LAPORAN PRAKTIKUM

IMPLEMENTASI CRUD DENGAN UPLOAD DAN DOWNLOAD CSV BERBASIS PBO

Mata Kuliah : Pemrograman Berorientasi Objek

Pertemuan Keempatbelas

Dosen Pengampu :

Bayu Adhi Nugroho, Ph.D
19790518201403100 1

UIN SUNAN AMPEL
S U R A B A Y A

Oleh :

Frisilia Vina Ariyanto (09010624009)

KELAS H7B.3

PROGRAM STUDI SISTEM INFORMASI

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS ISLAM NEGERI SUNAN AMPEL SURABAYA

2025

IMPLEMENTASI CRUD DENGAN UPLOAD DAN DOWNLOAD CSV BERBASIS PBO

A. Tujuan Praktikum

1. Mahasiswa mampu menerapkan konsep CRUD (Create, Read, Update, Delete) dalam aplikasi berbasis Pemrograman Berorientasi Objek (PBO).
2. Mahasiswa mampu mengimplementasikan fitur upload file CSV untuk membaca dan mengimpor data ke dalam aplikasi.
3. Mahasiswa mampu mengimplementasikan fitur download file CSV sebagai proses ekspor data dari aplikasi.
4. Mahasiswa mampu melatih penggunaan class, objek, dan enkapsulasi dalam pengelolaan data dan file berbasis PBO.
5. Mahasiswa mampu mengembangkan kemampuan menangani input/output file serta melakukan validasi data pada proses CSV

B. Petunjuk Praktikum

1. Membuat project baru di IDE Java (NetBeans/Eclipse/IntelliJ) dan menyiapkan struktur dasar program berbasis PBO (class, package, dan file utama).
2. Membuat struktur package seperti model, controller/dao, utils, dan view (opsional jika menggunakan GUI).
3. Membuat class Model (Entity) untuk merepresentasikan satu baris data yang akan disimpan ke dalam list.
4. Membuat class controller yang berisi fungsi CRUD.
5. Menambahkan data awal secara manual (opsional) pada list untuk kebutuhan pengujian fitur CRUD sebelum fitur CSV diimplementasikan.
6. Membuat fungsi Upload CSV menggunakan JFileChooser (memilih file CSV), BufferedReader (membaca isi file), split(",") (memisahkan kolom data dan menyimpannya ke ArrayList).
7. Membuat fitur Download CSV yang mengekspor seluruh list ke file .csv menggunakan FileWriter dengan format kolom yang rapi dan sesuai.
8. Membuat tampilan GUI.
9. Mengimplementasikan dan menguji alur CRUD serta Upload/Download CSV, memastikan data tersimpan, ditampilkan, diimpor, dan diekspor dengan benar tanpa eror.

C. Dasar Teori

1. File CSV (Comma-Separated Values)

CSV adalah format file teks sederhana yang digunakan untuk menyimpan data tabular (seperti spreadsheet) di mana setiap baris mewakili sebuah record, dan setiap record terdiri dari satu atau lebih field yang dipisahkan oleh koma (,) atau delimiter lainnya (seperti titik koma ;) tergantung pengaturan lokal.

Kelebihan Format CSV :

- Strukturnya mudah dipahami, dan file berukuran kecil karena hanya berisi data teks polos.
- Hampir semua aplikasi pengolah data (seperti Microsoft Excel, Google Sheets, database management systems, dan bahasa pemrograman) memiliki dukungan native atau library untuk membaca dan menulis file CSV.
- Tidak seperti file biner, data dalam CSV dapat dibuka dan diperiksa dengan text editor sederhana.
- Merupakan format *de facto* untuk proses impor/ekspor data antar sistem yang berbeda.

2. Upload/Import CSV

Upload CSV adalah proses membaca data dari sebuah file CSV eksternal dan memasukkannya ke dalam aplikasi (biasanya dengan operasi Create). Aplikasi akan membaca file baris per baris, mem-parsing setiap baris menjadi atribut-atribut objek, dan kemudian menyimpannya ke dalam memori (koleksi) atau langsung ke database.

Alur kerja Upload CSV :

- Pengguna memilih file CSV sumber yang akan diunggah.
- Aplikasi membuka file CSV menggunakan kelas seperti File Reader dan BufferedReader.
- File dibaca baris demi baris.
- Setiap baris yang dibaca (kecuali header, jika ada) di-*split* atau dipisahkan berdasarkan delimiter yang ditentukan (misalnya koma) menjadi sebuah array of string.
- Sebelum dibuat menjadi objek, data yang telah diparsing harus divalidasi. Ini mencegah error dan menjaga integritas data.

- Array of string yang telah divalidasi digunakan untuk menginstansiasi objek baru. Kemudian objek baru ditambahkan ke dalam koleksi (ArrayList).
- Setelah proses impor selesai, seringkali diperlukan untuk menulis ulang seluruh koleksi (yang sudah diperbarui) ke file CSV utama aplikasi untuk mempertahankan persistensi data.

