

TUGAS PERTEMUAN 3
PRAKTIKUM APP MOBILE

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



Disusun Oleh :

Yusuf Fatha Mubina Alexander (2211533017)

Departemen Informatika
Fakultas Teknologi Informasi
Universitas Andalas
2024

A. PELANGGAN

1. Class SQLiteHelper.java

```
package com.example.laundryapps.database;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

import java.util.ArrayList;
import java.util.List;

import com.example.laundryapps.model.ModelPelanggan;
import com.example.laundryapps.pelanggan.PelangganAddActivity;

public class SQLiteHelper extends SQLiteOpenHelper { 6 usages

    public static String DATABASE_NAME = "my_laundry.db"; 1 usage
    public static final int DATABASE_VERSION = 1; 1 usage
    public static final String TABLE_PELANGGAN = "pelanggan"; 4 usages
    public static final String KEY_PELANGGAN_ID = "pelanggan_id"; 2 usages
    public static final String KEY_PELANGGAN_NAMA = "nama"; 2 usages
    public static final String KEY_PELANGGAN_EMAIL = "email"; 2 usages
    public static final String KEY_PELANGGAN_HP = "hp"; 2 usages
    private static final String CREATE_TABLE_PELANGGAN = "CREATE TABLE " + 1 usage
        TABLE_PELANGGAN + "("
        + KEY_PELANGGAN_ID + " TEXT, " + KEY_PELANGGAN_NAMA + " TEXT, " +
        KEY_PELANGGAN_EMAIL + " TEXT, " + KEY_PELANGGAN_HP + " TEXT )";

    public SQLiteHelper(Context context) { 2 usages
        super(context, DATABASE_NAME, factory: null, DATABASE_VERSION);
    }
}
```

```

@Override
public void onCreate(SQLiteDatabase db) {
    db.execSQL(CREATE_TABLE_PELANGGAN);
}

@Override no usages
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
    db.execSQL("DROP TABLE IF EXISTS " + TABLE_PELANGGAN);
    onCreate(db);
}

public boolean insertPelanggan(ModelPelanggan mp) { 1 usage
    SQLiteDatabase database = this.getReadableDatabase();
    ContentValues contentValues = new ContentValues();
    contentValues.put(KEY_PELANGGAN_ID, mp.getId());
    contentValues.put(KEY_PELANGGAN_NAMA, mp.getNama());
    contentValues.put(KEY_PELANGGAN_EMAIL, mp.getEmail());
    contentValues.put(KEY_PELANGGAN_HP, mp.getHp());
    long id = database.insert(TABLE_PELANGGAN, nullColumnHack: null, contentValues);
    database.close();
    if (id != -1) {
        return true;
    } else {
        return false;
    }
}

public List<ModelPelanggan> getPelanggan() { 1 usage
    List<ModelPelanggan> pel = new ArrayList<>();
    String query = "SELECT * FROM " + TABLE_PELANGGAN;
    String query = "SELECT * FROM " + TABLE_PELANGGAN;
    SQLiteDatabase db = this.getWritableDatabase();
    Cursor cursor = db.rawQuery(query, selectionArgs: null);
    if (cursor.moveToFirst()) {
        do {
            ModelPelanggan k = new ModelPelanggan();
            k.setId(cursor.getString(i: 0));
            k.setNama(cursor.getString(i: 1));
            k.setEmail(cursor.getString(i: 2));
            k.setHp(cursor.getString(i: 3));
            pel.add(k);
        } while (cursor.moveToNext());
    }
    return pel;
}
}

```

Penjelasannya :

Nama Database dan Versi:

- DATABASE_NAME: Nama database yang digunakan adalah "LaundryDB".
- DATABASE_VERSION: Versi database adalah 11. Jika skema database berubah, versi ini akan ditingkatkan.

Nama Tabel dan Kolom:

- TABLE_PELANGGAN: Nama tabel yang digunakan adalah "pelanggan".
- Kolom di dalam tabel:

- o pelanggan_id: Kolom ID pelanggan, tipe data INTEGER, AUTO_INCREMENT, dan PRIMARY KEY.
- o nama: Kolom untuk menyimpan nama pelanggan, tipe data TEXT.
- o email: Kolom untuk menyimpan email pelanggan, tipe data TEXT.
- o hp: Kolom untuk menyimpan nomor HP pelanggan, tipe data TEXT.

