A. PENDAHULUAN

Program ini dibuat sebagai bagian dari *challenge* di kelas untuk membuat aplikasi pengelolaan debitur dan pinjaman sederhana. Program ini memungkinkan pengguna untuk menambah debitur, menambah pinjaman, serta menampilkan daftar pinjaman dengan rincian seperti bunga, lama pinjaman, dan angsuran bulanan.

B. LANDASAN TEORI

Program ini menggunakan konsep dasar Pemrograman Berorientasi Objek (OOP) dan fungsi dalam Python. Dengan OOP, data terkait debitur dan pinjaman diorganisir menggunakan kelas dan objek.

C. METODOLOGI

1. Analisis Kebutuhan: Menentukan fitur yang diperlukan seperti kelola debitur, kelola pinjaman, dan menampilkan data.
2. Perancangan: Membuat kelas Debitur dan Pinjaman serta fungsinya.
3. Implementasi: Menuliskan kode untuk menu dan fitur pengelolaan data.

D. IMPLEMENTASI

1. Kelas Debitur: Kelas ini digunakan untuk menyimpan informasi debitur, seperti nama dan nomor KTP.

```
class debitur:
    def __init__(self, nama, ktp, limit_pinjaman):
        self.nama = nama
        self.ktp = ktp
        self.limit_pinjaman = limit_pinjaman
```

2. Kelas Pinjaman: Kelas ini digunakan untuk menyimpan informasi pinjaman, seperti jumlah pinjaman, bunga, lama pinjaman, dan menghitung angsuran bulanan.

```
class pinjaman:
    def __init__(self, nama_debitur, jumlah, bunga, bulan):
        self.nama_debitur = nama_debitur
        self.jumlah = jumlah
        self.bunga = bunga
        self.bulan = bulan
        self.angsuran = self.hitung_angsuran()

    def hitung_angsuran(self):
        total_pinjaman = self.jumlah + (self.jumlah * self.bunga / 100)
        return total_pinjaman / self.bulan
```

3. Kelas Pinjol: Kelas ini memiliki dua atribut utama, yaitu daftar_debitur dan daftar_pinjaman, yang digunakan untuk menyimpan daftar debitur dan pinjaman.

   A. Fungsi tambah_debitur:

Fungsi ini digunakan untuk menambahkan debitur baru ke dalam daftar debitur. Sebelum menambahkan, sistem akan memeriksa apakah nomor KTP debitur sudah terdaftar. Jika sudah ada, maka debitur tidak akan ditambahkan.

```
def tambah_debitur(self, nama, ktp, limit_pinjaman):
        if any(debitur.ktp == ktp for debitur in self.daftar_debitur):
            print(f"KTP '{ktp}' sudah ada.")
        else:
            debitur_baru = debitur(nama, ktp, limit_pinjaman)
            self.daftar_debitur.append(debitur_baru)
            print(f"Debitur '{nama}' berhasil ditambahkan.")
```

   B. Fungsi tampilkan_debitur:

Fungsi ini menampilkan daftar debitur yang telah terdaftar dalam sistem. Jika tidak ada debitur, pesan khusus akan ditampilkan.

```
def tampilkan_debitur(self):
```

```
    if not self.daftar_debitur:
        print("Belum ada debitur yang terdaftar.")
    else:
        print("\n================= Daftar Debitur
=================")
        print(f"{'Nama Debitur':<20}{'No KTP':<15}{'Limit Pinjaman':>20}")
        for debitur in self.daftar_debitur:

print(f"{debitur.nama:<20}{debitur.ktp:<15}Rp.{debitur.limit_pinjaman:>15,}"
)
    input("\nTekan Enter untuk melanjutkan...")
```

C. Fungsi cari_debitur:

Fungsi ini digunakan untuk mencari debitur berdasarkan nomor KTP. Jika debitur ditemukan, informasi detail debitur akan ditampilkan.

```
def cari_debitur(self, ktp):
    debitur_ditemukan = next((debitur for debitur in self.daftar_debitur if
debitur.ktp == ktp), None)
    if debitur_ditemukan:
        print("\n================= Detail Debitur
=================")
        print(f"Nama Debitur: {debitur_ditemukan.nama}")
        print(f"No KTP: {debitur_ditemukan.ktp}")
        print(f"Limit Pinjaman: Rp.{debitur_ditemukan.limit_pinjaman:,}")
    else:
        print(f"Debitur dengan KTP '{ktp}' tidak ditemukan.")
    input("\nTekan Enter untuk melanjutkan...")
```

