LAPORAN PRAKTIKUM PEMROGRAMAN BERORIENTASI OBYEK PERTEMUAN EMPATBELAS

Disusun untuk memenuhi tugas : Halaman Login, Register, Lupa Password Koneksi IReport, Jasper, Download dan Upload File CSV dan JFrame Form Netbeans dengan JPA (Java Persistence API) dan Dihubungkan dengan Foreign Key di Kontainer Netbeans.

Dosen Pengampu :

Bayu Adhi Nugroho Ph. D

Disusun oleh :

Alif Faiz Fadhilah (09040624080)

Program Studi Sistem Informasi

Fakultas Sains dan Teknologi

Universitas Islam Negeri Sunan Ampel Surabaya

2025

A. PENDAHULUAN

1. HALAMAN LOGIN

Halaman login merupakan halaman awal yang digunakan pengguna untuk mengakses sistem dengan memasukkan username dan password. Pada aplikasi ini, proses autentikasi dilakukan dengan menghubungkan halaman login ke database menggunakan JPA (Java Persistence API). JPA berfungsi sebagai penghubung antara program Java dan database, sehingga data pengguna dapat diakses atau diverifikasi secara langsung melalui entitas (entity class).

Pada implementasinya di NetBeans, dibuat kelas entitas bernama Login yang berisi atribut seperti id, username, dan password. Kelas ini diberi anotasi @Entity agar dikenali sebagai representasi tabel pada database. JPA akan secara otomatis memetakan kelas tersebut ke tabel yang ada di database, misalnya tabel login.

Saat pengguna menekan tombol Login, sistem akan menjalankan query menggunakan EntityManager untuk memeriksa apakah kombinasi username dan password yang dimasukkan sesuai dengan data yang tersimpan di database.
Jika data valid, pengguna akan diarahkan ke halaman utama aplikasi. Namun, jika data tidak ditemukan atau salah, maka sistem akan menampilkan pesan kesalahan.

Dengan pendekatan ini, proses login menjadi lebih aman dan efisien karena seluruh interaksi data dikelola oleh JPA tanpa perlu menulis query SQL secara manual. Selain itu, penggunaan JPA memudahkan pengembangan aplikasi karena mendukung konsep Object Relational Mapping (ORM), yang memungkinkan objek Java langsung terhubung dengan tabel database.

2. PRIMARY KEY DAN FOREIGN KEY

Primary Key adalah atribut atau kombinasi atribut pada sebuah tabel yang berfungsi untuk membedakan setiap baris data agar bersifat unik dan tidak boleh kosong. Primary key digunakan sebagai identitas utama suatu record, sehingga tidak ada dua data yang memiliki nilai primary key yang sama.

Foreign Key adalah atribut pada suatu tabel yang digunakan untuk menghubungkan data dengan tabel lain melalui referensi ke primary key tabel tersebut. Foreign key berfungsi untuk menjaga hubungan antar-tabel dan memastikan konsistensi serta integritas data di dalam basis data.

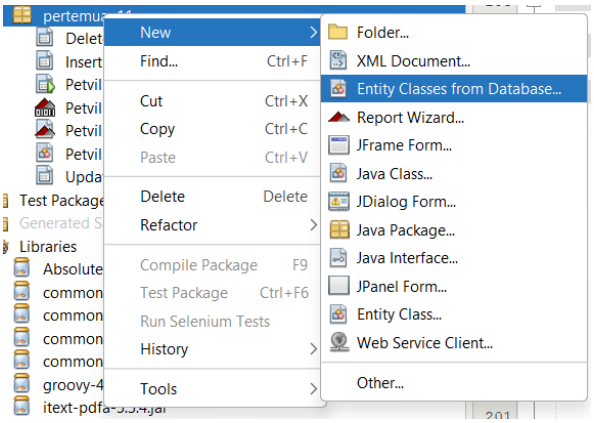
3. KONTAINER GUI

Kontainer dua tab pada NetBeans GUI menggunakan komponen JTabbedPane yang berfungsi untuk menampilkan beberapa panel dalam satu jendela aplikasi. Melalui komponen ini, pengguna dapat berpindah antar-tab dengan mudah tanpa membuka form baru. Setiap tab berisi tampilan atau fungsi yang berbeda, misalnya tab pertama menampilkan data pemilik, sedangkan tab kedua menampilkan data hewan. Dengan penggunaan JTabbedPane, tampilan aplikasi menjadi lebih rapi, terstruktur, dan efisien karena semua fitur dapat diakses dalam satu frame utama. Selain itu, komponen ini juga memudahkan pengelompokan data atau fitur yang saling berhubungan, sehingga meningkatkan kenyamanan dan kemudahan bagi pengguna dalam mengoperasikan aplikasi.

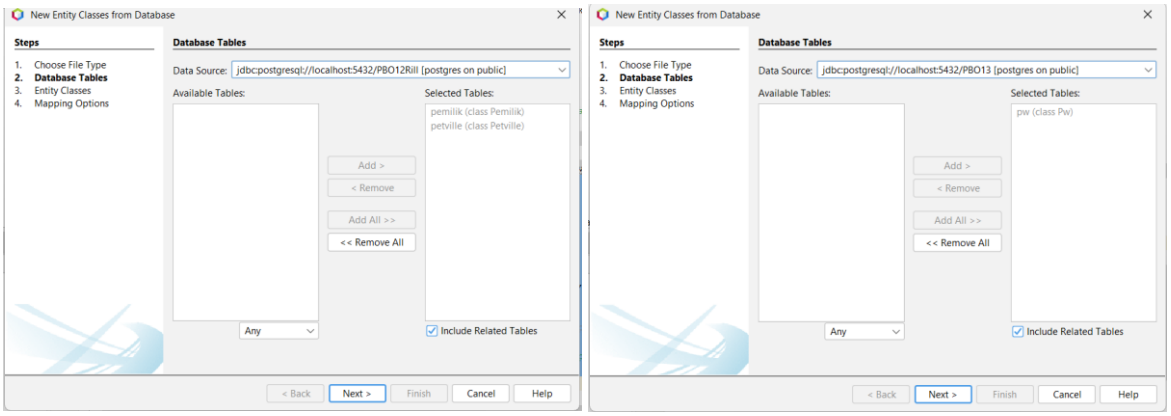
B. PRAKTIKUM MEMBUAT ENTITY CLASSES FROM DATABASE

I. Pembuatan Entity Classes From Database

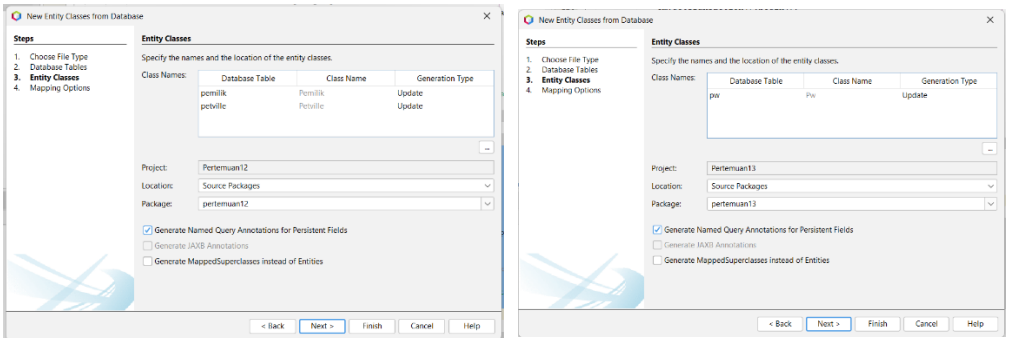
a. Klik New di package pertemuan Kontainer, lalu pilih Entity Classes from Database



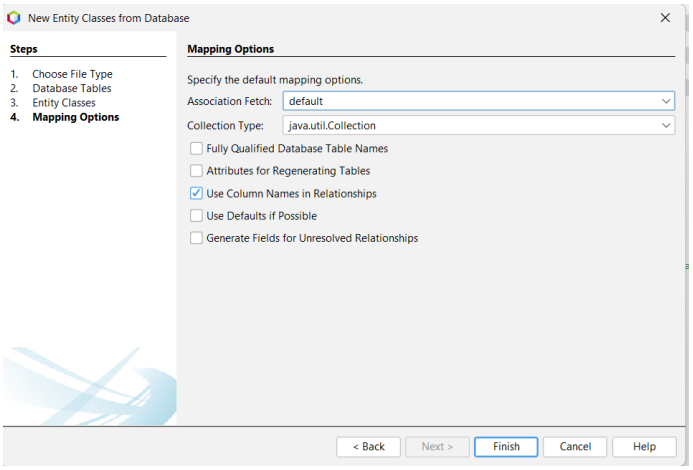
b. Pilih Data Source yang ingin kamu sambungkan sesuai database kamu, Setelah muncul tabel di avaible tables, pindahkan tabel nya ke kanan, lalu pilih Next



c. Centang pada bagian Generate Named Query Annotation for Persistent Field, Lalu next



d. Centang pada bagian Use Column Names in Relationship, lalu klik finish



II. Pembuatan Entity Classses From Database dan Persistence

a. Pembuatan Entity Classes From Database

```
package pertemuan11;

import java.io.Serializable;
import javax.persistence.Basic;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.NamedQueries;
import javax.persistence.NamedQuery;
import javax.persistence.Table;

/**
 *
 * @author Alif
 */
@Entity
@Table(name = "petville")
@NamedQueries({
    @NamedQuery(name = "Petville_1.findAll", query = "SELECT p FROM Petville_1 p ORDER BY p.idHewan ASC"),
    @NamedQuery(name = "Petville_1.findByIdHewan", query = "SELECT p FROM Petville_1 p WHERE p.idHewan = :idHewan"),
    @NamedQuery(name = "Petville_1.findByNamaHewan", query = "SELECT p FROM Petville_1 p WHERE p.namaHewan = :namaHewan"),
    @NamedQuery(name = "Petville_1.findByJenisHewan", query = "SELECT p FROM Petville_1 p WHERE p.jenisHewan = :jenisHewan"),
    @NamedQuery(name = "Petville_1.findByHarga", query = "SELECT p FROM Petville_1 p WHERE p.harga = :harga")})
public class Petville_1 implements Serializable {

    private static final long serialVersionUID = 1L;
    @Id
    @Basic(optional = false)
    @Column(name = "id_hewan")
    private String idHewan;
    @Column(name = "nama_hewan")
    private String namaHewan;
    @Column(name = "jenis_hewan")
    private String jenisHewan;
    @Column(name = "harga")
    private Integer harga;

    public Petville_1() {
    }

    public Petville_1(String idHewan) {
        this.idHewan = idHewan;
    }

    public String getIdHewan() {
        return idHewan;
    }

    public void setIdHewan(String idHewan) {
        this.idHewan = idHewan;
    }

    public String getNamaHewan() {
        return namaHewan;
    }

    public void setNamaHewan(String namaHewan) {
        this.namaHewan = namaHewan;
    }

    public String getJenisHewan() {
        return jenisHewan;
    }

    public void setJenisHewan(String jenisHewan) {
        this.jenisHewan = jenisHewan;
    }

    public Integer getHarga() {
        return harga;
    }

    public void setHarga(Integer harga) {
        this.harga = harga;
    }

    @Override
    public int hashCode() {
        int hash = 0;
        hash += (idHewan != null ? idHewan.hashCode() : 0);
        return hash;
    }

    @Override
    public boolean equals(Object object) {
        // TODO: Warning - this method won't work in the case the id fields are not set
        if (!(object instanceof Petville_1)) {
            return false;
        }
        Petville_1 other = (Petville_1) object;
        if ((this.idHewan == null && other.idHewan != null) || (this.idHewan != null && !this.idHewan.equals(other.idHewan))) {
            return false;
        }
        return true;
    }

    @Override
    public String toString() {
        return "pertemuan11.Petville_1[ idHewan=" + idHewan + " ]";
    }
```

