

LAPORAN PRATIKUM 4

APLIKASI MOBILE

Dosen Pengampu: Nurfiah, S.ST, M.Kom



DISUSUN OLEH :

Rehan Khairuno

2211533003

DEPARTEMEN INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI

UNIVERSITAS ANDALAS

2024

A. PelangganEditActivity

```
package com.rehan.laundryapp.pelanggan;

import ...

public class PelangganEditActivity extends AppCompatActivity { new*
    private String id, name, email, hp; 3 usages
    private EditText edPelEditNama, edPelEditEmail, edPelEditHp; 3 usages
    private Button btnPelEditSimpan, btnPelEditHapus, btnPelEditBatal; 2 usages
    private SQLiteHelper db; 3 usages

    @SuppressWarnings("MissingInflatedId") new*
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable(this);
        setContentView(R.layout.activity_pelanggan_edit);

        // Handle system window insets for better layout experience
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {
            Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());
            v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);
            return insets;
        });

        // Initialize SQLiteHelper for database operations
        db = new SQLiteHelper(context, this);

        // Retrieve intent extras
        id = getIntent().getStringExtra("id");
        name = getIntent().getStringExtra("name");
        email = getIntent().getStringExtra("email");
        hp = getIntent().getStringExtra("hp");

        // Initialize views
        edPelEditNama = findViewById(R.id.edPelEditNama);
        edPelEditEmail = findViewById(R.id.edPelEditEmail);
        edPelEditHp = findViewById(R.id.edPelEditHp);
        btnPelEditSimpan = findViewById(R.id.btnPelEditSimpan);
        btnPelEditHapus = findViewById(R.id.btnPelEditDelete);
        btnPelEditBatal = findViewById(R.id.btnPelEditBatal);

        // Set the existing customer data to EditText fields
        edPelEditNama.setText(name);
        edPelEditEmail.setText(email);
        edPelEditHp.setText(hp);

        // Set listeners for buttons
        btnPelEditSimpan.setOnClickListener(v -> updateCustomer());
        btnPelEditHapus.setOnClickListener(v -> deleteCustomer());
        btnPelEditBatal.setOnClickListener(v -> cancelEdit());
    }

    // Update customer in the database
    private void updateCustomer() { 1 usage new*
        String updatedName = edPelEditNama.getText().toString();
        String updatedEmail = edPelEditEmail.getText().toString();
        String updatedHp = edPelEditHp.getText().toString();

        // Check if fields are empty
        if (updatedName.isEmpty() || updatedEmail.isEmpty() || updatedHp.isEmpty()) {
            Toast.makeText(context, "All fields must be filled", Toast.LENGTH_SHORT).show();
            return;
        }

        // Update the customer in the database
        boolean isUpdated = db.updatePelanggan(id, updatedName, updatedEmail, updatedHp);
        if (isUpdated) {
            Toast.makeText(context, "Customer updated successfully", Toast.LENGTH_SHORT).show();
            PelangganActivity pa = new PelangganActivity();
            pa.getData();
            finish(); // Close the activity after successful update
        } else {
            Toast.makeText(context, "Failed to update customer", Toast.LENGTH_SHORT).show();
        }
    }

    // Delete customer from the database
    private void deleteCustomer() { 1 usage new*
        // Confirmation dialog before deleting
        new AlertDialog.Builder(context)
            .setTitle("Delete Confirmation")
            .setMessage("Are you sure you want to delete this customer?")
            .setPositiveButton("Yes", (dialog, which) -> {
                boolean isDeleted = db.deletePelanggan(id);
                if (isDeleted) {
                    Toast.makeText(context, "Customer deleted successfully", Toast.LENGTH_SHORT).show();
                    Intent intent = new Intent(context, PelangganActivity.class);
                    intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP | Intent.FLAG_ACTIVITY_NEW_TASK);
                    startActivity(intent);
                    finish(); // Close the current activity after deletion
                } else {
                    Toast.makeText(context, "Failed to delete customer", Toast.LENGTH_SHORT).show();
                }
            })
            .setNegativeButton("No", null)
            .create()
            .show();
    }
}
```

```

112         .setNegativeButton(text: "No", listener: null)
113         .show();
114     }
115
116     // Cancel edit and go back to the previous screen
117     private void cancelEdit() { finish(); // Close the activity without making any changes }
118
119
120
121     // Override onBackPressed to handle back button press
122     @Override new *
123     public void onBackPressed() {
124         cancelEdit(); // Use the cancelEdit() method to finish the activity and return
125     }
126 }