D. Fungsi tambah_pinjaman:

Fungsi ini menambah pinjaman baru untuk debitur yang sudah terdaftar. Pengguna harus memasukkan nama debitur, jumlah pinjaman, bunga, dan lama pinjaman.

```
def tambah_pinjaman(self, nama_debitur, jumlah, bunga, bulan):
    debitur_ditemukan = next((debitur for debitur in self.daftar_debitur if
debitur.nama.lower() == nama_debitur.lower()), None)
```

```
        if debitur_ditemukan:
            pinjaman_baru = pinjaman(nama_debitur, jumlah, bunga, bulan)
            self.daftar_pinjaman.append(pinjaman_baru)
            print(f"Pinjaman sebesar Rp.{jumlah:,} untuk '{nama_debitur}'
berhasil ditambahkan.")
        else:
            print(f"Debitur '{nama_debitur}' tidak ditemukan.")
```

E. Fungsi tampilkan_pinjaman:

Fungsi ini menampilkan daftar pinjaman yang sudah ada, lengkap dengan detail seperti nama debitur, jumlah pinjaman, bunga, lama pinjaman, dan angsuran bulanan.

```
def tampilkan_pinjaman(self):
        if not self.daftar_pinjaman:
            print("Belum ada pinjaman yang terdaftar.")
        else:

print("\n==================================================
Daftar Pinjaman
==================================================")
            print(f"{'Nama Debitur':<15}{'Jumlah Pinjaman':>25}{'Bunga
(%)':>20}{'Bulan':>15}{'Angsuran':>25}")
            for pinjaman in self.daftar_pinjaman:

print(f"{pinjaman.nama_debitur:<15}Rp.{pinjaman.jumlah:>22,}{pinjaman.b
unga:>18.2f}%{pinjaman.bulan:>15}Rp.{pinjaman.angsuran:>23,.2f}")
        input("\nTekan Enter untuk melanjutkan...")
```

F. Menu kelola_debitur:

Menu ini menyediakan opsi untuk mengelola data debitur, termasuk menampilkan, mencari, dan menambah debitur.

```
def menu_kelola_debitur(self):
        while True:
            print("\n================= Kelola Debitur
=================")
```

```
        print("1. Tampilkan Semua Debitur")
        print("2. Cari Debitur")
        print("3. Tambah Debitur")
        print("0. Kembali")
        pilihan = input("Masukan Pilihan Sub Menu: ")

        if pilihan == '1':
            self.tampilkan_debitur()
        elif pilihan == '2':
            ktp = input("Masukan No KTP Debitur: ")
            self.cari_debitur(ktp)
        elif pilihan == '3':
            nama = input("Masukan Nama Debitur: ")
            ktp = input("Masukan No KTP: ")
            limit_pinjaman = int(input("Masukan Limit Pinjaman (dalam
Rupiah): "))
            self.tambah_debitur(nama, ktp, limit_pinjaman)
        elif pilihan == '0':
            break
        else:
            print("Pilihan tidak valid, coba lagi!")
```

G. Menu pinjaman:

Menu ini memberikan pilihan untuk menambah atau menampilkan semua pinjaman yang ada.

```
def menu_pinjaman(self):
    while True:
        print("\n================= Kelola Pinjaman
=================")
        print("1. Tambah Pinjaman")
        print("2. Tampilkan Semua Pinjaman")
        print("0. Kembali")
        pilihan = input("Masukan Pilihan Sub Menu: ")

        if pilihan == '1':
            nama_debitur = input("Masukan Nama Debitur: ")
            jumlah = int(input("Masukan Jumlah Pinjaman (dalam Rupiah):
"))
            bunga = float(input("Masukan Bunga Pinjaman (dalam %): "))
            bulan = int(input("Masukan Lama Pinjaman (dalam bulan): "))
            self.tambah_pinjaman(nama_debitur, jumlah, bunga, bulan)
```

```
        elif pilihan == '2':
            self.tampilkan_pinjaman()
        elif pilihan == '0':
            break
        else:
            print("Pilihan tidak valid, coba lagi!")
```

