LAPORAN PRATIKUM OOP PEKAN 2

MEMBUAT FUNGSI CRUD (CREATE, READ, UPDATE, DELETE) USER DENGAN DATABASE MYSQL

MATA KULIAH PEMROGRAMAN BERBASIS OBJEK

DOSEN PENGAMPU : NURFIAH, S.ST, M.KOM.

OLEH:

AUFAN TAUFIQURRAHMAN

NIM 2411532011

FAKULTAS TEKNOLOGI INFORMASI

DEPARTEMEN INFORMATIKA

UNIVERSITAS ANDALAS

2025

BAB 1

PENDAHULUAN

1.1.Latar Belakang

Praktikum sebelumnya telah berhasil membangun fondasi aplikasi "Laundry Apps" dengan mendefinisikan struktur model data dasar (seperti User, Customer, Service) dan mengimplementasikan antarmuka pengguna (GUI) awal menggunakan Java Swing, khususnya untuk proses autentikasi (login) dan navigasi menu utama.

Namun, aplikasi tersebut masih bersifat statis. Proses login, misalnya, memvalidasi input pengguna terhadap data yang ditulis langsung di dalam kode (hardcoded) pada method User.login(). Aplikasi belum memiliki kemampuan untuk menyimpan data secara permanen, mengelola data (menambah, mengubah, atau menghapus), atau mengambil data dari sumber eksternal.

Praktikum pekan ini merupakan evolusi dari aplikasi tersebut, berfokus pada implementasi fungsionalitas CRUD (Create, Read, Update, Delete). Fungsionalitas ini diimplementasikan untuk mengelola data pengguna (User) dengan menghubungkan aplikasi Java ke database MySQL. Ini adalah langkah krusial untuk mengubah aplikasi dari prototipe statis menjadi aplikasi dinamis yang berbasis data.

1.2.Tujuan Praktikum

Tujuan dari praktikum ini adalah:

1. Memahami dan mampu membuat koneksi antara aplikasi Java dengan database MySQL menggunakan driver JDBC.
2. Mampu merancang dan mengimplementasikan Graphical User Interface (GUI) yang fungsional untuk operasi CRUD data user.
3. Memahami dan menerapkan pola desain DAO (Data Access Object) untuk memisahkan logika bisnis (aplikasi) dari logika akses data (database).
4. Mengimplementasikan keempat fungsi inti CRUD:
    - Create: Menyimpan data user baru ke database.
    - Read: Menampilkan data user dari database ke dalam tabel.
    - Update: Mengubah data user yang sudah ada di database.
    - Delete: Menghapus data user dari database.

BAB 2

DASAR TEORI

2.1. MySQL dan JDBC (MySQL Connector/J)

MySQL adalah sebuah Relational Database Management System (RDBMS) open-source yang populer digunakan untuk menyimpan, mengelola, dan mengambil data dalam format tabel terstruktur.

Untuk memungkinkan aplikasi berbasis Java berkomunikasi dengan database MySQL, diperlukan sebuah driver yang dikenal sebagai JDBC (Java Database

Connectivity). MySQL Connector/J adalah implementasi driver JDBC resmi dari MySQL yang menyediakan abstraksi dan method untuk membuka koneksi ke database, mengirimkan kueri SQL (seperti SELECT, INSERT, UPDATE, DELETE), dan menerima hasilnya.

2.2. Pola Desain DAO (Data Access Object)

DAO (Data Access Object) adalah sebuah pola desain struktural yang digunakan untuk memisahkan logika akses data dari logika bisnis utama aplikasi. Tujuan utamanya adalah untuk menyediakan antarmuka (Interface) abstrak yang mendefinisikan operasi-operasi terkait data (seperti save, show, delete).

Kelas implementasi (Repository) kemudian akan "mewarisi" kontrak dari interface tersebut dan berisi kode spesifik untuk berinteraksi dengan database (misalnya, kueri SQL dan JDBC). Dengan pola ini, jika suatu saat sumber data diganti (misalnya dari MySQL ke PostgreSQL), kita hanya perlu mengubah kelas implementasi DAO tanpa harus mengubah logika bisnis atau tampilan (UI) yang menggunakannya.

2.3. AbstractTableModel

Dalam Java Swing, JTable adalah komponen yang digunakan untuk menampilkan data dalam bentuk grid (baris dan kolom). Namun, JTable tidak tahu cara mengelola data secara langsung.

AbstractTableModel adalah sebuah kelas abstrak yang berfungsi sebagai jembatan (adapter) antara sumber data (seperti List<User>) dengan komponen JTable. Dengan meng-override method penting seperti getRowCount(), getColumnCount(), dan getValueAt(), kita memberi tahu JTable cara mengambil, menampilkan, dan menghitung data dari daftar yang kita miliki.

BAB 3

IMPLEMENTASI DAN PEMBAHASAN

3.1. Konfigurasi Koneksi (src.confg.Database)

Langkah pertama adalah membuat koneksi ke database. Sebuah paket baru src.confg dibuat untuk menampung kelas Database.java.