```

Penjelasan :

Kelas PelangganEditActivity di atas adalah bagian dari aplikasi Android yang memungkinkan pengguna untuk mengedit, menghapus, atau membatalkan pengeditan informasi pelanggan di aplikasi laundry. Kelas ini menggunakan komponen dari Android SDK seperti AlertDialog, Toast, Intent, dan EditText. Berikut adalah penjelasan lengkap dari kode tersebut:

1. Deklarasi Variabel

- **Variabel String:**
 - id: Menyimpan ID pelanggan.
 - name: Menyimpan nama pelanggan.
 - email: Menyimpan alamat email pelanggan.
 - hp: Menyimpan nomor telepon pelanggan.
- **EditText (Input Field):**
 - edPelEditNama: EditText untuk mengisi atau menampilkan nama pelanggan.
 - edPelEditEmail: EditText untuk mengisi atau menampilkan email pelanggan.
 - edPelEditHp: EditText untuk mengisi atau menampilkan nomor telepon pelanggan.
- **Button:**
 - btnPelEditSimpan: Tombol untuk menyimpan perubahan data pelanggan.
 - btnPelEditHapus: Tombol untuk menghapus pelanggan.
 - btnPelEditBatal: Tombol untuk membatalkan pengeditan dan kembali ke aktivitas sebelumnya.
- **SQLiteHelper:**
 - db: Objek dari SQLiteHelper yang digunakan untuk melakukan operasi pada database lokal SQLite.

2. Metode onCreate

- **Mengaktifkan Edge-to-Edge Display:** Metode ini menggunakan `EdgeToEdge.enable(this);` untuk mendukung tampilan penuh hingga tepi layar pada perangkat modern.
- **Membaca Data dari Intent:** Data seperti ID, nama, email, dan nomor telepon pelanggan diambil dari Intent yang dikirim dari aktivitas sebelumnya menggunakan `getIntent().getStringExtra()`.
- **Inisialisasi Views:** Menghubungkan variabel-variabel dengan elemen layout XML menggunakan `findViewById()`.
- **Setel Data ke EditText:** Menggunakan `setText()` untuk menampilkan data pelanggan yang ada ke dalam field EditText yang sesuai.
- **Listener untuk Tombol:**
 - btnPelEditSimpan: Menyimpan perubahan yang dilakukan oleh pengguna ke database dengan memanggil metode `updateCustomer()`.

- `btnPelEditHapus`: Menampilkan dialog konfirmasi sebelum menghapus data pelanggan dengan memanggil metode `deleteCustomer()`.
- `btnPelEditBatal`: Membatalkan pengeditan dan kembali ke layar sebelumnya dengan memanggil metode `cancelEdit()`.

3. Metode `updateCustomer`

- **Validasi Input:** Mengecek apakah semua input diisi sebelum melakukan update. Jika ada yang kosong, aplikasi akan menampilkan pesan menggunakan Toast.
- **Proses Update:** Memanggil `db.updatePelanggan()` untuk mengupdate data di database. Jika update berhasil, menampilkan pesan "Customer updated successfully" dan menutup aktivitas dengan `finish()`. Jika gagal, menampilkan pesan kesalahan.
- **Panggilan Metode `getData` pada `PelangganActivity`:** Setelah update berhasil, metode `getData` dari `PelangganActivity` dipanggil untuk memperbarui daftar pelanggan di layar sebelumnya.

