LAPORAN TUGAS PEMROGRAMAN BERORIENTASI OBYEK

Tugas ke-12 : Implementasi Fitur Persistence, Report, CSV, dan Keamanan Pada Aplikasi Manajemen Data Sembako Menggunakan Java.

Mata Kuliah : Pemrograman Berorientasi Obyek

Dosen Pengampu : Bayu Adhi Nugroho, Ph.D.

Oleh :

Erikhotun Najwa Aghnia (09010624005)

KELAS H7B.3

SISTEM INFORMASI

FAKULTAS SAINS DAN TEKNOLOGI

UIN SUNAN AMPEL SURABAYA

2025

A. PENDAHULUAN

Perkembangan teknologi informasi menuntut adanya sistem yang dapat mengelola data dengan efisien, aman, dan terstruktur. Pada mata kuliah Pemrograman Berorientasi Objek (PBO), kami diajak untuk mengimplementasikan berbagai konsep pemrograman berbasis objek dalam membangun aplikasi yang fungsional dan mudah dikembangkan. Laporan ini disusun untuk mendokumentasikan proses pengembangan aplikasi manajemen data sembako yang dilengkapi dengan fitur-fitur seperti:

- Pembuatan database dan desain antarmuka grafis (GUI) menggunakan JFrame dan JDialog.
- Implementasi persistence dengan Entity Class.
- Integrasi iReport untuk mencetak laporan.
- Fitur import dan export data dalam format CSV.
- Penggunaan Tabbed Pane untuk menampilkan multiple tabel.
- Fitur penghapus data (Clear) pada tabel.
- Sistem login dengan pertanyaan keamanan untuk meningkatkan keamanan akses.

Tujuan dari tugas ini adalah untuk melatih kemampuan dalam merancang dan mengimplementasikan aplikasi desktop yang terintegrasi dengan database, memiliki fitur laporan, serta menerapkan mekanisme keamanan sederhana.

B. PRAKTIKUM

1. **Membuat Database dan Desain Project dengan JFrame Form dan JDialog Form**

   Berikut link laporannya

   https://drive.google.com/file/d/1DV27FIII3DhBaU4tHaZMERiTTDr0PKFt/view?usp=drive_link

2. **Mengimplementasikan Persistence dengan Entity Class**

   Berikut link laporannya

   https://drive.google.com/file/d/1wFv8jXQpOh7NAVx59VVbbkP7Y-A7JHuH/view?usp=drive_link

3. **Menambahkan iReport dan Fitur Cetak untuk Mencetak Laporan dengan Report Wizard**

   Berikut link laporannya

   https://drive.google.com/file/d/15N6UNKHq1TkAbv-Tqc4TfGw3_aOhLf-y/view?usp=drive_link

4. **Menambahkan Fitur Upload File CSV untuk Mengimport Data**

   Berikut link laporannya

   https://drive.google.com/file/d/1_aA1GDkSJo4xNyf7JHqZK-1CQHvAu6vW/view?usp=drive_link

5. **Mengimplementasikan Tabbed Pane untuk Menambahkan Tabel Kedua**

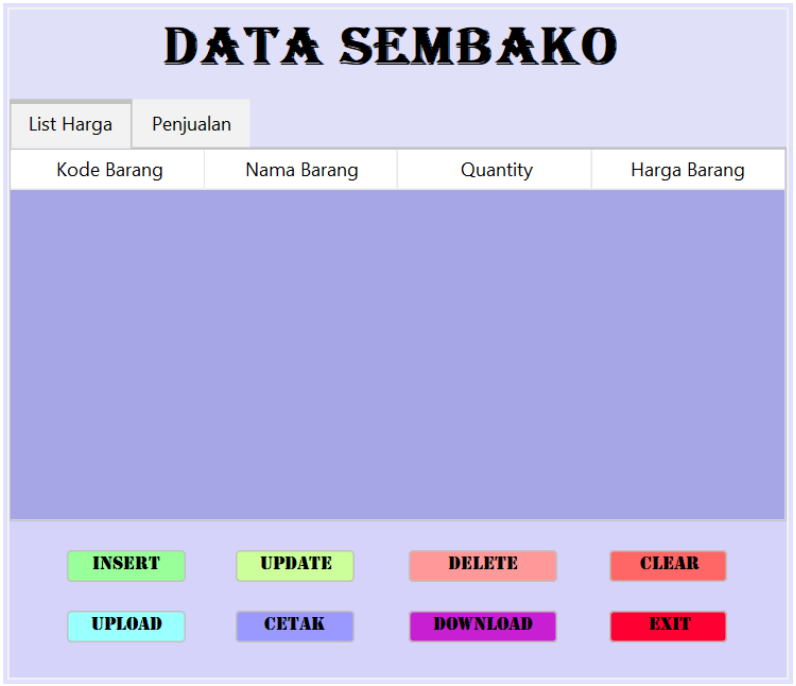
   Berikut link laporannya

   https://drive.google.com/file/d/1W_MpqsB4bQDFwJyMe24T3iZZ7-3lu-6o/view?usp=drive_link

