"Sistem Manajemen Perpustakaan dengan Python."

Tujuan Proyek
Terdapat beberapa tujuan dari proyek ini, antara lain:
Membuat program LMS menggunakan Python
Membuat program Python yang dapat terhubung ke basis data relasional
Menerapkan pemrograman berbasis fungsi atau pemrograman berorientasi objek
Menerapkan penulisan kode bersih (mengacu pada PEP 8)

2. Detail Tugas/Deskripsi
a. Gambaran Proyek Proyek ini terdiri dari Sistem Manajemen Perpustakaan Sederhana dengan beberapa fitur, seperti:
Pendaftaran anggota perpustakaan
Pendaftaran buku baru
Peminjaman
Menampilkan daftar anggota, buku, dan peminjaman
Pencarian buku
b. Alat Proyek Proyek ini menggunakan bahasa pemrograman (Python) dan database (MySQL).
c. Alur Kerja Proyek ini terdiri dari beberapa tahap, termasuk:

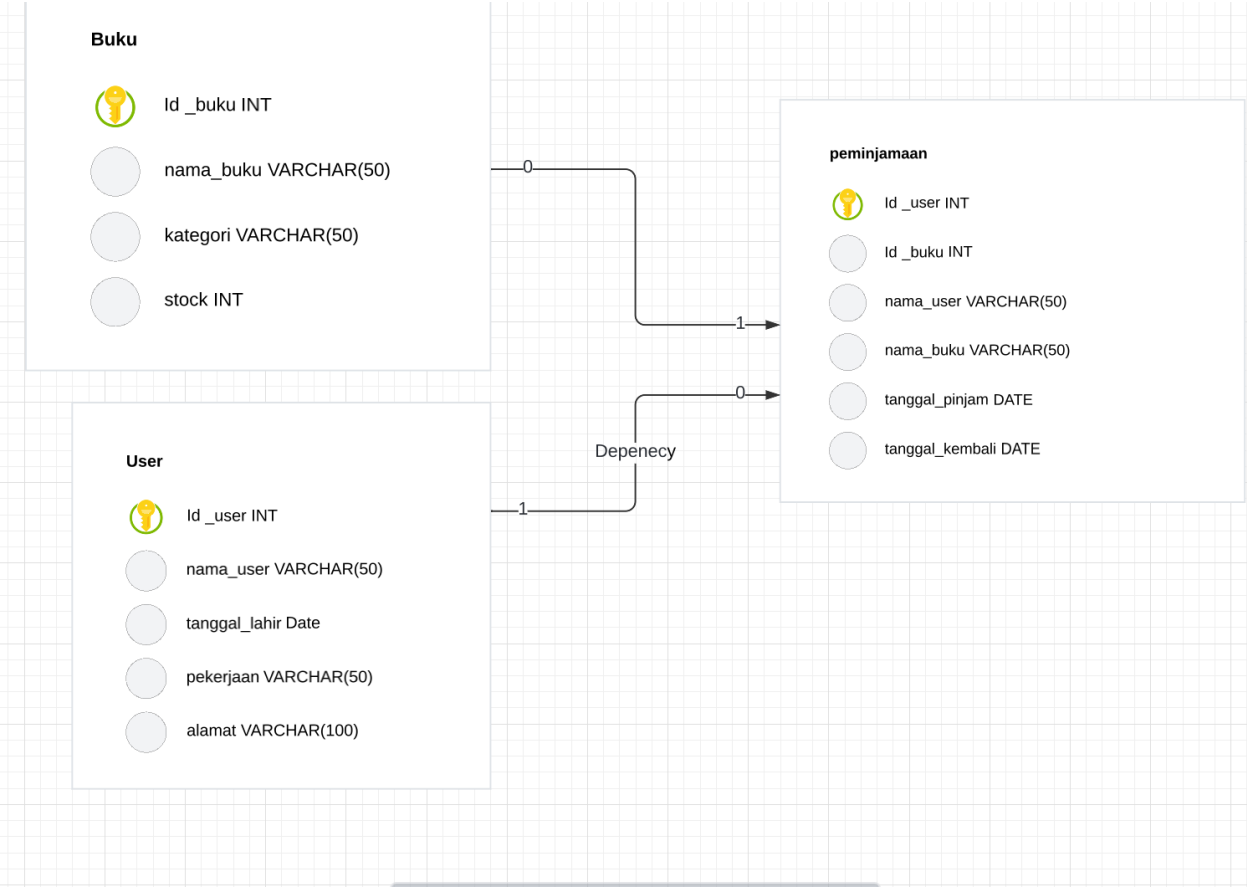
1. Pembuatan Database (MySQL) Membuat database dummy sebagai data yang akan diakses oleh pengguna. Tahap-tahap dalam MySQL adalah sebagai berikut:

```
# Membuat database
DROP DATABASE db_lib;

CREATE DATABASE db_lib;

USE db_lib;

# Membuat tabel daftar user perpustakaan
CREATE TABLE user(
    id_user INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    nama_user VARCHAR(50),
    tanggal_lahir DATE,
    pekerjaan VARCHAR(50),
    alamat VARCHAR(100)
```

```
);

# Membuat tabel daftar buku
CREATE TABLE buku(
    id_buku INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    nama_buku VARCHAR(50),
    kategori VARCHAR(50),
    stock INT
);

# Membuat tabel transaksi pinjaman buku
CREATE TABLE peminjaman(
    id_user INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    id_buku INT,
    nama_user VARCHAR(50),
    nama_buku VARCHAR(50),
    tanggal_pinjam DATE,
    tanggal_kembali DATE
);

ALTER TABLE peminjaman ADD FOREIGN KEY (id_user) REFERENCES user(id_user);
ALTER TABLE peminjaman ADD FOREIGN KEY (id_buku) REFERENCES buku(id_buku);