Query Membuat Tabel Pelanggan:

- Query CREATE_TABLE_PELANGGAN digunakan untuk membuat tabel pelanggan dengan 4 kolom: pelanggan_id,

nama, email, dan hp.

Konstruktor SQLiteHelper:

- SQLiteHelper(Context context): Konstruktor ini memanggil superclass SQLiteOpenHelper untuk menginisialisasi

database dengan nama dan versi yang telah didefinisikan.

Metode onCreate():

- Digunakan untuk membuat tabel pelanggan ketika database pertama kali dibuat dengan mengeksekusi query

CREATE_TABLE_PELANGGAN.

Metode onUpgrade():

- Jika versi database diperbarui, tabel lama akan dihapus dengan query DROP TABLE dan tabel baru dibuat ulang

dengan onCreate.

Fungsi insertPelanggan():

- Menggunakan ContentValues untuk menyimpan data pelanggan (nama, email, hp) ke dalam database.
- Fungsi ini mengembalikan true jika data berhasil disimpan (id tidak -1).

Fungsi getPelanggan():

- Mengambil semua data pelanggan dari tabel menggunakan query SELECT * FROM pelanggan.
- Data pelanggan disimpan dalam list ModelPelanggan dan dikembalikan dalam bentuk list berisi objek-objek pelanggan.

2. Class AdapterPelanggan.java

```
package com.example.laundryapps.adapter;

import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;

import com.example.laundryapps.R;
import com.example.laundryapps.model.ModelPelanggan;

import java.util.List;

public class AdapterPelanggan extends RecyclerView.Adapter<AdapterPelanggan.ViewHolder> {
    private static final String TAG = AdapterPelanggan.class.getSimpleName();
    private Context context;
    private List<ModelPelanggan> list;
    private View.OnClickListener onItemClick;
    public void setOnItemClickListener(View.OnClickListener itemClickListener) {
        onItemClick = itemClickListener;
    }
    public AdapterPelanggan(Context context, List<ModelPelanggan> list) {
        this.context = context;
        this.list = list;
    }
    @Override
    public ViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
```

```
        View view =
            LayoutInflater.from(parent.getContext()).inflate(R.layout.item_pelanggan,
                parent, attachToRoot: false);
        return new ViewHolder(view);
    }
    @Override
    public void onBindViewHolder(ViewHolder holder, int position) {
        ModelPelanggan item = list.get(position);
        holder.tvNama.setText(item.getNama());
        holder.tvHp.setText(item.getHp());
    }
    @Override
    public int getItemCount() {
        return list.size();
    }
    public void clear() {
        int size = this.list.size();
        this.list.clear();
        notifyItemRangeRemoved(0, size);
    }
    public class ViewHolder extends RecyclerView.ViewHolder {
        TextView tvNama, tvHp;
        public ViewHolder(View itemView) {
            super(itemView);
            tvNama = (TextView) itemView.findViewById(R.id.tvItemPelNama);
            tvHp = (TextView) itemView.findViewById(R.id.tvItemPelHp);
            itemView.setTag(this);
            itemView.setOnClickListener(onItemClick);
        }
    }
}
```

Penjelasan :

- Variabel Utama:
- TAG: Tag untuk logging dan debugging.