b. Pembuatan Persistence

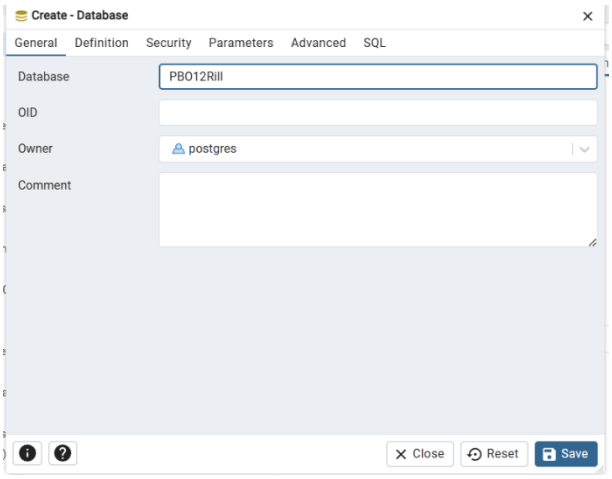
```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.2" xmlns="http://xmlns.jcp.org/xml/ns/persistence" xmlns:xsi="http://www.w3.org
  <!-- Database pertama -->
  <persistence-unit name="DBPertamaPU" transaction-type="RESOURCE_LOCAL">
    <provider>org.eclipse.persistence.jpa.PersistenceProvider</provider>
    <class>Kontainer.Pemilik</class>
    <class>Kontainer.Petville</class>
    <properties>
      <property name="javax.persistence.jdbc.url" value="jdbc:postgresql://localhost:5432/PBO12Rill"/>
      <property name="javax.persistence.jdbc.user" value="postgres"/>
      <property name="javax.persistence.jdbc.password" value="090406"/>
      <property name="javax.persistence.jdbc.driver" value="org.postgresql.Driver"/>
    </properties>
  </persistence-unit>

  <!-- Database kedua -->
  <persistence-unit name="DBKeduaPU" transaction-type="RESOURCE_LOCAL">
    <provider>org.eclipse.persistence.jpa.PersistenceProvider</provider>
    <class>Kontainer.Pw</class>
    <properties>
      <property name="javax.persistence.jdbc.url" value="jdbc:postgresql://localhost:5432/PBO13"/>
      <property name="javax.persistence.jdbc.user" value="postgres"/>
      <property name="javax.persistence.jdbc.password" value="090406"/>
      <property name="javax.persistence.jdbc.driver" value="org.postgresql.Driver"/>
    </properties>
  </persistence-unit>
</persistence>
```

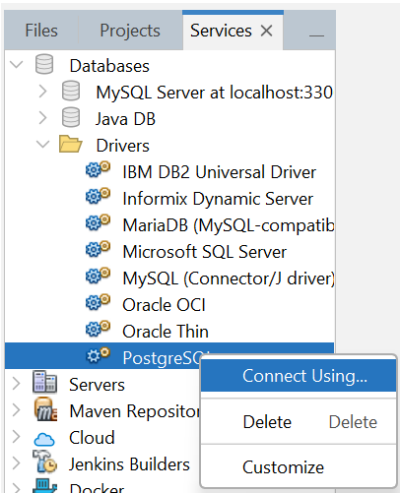
C. PRAKTIKUM CODING FORM GUI

III. Pembuatan DataBase di Postgresql dan Connection ke Netbeans

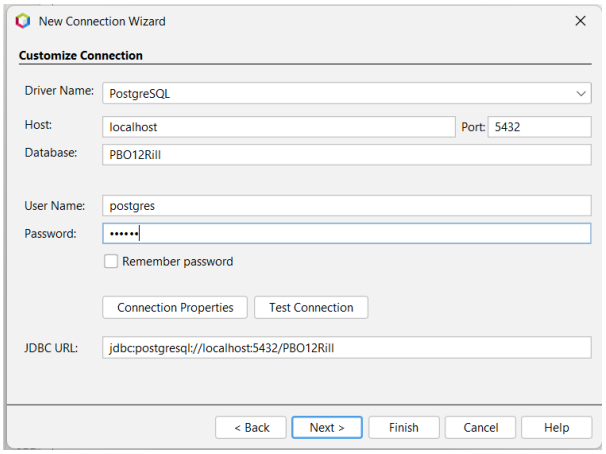
Sebelum mengconnectkan harus membuat Database baru di pg Admin terlebih dahulu, dengan cara klik Kanan pada Database, pilih **Create**, dan klik **Database… Alt + Shift + N**

Setelah diklik akan muncul jendela untuk memberi nama pada Database yang akan dibuat, setelah memberi nama lalu klik **Save**



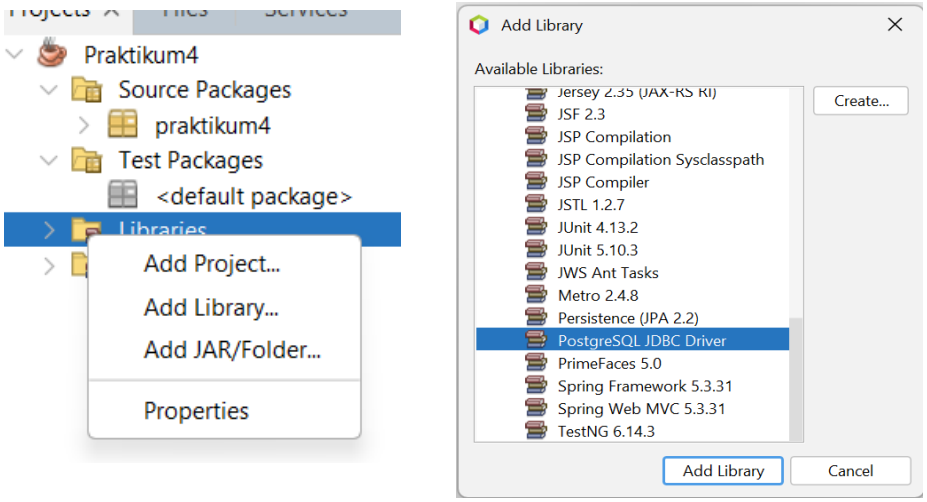


Selanjutnya setelah membuat Database maka baru dapat diconnectkan dengan Netbeans, dengan cara pada bagian **Services** di Netbeans, pilih **Database**, pilih **Drivers**, klik Kanan pada **PostgreSQL** dan pilih **Connect Using**. Kemudian akan muncul jendela Customize Connection, pada bagian **Database** sesuaikan dengan nama Database yang di pg Admin dan masukkan **Password** pg Admin anda, klik **Test Connection** jika sukses jangan lupa menyalin **JDBC URL** nya, lalu klik **Next**.

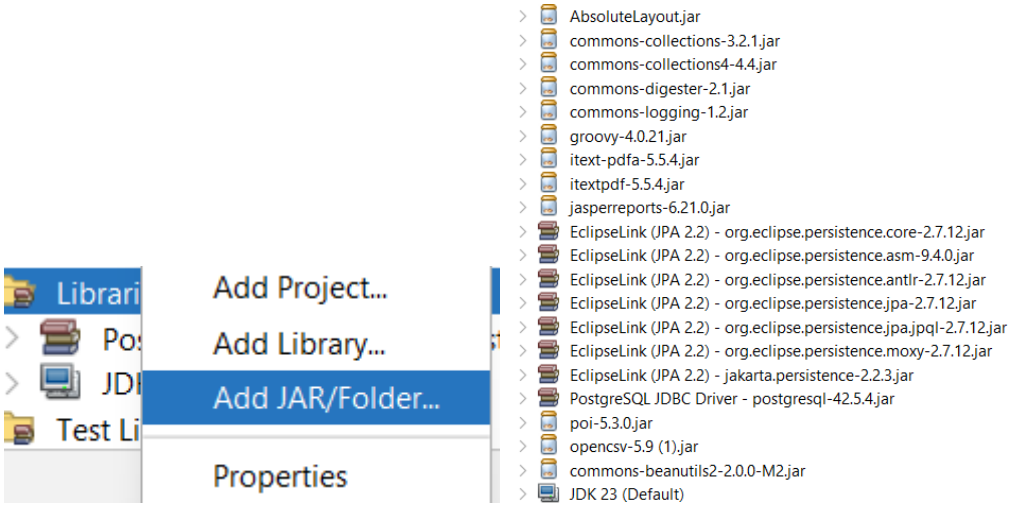


IV. Penambahan Library Postgresql dan Jasper ke Project nya

a. Buka Projectnya, klik Kanan pada **Library**, pilih **Add Library**. Kemudian muncul jendela untuk memilih library, cari **PostgreSQL JDBC Driver** dan klik **Add Library**.

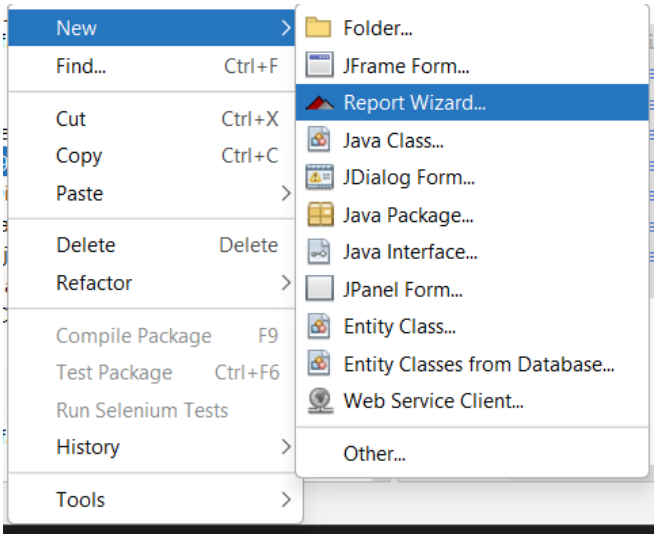


b. Buka Projectnya, klik Kanan pada **Library**, pilih **Add JAR/Folder**. Kemudian muncul jendela untuk memilih library, cari **Library yang sesuai dengan jasper (seperti gambar dibawah)** dan klik **Open**.