ALTER TABLE user AUTO_INCREMENT=15;

INSERT INTO user VALUES (01, 'Andi', '1995-02-15', 'Mahasiswi', 'Jakarta Pusat');
INSERT INTO user VALUES (02, 'Bambang', '1993-02-15', 'Pegawai', 'Cilincing');
INSERT INTO user VALUES (03, 'karlos', '1980-04-24', 'CEO', 'Sawangan');
INSERT INTO user VALUES (04, 'Davin', '2000-10-03', 'Pelajar', 'Grogol');
INSERT INTO user VALUES (05, 'Fahmi', '1986-06-27', 'Wiraswasta', 'Tanjung Barat');
INSERT INTO user VALUES (06, 'Geo', '1998-08-03', 'PNS', 'Pondok Kopi');
INSERT INTO user VALUES (07, 'Hafsah', '1989-01-17', 'PNS', 'Kalibata');
INSERT INTO user VALUES (08, 'Imran', '1990-11-04', 'Pegawai Swasta', 'Cibubur');
INSERT INTO user VALUES (09, 'Jaka', '1978-09-04', 'Wiraswasta', 'Rawamangun');
INSERT INTO user VALUES (10, 'Kido', '1995-09-02', 'Mahasiswa', 'Plumpang');
INSERT INTO user VALUES (11, 'Linda', '2002-11-03', 'Pelajar', 'Tebet');
INSERT INTO user VALUES (12, 'Adrian', '1990-05-29', 'Pegawai BUMN', 'Cawang');
INSERT INTO user(nama_user, tanggal_lahir, pekerjaan, alamat) VALUES ('Landy', '2005-
09-01', 'Pelajar', 'Duren Sawit');

INSERT INTO buku VALUES (1001, 'Belajar Python', 'Teknologi', 4);
INSERT INTO buku VALUES (1002, 'Harimau Harimau', 'Novel', 1);
INSERT INTO buku VALUES (1003, 'Cantik Itu Luka', 'Novel', 2);
INSERT INTO buku VALUES (1004, 'Rich Dad Poor Dad', 'Investasi', 1);
INSERT INTO buku VALUES (1005, 'Laa Tazhan', 'Agama', 2);
INSERT INTO buku VALUES (1006, 'Guns, Germs, and Steel', 'Sejarah', 4);
INSERT INTO buku VALUES (1007, '7 Habits', 'Self-Development', 3);
INSERT INTO buku VALUES (1008, 'Belajar Technical Analysis Saham', 'Invenstasi', 1);
INSERT INTO buku VALUES (1009, 'Data Wrangling in SQL', 'Teknologi', 3);
```

```sql
SELECT * FROM peminjaman;
SELECT * FROM buku;
WHERE nama_buku LIKE '%Belajar%';

UPDATE buku SET stock = stock - 1
WHERE id_buku = 1001;
```