- context: Menyimpan konteks dari aktivitas atau fragment yang menggunakan adapter ini.
- list: List yang berisi data pelanggan dari objek ModelPelanggan.
- onItemClick• Constructor (AdapterPelanggan):
- AdapterPelanggan(Context context, List<ModelPelanggan> list): Konstruktor untuk inialisasi context dan list data pelanggan.
- Metode setOnItemClickListener():
- Digunakan untuk menetapkan listener yang menangani ketika item di dalam RecyclerView diklik.
- Metode onCreateViewHolder():
- onCreateViewHolder(@NonNull ViewGroup parent, int viewType):
 - o Menginflate layout item pelanggan (item_pelanggan.xml) untuk setiap item di RecyclerView.
 - o Mengembalikan objek ViewHolder yang berisi referensi ke elemen tampilan.
- Metode onBindViewHolder():
- onBindViewHolder(@NonNull ViewHolder holder, int position):
 - o Mengambil data pelanggan dari list sesuai dengan posisi (position) dan mengikat data ke tampilan dalam ViewHolder.
 - o holder.tvNama.setText(item.getNama()): Menampilkan nama pelanggan.
 - o holder.tvHp.setText(item.getHp()): Menampilkan nomor HP pelanggan.
- Metode getItemCount():
- Mengembalikan jumlah item dalam list, digunakan untuk mengetahui berapa banyak data yang akan ditampilkan di RecyclerView.
- Metode clear():
- Membersihkan list data pelanggan dan memberi tahu RecyclerView bahwa item-item tersebut telah dihapus.
- notifyItemRangeRemoved() digunakan untuk memperbarui tampilan setelah penghapusan data.
- ViewHolder Class:
- ViewHolder(View itemView): Constructor untuk inialisasi referensi ke TextView di dalam item pelanggan.
- tvNama: TextView untuk menampilkan nama pelanggan, sesuai dengan ID dari item_pelanggan.xml.

- tvHp: TextView untuk menampilkan nomor HP pelanggan, sesuai dengan ID dari item_pelanggan.xml.
- itemView.setOnClickListener(onItemClicked): Menetapkan listener klik untuk item di RecyclerView.

3. Class ModelPelanggan.java

```
package com.example.laundryapps.model;

public class ModelPelanggan { 20 usages
    String id, nama, email, hp; 2 usages

    public String getId() {
        return id;
    }
    public void setId(String id) {
        this.id = id;
    }
    public String getNama() { 5 usages
        return nama;
    }
    public void setNama(String nama) { 3 usages
        this.nama = nama;
    }
    public String getEmail() { 3 usages
        return email;
    }
    public void setEmail(String email) { 3 usages
        this.email = email;
    }

    public String getHp() { 4 usages
        return hp;
    }
    public void setHp(String hp) { 3 usages
        this.hp = hp;
    }
}
```

Penjelasan :

Untuk mendapatkan suatu data yang kita input, diperlukan metode getter dan setter.

- Atribut Kelas:
- String id: Menyimpan ID pelanggan.
- String nama: Menyimpan nama pelanggan.
- String email: Menyimpan email pelanggan.
- String hp: Menyimpan nomor HP pelanggan.
- Metode getId():
- public String getId(): Mengembalikan nilai ID pelanggan.
- Metode setId():
- public void setId(String id): Mengatur nilai ID pelanggan.
- Metode getNama():
- public String getNama(): Mengembalikan nilai nama pelanggan.
- Metode setNama():
- public void setNama(String nama): Mengatur nilai nama pelanggan.
- Metode getEmail():
- public String getEmail(): Mengembalikan nilai email pelanggan.
- Metode setEmail():
- public void setEmail(String email): Mengatur nilai email pelanggan.
- Metode getHp():
- public String getHp(): Mengembalikan nilai nomor HP pelanggan.
- Metode setHp():
- public void setHp(String hp): Mengatur nilai nomor HP pelanggan.

4. Class PelangganActivity.java

```
package com.example.laundryapps.pelanggan;

import android.annotation.SuppressLint;
import android.app.ProgressDialog;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.animation.AlphaAnimation;
import android.widget.Button;
import android.widget.Toast;

import androidx.activity.EdgeToEdge;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.graphics.Insets;
import androidx.core.view.ViewCompat;
import androidx.core.view.WindowInsetsCompat;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import com.example.laundryapps.R;
import com.example.laundryapps.adapter.AdapterPelanggan;
import com.example.laundryapps.database.SQLiteHelper;
import com.example.laundryapps.model.ModelPelanggan;

import java.util.ArrayList;
import java.util.List;

public class PelangganActivity extends AppCompatActivity {
    SQLiteHelper db; 2 usages
    Button btnPelAdd; 2 usages
    RecyclerView rvPelanggan; 4 usages
    AdapterPelanggan adapterPelanggan; 4 usages
}
```

```

ArrayList<ModelPelanggan> list; 5 usages
ProgressDialog progressDialog; 6 usages
AlphaAnimation btnAnimasi = new AlphaAnimation(1F,0.5F); 1 usage

@SuppressLint("MissingInflatedId")
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    EdgeToEdge.enable( $this$enableEdgeToEdge; this);
    setContentView(R.layout.activity_pelanggan);
    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;
    });
    initView();
    eventHandling();
    getData();
}

private View.OnClickListener onClickListener = new 1 usage
    View.OnClickListener() {
        @Override
        public void onClick(View v) {
            v.startAnimation(btnAnimasi);
            RecyclerView.ViewHolder viewHolder = (RecyclerView.ViewHolder)
                v.getTag();
            int position = viewHolder.getAdapterPosition();
            ModelPelanggan mp = list.get(position);
            Toast.makeText( context: PelangganActivity.this, text: "" +mp.getNama(), Toast.LENGTH_SHORT).show();