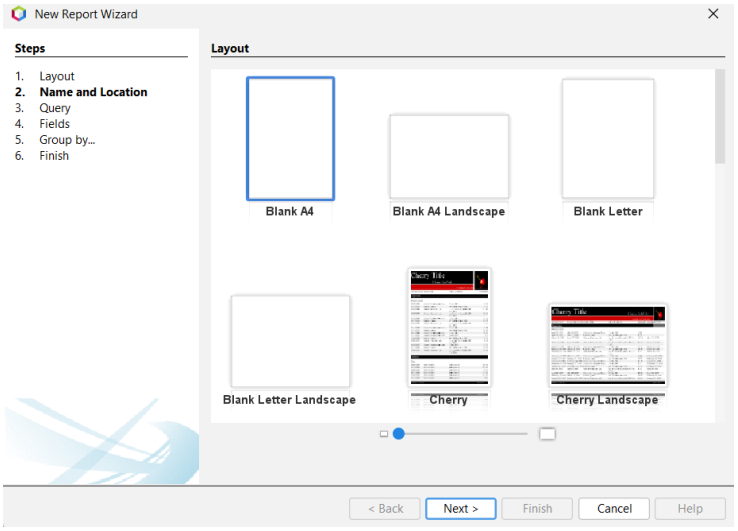


V. Penambahan Report Wizard ke Package

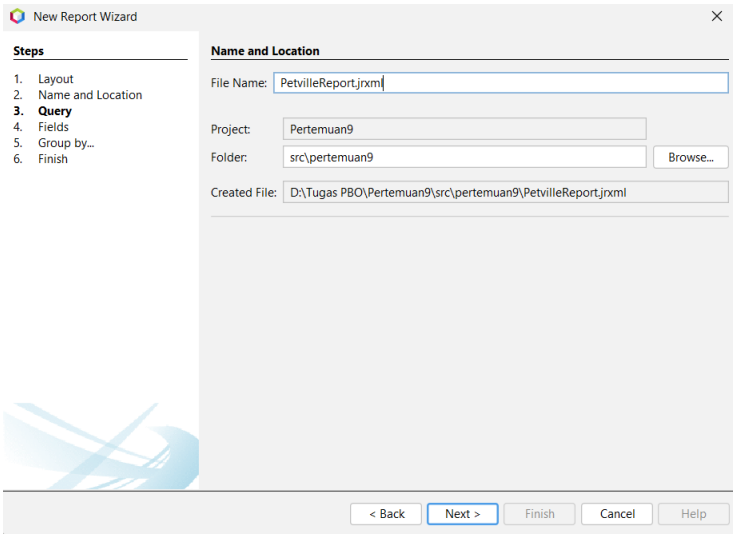
- Klik Kanan pada package lalu pilih New, Kemudian Pilih Menu Report Wizard.



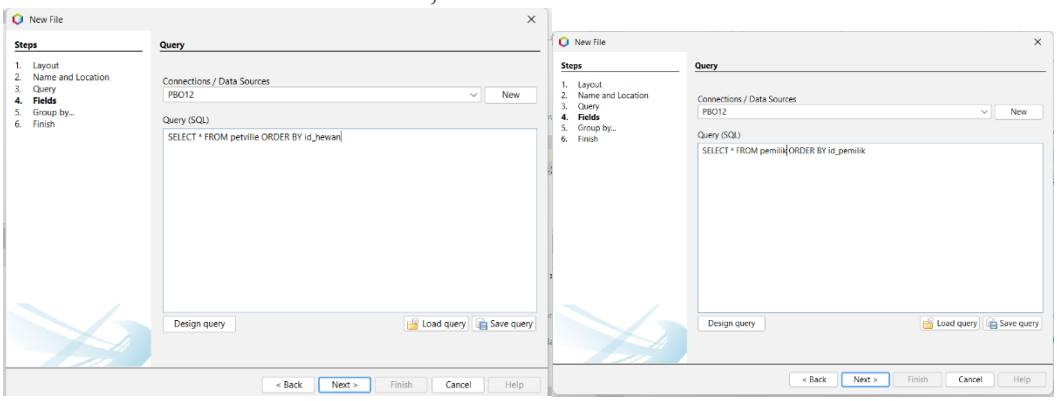
- Pilih jenis kertas sesuai dengan keinginan, lalu Klik Next.



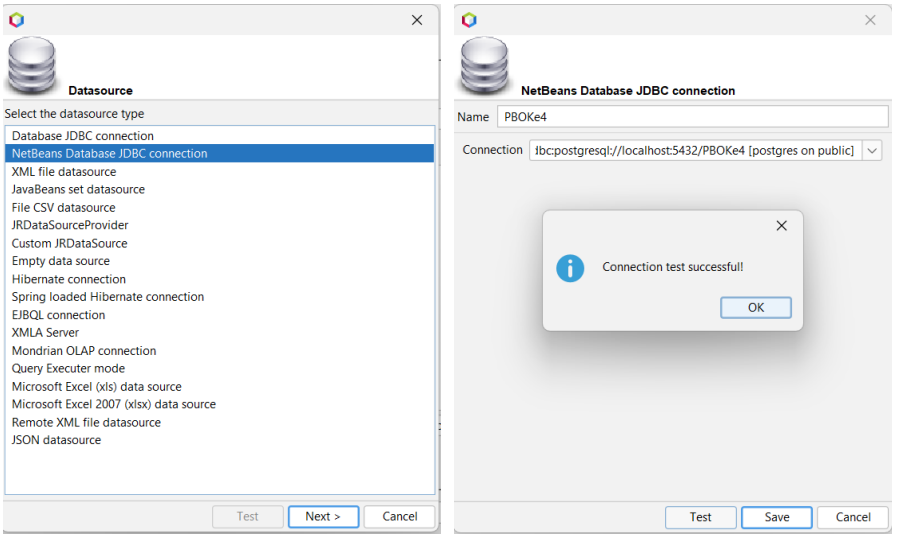
- Ganti nama file jaspernya sesuai keinginan, lalu Klik Next.



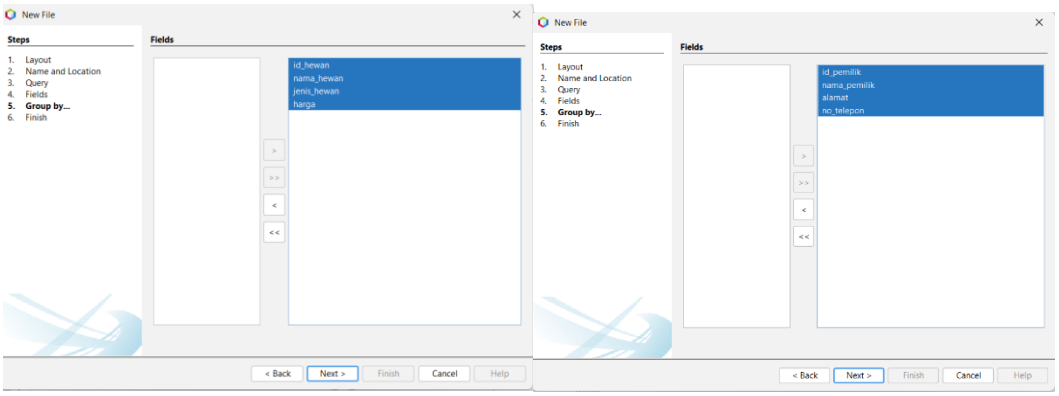
- Sesuaikan nama database kamu, kemudian klik New



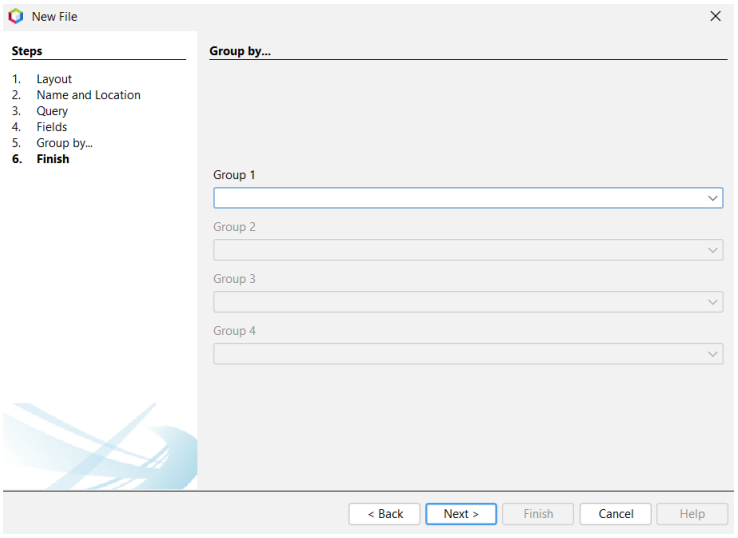
- Pilih Neatbeans Database JDBC Connecction, lalu pilih connection databasenya dan jangan lupa kasih nama, Klik test jika berhasil Klik Save.



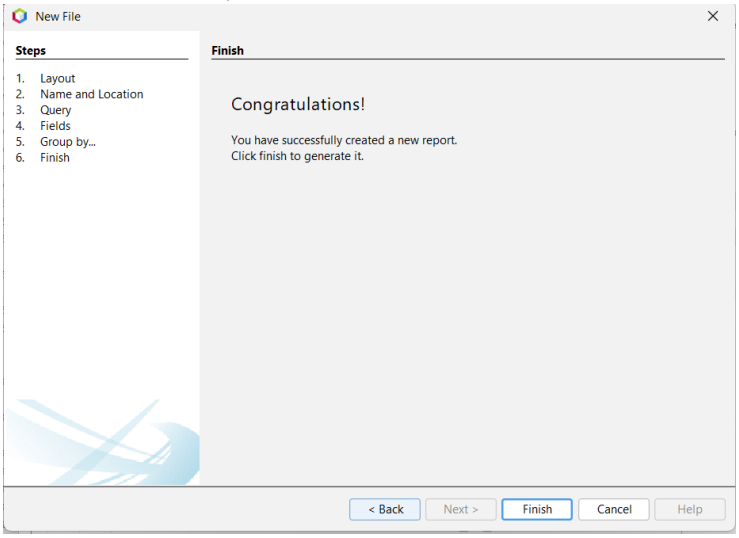
- Pindahkan semua fields ke kanan, lalu Klik Next.



- Next aja.

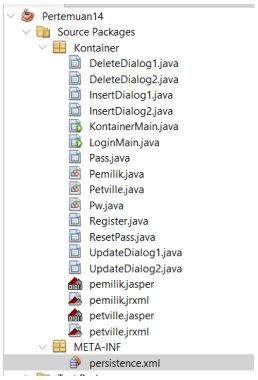


- Sudah selesaii, dan Klik Finish.



VI. Pembuatan Satu Package berisi Jframe Form dan Report Jasper dan JAP

Membuat satu package di Projek Pertemuan13 bernama Kontainer. Tambahkan satu JFrame Form yang ada kontainernya, enam JDialog masing-masing seperti (InsertDialog, UpdateDialog, DeleteDialog), tambahkan dua jasper dan dua jrmxl serta Entity Classes From Database dan persistence.