Keluaran ERM (Entity-Relationship Model) dari file SQL adalah sebagai berikut:



2. Mengimpor Modul yang Diperlukan
Berikut adalah modul-modul yang diperlukan untuk implementasi proyek ini:

```
import mysql.connector
from mysql.connector import Error
import pandas as pd
import datetime
```

3. Membuat Koneksi antara Python dan SQL
Untuk membuat koneksi antara Python dan SQL, kita menggunakan mysql.connector. Namun, sebelum itu, kita perlu membuat fungsi create_db_connection.

```
def create_db_connection(host_name, user_name, user_password, db_name):
    connection = None
    try:
        connection = mysql.connector.connect(
            host=host_name,
            user=user_name,
            passwd=user_password,
            database=db)
        print("MySQL database connection successfull")
    except Error as err:
        print(f"Error: {err}")
    return connection
```

Untuk membuat koneksi, penting untuk menentukan nama host (hostname), pengguna (user), kata sandi (password), dan basis data (database).

```
nama_host = "localhost"
user = "root"
password = "topg"
db = "db_lib"
```

4. Membuat Fungsi add_new_user (Mendaftarkan pengguna baru ke sistem perpustakaan)
Membuat koneksi dengan mysql.connector.connect.

```
def add_new_user():

    mydb = mysql.connector.connect(host=nama_host, user=user, passwd=password,
database=db)
    mycursor = mydb.cursor()
```

Fungsi ini akan menerima input pengguna, yang meliputi:
 a. name_user (str)
b. tgl_user / tanggal lahir pengguna (date)
 c. pek_user / pekerjaan pengguna (str)
 d. alamat_user (str)

Hasil input disimpan dalam variabel sql menggunakan sintaks SQL.

```
sql = f"INSERT INTO user(nama_user, tanggal_lahir, pekerjaan, alamat) VALUES
('{nama_user}', '{tgl_user}', '{pek_user}', '{alamat_user}')"
    mycursor.execute(sql)
    mydb.commit()
```

Kemudian, itu dieksekusi dengan mycursor.execute(sql) dan di-commit dengan mydb.commit().

Cetak "Query berhasil dieksekusi" dan "Data berhasil ditambahkan".

```
def add_new_user():

    mydb = mysql.connector.connect(host=nama_host, user=user, passwd=password,
database=db)
    mycursor = mydb.cursor()


    nama_user = input("Input Nama User: ")
    tgl_user = input("Input Tanggal Lahir(YYYY-MM-DD): ")
    pek_user = input("Input Pekerjaan: ")
    alamat_user = input("Input Alamat: ")


    sql = f"INSERT INTO user(nama_user, tanggal_lahir, pekerjaan, alamat) VALUES
('{nama_user}', '{tgl_user}', '{pek_user}', '{alamat_user}')"
    mycursor.execute(sql)
    mydb.commit()
    print("Query berhasil dieksekusi")
    print("-------------------------")
    print("Data berhasil ditambahkan")
```

5. Membuat Fungsi add_new_book (Menambahkan buku baru ke perpustakaan)
Membuat koneksi menggunakan mysql.connector.connect.
Fungsi ini akan menerima input pengguna:
a. book_id (int),
b. book_title (str),
c. book_category (str),
d. book_stock (int)

Hasil input dimasukkan ke dalam variabel sql menggunakan sintaks SQL.

```
sql = f"INSERT INTO buku VALUES ({id_buku}, '{nama_buku}', '{category_buku}',
{stock_buku})"
    mycursor.execute(sql)
    mydb.commit()
```

Kemudian, dijalankan dengan mycursor.execute(sql) dan di-commit dengan mydb.commit().