E. KODE PROGRAM

Berikut adalah kode lengkap program yang telah dibuat untuk aplikasi admin pinjaman online (pinjol) dengan semua kelas dan fungsionalitas yang telah dijelaskan sebelumnya:

```
class debitur:
    def __init__(self, nama, ktp, limit_pinjaman):
        self.nama = nama
        self.ktp = ktp
        self.limit_pinjaman = limit_pinjaman


class pinjaman:
    def __init__(self, nama_debitur, jumlah, bunga, bulan):
        self.nama_debitur = nama_debitur
        self.jumlah = jumlah
        self.bunga = bunga
        self.bulan = bulan
        self.angsuran = self.hitung_angsuran()

    def hitung_angsuran(self):
        total_pinjaman = self.jumlah + (self.jumlah * self.bunga / 100)
        return total_pinjaman / self.bulan


class pinjol:
    def __init__(self):
        self.daftar_debitur = []
        self.daftar_pinjaman = []

    def tambah_debitur(self, nama, ktp, limit_pinjaman):
        if any(debitur.ktp == ktp for debitur in self.daftar_debitur):
            print(f"KTP '{ktp}' sudah ada.")
        else:
            debitur_baru = debitur(nama, ktp, limit_pinjaman)
```

```
            self.daftar_debitur.append(debitur_baru)
            print(f"Debitur '{nama}' berhasil ditambahkan.")

    def tampilkan_debitur(self):
        if not self.daftar_debitur:
            print("Belum ada debitur yang terdaftar.")
        else:
            print("\n================= Daftar Debitur =================")
            print(f"{'Nama Debitur':<20}{'No KTP':<15}{'Limit Pinjaman':>20}")
            for debitur in self.daftar_debitur:

print(f"{debitur.nama:<20}{debitur.ktp:<15}Rp.{debitur.limit_pinjaman:>15,}")
        input("\nTekan Enter untuk melanjutkan...")

    def cari_debitur(self, ktp):
        debitur_ditemukan = next((debitur for debitur in self.daftar_debitur if
debitur.ktp == ktp), None)
        if debitur_ditemukan:
            print("\n================= Detail Debitur =================")
            print(f"Nama Debitur: {debitur_ditemukan.nama}")
            print(f"No KTP: {debitur_ditemukan.ktp}")
            print(f"Limit Pinjaman: Rp.{debitur_ditemukan.limit_pinjaman:,}")
        else:
            print(f"Debitur dengan KTP '{ktp}' tidak ditemukan.")
        input("\nTekan Enter untuk melanjutkan...")

    def tambah_pinjaman(self, nama_debitur, jumlah, bunga, bulan):
        debitur_ditemukan = next((debitur for debitur in self.daftar_debitur if
debitur.nama.lower() == nama_debitur.lower()), None)
        if debitur_ditemukan:
            pinjaman_baru = pinjaman(nama_debitur, jumlah, bunga, bulan)
            self.daftar_pinjaman.append(pinjaman_baru)
            print(f"Pinjaman sebesar Rp.{jumlah:,} untuk '{nama_debitur}' berhasil
ditambahkan.")
        else:
            print(f"Debitur '{nama_debitur}' tidak ditemukan.")

    def tampilkan_pinjaman(self):
        if not self.daftar_pinjaman:
            print("Belum ada pinjaman yang terdaftar.")
        else:
            print("\n==================================================
Daftar Pinjaman
==================================================")
            print(f"{'Nama Debitur':<15}{'Jumlah Pinjaman':>25}{'Bunga
(%)':>20}{'Bulan':>15}{'Angsuran':>25}")
            for pinjaman in self.daftar_pinjaman:
```

```
print(f"{pinjaman.nama_debitur:<15}Rp.{pinjaman.jumlah:>22,}{pinjaman.bunga:>1
8.2f}%{pinjaman.bulan:>15}Rp.{pinjaman.angsuran:>23,.2f}")
        input("\nTekan Enter untuk melanjutkan...")

    def menu_kelola_debitur(self):
        while True:
            print("\n================= Kelola Debitur ================")
            print("1. Tampilkan Semua Debitur")
            print("2. Cari Debitur")
            print("3. Tambah Debitur")
            print("0. Kembali")
            pilihan = input("Masukan Pilihan Sub Menu: ")

            if pilihan == '1':
                self.tampilkan_debitur()
            elif pilihan == '2':
                ktp = input("Masukan No KTP Debitur: ")
                self.cari_debitur(ktp)
            elif pilihan == '3':
                nama = input("Masukan Nama Debitur: ")
                ktp = input("Masukan No KTP: ")
                limit_pinjaman = int(input("Masukan Limit Pinjaman (dalam Rupiah): "))
                self.tambah_debitur(nama, ktp, limit_pinjaman)
            elif pilihan == '0':
                break
            else:
                print("Pilihan tidak valid, coba lagi!")

    def menu_pinjaman(self):
        while True:
            print("\n================= Kelola Pinjaman ================")
            print("1. Tambah Pinjaman")
            print("2. Tampilkan Semua Pinjaman")
            print("0. Kembali")
            pilihan = input("Masukan Pilihan Sub Menu: ")

            if pilihan == '1':
                nama_debitur = input("Masukan Nama Debitur: ")
                jumlah = int(input("Masukan Jumlah Pinjaman (dalam Rupiah): "))
                bunga = float(input("Masukan Bunga Pinjaman (dalam %): "))
                bulan = int(input("Masukan Lama Pinjaman (dalam bulan): "))
                self.tambah_pinjaman(nama_debitur, jumlah, bunga, bulan)
            elif pilihan == '2':
                self.tampilkan_pinjaman()
            elif pilihan == '0':
                break
            else:
```

```
            print("Pilihan tidak valid, coba lagi!")

    def menu_utama(self):
        while True:
            print("\n================= Admin Pinjol ================")
            print("1. Kelola Debitur")
            print("2. Pinjaman")
            print("0. Keluar")
            pilihan = input("Masukan Pilihan Menu: ")

            if pilihan == '1':
                self.menu_kelola_debitur()
            elif pilihan == '2':
                self.menu_pinjaman()
            elif pilihan == '0':
                print("Keluar dari sistem.")
                break
            else:
                print("Pilihan tidak valid, coba lagi!")


sistem_pinjol = pinjol()

sistem_pinjol.menu_utama()
```