4. Metode `deleteCustomer`

- **Dialog Konfirmasi:** Menampilkan dialog konfirmasi menggunakan `AlertDialog` sebelum menghapus pelanggan. Jika pengguna memilih "Yes", metode `db.deletePelanggan()` dipanggil untuk menghapus pelanggan dari database.
- **Proses Penghapusan:** Jika penghapusan berhasil, pelanggan akan dialihkan ke `PelangganActivity` menggunakan Intent dengan menambahkan flag `FLAG_ACTIVITY_CLEAR_TOP | FLAG_ACTIVITY_NEW_TASK` untuk memastikan aktivitas sebelumnya tidak ada lagi di stack.

5. Metode `cancelEdit`

- Menutup aktivitas dengan `finish()` tanpa melakukan perubahan, kembali ke layar sebelumnya.

6. Metode `onBackPressed`

- Override metode ini untuk menangani tombol back. Ketika tombol back ditekan, aplikasi akan memanggil metode `cancelEdit()`.

Struktur File XML Layout (Assumed)

- **`activity_pelanggan_edit.xml`:** File layout yang berisi elemen-elemen UI seperti `EditText` untuk nama, email, nomor telepon, serta tiga tombol (Simpan, Hapus, Batal).

Integrasi `SQLiteHelper`

- **`SQLiteHelper`** adalah kelas terpisah yang digunakan untuk operasi database seperti update dan delete data pelanggan. Metode seperti `updatePelanggan()` dan `deletePelanggan()` berinteraksi dengan database `SQLite` untuk menyimpan perubahan yang dilakukan oleh pengguna.

Dengan menggunakan kode di atas, aplikasi memungkinkan pengguna untuk mengedit dan menghapus data pelanggan dengan interaksi yang user-friendly, seperti validasi input dan dialog konfirmasi sebelum menghapus data.

B. activity_pelanggan_edit.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="16dp"
    android:background="@color/white"
    tools:context=".pelanggan.PelangganEditActivity">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:layout_margin="16dp"
        tools:ignore="MissingConstraints">

        <TextView
            android:textColor="@color/black"
            android:textStyle="bold"
            android:paddingBottom="16dp"
            android:textSize="24sp"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Edit Pelanggan" />

    </LinearLayout>

</androidx.constraintlayout.widget.ConstraintLayout>
```

```
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical">

            <TextView
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="Nama"
                android:textColor="@android:color/black" />

            <EditText
                android:drawablePadding="8dp"
                android:drawableLeft="@drawable/ic_username"
                android:id="@+id/edPelEditNama"
                android:padding="8dp"
                android:layout_marginTop="8dp"
                android:layout_width="match_parent"
                android:layout_height="48dp"
                android:background="@drawable/green_border" />

        </LinearLayout>

        <LinearLayout
            android:layout_marginTop="16dp"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical">

            <TextView
                android:layout_width="match_parent"
                android:layout_height="wrap_content"

</LinearLayout>
```

```
            <TextView
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="Email"
                android:textColor="@android:color/black" />

            <EditText
                android:inputType="textEmailAddress"
                android:id="@+id/edPelEditEmail"
                android:drawablePadding="8dp"
                android:drawableLeft="@drawable/ic_email"
                android:layout_marginTop="8dp"
                android:padding="16dp"
                android:layout_width="match_parent"
                android:layout_height="48dp"
                android:background="@drawable/green_border" />

        </LinearLayout>

        <LinearLayout
            android:layout_marginTop="16dp"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical">

            <TextView
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="HP"
                android:textColor="@android:color/black" />

        </LinearLayout>

    </LinearLayout>

</androidx.constraintlayout.widget.ConstraintLayout>
```

```

<Button
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginRight="4dp"
    android:layout_weight="1"
    android:background="#FFC107"
    android:text="Update"
    android:textColor="@android:color/white" />

<Button
    android:id="@+id/btnPelEditDelete"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginLeft="4dp"
    android:layout_marginRight="4dp"
    android:layout_weight="1"
    android:background="#F44336"
    android:text="Delete"
    android:textColor="@android:color/white" />