6. Menambahkan Fitur Download untuk Mengekspor Data Format CSV

6.1 Desain GUI

Pada desain GUI tambahkan button untuk tombol **Download**, tambahkan pada kedua table



6.2 Tambahkan Import dan Method pada class DataSembako

```
private void downloadToCSV(JTable table, String filename) {
    JFileChooser fileChooser = new JFileChooser();
    fileChooser.setDialogTitle("Simpan sebagai CSV");
    fileChooser.setSelectedFile(new File(filename));

    int userSelection = fileChooser.showSaveDialog(this);
    if (userSelection == JFileChooser.APPROVE_OPTION) {
        File fileToSave = fileChooser.getSelectedFile();
        // Pastikan ekstensi .csv
        if (!fileToSave.getAbsolutePath().toLowerCase().endsWith(".csv")) {
            fileToSave = new File(fileToSave.getAbsolutePath() + ".csv");
        }
        try (FileWriter writer = new FileWriter(fileToSave)) {
            // Tulis header
            DefaultTableModel model = (DefaultTableModel) table.getModel();
            int columnCount = model.getColumnCount();
            // Header
            for (int i = 0; i < columnCount; i++) {
                writer.write(model.getColumnName(i));
                if (i < columnCount - 1) {
                    writer.write(";");
                }
            }
            writer.write("\n");
            // Data
            for (int i = 0; i < model.getRowCount(); i++) {
                for (int j = 0; j < columnCount; j++) {
                    Object value = model.getValueAt(i, j);
                    writer.write(value != null ? value.toString() : "");
                    if (j < columnCount - 1) {
```

```
                    writer.write(";");
                }
            }
            writer.write("\n");
        }
        JOptionPane.showMessageDialog(this, "Data berhasil didownload ke: "
                + fileToSave.getAbsolutePath());
    } catch (IOException e) {
        JOptionPane.showMessageDialog(this, "Error saat mengdownload data: "
                + e.getMessage(),
                "Error", JOptionPane.ERROR_MESSAGE);
    }
}
}
```

6.3 Pada Button Download berikan source code berikut

Untuk Button Download pada Table List Harga/Sembako

```
downloadToCSV(jTable1, "data_sembako.csv");
```

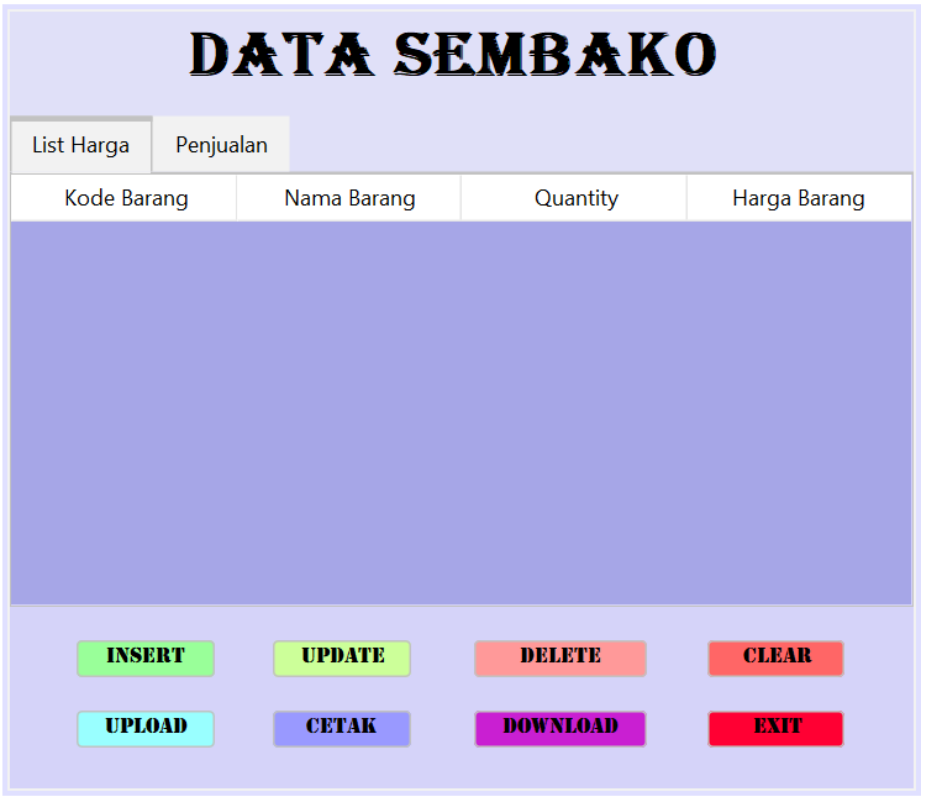
Untuk Button Download pada Table Penjualan/Terjual

```
downloadToCSV(jTable2, "data_penjualan.csv");
```