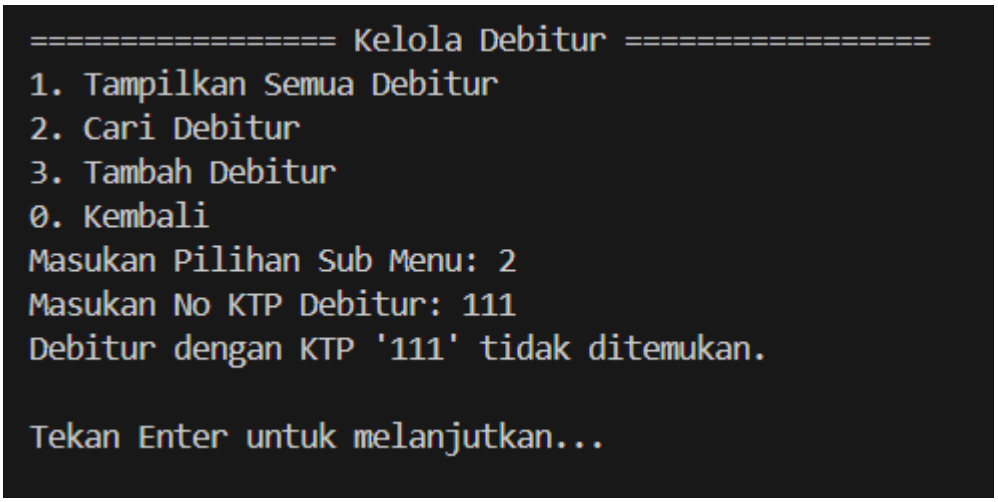
F. HASIL & PEMBAHASAN

```
================ Admin Pinjol ================
1. Kelola Debitur
2. Pinjaman
0. Keluar
Masukan Pilihan Menu: 1

================ Kelola Debitur ================
1. Tampilkan Semua Debitur
2. Cari Debitur
3. Tambah Debitur
0. Kembali
Masukan Pilihan Sub Menu: 1
Belum ada debitur yang terdaftar.

Tekan Enter untuk melanjutkan...
```

Hasil tersebut menunjukkan tampilan awal dari menu utama aplikasi Admin Pinjol. Pengguna memilih opsi 1 untuk mengelola debitur. Setelah masuk ke submenu Kelola Debitur, pengguna memilih opsi 1 untuk menampilkan semua debitur.