3. Download/Ekspor CSV

Download CSV adalah proses menulis data dari dalam aplikasi (dari koleksi objek) ke sebuah file CSV eksternal (biasanya dengan operasi Read). Aplikasi akan mengambil semua objek dari koleksi, mengonversi setiap objek menjadi sebuah baris string yang formatnya sesuai dengan CSV, dan menulisnya ke file.

Alur kerja Download CSV:

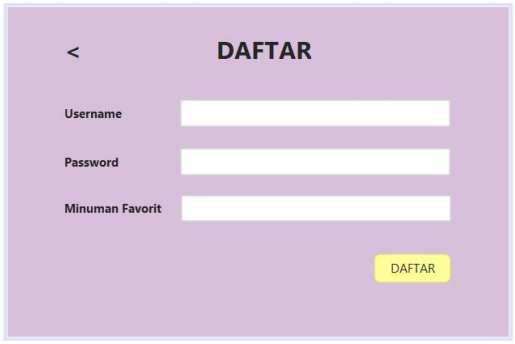
- Aplikasi membuka atau membuat file tujuan (misal: Data_Album.csv) menggunakan kelas seperti Filewritter dan BufferedWritter untuk efisiensi.
- Aplikasi melakukan iterasi (perulangan) melalui setiap objek dalam koleksi.
- String yang telah diformat untuk setiap objek ditulis ke dalam file sebagai baris baru.
- Setelah semua objek berhasil ditulis, file ditutup untuk memastikan semua data telah disimpan dengan benar.

Manfaat Download :

- Backup data.
- Menghasilkan file yang siap dicetak atau dianalisis lebih lanjut di aplikasi lain seprti Excel.
- Mengirim data ke sistem atau pihak lain yang membutuhkan.

D. PRAKTIKUM

1. Rename menjadi “PBOPertemuanEmpatbelas” pada project PBOPertemuanTigabelas.
2. Package Akun
    - Pada class Daftar tambahkan label dan textField untuk pertanyaan.



Kode ini berfungsi untuk proses pendaftaran akun baru. Sistem memeriksa apakah semua TextField termasuk TextField pertanyaan keamanan telah terisi. Pertanyaan keamanan (TfFav) digunakan sebagai data tambahan untuk verifikasi jika suatu saat pengguna perlu ubah password akun. Setelah validasi, sistem mengecek apakah username sudah ada di database. Jika belum, data username, password, dan pertanyaan keamanan disimpan, lalu pengguna dialihkan ke halaman login. Jika username sudah terdaftar, pendaftaran dibatalkan dan pengguna diminta mengganti username.

```
private void btnDaftarActionPerformed(java.awt.event.ActionEvent evt) {
    if (tfUsernameDaf.getText().equals("") || tfPasswordDaf.getText().equals("") || tfMinumanFav.getText().equals("")) {
        JOptionPane.showMessageDialog(null, "Isi semua data");
        return;
    } else {
        String user, pw, ktRhs;
        user = tfUsernameDaf.getText();
        pw = tfPasswordDaf.getText();
        ktRhs = tfMinumanFav.getText();

        EntityManagerFactory emf = Persistence.createEntityManagerFactory("PBOPertemuanDuabelasPU");
        EntityManager em = emf.createEntityManager();
        em.getTransaction().begin();

        Login_1 l = em.find(Login_1.class, user);
        if (l != null) {
            JOptionPane.showMessageDialog(null, "Username sudah ada, gunakan username lain");
            bersih();
            tfUsernameDaf.requestFocus();
        } else {
            Login_1 o = new Login_1();
            o.setUsername(user);
            o.setPassword(pw);
            o.setPertanyaan(tfMinumanFav.getText());
            em.persist(o);
            em.getTransaction().commit();

            JOptionPane.showMessageDialog(null, "Sukses dibuat");
            bersih();

            Login n = new Login();
            n.setVisible(true);
            this.dispose();
        }
        em.close();
        emf.close();
    }
}
```

- Tambahkan Dialog baaru untuk pertanyaannya.