7. Menambahkan Fitur Clear untuk Mengahapus Semua Data Sekaligus

7.1 Desain GUI

Pada desain GUI tambahkan button untuk tombol **Clear**, tambahkan pada kedua table



7.2 Tambahkan Method pada class DataSembako

```
private void clearSembako() {
    int confirm = JOptionPane.showConfirmDialog(this,
            "Apakah Anda yakin ingin menghapus SEMUA data sembako dan data penjualan?\n"
            + "Data penjualan ikut terhapus karena berikatan dengan data sembako.\n"
            + "Tindakan ini tidak dapat dibatalkan!",
            "Konfirmasi Hapus Semua Data",
            JOptionPane.YES_NO_OPTION,
            JOptionPane.WARNING_MESSAGE);
    if (confirm == JOptionPane.YES_OPTION) {
        EntityManagerFactory emf = Persistence.createEntityManagerFactory("PBOTugas12PU");
        EntityManager em = emf.createEntityManager();
        try {
            em.getTransaction().begin();
            // 1. Hapus semua data dari tabel Terjual terlebih dahulu
            Query deleteTerjual = em.createQuery("DELETE FROM Terjual");
            int deletedTerjualCount = deleteTerjual.executeUpdate();
            // 2. Hapus semua data dari tabel Sembako
            Query deleteSembako = em.createQuery("DELETE FROM Sembako");
            int deletedSembakoCount = deleteSembako.executeUpdate();

            em.getTransaction().commit();
            JOptionPane.showMessageDialog(this,
                    "Berhasil menghapus " + deletedSembakoCount + " data sembako dan "
                    + deletedTerjualCount + " data penjualan.",
                    "Sukses",
                    JOptionPane.INFORMATION_MESSAGE);
            showTable();
        } catch (Exception e) {
            if (em.getTransaction().isActive()) {
                em.getTransaction().rollback();
            }
            JOptionPane.showMessageDialog(this,
                    "Error saat menghapus data: " + e.getMessage(),
                    "Error",
                    JOptionPane.ERROR_MESSAGE);
            e.printStackTrace();
        } finally {
            if (em != null && em.isOpen()) {
                em.close();
            }
            if (emf != null && emf.isOpen()) {
                emf.close();
            }
        }
    }
}

private void clearTerjual() {
    int confirm = JOptionPane.showConfirmDialog(this,
            "Apakah Anda yakin ingin menghapus SEMUA data penjualan?\nTindakan ini tidak dapat dibatalkan!",
            "Konfirmasi Hapus Semua Data",
            JOptionPane.YES_NO_OPTION,
            JOptionPane.WARNING_MESSAGE);
    if (confirm == JOptionPane.YES_OPTION) {
        EntityManagerFactory emf = Persistence.createEntityManagerFactory("PBOTugas12PU");
        EntityManager em = emf.createEntityManager();
        try {
            em.getTransaction().begin();
            // Hapus semua data dari tabel Terjual
            Query deleteQuery = em.createQuery("DELETE FROM Terjual");
```

```
            int deletedCount = deleteQuery.executeUpdate();

            em.getTransaction().commit();
            JOptionPane.showMessageDialog(this,
                    "Berhasil menghapus " + deletedCount + " data penjualan.",
                    "Sukses",
                    JOptionPane.INFORMATION_MESSAGE);
            showTable();
        } catch (Exception e) {
            if (em.getTransaction().isActive()) {
                em.getTransaction().rollback();
            }
            JOptionPane.showMessageDialog(this,
                    "Error saat menghapus data: " + e.getMessage(),
                    "Error",
                    JOptionPane.ERROR_MESSAGE);
        } finally {
            if (em != null && em.isOpen()) {
                em.close();
            }
            if (emf != null && emf.isOpen()) {
                emf.close();
            }
        }
    }
}
```