Namun, aplikasi menginformasikan bahwa belum ada debitur yang terdaftar, yang menandakan bahwa belum ada data debitur yang dimasukkan ke dalam sistem. Hal ini menunjukkan bahwa pengguna perlu menambahkan debitur terlebih dahulu agar dapat melihat daftar debitur yang ada. Ini merupakan langkah awal yang penting dalam penggunaan aplikasi, untuk memastikan bahwa sistem dapat berfungsi dengan baik setelah data dimasukkan.

Setelah menampilkan pesan tersebut, aplikasi meminta pengguna untuk menekan Enter untuk melanjutkan, menunjukkan interaksi yang sederhana dan user-friendly.



Hasil berikutnya menunjukkan kembali tampilan submenu Kelola Debitur, di mana pengguna memilih opsi 2 untuk mencari debitur menggunakan nomor KTP. Pengguna memasukkan nomor KTP 111, tetapi aplikasi memberikan respons bahwa debitur dengan KTP '111' tidak ditemukan.

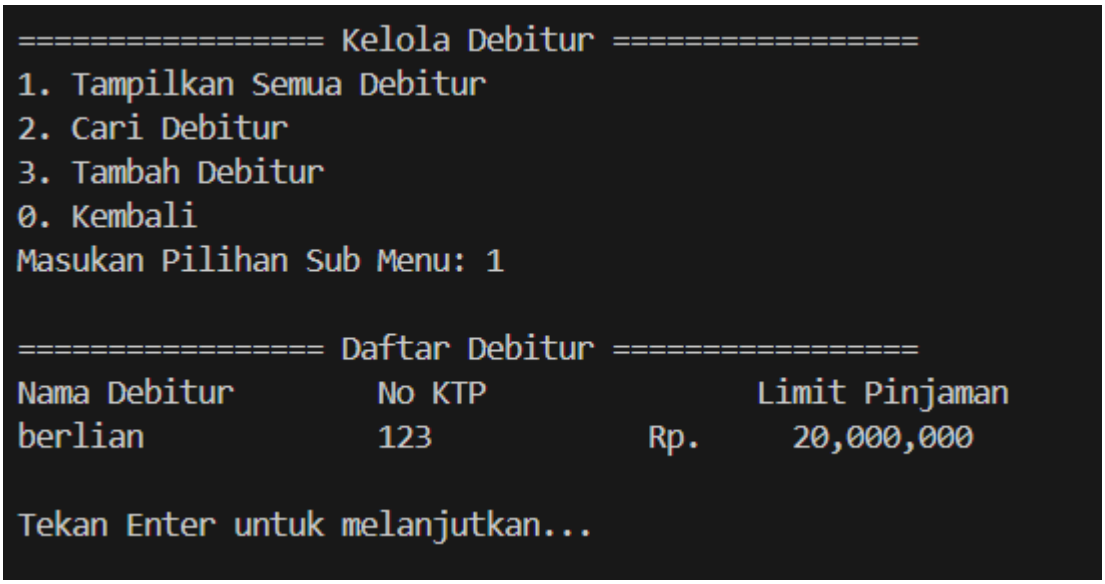
Hal ini menunjukkan bahwa sistem melakukan pencarian data debitur berdasarkan input pengguna dan menginformasikan jika tidak ada debitur yang terdaftar dengan KTP tersebut. Fitur ini penting karena memungkinkan pengguna untuk dengan cepat mengetahui apakah debitur tertentu sudah terdaftar dalam sistem. Jika tidak, pengguna bisa segera menambahkannya.

Setelah menampilkan pesan tersebut, aplikasi meminta pengguna untuk menekan Enter untuk melanjutkan, menjaga interaksi yang mudah dan responsif.

```
================= Kelola Debitur =================
1. Tampilkan Semua Debitur
2. Cari Debitur
3. Tambah Debitur
0. Kembali
Masukan Pilihan Sub Menu: 3
Masukan Nama Debitur: berlian
Masukan No KTP: 123
Masukan Limit Pinjaman (dalam Rupiah): 20000000
Debitur 'berlian' berhasil ditambahkan.
```

Pada hasil ini, pengguna kembali berada di submenu Kelola Debitur dan memilih opsi 3 untuk menambahkan debitur baru. Pengguna memasukkan nama debitur 'berlian', nomor KTP '123', dan limit pinjaman sebesar 20.000.000 Rupiah. Aplikasi kemudian mengonfirmasi bahwa debitur 'berlian' berhasil ditambahkan.

