Laporan Tugas Challenge Membuat Program Kelola Debitur dan Kelola Pinjaman Dengan Menerapkan Inheritance

Disusun oleh:

Nama : Taufik Yoga Pratama
NPM : 5230411269
Kelas : VIII

PROGRAM STUDI INFORMATIKA

UNIVERSITAS TEKNOLOGI YOGYAKARTA

2023/2024

Kata Pengantar

Puji syukur kehadirat Tuhan Yang Maha Esa atas segala rahmat dan karunia-Nya sehingga saya dapat menyelesaikan laporan ini dengan judul "Laporan Tugas Challenge Membuat Program Kelola Debitur dan Kelola Pinjaman Dengan Menerapkan Inheritance". Laporan ini disusun untuk memenuhi tugas mata kuliah Pemrograman Berorientasi Objek Praktik. Saya berharap laporan ini dapat bermanfaat bagi pengembangan kemampuan saya dalam memahami konsep pemrograman berorientasi objek dan penerapannya dalam program nyata.

Saya menyadari bahwa laporan ini masih memiliki banyak kekurangan, oleh karena itu saya sangat terbuka terhadap kritik dan saran yang membangun dari semua pihak untuk perbaikan di masa mendatang.

Daftar Isi

BAB I

PENDAHULUAN

1.1. Latar Belakang

Dalam dunia pemrograman, konsep Object-Oriented Programming (OOP) menjadi salah satu metode yang sangat efektif dalam mengelola dan mengembangkan perangkat lunak. Salah satu keunggulan dari OOP adalah penggunaan konsep inheritance (pewarisan) yang memudahkan dalam pengembangan kode yang lebih terstruktur dan modular. Oleh karena itu, tugas ini bertujuan untuk mengimplementasikan konsep pewarisan dalam pembuatan program untuk mengelola debitur dan pinjaman.

1.2. Rumusan Masalah

1. Bagaimana cara menerapkan konsep inheritance dalam pembuatan program yang dapat mengelola data debitur dan pinjaman?
2. Bagaimana cara membuat program yang dapat menambah, mencari, dan menampilkan data debitur serta mengelola pinjaman dengan batas limit tertentu?
3. Bagaimana struktur dan diagram dari program kelola debitur dan pinjaman tersebut?

1.3. Tujuan

1. Untuk memahami penerapan konsep inheritance dalam pemrograman berorientasi objek.
2. Untuk membuat program yang dapat mengelola data debitur dengan fungsi-fungsi seperti menambah, mencari, dan menampilkan debitur.
3. Untuk membuat program yang dapat mengelola pinjaman dengan memperhatikan limit yang telah ditentukan.