7.3 Pada Button Clear panggil masing-masing method

Untuk Button Clear pada Table List Harga/Sembako

```
clearSembako();
```

Untuk Button Clear pada Table Penjualan/Terjual

```
clearTerjual();
```

8. Menambahkan Login pada Awal Project

Berikut link laporannya

https://drive.google.com/file/d/1h_PUep9shUrkM5x8Qq_4HFoUp0v2L-C-/view?usp=drive_link

9. Menambahkan Pertanyaan Keamanan Pada Login

Menambahkan pertanyaan keamanan guna memperkuat keamanan untuk menghindari pengubahan password oleh orang yang tidak bertanggungjawab.

Alurnya: Ketika Daftar/membuat akun user akan diminta untuk membuiat pertanyaan dan jawabannya. Ketika user lupa password dan

ingin merubah password pertanyaan tersebut akan muncul denga JOptionPane dan meminta user untuk menjawabnya, jika jawabannya benar maka user dapat melanjutkan mengubah password namun jika jawabannya salah maka user tidak dapat mengubah password.

9.1 Menambahkan Kolom pada Database

Query | Query History

```
1  ALTER TABLE login ADD COLUMN pertanyaan VARCHAR(255);
2  ALTER TABLE login ADD COLUMN jawaban VARCHAR(100);
```

9.2 Menambahkan Field baru serta Getter dan Setter untuk Pertanyaan Keamanan di Entity Login

```
@Basic(optional = false)
@Column(name = "pertanyaan")
private String pertanyaan;

@Basic(optional = false)
@Column(name = "jawaban")
private String jawaban;
public String getPertanyaan() {
    return pertanyaan;
}

public void setPertanyaan(String pertanyaan) {
    this.pertanyaan = pertanyaan;
}

public String getJawaban() {
    return jawaban;
}

public void setJawaban(String jawaban) {
    this.jawaban = jawaban;
}
```

9.3 Desain GUI pada jDialog Daftar

Menggunakan TextField untuk Pertanyaan dan PasswordField untuk Jawaban agar orang lain tidak mengetahuinya

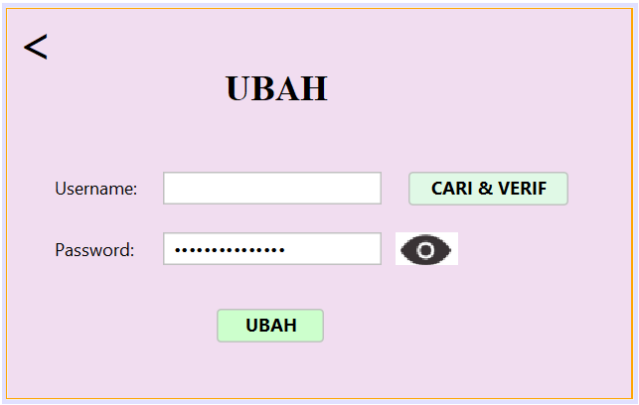
9.4 Modifikasi Source Code Button Buat

```
    private void btnBuatActionPerformed(java.awt.event.ActionEvent evt) {
        if (tfUser.getText().equals("") || jPasswordField1.getText().equals("")
            || tfPertanyaan.getText().equals("") || jPasswordField2.getPassword().length == 0) {
            JOptionPane.showMessageDialog(null, "Isi semua data");
        }else {
            String user, pw, pertanyaan, jawaban;
            user = tfUser.getText();
            pw = jPasswordField1.getText();
            pertanyaan = tfPertanyaan.getText();
            jawaban = new String(jPasswordField2.getPassword());

            EntityManagerFactory emf = Persistence.createEntityManagerFactory("PBOTugas12PU");
            EntityManager em = emf.createEntityManager();
            em.getTransaction().begin();

            Login_1 y = em.find(Login_1.class, user);
            if (y != null) {
                JOptionPane.showMessageDialog(null, "Username sudah ada, gunakan username lain");
                bersih();
                tfUser.requestFocus();
            } else {
                int confirm = JOptionPane.showConfirmDialog(
            null,
            "Pastikan Anda mengingat pertanyaan dan jawaban keamanan!\n\n" +
            "Pertanyaan: " + pertanyaan + "\n" +
            "Username: " + user,
            "Konfirmasi Pendaftaran",
            JOptionPane.YES_NO_OPTION,
            JOptionPane.WARNING_MESSAGE
        );
        if (confirm == JOptionPane.YES_OPTION) {
            Login_1 x = new Login_1();
            x.setUsername(user);
            x.setPassword(pw);
            x.setPertanyaan(pertanyaan);
            x.setJawaban(jawaban);
            em.persist(x);
            em.getTransaction().commit();

            JOptionPane.showMessageDialog(null, "Akun berhasil dibuat!");
            bersih();
            this.dispose();
        } else {
            em.getTransaction().rollback();
        }
            }
            em.close();
            emf.close();
        }
    }
```