Proses ini menunjukkan fungsi penambahan debitur yang berhasil. Dengan fitur ini, pengguna dapat memperluas daftar debitur dalam sistem dengan memasukkan informasi penting, seperti nama, KTP, dan limit pinjaman. Ini penting untuk pengelolaan debitur, karena setiap debitur harus terdaftar agar sistem dapat memproses pinjaman dan mengelola data mereka dengan baik.



Pada langkah ini, pengguna memilih opsi 1 untuk menampilkan semua debitur yang terdaftar dalam sistem. Aplikasi kemudian menampilkan daftar debitur, yang dalam hal ini hanya terdapat satu debitur bernama 'berlian' dengan nomor KTP '123' dan limit pinjaman sebesar 20.000.000 Rupiah.

Tampilan ini menunjukkan bahwa sistem berhasil menyimpan data debitur yang sebelumnya ditambahkan. Informasi disajikan dengan format yang rapi, mencakup nama debitur, nomor KTP, dan limit pinjaman. Ini penting untuk memudahkan admin dalam melakukan pengelolaan debitur, serta memberikan transparansi tentang siapa saja yang terdaftar dan berapa limit pinjaman mereka. Dengan adanya daftar ini, admin dapat dengan mudah melakukan verifikasi dan pengawasan terhadap debitur yang ada.

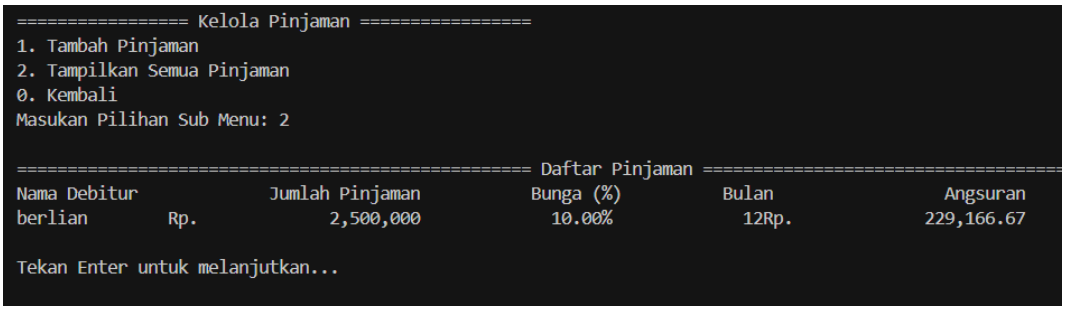
Setelah menampilkan daftar debitur, aplikasi meminta pengguna untuk menekan Enter untuk melanjutkan, menjaga interaksi pengguna tetap responsif dan user-friendly.

```
================= Kelola Debitur =================
1. Tampilkan Semua Debitur
2. Cari Debitur
3. Tambah Debitur
0. Kembali
Masukan Pilihan Sub Menu: 0

================= Admin Pinjol =================
1. Kelola Debitur
2. Pinjaman
0. Keluar
Masukan Pilihan Menu: 2

================= Kelola Pinjaman =================
1. Tambah Pinjaman
2. Tampilkan Semua Pinjaman
0. Kembali
Masukan Pilihan Sub Menu: 1
Masukan Nama Debitur: berlian
Masukan Jumlah Pinjaman (dalam Rupiah): 2500000
Masukan Bunga Pinjaman (dalam %): 10
Masukan Lama Pinjaman (dalam bulan): 12
Pinjaman sebesar Rp.2,500,000 untuk 'berlian' berhasil ditambahkan.
```

Setelah pengguna memilih opsi 0 pada menu Kelola Debitur, mereka kembali ke menu utama Admin Pinjol dan melanjutkan dengan memilih opsi 2 untuk Pinjaman. Pada menu Kelola Pinjaman, pengguna memilih opsi 1 untuk menambah pinjaman.

Pada langkah ini, pengguna memasukkan nama debitur 'berlian', jumlah pinjaman sebesar 2.500.000 Rupiah, bunga pinjaman 10%, dan lama pinjaman 12 bulan. Aplikasi berhasil menambahkan pinjaman tersebut dan mengkonfirmasi dengan pesan bahwa pinjaman telah berhasil ditambahkan untuk debitur yang bersangkutan.