Kode ini digunakan saat tombol **Selesai** diklik pada dialog input pertanyaan keamanan. Program memeriksa apakah TextField **TfFav** berisi teks atau tidak. Jika masih kosong, muncul pesan peringatan dan kursor diarahkan kembali ke TextField agar pengguna mengisinya. Jika sudah terisi, teks dari TfFav disimpan ke variabel **jawaban**, kemudian dialog ditutup dengan dispose() sehingga pengguna kembali ke tampilan utama.

```
private void btnSelesaiActionPerformed(java.awt.event.ActionEvent evt) {
    if (tfMinumanFav.getText().trim().isEmpty()) {
        JOptionPane.showMessageDialog(parentComponent: this, message: "Pertanyaan wajib diisi!");
        tfMinumanFav.requestFocus();
        return;
    }

    jawaban = tfMinumanFav.getText().trim();
    dispose(); // Tutup dialog dan kembali ke frame utama
}
```

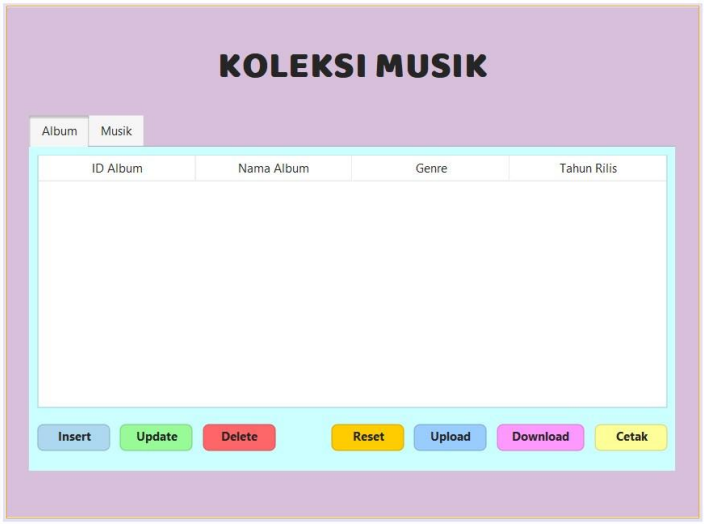
- Pada class “ResetPassword” kode ini berfungsi untuk mencari akun berdasarkan username lalu menampilkan dialog pertanyaan keamanan. Jika jawaban yang diberikan sesuai dengan data di database, maka pengguna diizinkan melanjutkan proses reset password. Jika tidak sesuai, proses dihentikan dan reset password tidak dapat dilakukan.

```
private void btnCariActionPerformed(java.awt.event.ActionEvent evt) {
    if (tfUsername.getText().equals(anObject: "")) {
        JOptionPane.showMessageDialog(parentComponent: null, message: "Isi Username Terlebih Dahulu");
    } else {
        EntityManagerFactory emf = Persistence.createEntityManagerFactory(persistenceUnitName:"PBOPertemuanDuabelasPU");
        EntityManager em = emf.createEntityManager();

        em.getTransaction().begin();

        String user = tfUsername.getText();
        Login_1 l = em.find(type: Login_1.class, o: user);

        if (l == null) {
            JOptionPane.showMessageDialog(parentComponent: null, message: "Data tidak ditemukan");
            tfUsername.requestFocus();
        } else {

            Pertanyaan dialog = new Pertanyaan(
                    (java.awt.Frame) SwingUtilities.getWindowAncestor(c: this),
                    modal: true
            );
            dialog.setLocationRelativeTo(c: this); // agar muncul di tengah
            dialog.setVisible(b: true);

            String input = dialog.getPertanyaan();
            // Cek jawaban dari database
            if (!input.equals(anObject: l.getPertanyaan())) {
                JOptionPane.showMessageDialog(parentComponent: null,
                        message: "Jawaban pertanyaan salah! Tidak bisa reset password.");
```

```
            tfUsername.setEditable(b: true); // username bisa diubah lagi
            tfUsername.requestFocus();
            return; // Stop proses
        }
        jlPasswordBaru.setVisible(aFlag: true);
        btnSimpan.setVisible(aFlag: true);
        tfPassBaru.setVisible(aFlag: true);
        tfUsername.setEditable(b: false);
        tfPassBaru.requestFocus();
    }
    em.getTransaction().commit();
    em.close();
    emf.close();
}
}
```

3. Pakcage DataMusik

- Pada class “Utama” tambahkan button download untuk tabel album.