Kelas ini memiliki satu method statis public static Connection koneksi(). Method ini bertanggung jawab untuk:

1. Memuat driver MySQL: Class.forName("com.mysql.cj.jdbc.Driver");
2. Membuka koneksi ke database menggunakan DriverManager.getConnection() dengan URL JDBC (jdbc:mysql://localhost/laundry_apps), username ("root"), dan password ("").
3. Mengembalikan objek Connection yang akan digunakan oleh *repository* untuk mengeksekusi kueri.

3.2. Implementasi Pola DAO (src.DAO)

Untuk memisahkan logika, dibuat paket src.DAO yang berisi *interface* dan implementasinya.

1. Interface (UserDAO.java): File ini bertindak sebagai "kontrak" yang mendefinisikan operasi apa saja yang bisa dilakukan terkait data User. Ini berisi definisi method abstrak seperti save(User user), show(), delete(String id), dan update(User user).
2. Implementasi (UserRepo.java): Kelas ini implements UserDAO dan berisi logika SQL yang sebenarnya.
    - Koneksi: Di dalam konstruktor, kelas ini memanggil Database.koneksi() untuk mendapatkan objek Connection.
    - save(User user): Menggunakan kueri INSERT INTO user (...) VALUES (?,?,?). Data dari objek user dimasukkan ke dalam PreparedStatement untuk mengeksekusi perintah.
    - show(): Menggunakan kueri SELECT * FROM user. Hasil ResultSet di-looping, diubah menjadi objek User, dan dimasukkan ke dalam List<User> untuk dikembalikan.
    - update(User user): Menggunakan kueri UPDATE user SET name=?, username=?, password=? WHERE id=?.
    - delete(String id): Menggunakan kueri DELETE FROM user WHERE id=?.

3.3. Model Tabel (src.table.TableUser)

Untuk menampilkan List<User> yang didapat dari UserRepo.show() ke dalam JTable di GUI, dibuat kelas TableUser.java yang extends AbstractTableModel.

Kelas ini memetakan data dari list ke tabel. Method getValueAt(int rowIndex, int columnIndex) menggunakan switch-case untuk menentukan data mana yang harus ditampilkan; case 0 mengembalikan ls.get(rowIndex).getId(), case 1 mengembalikan getNama(), dan seterusnya.

3.4. Antarmuka CRUD (src.ui.UserFrame)

File UserFrame.java adalah implementasi GUI (JFrame) yang baru. Tampilan ini berisi form input (JTextFields untuk nama, username, password), tombol-tombol operasi (Save, Update, Delete, Cancel), dan sebuah JTable untuk menampilkan data.

Kelas ini mengikat semua logika yang telah dibuat:

1. Inisialisasi: UserRepo usr diinisialisasi sebagai properti kelas.
2. Load Data (Read): Terdapat method loadTable() yang memanggil usr.show() untuk mengambil semua data user. Data tersebut kemudian dibungkus dengan TableUser dan diatur sebagai model untuk tableUsers (tableUsers.setModel(tu)). Method ini dipanggil saat frame pertama kali dibuka.

3. Simpan Data (Create): btnSave memiliki ActionListener yang mengambil teks dari txtName, txtUsername, dan txtPassword. Data ini dimasukkan ke objek User baru, lalu dikirim ke usr.save(user). Setelah disimpan, form dibersihkan (reset()) dan tabel dimuat ulang (loadTable()).
4. Seleksi Data (Event Handling): tableUsers diberi MouseListener. Ketika sebuah baris di tabel diklik, data dari baris tersebut (termasuk id yang tersembunyi) diambil dan digunakan untuk mengisi form input di atasnya. id user juga disimpan dalam variabel global id di dalam frame.
5. Ubah Data (Update): btnUpdate memiliki ActionListener. Logikanya mirip dengan save, namun ia juga menyertakan id yang telah disimpan (user.setId(id)) sebelum memanggil usr.update(user).
6. Hapus Data (Delete): btnDelete memiliki ActionListener yang memeriksa apakah id sudah dipilih (tidak null). Jika ya, ia langsung memanggil usr.delete(id) dan memuat ulang tabel.

BAB 4

KESIMPULAN

Praktikum pekan 2 ini berhasil mengimplementasikan fungsionalitas penuh CRUD (Create, Read, Update, Delete) untuk entitas User pada aplikasi Laundry Apps.

Pemisahan logika telah diterapkan dengan baik menggunakan pola desain DAO (UserDAO dan UserRepo), yang memisahkan kueri SQL dan logika database dari antarmuka pengguna (UserFrame). Aplikasi kini bersifat dinamis, mampu berinteraksi secara penuh dengan database MySQL menggunakan JDBC, menggantikan data statis (hardcoded) yang digunakan pada praktikum sebelumnya.

Penggunaan AbstractTableModel (TableUser) juga berhasil diimplementasikan sebagai perantara yang efektif untuk menampilkan data dari List<User> ke dalam komponen JTable Swing.