BAB II

PEMBAHASAN

2.1. Codingan

```
class kelolo_debit():
    def __init__(self, nama, ktp, limit_pinjaman):
        self.nama = nama
        self.__ktp = ktp
        self._limit_pinjawam = limit_pinjaman

    def tampilkan_semua_debitur(self):
        print(f"{self.nama}               {self.__ktp}
 {self._limit_pinjawam}")

    def cari_debitur(self, nama):
        print(f"{nama}               {self.__ktp}               {se
lf._limit_pinjawam}")

    def get_nama(self):
        return self.nama

    def get_ktp(self):
        return self.__ktp

    def get_limit_pinjaman(self):
        return self._limit_pinjawam

def tambah_debitur(data):
    try:
        status = True
        while status:
            print(">>>>> Tambah Debitur")
            print(" ")
            ktp = int(input("Masukkan No KTP baru: "))
            ktp_terdaftar = False
            for user in data:
                if user.get_ktp() == ktp:
                    print("KTP sudah ada")
                    ktp_terdaftar = True
                    break
            if not ktp_terdaftar:
                nama = input("Masukkan nama debitur: ")
                limit_pinjaman = int(input("Masukkan limit pinjaman
debitur: "))
                debitur_baru = kelolo_debit(nama, ktp,
limit_pinjaman)
                data.append(debitur_baru)
                print("Data berhasil ditambahkan")
```

```
                status = False
    except ValueError:
            if ktp != int:
                print("KTP harus berupa angka")
            elif nama != str:
                print("Nama harus berupa huruf")

class menu_pinjaman(kelolo_debit):
    def __init__(self, nama):
        self.nama = nama

def tambah_pinjaman(data):
    try:
        print(">>>>> Tambah Pinjaman")
        print(" ")
        nama = input("Siapa yg ingin meminjam: ")
        debitur_ditemukan = False

        for user in data:
            if user.get_nama() == nama:
                debitur_ditemukan = True
                jumlah = int(input("Masukan jumlah pinjaman: "))
                if jumlah > user.get_limit_pinjaman():
                    print("Pinjaman melebihi limit")
                else:
                    bunga = int(input("Masukkan suku bunga: "))
                    limit = int(input("Masukan limit waktu dalam
bulan: "))
                    angsuranPokok = jumlah * bunga / 100
                    angsuranBln = angsuranPokok / limit
                    totalAngsuran = angsuranPokok + angsuranBln
                    list_pinjaman.append({
                        "nama": nama,
                        "jumlah": jumlah,
                        "bunga": bunga,
                        "limit_waktu": limit,
                        "angsuran": totalAngsuran
                    })
                    print(f"Pinjaman {nama} telah ditambahkan")
                break

        if not debitur_ditemukan:
            print("Nama belum terdaftar")

    except ValueError:
            if nama != str:
                print("Nama harus berupa string")
            elif jumlah != int:
```

```
                print("Jumlah pinjaman harus berupa integer")
            elif bunga != int:
                print("Suku bunga harus berupa integer")
            elif limit != int:
                print("Limit waktu harus berupa integer")

def tampilkan_pinjaman():
    print(">>>>> Tampilkan Pinjaman")
    print(" ")
    print("Nama
Debitur        Pinjaman        Bunga          Bulan          Angsura
n")
    for pinjaman in list_pinjaman:
        print(f"{pinjaman["nama"]}              Rp.{pinjaman["jumla
h"]}              {pinjaman["bunga"]}%                  {pinjaman["
limit_waktu"]}               Rp.{pinjaman["angsuran"]}")

def display(data):
    print("========== Daftar Debitur ==========")
    print(" ")
    print(" ")
    print("Nama Debitur          No KTP           Limit Pinjaman")
    print("=======================================================")
    for user in data:
        user.tampilkan_semua_debitur()

def cari_debitur(data):
    try:
        nama = input("Masukkan nama debitur yang ingin dicari: ")
        dataUser = False
        for i in data:
            if i.get_nama() == nama:
                print("========== Hasil Cari Debitur ==========")
                print(" ")
                print(" ")
                print("Nama Debitur          No KTP           Limit
Pinjaman")
                print("=============================================
=========")
                i.tampilkan_semua_debitur()
                dataUser = True
                break

        if not dataUser:
            print(f"{nama} tidak ditemukan")
    except ValueError:
        if nama != str:
            print("Nama harus berupa string")
```

```
user1 = kelolo_debit("naruto", 326, 5000000)
user2 = kelolo_debit("sakura", 545, 2500000)
user3 = kelolo_debit("sasuke", 356, 2000000)
user4 = kelolo_debit("garra", 267, 1000000)
user5 = kelolo_debit("shikamaru", 666, 1000000)

data = [user1, user2, user3, user4, user5]
list_pinjaman = []

def main():
    try:
        status = True
        while status:
            print("================== Sub Menu ==================")
            print("1. Kelola debitur")
            print("2. Menu pinjaman")
            print("0. Keluar")
            inputan = input("Pilih menu: ")
            if inputan == "1":
                statusKD = True
                while statusKD:
                    print("================ Kelola Debitur
================")
                    print("1. Tampilkan semua debitur")
                    print("2. Cari debitur")
                    print("3. Tambah debitur")
                    print("0. Kembali")
                    inputanKD = input("Pilih menu: ")
                    if inputanKD == "1":
                        display(data)
                    elif inputanKD == "2":
                        cari_debitur(data)
                    elif inputanKD == "3":
                        tambah_debitur(data)
                    elif inputanKD == "0":
                        statusKD = False
                    else:
                        print("Pilihan tidak ada")
            elif inputan == "2":
                statusMP = True
                while statusMP:
                    print("================== Menu Pinjaman
==================")
                    print("1. Tambah Pinjaman")
                    print("2. Tampilkan Pinjaman")
                    print("0. Kembali")
```

```
                    inputanMP = input("Pilih menu: ")
                    if inputanMP == "1":
                        tambah_pinjaman(data)
                    elif inputanMP == "2":
                        tampilkan_pinjaman()
                    elif inputanMP == "0":
                        statusMP = False
                    else:
                        print("Pilihan tidak ada")
            elif inputan == "0":
                status = False
            else:
                print("Pilihan tidak ada")
    except ValueError:
        if inputan != str:
            print("harus berupa angka")
        elif inputanKD != str:
            print("harus berupa angka")
        elif inputanMP != str:
            print("harus berupa angka")

main()
```