```

```

private void getData() { 1 usage
    list.clear();
    showMsg();
    progressDialog.dismiss();
    try {
        List<ModelPelanggan> p = db.getPelanggan();
        if (p.size() > 0){
            for (ModelPelanggan pel : p){
                ModelPelanggan mp = new ModelPelanggan();
                mp.setId(pel.getId());
                mp.setNama(pel.getNama());
                mp.setEmail(pel.getEmail());
                mp.setHp(pel.getHp());
                list.add(mp);
            }
            adapterPelanggan = new AdapterPelanggan( context: this, list);
            adapterPelanggan.notifyDataSetChanged();
            rvPelanggan.setAdapter(adapterPelanggan);
            adapterPelanggan.setOnItemClickListener(onClickListener);
        }else{
            Toast.makeText( context: this, text: "Data tidak ditemukan",
                Toast.LENGTH_SHORT).show();
        }
    }catch (Exception e){
    }
}

private void eventHandling() { 1 usage
    btnPelAdd.setOnClickListener(new View.OnClickListener() {

```

```

private void eventHandling() { 1 usage
    btnPelAdd.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            startActivity(new Intent( packageName: PelangganActivity.this, PelangganAddActivity.class));
        }
    });
}

private void setView() { 1 usage
    db = new SQLiteHelper( context: this);
    progressDialog = new ProgressDialog( context: this);
    btnPelAdd = (Button) findViewById(R.id.btnPelAdd);
    rvPelanggan = (RecyclerView) findViewById(R.id.rvPelanggan);
    list = new ArrayList<>();
    LinearLayoutManager llm = new LinearLayoutManager( context: this);
    llm.setOrientation(LinearLayoutManager.VERTICAL);
    rvPelanggan.setHasFixedSize(true);
    rvPelanggan.setLayoutManager(llm);
}

private void showMsg() { 1 usage
    progressDialog.setTitle("Informasi");
    progressDialog.setMessage("Loading Data..");
    progressDialog.setCancelable(false);
    progressDialog.show();
}
}

```

Penjelasan :

- Deklarasi Variabel:
- SQLiteHelper db: Digunakan untuk operasi database SQLite.
- Button btnPelAdd: Tombol untuk menambahkan pelanggan baru.
- RecyclerView rvPelanggan: RecyclerView untuk menampilkan daftar pelanggan.
- AdapterPelanggan adapterPelanggan: Adapter untuk RecyclerView, yang mengatur tampilan dari data.
- ArrayList<ModelPelanggan> list: List yang digunakan untuk menyimpan data pelanggan dari database.
- ProgressDialog progressDialog: Dialog yang menampilkan pesan "loading" saat data sedang diambil.
- AlphaAnimation btnAnimasi: Animasi yang digunakan untuk tombol, mengubah alpha (transparansi) dari 1 menjadi 0.5 saat ditekan.
- Metode onCreate():
- EdgeToEdge.enable(this): Mengaktifkan dukungan untuk tampilan tepi ke tepi.
- setContentView(R.layout.activity_pelanggan): Mengatur layout yang digunakan untuk aktivitas ini.
- ViewCompat.setOnApplyWindowInsetsListener: Menyusun padding dari tampilan utama berdasarkan sistem insets (misal status bar, navigation bar).
- setView(): Inisialisasi tampilan (RecyclerView, tombol, dan lain-lain).
- eventHandling(): Menetapkan listener untuk event klik.