VII. Pembuatan Satu Class dan Satu JFrame Form

1. Pembuatan Jframe Form

Kelas ini bertanggung jawab atas design dan proses interaktif terhadap apa yang di lakukan user kepada form tersebut. Berisi import library, public class JualBeliHewan yang extends Jframe, method koneksi, method bersih dan method tampilkan data.

```
 */
import java.io.*;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.persistence.*;
import javax.swing.*;
import javax.swing.filechooser.FileSystemView;
import javax.swing.table.DefaultTableModel;
import net.sf.jasperreports.engine.*;
import net.sf.jasperreports.engine.util.JRLoader;
import net.sf.jasperreports.view.JasperViewer;

public class Petville extends javax.swing.JFrame {

    // EntityManager untuk JPA
    private EntityManagerFactory emf;
    private EntityManager em;

    // Variabel data
    String id_hewan, nama_hewan, jenis_hewan;
    int harga;

    // Konfigurasi PostgreSQL (khusus untuk laporan Jasper)
    String DB_URL = "jdbc:postgresql://localhost:5432/PBOKe4";
    String USER = "postgres";
    String PASS = "090406";
    String DRIVER = "org.postgresql.Driver";
```

```
public KontainerMain() {

    initComponents();
    initJPA();

    tabelPetville = jTable1;
    tabelPemilik = jTable2;

    loadDataPetville();
    loadDataPemilik();
}

private void initJPA() {
    try {
        emf = Persistence.createEntityManagerFactory("DBPertamaPU");
        em = emf.createEntityManager();
        System.out.println("Koneksi ke PostgreSQL berhasil!");
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, "Koneksi ke database gagal: " + e.getMessage());
    }
}
```

```
private void loadDataPetville() {
    EntityManager em = emf.createEntityManager();
    try {
        List<Petville> listPet = em.createQuery(
                "SELECT p FROM Petville p ORDER BY p.idHewan", Petville.class
        ).getResultList();

        DefaultTableModel model = (DefaultTableModel) jTable1.getModel();
        model.setRowCount(0); // hapus semua baris lama

        for (Petville p : listPet) {
            model.addRow(new Object[]{
                p.getIdHewan(),
                p.getNamaHewan(),
                p.getJenisHewan(),
                p.getHarga(),
                p.getIdPemilik().getIdPemilik()
            });
        }

    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, "Gagal memuat data: " + e.getMessage());
    } finally {
        em.close();
    }
}
```

```
private void loadDataPemilik() {


    EntityManager em = emf.createEntityManager();
    try {
        List<Pemilik> list = em.createQuery("SELECT p FROM Pemilik p ORDER BY p.idPemilik", Pemilik.class).getResultList();

        DefaultTableModel model = (DefaultTableModel) jTable2.getModel();
        model.setRowCount(0); // hapus semua baris lama


        for (Pemilik p : list) {

            model.addRow(new Object[]{
                p.getIdPemilik(),
                p.getNamaPemilik(),
                p.getAlamat(),
                p.getNoTelepon(),});
        }

        jTable2.setModel(model);
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, "Gagal menampilkan data Pemilik: " + e.getMessage());
    } finally {
        em.close();
    }

}
```

2. Memprogram Button Insert dan Update yang terhubung ke JDialog

Method ini memungkinkan pengguna untuk memasukkan data baru ke dalam tabel. Menyediakan fungsionalitas “Insert” dari operasi CRUD, memungkinkan penambahan data baru ke database. Kelas ini memungkinkan pengguna untuk memperbarui data yang sudah ada di tabel. Menyediakan fungsionalitas “Update” dari operasi CRUD.

```
private void InsertButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    InsertDialog1 dialog = new InsertDialog1(this, true); // true = modal
    dialog.setLocationRelativeTo(this); // supaya muncul di tengah
    dialog.setVisible(true);

    loadDataPetville();
}
```

```
private void UpdateButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    int row = jTable1.getSelectedRow();
    if (row != -1) {
        String idHewan = jTable1.getValueAt(row, 0).toString();
        String namaHewan = jTable1.getValueAt(row, 1).toString();
        String jenisHewan = jTable1.getValueAt(row, 2).toString();
        int harga = Integer.parseInt(jTable1.getValueAt(row, 3).toString());
        String idPemilik = jTable1.getValueAt(row, 4).toString();

        UpdateDialog1 dialog = new UpdateDialog1(this, true, idHewan, namaHewan, jenisHewan, harga, idPemilik);
        dialog.setVisible(true);

        loadDataPetville();
    }else {
        JOptionPane.showMessageDialog(this, "Pilih data yang ingin dihapus terlebih dahulu!");
    }
}
```

- Insert dan Update Kedua

```
private void InsertButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    InsertDialog2 dialog = new InsertDialog2(this, true); // true = modal
    dialog.setLocationRelativeTo(this); // supaya muncul di tengah
    dialog.setVisible(true);

    loadDataPemilik();
}
```

```
private void UpdateButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    int row = jTable2.getSelectedRow();
    if (row != -1) {
        String idPemilik = jTable2.getValueAt(row, 0).toString();
        String namaPemilik = jTable2.getValueAt(row, 1).toString();
        String alamat = jTable2.getValueAt(row, 2).toString();
        String noTelepon = jTable2.getValueAt(row, 3).toString();

        UpdateDialog2 dialog = new UpdateDialog2(this, true, idPemilik, namaPemilik, alamat, noTelepon);
        dialog.setVisible(true);

        loadDataPemilik(); // ganti sesuai method untuk refresh tabel pemilik
    }else {
        JOptionPane.showMessageDialog(this, "Pilih data yang ingin diupdate terlebih dahulu!");
    }

}
```

3. **Memprogram Button Delete yang terhubung dengan Jdialog dan Button Cetak**
Kelas ini memungkinkan pengguna untuk menghapus satu baris data dari tabel dan untuk mencetak kertas jasper yang sudah tersambung.

```
    private void DeleteButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        int selectedRow = jTable1.getSelectedRow();
        if (selectedRow != -1) {
            // Ambil data dari JTable
            String id_hewan = jTable1.getValueAt(selectedRow, 0).toString();
            String nama_hewan = jTable1.getValueAt(selectedRow, 1).toString();
            String jenis_hewan = jTable1.getValueAt(selectedRow, 2).toString();
            int harga = Integer.parseInt(jTable1.getValueAt(selectedRow, 3).toString());
            String id_pemilik = jTable1.getValueAt(selectedRow, 4).toString(); // kolom FK id_pemilik

            // Kirim data ke DeleteDialog1 (yang sudah kamu modifikasi)
            DeleteDialog1 dialog = new DeleteDialog1(this, true, id_hewan, nama_hewan, jenis_hewan, harga, id_pemilik);
            dialog.setLocationRelativeTo(this); // supaya muncul di tengah layar
            dialog.setVisible(true);

            // Refresh data tabel setelah hapus
            loadDataPetville();
        } else {
            JOptionPane.showMessageDialog(this, "Pilih data yang ingin diupdate terlebih dahulu!");
        }
    }

    private void CetakButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        JasperReport reports;
        String path = ".\\src\\pertemuan12\\petville.jasper";
        try {
            Class.forName(DRIVER);
            java.sql.Connection conn = java.sql.DriverManager.getConnection(DB_URL, USER, PASS);
            reports = (JasperReport) JRLoader.loadObjectFromFile(path);
            JasperPrint jprint = JasperFillManager.fillReport(reports, null, conn);
            JasperViewer viewer = new JasperViewer(jprint, false);
            viewer.setDefaultCloseOperation(DISPOSE_ON_CLOSE);
            viewer.setVisible(true);
        } catch (ClassNotFoundException | java.sql.SQLException | JRException ex) {
            Logger.getLogger(Petville.class.getName()).log(Level.SEVERE, null, ex);
            JOptionPane.showMessageDialog(this, "Gagal mencetak laporan: " + ex.getMessage());
        }
    }
```

- Delete dan Cetak Kedua

```
    private void DeleteButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        int selectedRow = jTable2.getSelectedRow();
        if (selectedRow != -1) {
            // Ambil data dari JTable
            String id_pemilik = jTable2.getValueAt(selectedRow, 0).toString();
            String nama_pemilik = jTable2.getValueAt(selectedRow, 1).toString();
            String alamat = jTable2.getValueAt(selectedRow, 2).toString();
            String no_telepon = jTable2.getValueAt(selectedRow, 3).toString();

            // Kirim data ke DeleteDialog2 (yang sudah kamu modifikasi)
            DeleteDialog2 dialog = new DeleteDialog2(this, true, id_pemilik, nama_pemilik, alamat, no_telepon);
            dialog.setLocationRelativeTo(this); // supaya muncul di tengah layar
            dialog.setVisible(true);

            // Refresh data tabel setelah hapus
            loadDataPemilik();
        } else {
            JOptionPane.showMessageDialog(this, "Pilih data yang ingin dihapus terlebih dahulu!");
        }
    }
```

```
    private void CetakButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        JasperReport reports;
        String path = ".\\src\\pertemuan12\\pemilik.jasper";
        try {
            Class.forName(DRIVER);
            java.sql.Connection conn = java.sql.DriverManager.getConnection(DB_URL, USER, PASS);
            reports = (JasperReport) JRLoader.loadObjectFromFile(path);
            JasperPrint jprint = JasperFillManager.fillReport(reports, null, conn);
            JasperViewer viewer = new JasperViewer(jprint, false);
            viewer.setDefaultCloseOperation(DISPOSE_ON_CLOSE);
            viewer.setVisible(true);
        } catch (ClassNotFoundException | java.sql.SQLException | JRException ex) {
            Logger.getLogger(Petville.class.getName()).log(Level.SEVERE, null, ex);
            JOptionPane.showMessageDialog(this, "Gagal mencetak laporan: " + ex.getMessage());
        }
    }
```

4. Pembuatan Button Upload

Kelas ini memungkinkan pengguna untuk mengupload satu file bertipe csv.

```
private void UploadButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    JFileChooser jfc = new JFileChooser(FileSystemView.getFileSystemView().getHomeDirectory());
    int returnValue = jfc.showOpenDialog(null);
    if (returnValue == JFileChooser.APPROVE_OPTION) {
        File filePilihan = jfc.getSelectedFile();
        try (BufferedReader br = new BufferedReader(new FileReader(filePilihan))) {
            String baris;
            String pemisah = ";";
            em.getTransaction().begin();
            while ((baris = br.readLine()) != null) {
                baris = baris.replace("\uFEFF", "").trim();
                String[] data = baris.split(pemisah);

                if (data.length < 5 || data[0].equalsIgnoreCase("id_hewan")) {
                    continue;
                }

                String idHewan = data[0].trim();
                String namaHewan = data[1].trim();
                String jenisHewan = data[2].trim();
                int harga = Integer.parseInt(data[3].trim());
                String idPemilikStr = data[4].trim();

                // 🔍 Cari objek Pemilik berdasarkan ID
                Pemilik pemilik = em.find(Pemilik.class, idPemilikStr);
                if (pemilik == null) {
                    System.out.println("⚠ Pemilik tidak ditemukan untuk ID: " + idPemilikStr);
                    continue;
                }

                // Buat entitas Petville baru
                Petville p = new Petville(idHewan);
                p.setNamaHewan(namaHewan);
                p.setJenisHewan(jenisHewan);
                p.setHarga(harga);
                p.setIdPemilik(pemilik); // ✅ gunakan objek Pemilik, bukan String

                em.persist(p);
            }

            em.getTransaction().commit();
            JOptionPane.showMessageDialog(this, "Upload selesai!");
            loadDataPetville();

        } catch (Exception ex) {
            JOptionPane.showMessageDialog(this, "Gagal upload: " + ex.getMessage());
            ex.printStackTrace();
        }
    }
```

- Upload Kedua

```
    private void UploadButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        JFileChooser jfc = new JFileChooser(FileSystemView.getFileSystemView().getHomeDirectory());
        int returnValue = jfc.showOpenDialog(null);

        if (returnValue == JFileChooser.APPROVE_OPTION) {
            File filePilihan = jfc.getSelectedFile();

            EntityManagerFactory emf = Persistence.createEntityManagerFactory("Pertemuan12PU");
            EntityManager em = emf.createEntityManager();

            try (BufferedReader br = new BufferedReader(new FileReader(filePilihan))) {
                String baris;
                String pemisah = ";";

                em.getTransaction().begin();

                while ((baris = br.readLine()) != null) {
                    // Hilangkan karakter BOM dan spasi
                    baris = baris.replace("\uFEFF", "").trim();
                    String[] data = baris.split(pemisah);

                    // Lewati header dan baris tidak lengkap
                    if (data.length < 4 || data[0].equalsIgnoreCase("id_pemilik")) {
                        continue;
                    }

                    // Ambil data dari CSV
                    String idPemilik = data[0].trim();
                    String namaPemilik = data[1].trim();
                    String alamat = data[2].trim();
                    String noTelepon = data[3].trim();

                    // Cek apakah data dengan ID ini sudah ada
                    Pemilik existing = em.find(Pemilik.class, idPemilik);
                    if (existing == null) {
                        // Buat objek baru
                        Pemilik p = new Pemilik(idPemilik);
                        p.setNamaPemilik(namaPemilik);
                        p.setAlamat(alamat);
                        p.setNoTelepon(noTelepon);

                        em.persist(p);
                    } else {
                        // Jika sudah ada, update data-nya
                        existing.setNamaPemilik(namaPemilik);
                        existing.setAlamat(alamat);
                        existing.setNoTelepon(noTelepon);
                        em.merge(existing);
                    }
```

```
                }

                em.getTransaction().commit();
                JOptionPane.showMessageDialog(this, "Upload data Pemilik selesai!");

                // Refresh tabel setelah upload
                loadDataPemilik(); // pastikan kamu punya method ini untuk refresh JTable

            } catch (Exception ex) {
                JOptionPane.showMessageDialog(this, "Gagal upload: " + ex.getMessage());
                ex.printStackTrace();
            } finally {
                em.close();
                emf.close();
            }
        }
    }
```