Jangan lupa method bersihnya

```
public void bersih() {
    tfUser.setText("");
    jPasswordField1.setText("");
    tfPertanyaan.setText("");
    jPasswordField2.setText("");
}
```

9.5 Desain GUI pada jDialog Ubah



9.6 Modifikasi Source Code Cari&Verif

```
private void btnCariActionPerformed(java.awt.event.ActionEvent evt) {
    if (tfUser.getText().equals("")) {
        JOptionPane.showMessageDialog(null, "Isi Username Terlebih Dahulu");
        return;
    }
        EntityManagerFactory emf = Persistence.createEntityManagerFactory("PBOTugas12PU");
        EntityManager em = emf.createEntityManager();
    try {
        em.getTransaction().begin();
        String user = tfUser.getText();
        Login_1 y = em.find(Login_1.class, user);
        if (y == null) {
            JOptionPane.showMessageDialog(null, "Username tidak ditemukan");
            bersih();
            tfUser.requestFocus();
        } else {
            JPasswordField isiJawaban = new JPasswordField();
            Object[] message = {
                "Pertanyaan Keamanan Anda:",
                y.getPertanyaan(),
                "Jawaban:", isiJawaban
            };
            int option = JOptionPane.showConfirmDialog(
                    null,
                    message,
                    "Verifikasi Keamanan",
                    JOptionPane.OK_CANCEL_OPTION,
                    JOptionPane.QUESTION_MESSAGE
            );
            if (option == JOptionPane.OK_OPTION) {
                String jawaban = new String(isiJawaban.getPassword());
                if (jawaban.trim().equalsIgnoreCase(y.getJawaban())) {
                    lblPass.setVisible(true);
                    btnUbah.setVisible(true);
                    jPasswordField1.setVisible(true);
                    lblBuka1.setVisible(true);
                    lblTutup.setVisible(false);
                    jPasswordField1.requestFocus();
                      JOptionPane.showMessageDialog(null,
                              "Verifikasi berhasil!\nSilakan masukkan password baru di form.",
                              "Berhasil",
                              JOptionPane.INFORMATION_MESSAGE);
                } else {
                    JOptionPane.showMessageDialog(null,
                            "Jawaban salah!\nPastikan jawaban sesuai dengan yang Anda buat saat pendaftaran.",
                            "Error",
                            JOptionPane.ERROR_MESSAGE);
                    tfUser.requestFocus();
                }
            }
        }
        em.getTransaction().commit();
    } finally {
        em.close();
        emf.close();
    }
}
```

9.7 Update Persistence.xml

```
<property name="javax.persistence.schema-generation.database.action" value="update"/>
```

C. KESIMPULAN

Berdasarkan pelaksanaan tugas ini, dapat disimpulkan bahwa:

1. Penggunaan JFrame dan JDialog Form memudahkan dalam merancang antarmuka pengguna yang interaktif dan terstruktur.
2. Entity Class berperan penting dalam mengelola persistensi data ke database secara efisien.
3. Integrasi iReport memungkinkan pembuatan laporan cetak yang rapi dan profesional.
4. Fitur import dan export CSV meningkatkan fleksibilitas dalam pertukaran data dengan sistem lain.
5. Tabbed Pane memudahkan pengguna dalam mengelola banyak data dalam satu jendela.
6. Fitur Clear dan Download memberikan kemudahan dalam manajemen data secara massal.
7. Implementasi login dengan pertanyaan keamanan berhasil meningkatkan lapisan keamanan aplikasi, terutama dalam hal pengubahan password/kata sandi.