- `getData()`: Mengambil data pelanggan dari database dan menampilkannya di `RecyclerView`.
- Metode `setView()`:
- `db = new SQLiteHelper(this)`: Inisialisasi objek `SQLiteHelper` untuk operasi database.
- `progressDialog = new ProgressDialog(this)`: Inisialisasi `ProgressDialog` untuk menampilkan loading saat data diambil.
- `btnPelAdd = findViewById(R.id.btnPLAdd)`: Menghubungkan tombol tambah pelanggan dengan XML layout.
- `rvPelanggan = findViewById(R.id.rvPelanggan)`: Menghubungkan `RecyclerView` dengan XML layout.
- `list = new ArrayList<>()`: Inisialisasi list untuk menyimpan data pelanggan.
- `LinearLayoutManager llm = new LinearLayoutManager(this)`: Mengatur layout manager untuk `RecyclerView`, menggunakan layout vertikal.
- `rvPelanggan.setHasFixedSize(true)`: Mengoptimalkan ukuran `RecyclerView`.
- `rvPelanggan.setLayoutManager(llm)`: Menetapkan layout manager ke `RecyclerView`.
- Metode `eventHandling()`:
- `btnPelAdd.setOnClickListener`: Menetapkan listener klik untuk tombol tambah pelanggan.
- Saat tombol ditekan, `startActivity()` memulai aktivitas `PelangganAddActivity` untuk menambah pelanggan baru.
- Metode `getData()`:
- `list.clear()`: Membersihkan list sebelum menambahkan data baru.
- `showMsg()`: Menampilkan dialog loading sebelum mengambil data.
- `progressDialog.dismiss()`: Menutup dialog loading setelah data berhasil diambil.
- `db.getPelanggan()`: Mengambil data pelanggan dari database.
- `list.add(mp)`: Menambahkan data pelanggan ke list.
- `adapterPelanggan.notifyDataSetChanged()`: Beritahu adapter bahwa data telah berubah.
- `rvPelanggan.setAdapter(adapterPelanggan)`: Menetapkan adapter ke `RecyclerView`.
- `adapterPelanggan.setOnItemClickListener(onClickListener)`: Menetapkan event klik pada item `RecyclerView`.
- Metode `showMsg()`:

- Membuat dan menampilkan ProgressDialog dengan pesan "Loading Data...".
- progressDialog.setCancelable(false): Dialog tidak bisa dibatalkan dengan klik di luar.
- Metode onClickListener:
- Ketika item di RecyclerView diklik, menampilkan nama pelanggan dalam bentuk Toast.
- v.startAnimation(btnAnimasi): Menjalankan animasi pada item yang diklik.

5. Class PelangganAddActivity.java

```
package com.example.laundryapps.pelanggan;

import android.annotation.SuppressLint;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import androidx.activity.EdgeToEdge;
import androidx.appcompat.app.AppCompatActivity;

import com.example.laundryapps.R;
import com.example.laundryapps.database.SQLiteHelper;
import com.example.laundryapps.model.ModelPelanggan;

import java.util.UUID;

public class PelangganAddActivity extends AppCompatActivity {
    EditText nama, email, hp; 2 usages
    Button btnSimpan, btnBatal; 2 usages
    SQLiteHelper db; 2 usages

    @SuppressLint("MissingInflatedId")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable(this);
        setContentView(R.layout.activity_pelanggan_add);
    }
}
```

```

        nama = (EditText) findViewById(R.id.edPelNama);
        email = (EditText) findViewById(R.id.edPelEmail);
        hp = (EditText) findViewById(R.id.edPelAddHp);
        btnSimpan = (Button) findViewById(R.id.btnPelAddSimpan);
        btnBatal = (Button) findViewById(R.id.btnPelAddBatal);

        db = new SQLiteHelper(context: PelangganAddActivity.this);

        btnSimpan.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                ModelPelanggan mp = new ModelPelanggan();
                String uniqueID = UUID.randomUUID().toString();
                mp.setNama(nama.getText().toString());
                mp.setEmail(email.getText().toString());
                mp.setHp(hp.getText().toString());

                Toast.makeText(context: PelangganAddActivity.this, text: "Nama: " + mp.getNama() + " Email: " + mp.getEmail() + " Hp: " + mp.getHp(), Toast.LENGTH_SHORT).show();

                boolean cek = db.insertPelanggan(mp);
                if (cek == true) {
                    Toast.makeText(context: PelangganAddActivity.this, text: "Data berhasil disimpan", Toast.LENGTH_SHORT).show();
                    startActivity(new Intent(context: PelangganAddActivity.this, PelangganActivity.class));
                    finish();
                } else {
                    Toast.makeText(context: PelangganAddActivity.this, text: "Data gagal disimpan", Toast.LENGTH_SHORT).show();
                }
            }
        });

        btnBatal.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }
        });
    }
}