5. Pembuatan Button Download

- Buat Method Downloadnya

```
public void exportTableToCSV(JTable table, String fileName) {
    JFileChooser fileChooser = new JFileChooser();
    fileChooser.setSelectedFile(new File(fileName));

    int option = fileChooser.showSaveDialog(null);

    if (option == JFileChooser.APPROVE_OPTION) {
        File file = fileChooser.getSelectedFile();

        try (BufferedWriter bw = new BufferedWriter(new FileWriter(file))) {

            TableModel model = table.getModel();
            int colCount = model.getColumnCount();

            // Tulis header
            for (int i = 0; i < colCount; i++) {
                bw.write(model.getColumnName(i));
                if (i < colCount - 1) {
                    bw.write(";");
                }
            }
            bw.newLine();

            // Tulis isi baris
            for (int row = 0; row < model.getRowCount(); row++) {
                for (int col = 0; col < colCount; col++) {
                    Object value = model.getValueAt(row, col);
                    bw.write(value == null ? "" : value.toString());
                    if (col < colCount - 1) {
                        bw.write(";");
                    }
                }
                bw.newLine();
            }

            JOptionPane.showMessageDialog(null,
                    "File CSV berhasil disimpan:\n" + file.getAbsolutePath());

        } catch (Exception e) {
            JOptionPane.showMessageDialog(null,
                    "Gagal menyimpan CSV: " + e.getMessage());
        }
    }
}
```

- Button Download 1

```
private void DownloadButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    exportTableToCSV(jTable1, "data_hewan.csv");
}
```

- Button Download 2

```
private void DownloadButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    exportTableToCSV(jTable2, "data_pemilik.csv");
}
```

6. Pembuatan Button Delete All

- Button Delete All 1 dan 2

```
private void DeleteAllButtonActionPerformed(java.awt.event.ActionEvent evt) {
    int confirm = JOptionPane.showConfirmDialog(this,
            "Yakin ingin menghapus SEMUA data akun (tabel Hewan)?",
            "Konfirmasi Hapus Semua",
            JOptionPane.YES_NO_OPTION);

    if (confirm != JOptionPane.YES_OPTION) {
        return;
    }

    EntityManagerFactory emf = Persistence.createEntityManagerFactory("DBPertamaPU");
    EntityManager em = emf.createEntityManager();

    try {

        em.getTransaction().begin();

        // Ambil semua data dari Entity Pw
        List<Petville> data = em.createQuery("SELECT p FROM Petville p", Petville.class).getResultList();

        // Hapus satu per satu
        for (Petville item : data) {
            em.remove(em.contains(item) ? item : em.merge(item));
        }

        em.getTransaction().commit();

        // Jika kamu punya fungsi refresh tabel, panggil di sini:
        // showTable();
        JOptionPane.showMessageDialog(this, "Semua data tabel Petville berhasil dihapus!");

        loadDataPetville();

    } catch (Exception e) {
        e.printStackTrace();
        JOptionPane.showMessageDialog(this, "Gagal menghapus data PW!");

    } finally {
        em.close();
        emf.close();
    }
```

```
private void DeleteAllButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    int confirm = JOptionPane.showConfirmDialog(this,
            "Yakin ingin menghapus SEMUA data akun (tabel Pemilik)?",
            "Konfirmasi Hapus Semua",
            JOptionPane.YES_NO_OPTION);

    if (confirm != JOptionPane.YES_OPTION) {
        return;
    }

    EntityManagerFactory emf = Persistence.createEntityManagerFactory("DBPertamaPU");
    EntityManager em = emf.createEntityManager();

    try {

        em.getTransaction().begin();

        // Ambil semua data dari Entity Pw
        List<Pemilik> data = em.createQuery("SELECT p FROM Pemilik p", Pemilik.class).getResultList();

        // Hapus satu per satu
        for (Pemilik item : data) {
            em.remove(em.contains(item) ? item : em.merge(item));
        }

        em.getTransaction().commit();

        // Jika kamu punya fungsi refresh tabel, panggil di sini:
        // showTable();
        JOptionPane.showMessageDialog(this, "Semua data tabel Pemilik berhasil dihapus!");

        loadDataPetville();

    } catch (Exception e) {
        e.printStackTrace();
        JOptionPane.showMessageDialog(this, "Gagal menghapus data PW!");

    } finally {
        em.close();
        emf.close();
    }
}
```

7. Pembuatan Main Class

Kode main tersebut berfungsi untuk mengatur tampilan aplikasi menggunakan Nimbus (jika tersedia), kemudian menjalankan dan menampilkan form utama JualBeliHewan di layar melalui thread khusus GUI agar aplikasi tetap responsif.

```
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    Look and feel setting code (optional)

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new Petville().setVisible(true);
        }
    });
}
```

VIII. Pembuatan JFRAME Lupa Password

Kelas ini bertanggung jawab atas design dan proses interaktif terhadap apa yang di lakukan user kepada form tersebut. Memungkinkan pengguna untuk mencari data username ke dalam tabel.

```
*/
import javax.persistence.*;
import javax.swing.JOptionPane;
public class Pass extends javax.swing.JFrame {

    /**
     * Creates new form Pass
     */
    public Pass() {
        initComponents();
    }

    @SuppressWarnings("unchecked")
    Generated Code

    private void tfUsernameActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void ButtonForgotActionPerformed(java.awt.event.ActionEvent evt) {
        EntityManagerFactory emf = Persistence.createEntityManagerFactory("Pertemuan13PU");
        EntityManager em = emf.createEntityManager();

        try {
            String username = tfUsername.getText();

            if (username.isEmpty()) {
                JOptionPane.showMessageDialog(this, "Silakan isi username terlebih dahulu!");
                return;
            }

            Pw user = em.find(Pw.class, username);

            if (user != null) {
                JOptionPane.showMessageDialog(this, "Username ditemukan. Silakan ubah password Anda.");
                // buka form reset password dan kirim username
                new ResetPass(username).setVisible(true);
                this.dispose();
            } else {
                JOptionPane.showMessageDialog(this, "Username tidak ditemukan!");
            }

        } finally {
            em.close();
            emf.close();
        }
    }
```

IX. Pembuatan Jframe Reset Password

Kelas ini bertanggung jawab atas design dan proses interaktif terhadap apa yang di lakukan user kepada form tersebut. Berisi import library, public class. Memungkinkan pengguna untuk memperbarui data password lama menjadi yang baru.

```
 */
import javax.persistence.*;
import javax.swing.JOptionPane;

public class ResetPass extends javax.swing.JFrame {

    private String username;

    /**
     * Creates new form ResetPass
     */
    public ResetPass(String username) {
        initComponents();
        setLocationRelativeTo(null);
        this.username = username;
    }
```

```
    private void ButtonResetPasswordActionPerformed(java.awt.event.ActionEvent evt) {
        EntityManagerFactory emf = Persistence.createEntityManagerFactory("Pertemuan13PU");
        EntityManager em = emf.createEntityManager();

        try {
            String newPass1 = new String(pfPassword.getPassword());
            String newPass2 = new String(pfConfirmPassword.getPassword());

            if (newPass1.isEmpty() || newPass2.isEmpty()) {
                JOptionPane.showMessageDialog(this, "Password tidak boleh kosong!");
                return;
            }

            if (!newPass1.equals(newPass2)) {
                JOptionPane.showMessageDialog(this, "Password tidak cocok!");
                return;
            }

            pertemuan13.Pw user = em.find(pertemuan13.Pw.class, username);

            if (user != null) {
                em.getTransaction().begin();
                user.setPassword(newPass1);
                em.getTransaction().commit();

                JOptionPane.showMessageDialog(this, "Password berhasil diubah!");
                this.dispose();
                new LoginMain().setVisible(true); // ganti dengan nama form login kamu
            } else {
                JOptionPane.showMessageDialog(this, "Username tidak ditemukan!");
            }

        } finally {
            em.close();
            emf.close();
        }
    }
```

X. Pembuatan Jframe Register Akun dan Dashboard

Kelas ini bertanggung jawab atas design dan proses interaktif terhadap apa yang di lakukan user kepada form tersebut. Memungkinkan pengguna untuk menambah atau membuat akun baru dan untuk akses halaman setelah Login.

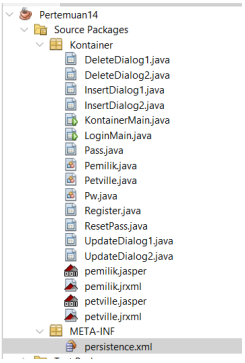
```
private void ButtonSubmitActionPerformed(java.awt.event.ActionEvent evt) {
     EntityManagerFactory emf = Persistence.createEntityManagerFactory("DBKedu
    EntityManager em = emf.createEntityManager();

    try {
        String username = tfUsername.getText();
        String hobi = tfHobi.getText();
        String password = new String(pfPassword.getPassword());


        Kontainer.Pw existingUser = em.find(Kontainer.Pw.class, username);

        if (existingUser != null) {
            JOptionPane.showMessageDialog(this, "Username sudah digunakan!");
            return;
        }

        Kontainer.Pw newUser = new Kontainer.Pw();
        newUser.setUsername(username);
        newUser.setHobi(hobi);
        newUser.setPassword(password);

        em.getTransaction().begin();
        em.persist(newUser);
        em.getTransaction().commit();

        JOptionPane.showMessageDialog(this, "Pendaftaran berhasil!");
        LoginMain l = new LoginMain();
        l.setVisible(true);
        this.dispose();
    } finally {
        em.close();
        emf.close();
    }
}
```

```
public class Dashboard extends javax.swing.JFrame {

    /**
     * Creates new form Dashboard
     */
    public Dashboard() {
        initComponents();
    }
```

D. PRAKTIKUM CODING JDIALOG

XI. Pembuatan Satu Package berisi Jframe Form dan Report Jasper dan JAP

Membuat satu package di Projek Pertemuan14 bernama Kontainer. Tambahkan satu JFrame Form, tiga JDialog seperti (InsertDialog, UpdateDialog, DeleteDialog), tambahkan satu jasper dan satu jrmxl serta Entity Classes From Database dan peersistence.