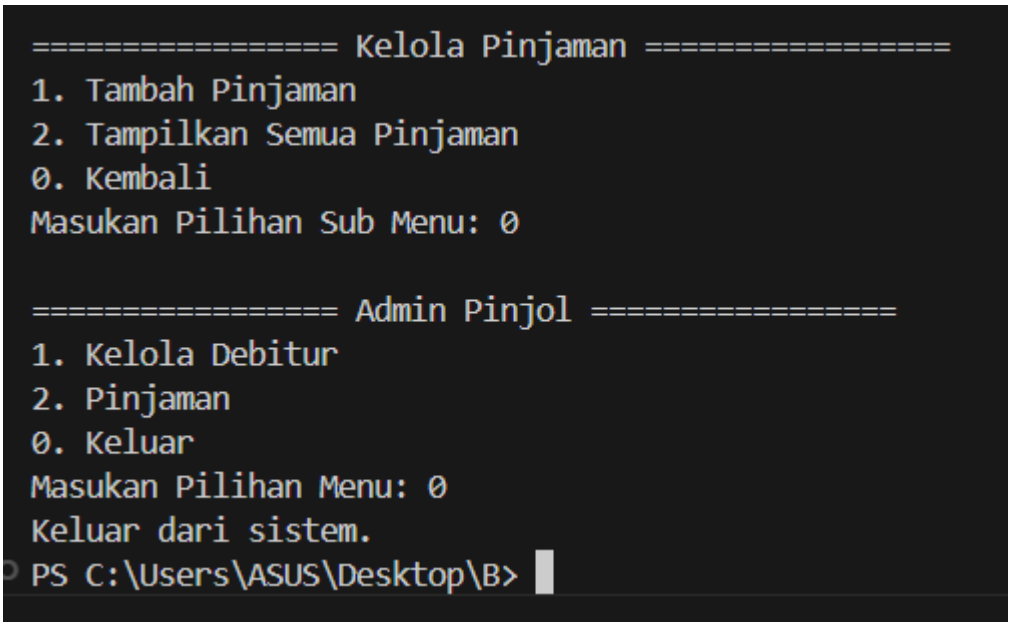
Proses ini menunjukkan bagaimana sistem mendukung admin dalam menambah informasi pinjaman bagi debitur. Dengan adanya fitur ini, admin dapat melakukan manajemen pinjaman secara efektif, termasuk mencatat jumlah pinjaman, suku bunga, dan durasi pinjaman. Data ini penting untuk perhitungan angsuran dan monitoring pinjaman debitur. Selain itu, sistem memberikan umpan balik yang jelas kepada pengguna, memastikan bahwa setiap tindakan yang diambil tercatat dengan baik dalam sistem.



Setelah pengguna memilih opsi 2 pada menu Kelola Pinjaman, sistem menampilkan daftar semua pinjaman yang telah dicatat. Dalam output yang ditampilkan, terdapat informasi mengenai pinjaman debitur 'berlian', termasuk:

- Jumlah Pinjaman: Rp. 2.500.000
- Bunga (%): 10.00%
- Bulan: 12
- Angsuran: Rp. 229.166,67

Tabel ini memberikan gambaran yang jelas mengenai pinjaman yang diambil oleh debitur, mencakup rincian jumlah pinjaman, suku bunga, durasi pinjaman dalam bulan, dan jumlah angsuran yang harus dibayar setiap bulannya.



Setelah pengguna memilih opsi 0 pada menu Kelola Pinjaman, sistem akan kembali ke menu utama Admin Pinjol. Pengguna kemudian diberikan pilihan untuk mengelola debitur, mengelola pinjaman, atau keluar dari sistem.

Pada langkah terakhir, pengguna memilih untuk keluar dari sistem dengan memilih 0 pada menu utama. Output menunjukkan pesan "Keluar dari sistem." Ini menandakan bahwa pengguna telah berhasil menyelesaikan sesi dan keluar dari aplikasi dengan aman.

G. KESIMPULAN

Program ini berhasil dikembangkan untuk mengelola data debitur dan pinjaman dalam sistem pinjaman online (pinjol). Dengan menggunakan kelas debitur dan pinjaman, pengguna dapat menambah, mencari, dan menampilkan debitur serta pinjaman dengan mudah. Menu interaktif yang sederhana serta validasi input membantu pengguna menghindari kesalahan saat memasukkan data.

Output program menunjukkan informasi debitur dan pinjaman secara rapi, memberikan gambaran jelas tentang status pinjaman. Dengan demikian, program ini tidak hanya efektif dalam pengelolaan data, tetapi juga memberikan pengalaman pengguna yang intuitif.