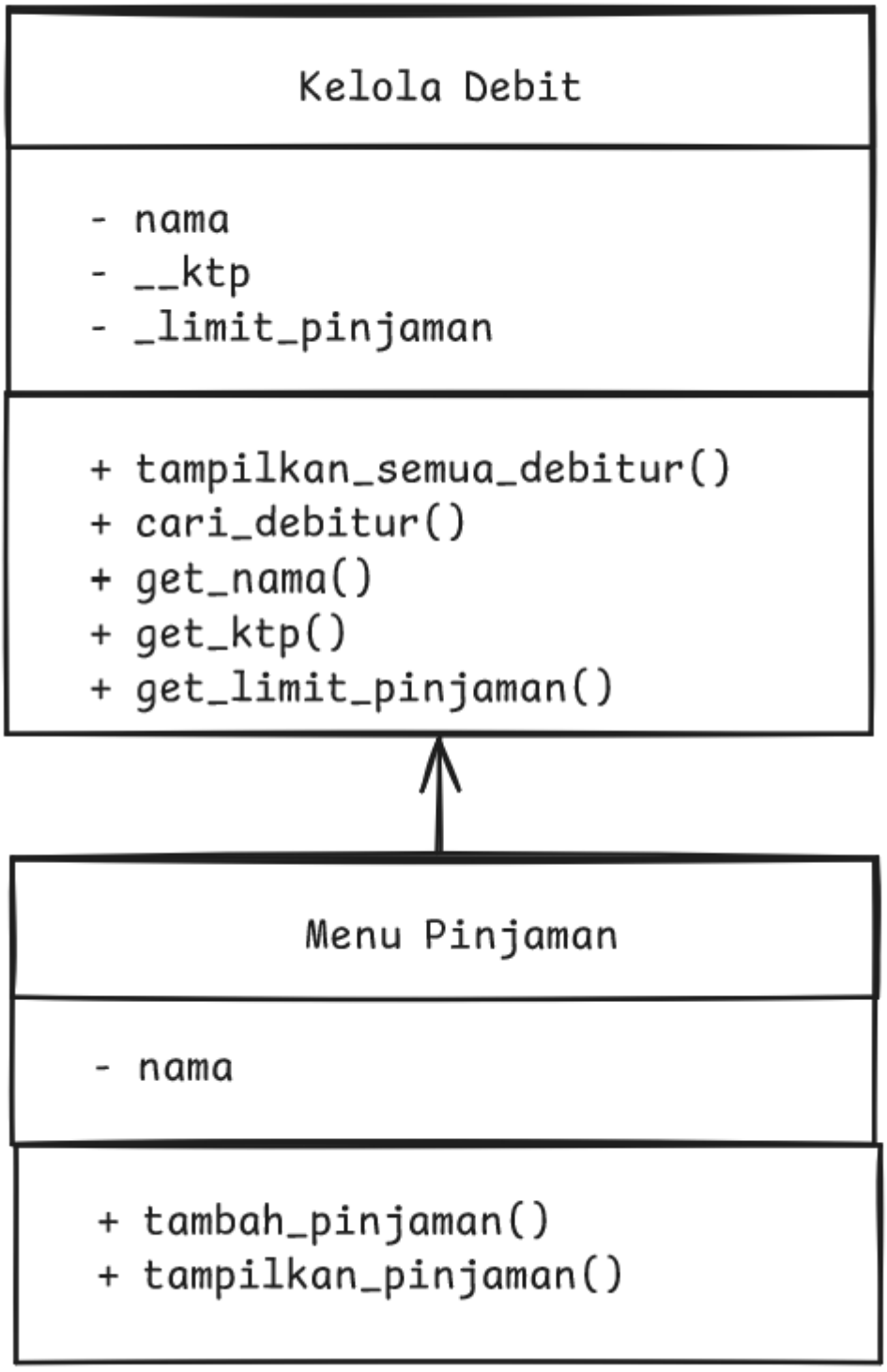
2.2. Pembahasan

Program ini menggunakan konsep OOP dengan menerapkan inheritance. Berikut adalah penjelasan singkat dari setiap bagian program:

- Class kelola_debit: Class ini digunakan untuk menyimpan informasi tentang debitur, seperti nama, KTP, dan limit pinjaman. Terdapat beberapa metode yang digunakan untuk menampilkan dan mengakses data debitur.
- Method tambah_debitur: Fungsi ini digunakan untuk menambahkan debitur baru ke dalam sistem. Program akan mengecek apakah KTP yang dimasukkan sudah terdaftar sebelumnya atau belum.
- Class menu_pinjaman: Class ini merupakan subclass dari kelolo_debit yang digunakan untuk mengelola pinjaman. Terdapat fungsi untuk menambah pinjaman dengan memperhatikan limit pinjaman yang dimiliki oleh debitur.
- Method tambah_pinjaman: Fungsi ini digunakan untuk menambah pinjaman ke debitur yang sudah terdaftar. Program akan memeriksa apakah jumlah pinjaman melebihi limit pinjaman yang tersedia.
- Method tampilkan_pinjaman: Fungsi ini digunakan untuk menampilkan daftar pinjaman yang sudah diinputkan beserta rincian bunga, waktu, dan jumlah angsuran yang harus dibayar.

2.3. Class Diagram

Class Diagram



BAB III

PENUTUP

3.1. Kesimpulan

Dari tugas ini, dapat disimpulkan bahwa penerapan konsep inheritance dalam pemrograman berorientasi objek sangat membantu dalam pengembangan program yang modular dan terstruktur. Dengan menggunakan inheritance, kita dapat memanfaatkan sifat pewarisan untuk memperluas fungsionalitas tanpa harus membuat ulang kode yang sama. Program kelola debitur dan pinjaman yang telah dibuat ini merupakan contoh nyata dari penerapan konsep tersebut untuk mengelola data dengan lebih efisien.