<Button
    android:id="@+id/btnPelEditBatal"
    android:layout_marginLeft="4dp"
    android:background="#FFEB3B"
    android:textColor="@android:color/white"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    android:text="Batal" />
</LinearLayout>

```

Penjelasan :

Berikut adalah penjelasan lengkap dari file XML layout activity_pelanggan_edit.xml, yang merupakan tampilan dari aktivitas PelangganEditActivity di aplikasi laundry Android. Layout ini menggunakan ConstraintLayout sebagai root view dengan elemen-elemen yang diatur menggunakan LinearLayout untuk membentuk UI yang terorganisir.

Struktur Layout

1. Root Layout: ConstraintLayout

- androidx.constraintlayout.widget.ConstraintLayout: Root dari seluruh layout ini.
- **Attributes:**
 - xmlns:android, xmlns:app, dan xmlns:tools: Deklarasi namespace XML.
 - android:id="@+id/main": ID untuk root layout ini, yang digunakan di kode Java untuk pengaturan padding dengan window insets.
 - android:layout_width="match_parent" dan android:layout_height="match_parent": Menjadikan root layout ini memenuhi seluruh lebar dan tinggi layar.
 - android:padding="16dp": Memberikan padding di semua sisi root layout.
 - android:background="@color/white": Mengatur warna latar belakang menjadi putih.
 - tools:context=".pelanggan.PelangganEditActivity": Menentukan konteks aktivitas untuk membantu dalam preview layout di Android Studio.

2. LinearLayout untuk Konten Utama

- **Attributes:**
 - android:layout_width="match_parent": Lebar layout akan mengikuti lebar layar.
 - android:layout_height="wrap_content": Tinggi layout menyesuaikan kontennya.
 - android:orientation="vertical": Elemen-elemen di dalam layout ini diatur secara vertikal (dari atas ke bawah).
 - android:layout_margin="16dp": Memberikan margin di sekitar layout ini.

Elemen di dalam LinearLayout ini:

2. **Judul: TextView**

- `android:text="Edit Pelanggan"`: Menampilkan teks "Edit Pelanggan" sebagai judul.
- `android:textSize="24sp"`: Mengatur ukuran teks menjadi 24sp, lebih besar dari teks normal.
- `android:textStyle="bold"`: Teks ditampilkan dengan gaya tebal.
- `android:paddingBottom="16dp"`: Memberikan jarak vertikal antara judul dan elemen berikutnya.

3. **Form Input untuk Nama Pelanggan**

- **LinearLayout:**
 - `android:orientation="vertical"`: Mengatur teks label dan input field di dalam layout ini secara vertikal.
- **TextView untuk label:**
 - `android:text="Nama"`: Menampilkan label "Nama".
- **EditText untuk input:**
 - `android:id="@+id/edPelEditNama"`: ID yang digunakan untuk merujuk ke field ini dari Java.
 - `android:drawableLeft="@drawable/ic_ussername"`: Menambahkan ikon di kiri input field.
 - `android:background="@drawable/green_border"`: Mengatur background dengan border hijau.