Cetak "Query berhasil dieksekusi" dan "Data berhasil ditambahkan".

```
def add_new_book():

    mydb = mysql.connector.connect(host=nama_host, user=user, passwd=password,
database=db)
    mycursor = mydb.cursor()


    id_buku = input("Input Kode Buku: ")
    nama_buku = input("Input Nama Buku: ")
    category_buku = input("Input Kategori Buku: ")
    stock_buku = input("Input Stock Buku: ")


    sql = f"INSERT INTO buku VALUES ({id_buku}, '{nama_buku}', '{category_buku}',
{stock_buku})"
    mycursor.execute(sql)
    mydb.commit()


    print("Query berhasil dieksekusi")
    print("-------------------------")
    print("Data berhasil ditambahkan")
```

6. Membuat Fungsi add_new_trans (Melakukan input untuk peminjaman buku)

Membuat koneksi dengan mysql.connector.connect.

```
mydb = mysql.connector.connect(host=nama_host, user=user, passwd=password,
database=db)
    mycursor = mydb.cursor()
```

```
Fungsi ini akan menerima input pengguna:
a. id_user_borrow (int),
b. id_book_borrow (int),
c. user_name_borrow (str),
d. book_name_borrow (str)
Sementara variabel berikut akan diisi secara otomatis:
a. borrow_date (date) → otomatis diatur sebagai tanggal hari ini,
b. return_date (date) → otomatis diatur sebagai 3 hari dari tanggal peminjaman.

Hasil input akan dimasukkan ke dalam variabel sql menggunakan SQL.
```

```
sql = f"INSERT INTO peminjaman VALUES ('{id_user_pinjam}', '{id_buku_pinjam}',
'{nama_user_pinjam}', '{nama_buku_pinjam}','{tanggal_pinjam.strftime('%Y-%m-%d')}',
'{tanggal_kembali.strftime('%Y-%m-%d')}')"
    sql_update = f"UPDATE buku SET stock = stock - 1 WHERE id_buku = {id_buku_pinjam}"
    mycursor.execute(sql)
    mycursor.execute(sql_update)
    mydb.commit()
```

Untuk mengakomodasi fungsi stok buku yang akan berkurang, variabel sql_update akan digunakan untuk memperbarui nilai stok buku, yaitu:

```
sql_update = f"UPDATE buku SET stock = stock + 1 WHERE id_buku = {id_buku_pinjam}"
```

```
Berikutnya, jalankan mycursor.execute(sql), mycursor.execute(sql_update), dan commit
mydb.commit().

Cetak "Query berhasil dieksekusi" dan print(f"Buku telah dipinjam oleh:
{user_name_borrow}")".
```

```
def add_new_book():

    mydb = mysql.connector.connect(host=nama_host, user=user, passwd=password,
database=db)
    mycursor = mydb.cursor()


    id_buku = input("Input Kode Buku: ")
    nama_buku = input("Input Nama Buku: ")
    category_buku = input("Input Kategori Buku: ")
    stock_buku = input("Input Stock Buku: ")


    sql = f"INSERT INTO buku VALUES ({id_buku}, '{nama_buku}', '{category_buku}',
{stock_buku})"
    mycursor.execute(sql)
    mydb.commit()


    print("Query berhasil dieksekusi")
    print("--------------------------")
    print("Data berhasil ditambahkan")
```

7. Membuat Fungsi show_buku (Menampilkan semua buku yang terdaftar)
Atur variabel 'no' menjadi 0 sebagai penghitung.

Membuat koneksi dengan mysql.connector.connect.

```
mydb = mysql.connector.connect(host=nama_host, user=user, passwd=password,
database=db)
mycursor = mydb.cursor()
```

Untuk menampilkan semua buku, masukkan sintaks kueri SQL 'SELECT * FROM buku;' dalam mycursor.execute.

Tampilkan hasil kueri menggunakan mycursor.fetchall().

Tampilkan header kolom menggunakan print(mycursor.column_names).

