beberapa fitur utama. Pertama, pengguna dapat melakukan **login** menggunakan akun yang sudah ada atau mendaftarkan diri sebagai pengguna baru. Untuk pendaftaran, pengguna harus membuat nama pengguna (*username*) dan kata sandi (*password*), yang kemudian akan disimpan dalam basis data berupa file teks. Sistem juga memastikan bahwa nama pengguna tidak duplikat. Setelah masuk, pengguna dapat memilih untuk memesan menu dari berbagai kategori seperti makanan, minuman, atau camilan. Data menu diambil dari file teks terpisah sesuai kategori, dan pengguna dapat memesan beberapa item hingga selesai, dengan sistem yang otomatis menghitung total harga.

Setelah pemesanan selesai, sistem secara otomatis mencatat transaksi ke dalam log penjualan, menambahkan jumlah uang ke kas restoran, dan memberikan poin kepada pelanggan. Poin dihitung sebesar **10% dari total harga pesanan** dan disimpan dalam file khusus untuk setiap pelanggan. Selain itu, pelanggan juga dapat memeriksa jumlah poin yang telah mereka kumpulkan. Program ini memanfaatkan

file teks untuk menyimpan semua data, seperti akun pengguna, poin pelanggan, menu restoran, dan catatan penjualan, sehingga mudah diakses dan dikelola. Dengan fitur ini, *FliFood* dirancang untuk memberikan pengalaman yang terorganisir dan efisien dalam mengelola restoran.

BAB IV
PENUTUP

4.1 Kesimpulan

Aplikasi **Sistem Informasi Restaurant Fli Food** dirancang untuk mempermudah seluruh aktivitas operasional restoran, dari manajemen menu hingga proses penjualan, dengan tujuan utama untuk meningkatkan efisiensi dan transparansi. Dengan adanya aplikasi ini, setiap transaksi, perubahan menu, dan data pelanggan dapat tercatat dengan jelas dan terorganisir. Hal ini berbeda dengan sistem manual yang rentan terhadap kesalahan manusia dan kesulitan dalam pencatatan data.

Sistem informasi ini memberikan kontribusi yang sangat besar dalam meningkatkan pengelolaan restoran secara keseluruhan. Melalui pengelolaan menu yang fleksibel, pencatatan transaksi yang akurat, serta pemantauan poin loyalitas pelanggan, aplikasi ini mampu mempercepat layanan dan memberikan pengalaman yang lebih baik bagi pelanggan. Pengelola restoran pun mendapatkan wawasan yang lebih baik mengenai keuangan, transaksi, dan preferensi pelanggan, yang membantu dalam pengambilan keputusan bisnis yang lebih strategis.

Pencatatan transaksi dan data pelanggan secara otomatis juga memudahkan pengelola restoran dalam memantau perkembangan bisnis dan merancang promosi yang lebih tepat sasaran. Dengan adanya sistem poin loyalitas, aplikasi ini dapat meningkatkan keterlibatan pelanggan dan mendorong mereka untuk kembali ke restoran, yang pada akhirnya berkontribusi pada peningkatan loyalitas dan retensi pelanggan.

Namun, meskipun aplikasi ini telah memberikan banyak manfaat bagi operasional restoran, pengembangannya menghadapi beberapa tantangan, terutama dalam hal biaya operasional yang tinggi, khususnya untuk server dan infrastruktur yang diperlukan jika aplikasi ini nantinya dapat diakses secara global. Seiring dengan itu, masih banyak fitur yang perlu dikembangkan untuk memaksimalkan potensi aplikasi ini dan memberikan solusi yang lebih lengkap bagi pengelola restoran.

4.2 Saran

Berdasarkan pengamatan terhadap pengembangan aplikasi ini, terdapat beberapa saran yang dapat membantu dalam meningkatkan fungsionalitas dan efisiensi aplikasi:

1. Pengembangan Akses Global

   Salah satu saran utama adalah untuk mengembangkan kemampuan aplikasi agar dapat diakses secara global. Ini akan membuka peluang bagi pelanggan untuk memesan makanan dari luar restoran, baik untuk dinikmati di rumah atau melalui layanan pengantaran. Meskipun demikian, ini juga akan membutuhkan peningkatan infrastruktur server yang lebih besar dan biaya operasional yang lebih tinggi. Oleh karena itu, diperlukan perencanaan matang agar aplikasi dapat berjalan dengan lancar di tingkat global.

2. Optimasi Biaya Operasional

   Tantangan besar yang dihadapi dalam pengembangan aplikasi ini adalah biaya operasional yang tinggi, terutama untuk server yang mendukung akses global. Oleh karena itu, disarankan untuk mempertimbangkan opsi-opsi solusi cloud yang lebih efisien dan

skalabel untuk mengurangi beban biaya operasional dalam jangka panjang. Penggunaan teknologi server yang lebih efisien dan penyesuaian kapasitas sesuai dengan kebutuhan dapat membantu mengurangi biaya yang tidak perlu.

3. Integrasi Pembayaran Digital

   Untuk mempermudah transaksi, aplikasi ini sebaiknya diintegrasikan dengan sistem pembayaran digital yang lebih modern, seperti e-wallet, kartu kredit, atau metode pembayaran lainnya. Dengan demikian, pelanggan dapat melakukan pembayaran dengan lebih cepat dan praktis, yang tentunya juga mempercepat proses transaksi di restoran.

4. Manajemen Stok Bahan Baku

   Sebagai bagian dari pengembangan lebih lanjut, disarankan untuk menambahkan fitur manajemen stok bahan baku yang memungkinkan pengelola restoran untuk memantau persediaan bahan makanan secara lebih efisien. Fitur ini akan membantu restoran mengurangi pemborosan dan memastikan bahwa bahan baku yang dibutuhkan selalu tersedia untuk mendukung operasional harian.

5. Fitur Analitik dan Laporan Lebih Lanjut

   Pengelola restoran bisa mendapatkan manfaat lebih banyak jika aplikasi ini dilengkapi dengan fitur analitik yang lebih mendalam. Analisis penjualan, pola pembelian pelanggan, dan laporan kinerja restoran berdasarkan data yang ada dapat memberikan wawasan yang lebih baik untuk strategi bisnis ke depan. Fitur laporan yang

lebih rinci, seperti analisis tren atau prediksi permintaan menu tertentu, akan sangat membantu dalam perencanaan dan pengelolaan stok.

6. Pengembangan Pengalaman Pengguna (User Experience)

   Untuk memastikan aplikasi mudah digunakan oleh pengelola restoran dan pelanggan, antarmuka pengguna (user interface) perlu lebih ditingkatkan. Desain yang lebih intuitif, serta proses navigasi yang lebih sederhana, akan meningkatkan kenyamanan pengguna dan mempercepat adaptasi aplikasi dalam operasional restoran.

7. Peningkatan Keamanan Sistem

   Meskipun aplikasi ini sudah memiliki mekanisme login dasar, pengembangan lebih lanjut dalam hal keamanan, seperti otentikasi dua faktor (2FA) untuk akses admin, sangat disarankan. Dengan menambahkan lapisan keamanan ekstra, aplikasi ini akan lebih aman dari potensi kebocoran data dan akses tidak sah.

DAFTAR PUSTAKA

https://aws.amazon.com/id/what-is/python/

https://dqlab.id/mengenal-fungsi-fungsi-python-beserta-contohnya-4478

https://www.w3schools.com/python/python_file_handling.asp

https://www.geeksforgeeks.org/file-handling-python/

https://builtin.com/data-science/python-list

https://sainsdata.id/pemrograman/python/4504/python-percabangan-dan-perulangan/