4. **Form Input untuk Email**

- Mirip dengan bagian nama, tetapi:
 - `android:inputType="textEmailAddress"`: Mengatur tipe input sebagai alamat email untuk keyboard yang relevan.
 - `android:id="@+id/edPelEditEmail"`: ID yang merujuk ke field ini di Java.
 - `android:drawableLeft="@drawable/ic_email"`: Menambahkan ikon email di sebelah kiri input field.

5. **Form Input untuk Nomor Telepon (HP)**

- Sama dengan dua bagian sebelumnya, namun:
 - `android:inputType="number"`: Mengatur tipe input untuk hanya menerima angka (nomor telepon).
 - `android:id="@+id/edPelEditHp"`: ID untuk merujuk ke field input nomor telepon di Java.
 - `android:drawableLeft="@drawable/ic_hp"`: Menambahkan ikon telepon di sebelah kiri input field.

3. **Tombol Aksi (Button)**

Semua tombol ini ditempatkan di dalam `LinearLayout` horizontal, dengan tiga tombol yang memiliki berat (weight) yang sama untuk membagi lebar layar secara proporsional.

- **Attributes untuk `LinearLayout` ini:**

- `android:orientation="horizontal"`: Elemen-elemen diatur secara horizontal (berdampingan).
- `android:weightSum="3"`: Mengatur total berat layout ini sebagai 3, yang berarti setiap tombol akan mendapatkan lebar yang proporsional terhadap berat yang ditentukan.

Tombol-tombol di dalamnya:

1. Button untuk Simpan/Update:

- `android:id="@+id/btnPelEditSimpan"`: ID yang merujuk ke tombol ini di Java.
- `android:background="#FFC107"`: Warna latar belakang kuning (#FFC107).
- `android:text="Update"`: Menampilkan teks "Update".
- `android:textColor="@android:color/white"`: Mengatur warna teks menjadi putih.

2. Button untuk Hapus:

- `android:id="@+id/btnPelEditDelete"`: ID yang merujuk ke tombol ini di Java.
- `android:background="#F44336"`: Warna latar belakang merah (#F44336).
- `android:text="Delete"`: Menampilkan teks "Delete".
- `android:textColor="@android:color/white"`: Mengatur warna teks menjadi putih.

3. Button untuk Membatalkan:

- `android:id="@+id/btnPelEditBatal"`: ID yang merujuk ke tombol ini di Java.
- `android:background="#FFEB3B"`: Warna latar belakang kuning cerah (#FFEB3B).
- `android:text="Batal"`: Menampilkan teks "Batal".
- `android:textColor="@android:color/white"`: Warna teks putih.

Summary

Layout ini berfungsi sebagai form untuk mengedit pelanggan, dengan tiga input field (nama, email, nomor HP) dan tiga tombol aksi (Update, Delete, Batal). Tombol "Update" digunakan untuk menyimpan perubahan, tombol "Delete" untuk menghapus pelanggan, dan tombol "Batal" untuk membatalkan proses pengeditan. Layout ini dirancang menggunakan pendekatan yang responsif dengan ConstraintLayout sebagai root, dan LinearLayout di dalamnya untuk mengatur elemen-elemen secara vertikal.

C. SQLite