XII. Pembuatan JDialog Insert

Kelas ini bertanggung jawab atas design dan proses interaktif terhadap apa yang di lakukan user kepada form tersebut.

```
public InsertDialog1(java.awt.Frame parent, boolean modal) {
    super(parent, modal);
    initComponents();

    EntityManagerFactory emf = Persistence.createEntityManagerFactory("DBPertamaPU");
    em = emf.createEntityManager();

    loadPemilikCombo(); // = tambahkan ini

    System.out.println("Koneksi ke PostgreSQL berhasil!");
}

public void bersih() {
    tfIDHewan.setText("");
    tfNamaHewan.setText("");
    tfJenisHewan.setText("");
    tfHarga.setText("");
    IDPemilikComboBox1.setSelectedIndex(-1);

}

private void loadPemilikCombo() {
    IDPemilikComboBox1.removeAllItems();

    try {
        TypedQuery<Pemilik> query = em.createQuery("SELECT p FROM Pemilik p ORDER BY p.idPemilik ASC", Pemilik.class);
        for (Pemilik p : query.getResultList()) {
            // Format: ID - Nama
            IDPemilikComboBox1.addItem(p.getIdPemilik() + " - " + p.getNamaPemilik());
        }
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, "Gagal memuat data pemilik: " + e.getMessage());
    }
}
```

- Dialog Insert Kedua

```
import java.sql.*;
import javax.swing.JOptionPane;
import javax.persistence.*;

public class InsertDialog2 extends javax.swing.JDialog {

    // --- EntityManagerFactory dibuat sekali untuk seluruh kelas ---
    private static final EntityManagerFactory emf
            = Persistence.createEntityManagerFactory("Pertemuan12PU");
    private EntityManager em;

    public InsertDialog2(java.awt.Frame parent, boolean modal) {
        super(parent, modal);
        initComponents();
        em = emf.createEntityManager();
        System.out.println("Koneksi ke PostgreSQL berhasil!");
    }

    public void bersih() {

        tfNamaPemilik.setText("");
        tfAlamat.setText("");
        tfNoTelepon.setText("");
        tfIDPemilik.setText("");
    }
```

Memungkinkan pengguna untuk memasukkan data baru ke dalam tabel. Menyediakan fungsionalitas “Insert” dari operasi CRUD, memungkinkan penambahan data baru ke database

```
private void ButtonInsertActionPerformed(java.awt.event.ActionEvent evt) {
    if (tfIDHewan.getText().isEmpty()
            || tfNamaHewan.getText().isEmpty()
            || tfJenisHewan.getText().isEmpty()
            || tfHarga.getText().isEmpty()
            || IDPemilikComboBox1.getSelectedItem() == null)
             {

        JOptionPane.showMessageDialog(this, "Isi semua data terlebih dahulu!");
        return;
    }

    try {
        int harga = Integer.parseInt(tfHarga.getText());

        em.getTransaction().begin();

        // Cek apakah ID Hewan sudah ada di database
        Petville existingHewan = em.find(Petville.class, tfIDHewan.getText());
        if (existingHewan != null) {
            JOptionPane.showMessageDialog(this, "ID Hewan sudah ada! Gunakan ID lain.");
            em.getTransaction().rollback();
            return;
        }

        // Ambil entity Pemilik berdasarkan id_pemilik yang diinput
        String selected = (String) IDPemilikComboBox1.getSelectedItem();
        if (selected == null || selected.isEmpty()) {
            JOptionPane.showMessageDialog(this, "Silakan pilih pemilik!");
            em.getTransaction().rollback();
            return;
        }

        // Pecah "ID - Nama"
        String idPemilik = selected.split(" - ")[0];

        Pemilik pemilik = em.find(Pemilik.class, idPemilik);
        if (pemilik == null) {
            JOptionPane.showMessageDialog(this, "Pemilik tidak ditemukan!");
            em.getTransaction().rollback();
            return;
        }
```

```
    if (pemilik == null) {
        JOptionPane.showMessageDialog(this, "ID Pemilik tidak ditemukan di database!");
        em.getTransaction().rollback();
        return;
    }

    // Buat entity Petville baru
    Petville hewan = new Petville();
    hewan.setIdHewan(tfIDHewan.getText());
    hewan.setNamaHewan(tfNamaHewan.getText());
    hewan.setJenisHewan(tfJenisHewan.getText());
    hewan.setHarga(harga);
    hewan.setIdPemilik(pemilik); // relasi foreign key

    em.persist(hewan);
    em.getTransaction().commit();

    JOptionPane.showMessageDialog(this, "Data berhasil disimpan ke database!");
    bersih();
    dispose();
} catch (NumberFormatException e) {
    JOptionPane.showMessageDialog(this, "Harga harus berupa angka!");
} catch (Exception e) {
    if (em.getTransaction().isActive()) {
        em.getTransaction().rollback();
    }
    JOptionPane.showMessageDialog(this, "Gagal menyimpan data: " + e.getMessage());
}
```

- SAVE Insert Kedua

```
private void ButtonInsertActionPerformed(java.awt.event.ActionEvent evt) {
    // Validasi input kosong
    if (tfIDPemilik.getText().isEmpty()
            || tfNamaPemilik.getText().isEmpty()
            || tfAlamat.getText().isEmpty()
            || tfNoTelepon.getText().isEmpty()) {

        JOptionPane.showMessageDialog(this, "Isi semua data terlebih dahulu!");
        return;
    }

    try {
        em.getTransaction().begin();

        // Cek apakah ID Pemilik sudah ada di database
        Pemilik existingPemilik = em.find(Pemilik.class, tfIDPemilik.getText());
        if (existingPemilik != null) {
            JOptionPane.showMessageDialog(this, "ID Pemilik sudah ada! Gunakan ID lain.");
            em.getTransaction().rollback();
            return;
        }

        // Buat entity Pemilik baru
        Pemilik pemilik = new Pemilik();
        pemilik.setIdPemilik(tfIDPemilik.getText());
        pemilik.setNamaPemilik(tfNamaPemilik.getText());
        pemilik.setAlamat(tfAlamat.getText());
        pemilik.setNoTelepon(tfNoTelepon.getText());

        // Simpan ke database
        em.persist(pemilik);
        em.getTransaction().commit();

        JOptionPane.showMessageDialog(this, "Data Pemilik berhasil disimpan!");
        bersih();
        dispose();

    } catch (Exception e) {
        if (em.getTransaction().isActive()) {
            em.getTransaction().rollback();
        }
        JOptionPane.showMessageDialog(this, "Gagal menyimpan data: " + e.getMessage());
    }
}
```

XIII. Pembuatan JDialog Update

Kelas ini bertanggung jawab atas design dan proses interaktif terhadap apa yang di lakukan user kepada form tersebut. Berisi import library, public class UpdatetDialog yang extends Jdialog, method koneksi, method bersih dan Konstruktor DialogUpdate.

```
public UpdateDialog1(java.awt.Frame parent, boolean modal, String idHewan, String namaHewan, String jenisHewan, int harga, String idPemilik) {
    super(parent, modal);
    initComponents();

    // Simpan ID lama untuk acuan update
    this.idHewanLama = idHewan;

    // Isi field dengan data lama
    tfIDHewan.setText(idHewan);
    tfNamaHewan.setText(namaHewan);
    tfJenisHewan.setText(jenisHewan);
    tfHarga.setText(String.valueOf(harga));

    // Load combo box dan otomatis pilih yang sesuai
    loadPemilikCombo(idPemilik);

}

public void bersih() {
    tfIDHewan.setText("");
    tfNamaHewan.setText("");
    tfJenisHewan.setText("");
    tfHarga.setText("");
    UpdateComboBox1.setSelectedIndex(-1);
}

private void loadPemilikCombo(String idPemilikLama) {
    UpdateComboBox1.removeAllItems();

    EntityManagerFactory emf = Persistence.createEntityManagerFactory("DBPertamaPU");
    EntityManager em = emf.createEntityManager();

    try {
        var list = em.createQuery(
                "SELECT p FROM Pemilik p ORDER BY p.idPemilik ASC",
                Pemilik.class
        ).getResultList();

        for (Pemilik p : list) {
            String item = p.getIdPemilik() + " - " + p.getNamaPemilik();
            UpdateComboBox1.addItem(item);

            // Auto-select jika ID pemilik cocok dengan hewan lama
            if (p.getIdPemilik().equals(idPemilikLama)) {
                UpdateComboBox1.setSelectedItem(item);
            }
        }
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, "Gagal memuat pemilik: " + e.getMessage());
    } finally {
```

- Dialog Update Kedua

```
import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Persistence;
import javax.swing.JOptionPane;

public class UpdateDialog2 extends javax.swing.JDialog {

    // EntityManagerFactory hanya dibuat sekali untuk seluruh aplikasi
    private static final EntityManagerFactory emf = Persistence.createEntityManagerFactory("Pertemuan12PU");

    private String idPemilikLama; // Simpan ID lama untuk acuan update

    // Konstruktor: menerima data lama dari tabel Pemilik
    public UpdateDialog2(java.awt.Frame parent, boolean modal, String idPemilik, String namaPemilik, String alamat, String
        super(parent, modal);
        initComponents();

        // Simpan ID lama untuk referensi update
        this.idPemilikLama = idPemilik;

        // Isi field dengan data lama
        tfIDPemilik.setText(idPemilik);
        tfNamaPemilik.setText(namaPemilik);
        tfAlamat.setText(alamat);
        tfNoTelepon.setText(noTelepon);
    }

    public void bersih() {
        tfIDPemilik.setText("");
        tfNamaPemilik.setText("");
        tfAlamat.setText("");
        tfNoTelepon.setText("");
    }
```

Memungkinkan pengguna untuk memperbarui data yang sudah ada di tabel. Menyediakan fungsionalitas “Update” dari operasi CRUD.

```
private void ButtonUpdateActionPerformed(java.awt.event.ActionEvent evt) {
    if (tfIDHewan.getText().isEmpty()
            || tfNamaHewan.getText().isEmpty()
            || tfJenisHewan.getText().isEmpty()
            || tfHarga.getText().isEmpty()
            || UpdateComboBox1.getSelectedItem() == null) {

        JOptionPane.showMessageDialog(this, "Isi semua data terlebih dahulu!");
        return;
    }

    try {
        EntityManagerFactory emf = Persistence.createEntityManagerFactory("DBPertamaPU");
        EntityManager em = emf.createEntityManager();

        Petville hewan = em.find(Petville.class, idHewanLama);

        if (hewan != null) {
            em.getTransaction().begin();

            hewan.setIdHewan(tfIDHewan.getText());
            hewan.setNamaHewan(tfNamaHewan.getText());
            hewan.setJenisHewan(tfJenisHewan.getText());
            hewan.setHarga(Integer.parseInt(tfHarga.getText()));

            // Ambil ID Pemilik dari combo box
            String selected = (String) UpdateComboBox1.getSelectedItem();
            String idPemilikBaru = selected.split(" - ")[0];

            Pemilik pemilikBaru = em.find(Pemilik.class, idPemilikBaru);

            if (pemilikBaru == null) {
                JOptionPane.showMessageDialog(this, "ID Pemilik tidak ditemukan!");
                em.getTransaction().rollback();
                return;
            }

            hewan.setIdPemilik(pemilikBaru);
            em.getTransaction().commit();
            JOptionPane.showMessageDialog(this, "Data berhasil diperbarui!");
        } else {
            JOptionPane.showMessageDialog(this, "Data tidak ditemukan!");
        }

        em.close();
        emf.close();
        dispose();
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, "Terjadi kesalahan: " + e.getMessage());
        e.printStackTrace();
    }
```