Lakukan iterasi melalui setiap baris hasil yang diambil menggunakan sintaks berikut:

```
for data in cursor:
        no += 1
        print(data)
```

```
def show_buku():
    no = 0
    mydb = mysql.connector.connect(host=nama_host, user=user, passwd=password,
database=db)
    mycursor = mydb.cursor()
    mycursor.execute('SELECT * FROM buku;')
    cursor = mycursor.fetchall()
    print(mycursor.column_names)
    for data in cursor:
        no += 1
        print(data)
```

8. Membuat Fungsi show_user (Menampilkan semua pengguna yang terdaftar)

Atur variabel no = 0 sebagai penghitung.

Membuat koneksi dengan mysql.connector.connect.

```
mydb = mysql.connector.connect(host=nama_host, user=user, passwd=password,
database=db)
    mycursor = mydb.cursor()
```

Untuk menampilkan semua pengguna, masukkan sintaks kueri SQL 'SELECT * FROM user;' pada mycursor.execute.
Tampilkan hasil kueri dengan mycursor.fetchall().

Tampilkan header kolom menggunakan print(mycursor.column_names).

Lakukan iterasi melalui setiap baris hasil yang diambil menggunakan sintaks berikut:

```
for data in cursor:
        no += 1
        print(data)
```

```
def show_user():

    no = 0


    mydb = mysql.connector.connect(host=nama_host, user=user, passwd=password,
database=db)
    mycursor = mydb.cursor()


    mycursor.execute('SELECT * FROM user;')
    cursor = mycursor.fetchall()


    print(mycursor.column_names)


    for data in cursor:
        no += 1
        print(data)
```

9. Membuat Fungsi show_trans (Menampilkan semua buku yang sedang dipinjam saat ini)
Atur variabel no = 0 sebagai penghitung.

Membuat koneksi dengan mysql.connector.connect.

```
   mydb = mysql.connector.connect(host=nama_host, user=user, passwd=password,
database=db)
    mycursor = mydb.cursor()
```

Untuk menampilkan semua pinjaman buku
masukkan sintaks kueri SQL 'SELECT * FROM peminjaman

pada mycursor.execute. Tampilkan hasil kueri dengan mycursor.fetchall()

Tampilkan header kolom menggunakan print(mycursor.column_names).

Lakukan pemrograman defensif jika tidak ada pinjaman yang ada.

```
if mycursor.rowcount == 0:
        print("Belum terdapat peminjaman")
    else:
        for data in result:
```

```
            no += 1
            print(data)


def show_trans():

    no = 0


    mydb = mysql.connector.connect(host=nama_host, user=user, passwd=password,
database=db)
    mycursor = mydb.cursor()


    mycursor.execute('SELECT * FROM peminjaman;')
    result = mycursor.fetchall()


    print(mycursor.column_names)


    if mycursor.rowcount == 0:
        print("Belum terdapat peminjaman")
    else:
        for data in result:
            no += 1
            print(data)
```

10. Membuat Fungsi search_books (Menampilkan daftar buku yang dicari berdasarkan nama) Menerima input untuk nama buku dan menetapkannya ke variabel search_book.

Membuat koneksi dengan mysql.connector.connect.

```
   mydb = mysql.connector.connect(host=nama_host, user=user, passwd=password,
database=db)
 mycursor = mydb.cursor()
```

Untuk menampilkan buku yang dicari, gunakan sintaks SQL f"SELECT * FROM buku WHERE nama_buku LIKE '%{seach_book}%';" dalam mycursor.execute.

Tampilkan hasil kueri dengan mycursor.fetchall().

Tampilkan header kolom menggunakan print(mycursor.column_names).

Lakukan pemrograman defensif jika tidak ada hasil yang tersedia.

```
if mycursor.rowcount == 0:
        print("Buku tidak tersedia")
    else:
        columns = mycursor.description
        result = []
        for value in cursor:
            tmp = {}
            for (index,column) in enumerate(value):
                tmp[columns[index][0]] = column
            result.append(tmp)
        print(pd.DataFrame(result))

def cari_buku():

    seach_book = input("Masukkan Nama Buku: ")


    mydb = mysql.connector.connect(host=nama_host, user=user, passwd=password,
database=db)
    mycursor = mydb.cursor()


    mycursor.execute(f"SELECT * FROM buku WHERE nama_buku LIKE '%{seach_book}%';")
    cursor = mycursor.fetchall()


    if mycursor.rowcount == 0:
        print("Buku tidak tersedia")
    else:
        columns = mycursor.description
        result = []
        for value in cursor:
            tmp = {}
            for (index,column) in enumerate(value):
                tmp[columns[index][0]] = column
            result.append(tmp)
        print(pd.DataFrame(result))
```