Kode ini digunakan untuk mengekspor data album dari tabel ke file CSV. Pengguna memilih lokasi penyimpanan menggunakan JFileChooser, lalu program menuliskan setiap data dalam tabel ke dalam file dengan pemisah titik koma. Jika file sudah ada, pengguna diminta konfirmasi untuk menimpa. Setelah berhasil, muncul pesan bahwa data berhasil disimpan. Jika gagal, ditampilkan pesan error.

```
private void btnDownloadAlbumActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    JFileChooser fileChooser = new JFileChooser();
    fileChooser.setDialogTitle("Simpan sebagai CSV");

    // Nama default
    fileChooser.setSelectedFile(new File("Data_Album.csv"));

    int userSelection = fileChooser.showSaveDialog(this);

    if (userSelection == JFileChooser.APPROVE_OPTION) {

        File fileToSave = fileChooser.getSelectedFile();

        // Cek jika file sudah ada --> konfirmasi ke user
        if (fileToSave.exists()) {
            int confirm = JOptionPane.showConfirmDialog(
                    this,
                    "File sudah ada.\nApakah Anda ingin menimpanya?",
                    "Konfirmasi",
                    JOptionPane.YES_NO_OPTION,
                    JOptionPane.WARNING_MESSAGE
            );
            if (confirm != JOptionPane.YES_OPTION) {
                return; // batalkan
            }
        }

        try (FileWriter fw = new FileWriter(fileToSave)) {

            DefaultTableModel model = (DefaultTableModel) jTblAlbum.getModel();
            int colCount = model.getColumnCount();
            int rowCount = model.getRowCount();

            // Tulis isi tabel
            for (int r = 0; r < rowCount; r++) {
                for (int c = 0; c < colCount; c++) {
                    Object value = model.getValueAt(r, c);
                    String cell = (value == null) ? "" : value.toString().replace(";", ",");
                    fw.write(cell);

                    if (c < colCount - 1) {
                        fw.write(";");
                    }
                }
                fw.write("\n");
            }

            JOptionPane.showMessageDialog(
                    this,
                    "Data album berhasil diekspor ke file CSV!\n" + fileToSave.getAbsolutePath(),
                    "Sukses",
                    JOptionPane.INFORMATION_MESSAGE
            );

        } catch (Exception e) {
            JOptionPane.showMessageDialog(
                    this,
                    "Terjadi kesalahan saat menyimpan:\n" + e.getMessage(),
                    "Error",
                    JOptionPane.ERROR_MESSAGE
            );
            e.printStackTrace();
        }
    }
}
```

- Pada class “Utama” tambahkan button Download untuk tabel musik.

Kode ini digunakan untuk mengekspor data musik dari tabel ke file CSV. Pengguna memilih lokasi penyimpanan melalui JFileChooser, kemudian program menulis setiap baris tabel ke file dengan pemisah titik koma. Jika file sudah ada, akan muncul konfirmasi untuk menimpanya.

```
private void btnDownloadMusikActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    JFileChooser fileChooser = new JFileChooser();
    fileChooser.setDialogTitle("Simpan sebagai CSV");

    // Nama default
    fileChooser.setSelectedFile(new File("Data_Musik.csv"));

    int userSelection = fileChooser.showSaveDialog(this);

    if (userSelection == JFileChooser.APPROVE_OPTION) {

        File fileToSave = fileChooser.getSelectedFile();

        // Cek jika file sudah ada
        if (fileToSave.exists()) {
            int confirm = JOptionPane.showConfirmDialog(
                    this,
                    "File sudah ada.\nApakah Anda ingin menimpanya?",
                    "Konfirmasi",
                    JOptionPane.YES_NO_OPTION,
                    JOptionPane.WARNING_MESSAGE
            );
            if (confirm != JOptionPane.YES_OPTION) {
                return;
            }
        }

        try (FileWriter fw = new FileWriter(fileToSave)) {

            DefaultTableModel model = (DefaultTableModel) jTblMusik.getModel();
            int colCount = model.getColumnCount();
            int rowCount = model.getRowCount();

            // Tulis isi tabel
            for (int r = 0; r < rowCount; r++) {
                for (int c = 0; c < colCount; c++) {
                    Object value = model.getValueAt(r, c);
                    String cell = (value == null) ? "" : value.toString().replace(";", ",");
                    fw.write(cell);
                    if (c < colCount - 1) {
                        fw.write(";");
                    }
                }
                fw.write("\n");
            }

            JOptionPane.showMessageDialog(
                    this,
                    "Data musik berhasil diekspor ke file CSV!\n" + fileToSave.getAbsolutePath(),
                    "Sukses",
                    JOptionPane.INFORMATION_MESSAGE
            );

        } catch (Exception e) {
            JOptionPane.showMessageDialog(
                    this,
                    "Terjadi kesalahan saat menyimpan:\n" + e.getMessage(),
                    "Error",
                    JOptionPane.ERROR_MESSAGE
            );
            e.printStackTrace();
        }
    }
}
```

KESIMPULAN

Dari praktikum ke 14 ini, dapat disimpulkan bahwa implementasi CRUD dengan fitur upload dan download CSV berbasis Pemrograman Berorientasi Objek (PBO) merupakan cara yang efektif untuk mengelola data secara terstruktur pada aplikasi Java. Dengan menggunakan PBO, setiap entitas data direpresentasikan sebagai objek, sehingga proses Create, Read, Update, dan Delete dapat dilakukan dengan lebih rapi dan mudah dipelihara.

Selain itu, integrasi upload dan download CSV memungkinkan data dimasukkan dari file eksternal maupun diekspor kembali ke file CSV dengan aman dan efisien. Hal ini memudahkan pertukaran data antar sistem, menjaga integritas data, serta meningkatkan fleksibilitas pengelolaan informasi dalam aplikasi.