- SAVE Update Kedua

```
private void ButtonUpdateActionPerformed(java.awt.event.ActionEvent evt) {

if (tfIDPemilik.getText().isEmpty() ||
        tfNamaPemilik.getText().isEmpty() ||
        tfAlamat.getText().isEmpty() ||
        tfNoTelepon.getText().isEmpty()) {

        JOptionPane.showMessageDialog(this, "Isi semua data terlebih dahulu!");
        return;
    }

    EntityManager em = emf.createEntityManager();

    try {
        // Cari data pemilik lama berdasarkan ID
        Pemilik pemilik = em.find(Pemilik.class, idPemilikLama);

        if (pemilik != null) {
            em.getTransaction().begin();

            pemilik.setIdPemilik(tfIDPemilik.getText());
            pemilik.setNamaPemilik(tfNamaPemilik.getText());
            pemilik.setAlamat(tfAlamat.getText());
            pemilik.setNoTelepon(tfNoTelepon.getText());

            em.getTransaction().commit();
            JOptionPane.showMessageDialog(this, "Data pemilik berhasil diperbarui!");
            dispose();
        } else {
            JOptionPane.showMessageDialog(this, "Data tidak ditemukan!");
        }

    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, "Terjadi kesalahan: " + e.getMessage());
        e.printStackTrace();
    } finally {
        em.close();
    }
```

XIV. Pembuatan Jdialog Delete

Kelas ini bertanggung jawab atas design dan proses interaktif terhadap apa yang di lakukan user kepada form tersebut. Berisi import library, public class UpdatetDialog yang extends Jdialog, method koneksi, method bersih dan Konstruktor DialogUpdate.

```
 */
import java.awt.Frame;
import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Persistence;
import javax.swing.JOptionPane;

public class DeleteDialog1 extends javax.swing.JDialog {


    private static final EntityManagerFactory emf = Persistence.createEntityManagerFactory("Pertemuan12PU");

    String id_hewan, nama_hewan, jenis_hewan, id_pemilik;
    int harga;

    public DeleteDialog1(java.awt.Frame parent, boolean modal, String idhewan, String namahewan, String jenishewan, int ha
        super(parent, modal);
        initComponents();

        this.id_hewan = idhewan;
        this.nama_hewan = namahewan;
        this.jenis_hewan = jenishewan;
        this.harga = harga;
        this.id_pemilik = idpemilik;

        DataDelete.setText(id_hewan + " - " + nama_hewan + " - " + jenis_hewan + " - " + harga + " - " + id_pemilik);
    }

    /**
```

- Dialog Delete Kedua

```java
import java.awt.Frame;
import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Persistence;
import javax.swing.JOptionPane;

public class DeleteDialog2 extends javax.swing.JDialog {

// ✅ Gunakan satu EntityManagerFactory global agar efisien
private static final EntityManagerFactory emf = Persistence.createEntityManagerFactory("Pertemuan12PU");

private String id_pemilik;
private String nama_pemilik;
private String alamat;
private String no_telepon;

public DeleteDialog2(java.awt.Frame parent, boolean modal, String id_pemilik, String nama_pemilik, String alamat, String
    super(parent, modal);
    initComponents();

    // Simpan data sementara (untuk tampil di label konfirmasi)
    this.id_pemilik = id_pemilik;
    this.nama_pemilik = nama_pemilik;
    this.alamat = alamat;
    this.no_telepon = no_telepon;

    // Tampilkan ringkasan data yang akan dihapus
    DataDelete.setText(id_pemilik + " - " + nama_pemilik + " - " + alamat + " - " + no_telepon);
}
```

Memungkinkan pengguna untuk memperbarui data yang sudah ada di tabel. Menyediakan fungsionalitas “Delete” dari operasi CRUD.

```java
private void ButtonDeleteActionPerformed(java.awt.event.ActionEvent evt) {
  EntityManager em = null;  // ✅ Buat variabel EntityManager di luar try
    try {
        em = emf.createEntityManager();

        Petville hewan = em.find(Petville.class, id_hewan);
        if (hewan != null) {
            em.getTransaction().begin();
            em.remove(hewan);
            em.getTransaction().commit();
            JOptionPane.showMessageDialog(this, "Data hewan berhasil dihapus!");
        } else {
            JOptionPane.showMessageDialog(this, "Data hewan tidak ditemukan di database!");
        }


        dispose();

    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, "Gagal menghapus data: " + e.getMessage());
        e.printStackTrace();
    } finally {
        // ✅ Pastikan EntityManager ditutup agar koneksi tidak bocor
        if (em != null && em.isOpen()) {
            em.close();
        }
    }

}
```

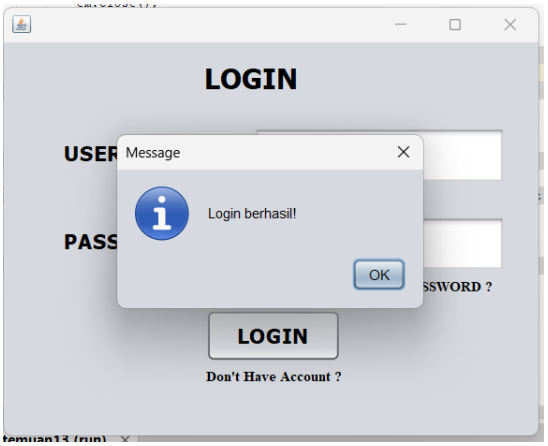
- SAVE Delete Kedua

```java
private void ButtonDeleteActionPerformed(java.awt.event.ActionEvent evt) {
  EntityManager em = null;  // ✅ Buat variabel EntityManager di luar try
    try {
        em = emf.createEntityManager();

        Pemilik pemilik = em.find(Pemilik.class, id_pemilik);
        if (pemilik != null) {
            em.getTransaction().begin();
            em.remove(pemilik);
            em.getTransaction().commit();
            JOptionPane.showMessageDialog(this, "Data Pemilik berhasil dihapus!");
        } else {
            JOptionPane.showMessageDialog(this, "Data Pemilik tidak ditemukan di database!");
        }


        dispose();

    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, "Gagal menghapus data: " + e.getMessage());
        e.printStackTrace();
    } finally {
        // ✅ Pastikan EntityManager ditutup agar koneksi tidak bocor
        if (em != null && em.isOpen()) {
            em.close();
        }
    }

}
```

E. PRAKTIKUM OUPUT FORM GUI dan KERTAS JASPER

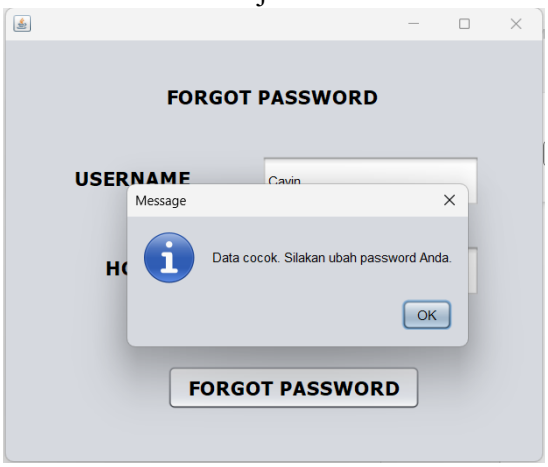
I. Jika Sudah Punya akun Login dan Datanya benar

Berisi data username dan password di dalam PostgreSQL.

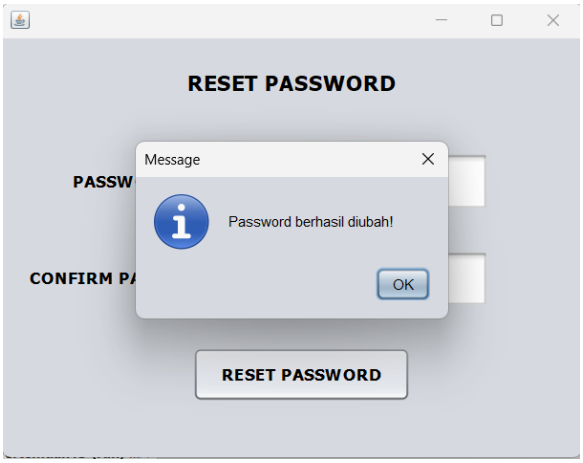


II. Jika Lupa Password Akunnya

- Masukkan Username jika benar maka akan diarahkan ke reset password

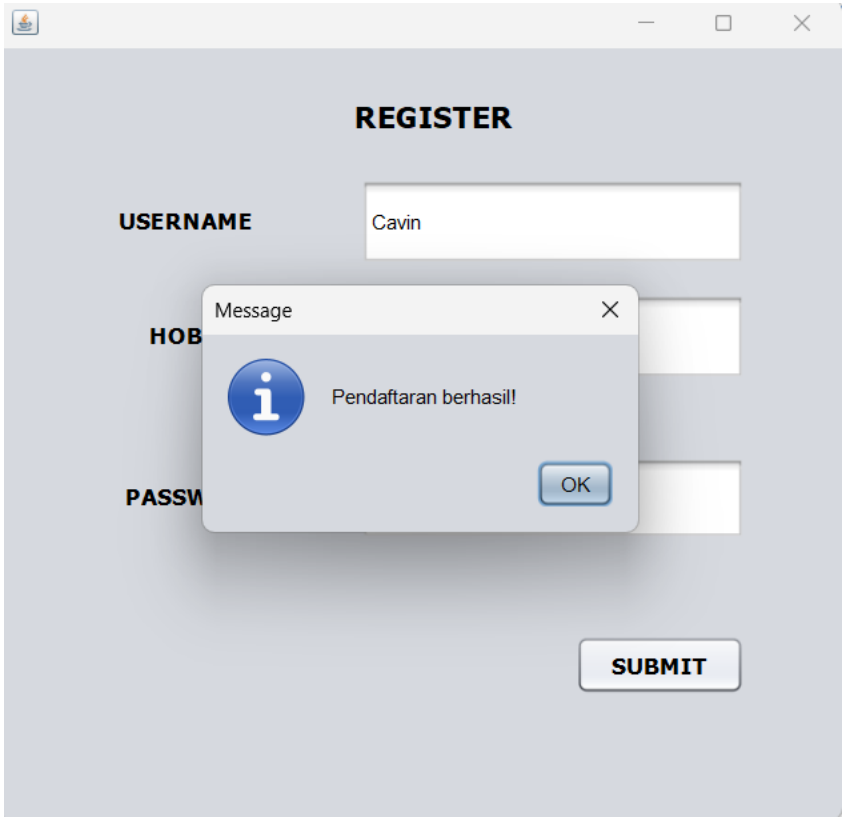


- Ubah Passwordmu jadi password yang baru.



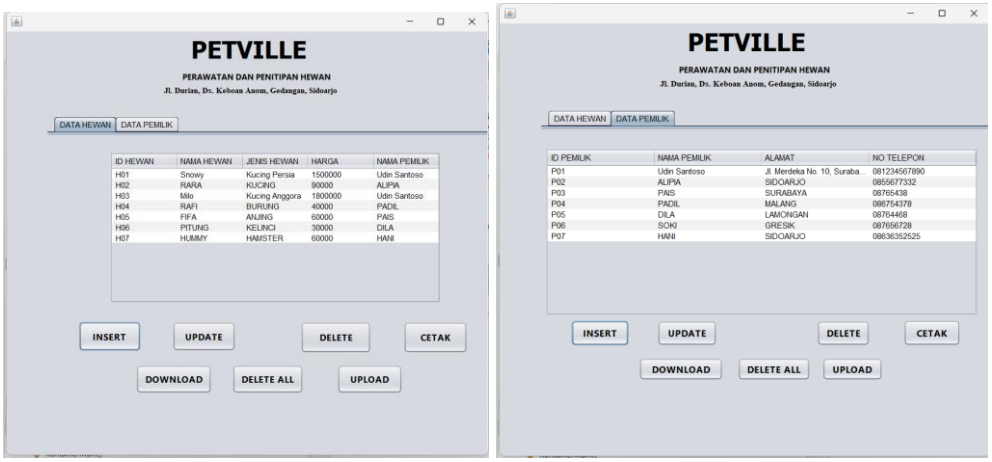
III. Jika Tidak Punya Akun atau Ingin Buat Akun Baru

Masukkan Username dan Password yang ingin kamu jadikan Akun dan pastikan jika pendaftaran atau register akunnya berhasil.