11. Membuat Fungsi return_book (Mengembalikan buku yang dipinjam)

Membuat koneksi dengan mysql.connector.connect.

```
mydb = mysql.connector.connect(host=nama_host, user=user, passwd=password,
database=db)
    mycursor = mydb.cursor()
```

Menerima input untuk id_user_pinjam (int) dan id_buku_pinjam (int).

Hapus transaksi peminjaman untuk buku yang dikembalikan, masukkan sintaks dalam variabel sql f"DELETE FROM peminjaman WHERE id_user = {id_user_pinjam} AND id_buku = {id_buku_pinjam}".

Perbarui stok buku yang dikembalikan, masukkan sintaks dalam variabel sql_update = f"UPDATE buku SET stok = stok + 1 WHERE id_buku = {id_buku_pinjam}".

Jalankan sintaks Python ke SQL menggunakan mycursor.execute(sql) dan mycursor.execute(sql_update), dan commit mydb.commit().

Tampilkan header kolom dengan print(mycursor.column_names).

Cetak pengembalian yang berhasil dengan print("Query berhasil dieksekusi") dan sebagai pemisah cetak(" – – – – – – – – – – – – –").

Lakukan pemrograman defensif jika buku yang dikembalikan tidak tersedia.

```
if mycursor.rowcount == 0:
        print("Tidak ada peminjaman buku tersebut")
    else:
        print(f"Buku telah dikembalikan")

def kembalikan_buku():

    mydb = mysql.connector.connect(host=nama_host, user=user, passwd=password,
database=db)
    mycursor = mydb.cursor()


    id_user_pinjam = input("Input ID Peminjaman: ")
    id_buku_pinjam = input("Input ID Buku: ")


    sql = f"DELETE FROM peminjaman WHERE id_user = {id_user_pinjam} AND id_buku =
{id_buku_pinjam}"
```

```
    sql_update = f"UPDATE buku SET stock = stock + 1 WHERE id_buku = {id_buku_pinjam}"


    mycursor.execute(sql)
    mycursor.execute(sql_update)
    mydb.commit()


    print("Query berhasil dieksekusi")
    print("-------------------------")
    if mycursor.rowcount == 0:
        print("Tidak ada peminjaman buku tersebut")
    else:
        print(f"Buku telah dikembalikan")
```

12. Membuat Interface Program

```
finished = False
while not finished:


    interface = """
    ..................LIBRARY MANAGEMENT...................
          1. Pendaftaran User Baru
          2. Pendaftaran Buku Baru
          3. Peminjaman Buku
          4. Tampilkan Daftar Buku
          5. Tampilkan Daftar User
          6. Tampilkan Daftar Peminjaman
          7. Cari Buku
          8. Pengembalian
          9. Exit
    """
    print(interface)


    choice = int(input('Masukkan nomor tugas:'))

    if choice == 1:

        print("-------------------------------------")
        add_new_user()
```

```
elif choice == 2:

    print("-------------------------------------")
    add_new_book()

elif choice == 3:

    print("-------------------------------------")
    add_new_trans()

elif choice == 4:

    print("-------------------------------------")
    show_buku()

elif choice == 5:

    print("-------------------------------------")
    show_user()

elif choice == 6:

    print("-------------------------------------")
    show_trans()

elif choice == 7: # Cari Buku
    #Tampilkan data
    print("-------------------------------------")
    cari_buku()

elif choice == 8:

    print("-------------------------------------")
    kembalikan_buku()


elif choice == 9:
    is_finished = input('Apakah anda ingin keluar? (Y/N) ').upper()

    if is_finished == 'Y':
        finished == True
        break
    else:
        pass

else:
    print("Masukan angka sesuai dengan menu!")
```