```
// Fungsi untuk memperbarui pelanggan
public boolean updatePelanggan(String id, String name, String email, String hp) {
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues contentValues = new ContentValues();
    contentValues.put(KEY_PELANGGAN_NAMA, name);
    contentValues.put(KEY_PELANGGAN_EMAIL, email);
    contentValues.put(KEY_PELANGGAN_HP, hp);

    // Update the row where the id matches
    int result = db.update(TABLE_PELANGGAN, contentValues, "KEY_PELANGGAN_ID = ?", new String[]{id});
    db.close();
    return result > 0; // Mengembalikan true jika update berhasil
}

// Fungsi untuk menghapus pelanggan
public boolean deletePelanggan(String id) {
    SQLiteDatabase db = this.getWritableDatabase();
    int result = db.delete(TABLE_PELANGGAN, "KEY_PELANGGAN_ID = ?", new String[]{id});
    db.close();
    return result > 0; // Mengembalikan true jika penghapusan berhasil
}
}
```

Penjelasan :


1. **SQLiteDatabase db = this.getWritableDatabase();**: Mendapatkan instance database dalam mode tulis karena kita ingin memperbarui data.
2. **ContentValues contentValues = new ContentValues();**: Membuat objek ContentValues untuk menyimpan data pelanggan yang baru. ContentValues berfungsi untuk menyimpan pasangan key-value, di mana key adalah nama kolom dalam tabel, dan value adalah nilai yang baru.
3. **contentValues.put(KEY_PELANGGAN_NAMA, name);**: Memasukkan nilai baru untuk kolom name.
 - o Begitu juga dengan email dan hp, masing-masing akan disimpan di kolom yang sesuai.
4. **db.update(...)**: Memperbarui data pelanggan di database berdasarkan id pelanggan. Pada klausa WHERE, KEY_PELANGGAN_ID + " = ?" berarti kita mencari baris di mana kolom id sesuai dengan id yang diberikan sebagai argumen.
5. **db.close();**: Menutup database setelah operasi selesai untuk membebaskan sumber daya.
6. **return result > 0;**: Mengembalikan true jika jumlah baris yang diupdate lebih besar dari 0, artinya update berhasil. Jika tidak, mengembalikan false (update gagal).
7. ☐ **SQLiteDatabase db = this.getWritableDatabase();**: Mendapatkan instance database dalam mode tulis karena kita ingin menghapus data.
8. ☐ **db.delete(...)**: Menghapus baris dari tabel TABLE_PELANGGAN berdasarkan id pelanggan. Sama seperti pada fungsi updatePelanggan, klausa WHERE di sini memastikan hanya baris yang sesuai dengan id yang diberikan yang akan dihapus.
9. ☐ **db.close();**: Menutup koneksi database setelah operasi selesai.
10. ☐ **return result > 0;**: Mengembalikan true jika jumlah baris yang dihapus lebih besar dari 0, artinya penghapusan berhasil. Jika tidak, mengembalikan false.

Output :


4:20 ⓘ 🔒 🔋

Edit Pelanggan


Nama

 Asep

Email

 A@gmail.com

HP

 0843414134

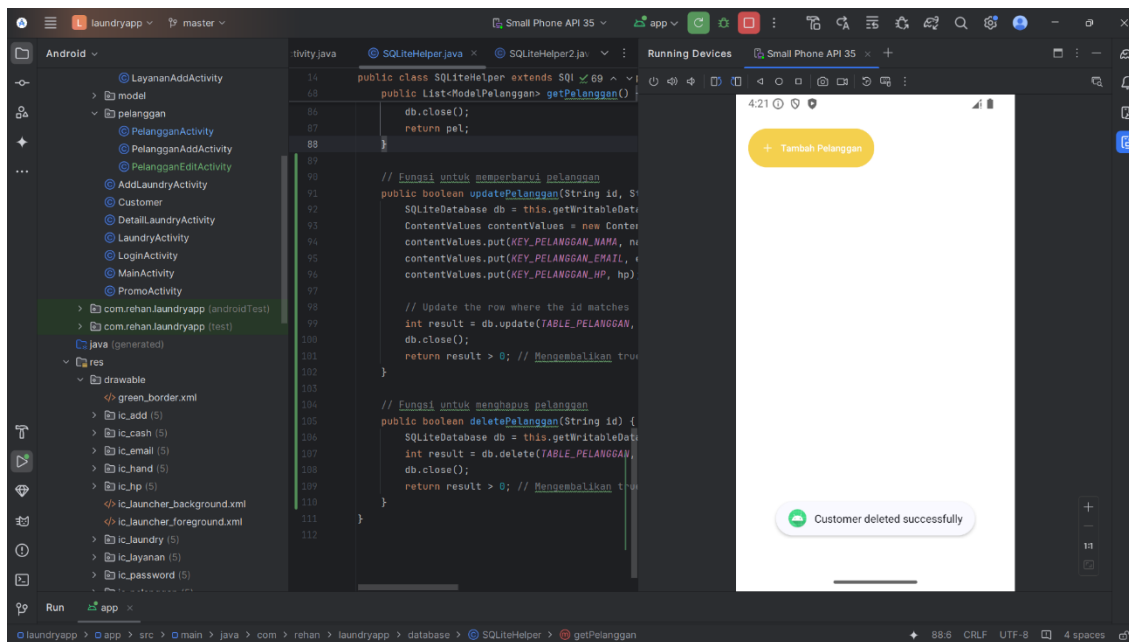
Update Delete Batal

4:20 ⓘ 🔒 🔋

[+ Tambah Pelanggan](#)

Rehan
08314341414

Asep
0843414134



Daftar Pustaka

<https://hackernoon.com/top-10-android-studio-tips-and-tricks-for-faster-development>

<https://chatgpt.com/>