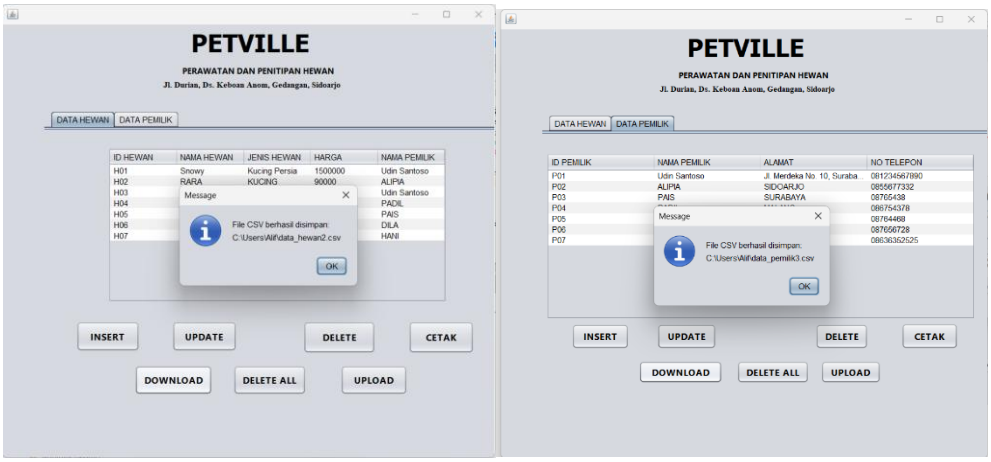


IV. Atribut Tabel dalam JFrame

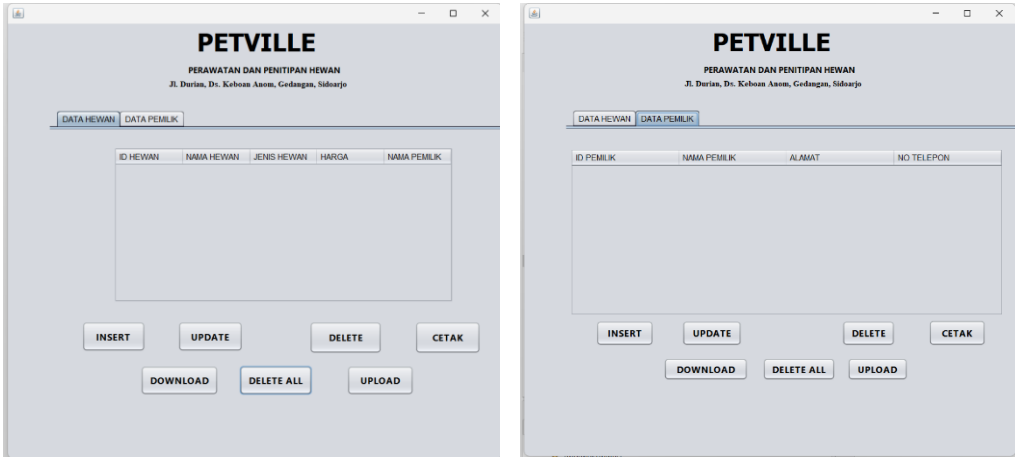
Berisi semua data table yang terkoneksi dalam PostgreSQL.



V. Mendownload Tabel ke File CSV



VI. Menghapus Semua data di tabel



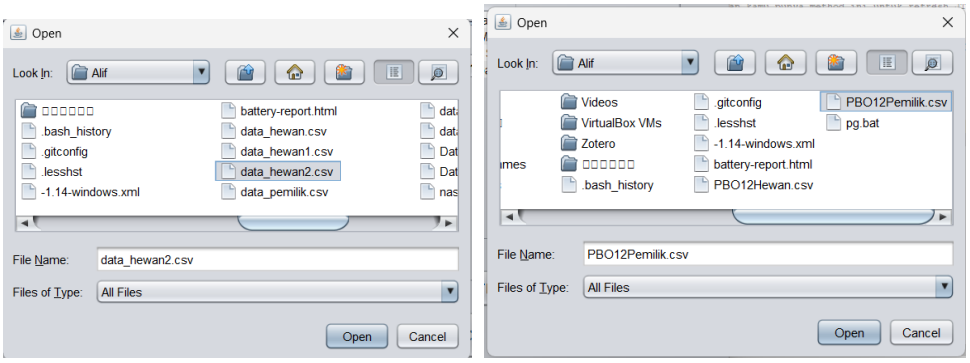
VII. Mengupload File bertipe CSV

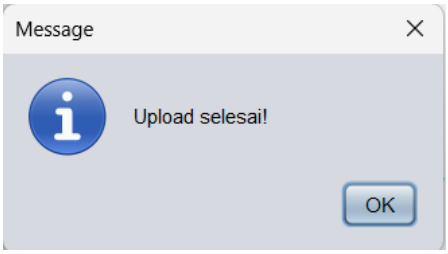
- Membuat Tabel di Excel dan di Save As berbentuk file CSV

```
ID HEWAN;NAMA HEWAN;JENIS HEWAN;HARGA;NAMA PEMILIK
H01;Snowy;Kucing Persia;1500000;Udin Santoso
H02;RARA;KUCING;90000;ALIPIA
H03;Milo;Kucing Anggora;1800000;Udin Santoso
H04;RAFI;BURUNG;40000;PADIL
H05;FIFA;ANJING;60000;PAIS
H06;PITUNG;KELINCI;30000;DILA
H07;HUMMY;HAMSTER;60000;HANI
```

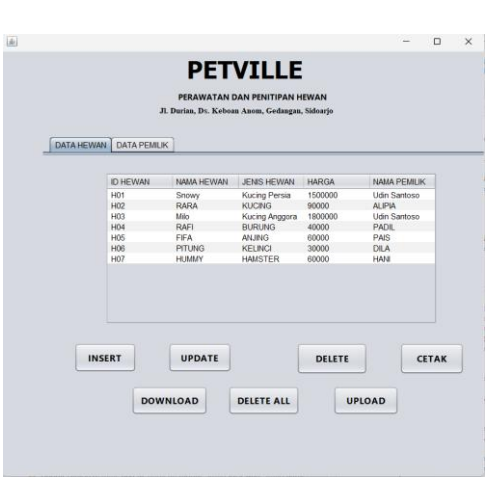
```
ID PEMILIK;NAMA PEMILIK;ALAMAT;NO TELEPON
P01;Udin Santoso;Jl. Merdeka No. 10, Surabaya;081234567890
P02;ALIPIA;SIDOARJO;0855677332
P03;PAIS;SURABAYA;08765438
P04;PADIL;MALANG;086754378
P05;DILA;LAMONGAN;08764468
P06;SOKI;GRESIK;087656728
P07;HANI;SIDOARJO;08636352525
```

- Klik Button Upload dan dan pilih dile CSV sampai ada notifikasi Upload Selesai.



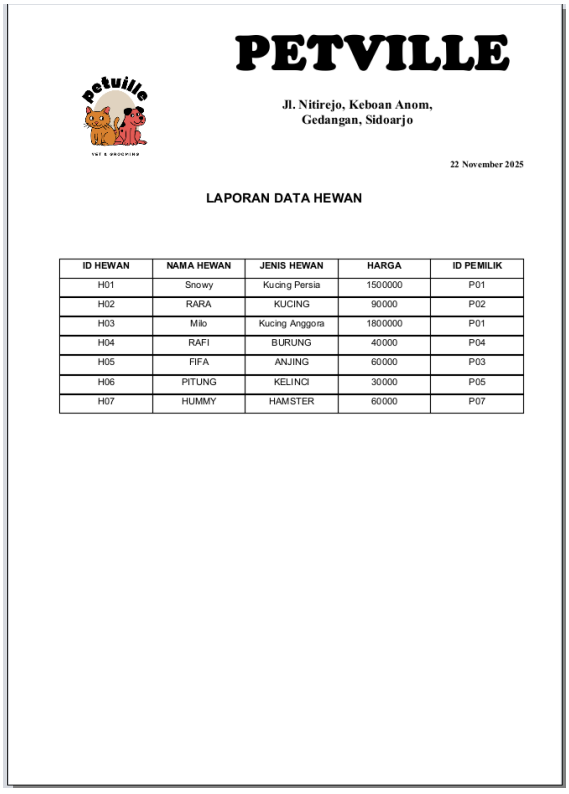


- Data CSV sudah berhasil ditambahkan



VIII. Mencetak Semua Data Tabel di Kertas Jasper

Untuk Mencetaknya tinggal Klik Button Cetak yang ada di JFrame kemudian Jasper akan terpanggil dan mencetak semua data yang ada di dalam table JFrame.



PETVILLE

Jl. Nitirejo, Keboan Anom,
Gedangan, Sidoarjo

22 November 2025

LAPORAN DATA HEWAN

ID HEWAN	NAMA HEWAN	JENIS HEWAN	HARGA	ID PEMILIK
H01	Snowy	Kucing Persia	1500000	P01
H02	RARA	KUCING	90000	P02
H03	Milo	Kucing Anggora	1800000	P01
H04	RAFI	BURUNG	40000	P04
H05	FIFA	ANJING	60000	P03
H06	PITUNG	KELINCI	30000	P05
H07	HUMMY	HAMSTER	60000	P07

PETVILLE

Jl. Nitirejo, Keboan Anom,
Gedangan, Sidoarjo

22 November 2025

LAPORAN DATA HEWAN

ID PEMILIK	NAMA PEMILIK	ALAMAT	NO TELEPON
ID PEMILIK	NAMA PEMILIK	ALAMAT	NO TELEPON
P01	Udin Santoso	Jl. Merdeka No. 10,	081234567890
P02	ALIPIA	SIDOARJO	0855677332
P03	PAIS	SURABAYA	08765438
P04	PADIL	MALANG	086754378
P05	DILA	LAMONGAN	08764468
P06	SOKI	GRESIK	087656728
P07	HANI	SIDOARJO	08636352525

KESIMPULAN PRAKTIKUM

Praktikum ini berhasil menunjukkan bagaimana membangun aplikasi Jual Beli Hewan menggunakan Java NetBeans dengan konsep GUI, JPA, PostgreSQL, dan Jasper Report. Melalui proses pembuatan entity class, pengaturan koneksi database, perancangan JFrame dan JDialog, serta implementasi operasi CRUD (Insert, Update, Delete), aplikasi dapat berjalan secara terstruktur dan efisien. Selain itu, fitur tambahan seperti login, register, reset password, upload dan download CSV, serta pencetakan laporan melalui Jasper berhasil diterapkan untuk melengkapi fungsionalitas sistem. Secara keseluruhan, praktikum ini memberikan pemahaman menyeluruh tentang integrasi GUI Java dengan database dan laporan otomatis, sekaligus melatih kemampuan dalam membuat aplikasi berbasis objek yang lengkap dan terhubung dengan database.