```

Penjelasan :

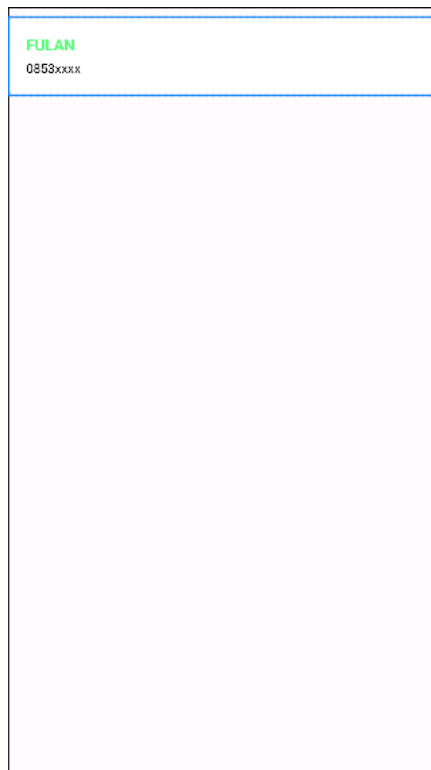
- Deklarasi Variabel:
- EditText edtNama, edtEmail, edtTelp: Input form untuk nama, email, dan nomor telepon pelanggan.
- Button btnSimpan, btnBatal: Tombol untuk menyimpan data atau membatalkan aksi.
- SQLiteHelper db: Objek untuk operasi database SQLite.
- Metode onCreate():
- setContentView(R.layout.activity_pelanggan_add): Mengatur layout XML yang digunakan untuk aktivitas ini.
- findViewById(R.id.edPelAddNama): Inisialisasi EditText untuk nama pelanggan.
- findViewById(R.id.edPelAddEmail): Inisialisasi EditText untuk email pelanggan.

- `findViewById(R.id.edPelAddHp)`: Inisialisasi EditText untuk nomor HP pelanggan.
- `findViewById(R.id.btnPelAddSimpan)`: Inisialisasi tombol "Simpan".
- `findViewById(R.id.btnPelAddBatal)`: Inisialisasi tombol "Batal".
- `db = new SQLiteHelper(this)`: Inisialisasi database helper untuk melakukan operasi database.
- `btnSimpan.setOnClickListener()`:
- `ModelPelanggan mp = new ModelPelanggan()`: Membuat objek ModelPelanggan baru.
- `String uuid = UUID.randomUUID().toString()`: Membuat ID unik untuk pelanggan.
- `mp.setId(uuid)`: Mengatur ID pelanggan.
- `mp.setNama(edtNama.getText().toString())`: Mengambil nilai input dari EditText dan menyetrnya ke atribut nama pelanggan.
- `mp.setEmail(edtEmail.getText().toString())`: Mengambil nilai input dari EditText dan menyetrnya ke atribut email pelanggan.
- `mp.setHp(edtTelp.getText().toString())`: Mengambil nilai input dari EditText dan menyetrnya ke atribut nomor HP pelanggan.
- `Toast.makeText()`: Menampilkan informasi nama, email, dan nomor telepon pelanggan dalam pesan Toast.
- Menyimpan Data ke Database:
- `boolean cek = db.insertPelanggan(mp)`: Memasukkan data pelanggan ke database menggunakan metode `insertPelanggan` dari `SQLiteHelper`.
- `if (cek)`: Jika penyimpanan berhasil, menampilkan pesan "Data berhasil disimpan" dan beralih ke `PelangganActivity`.
- `else`: Jika penyimpanan gagal, menampilkan pesan "Data gagal disimpan".
- `btnBatal.setOnClickListener()`:
- Ketika tombol "Batal" diklik, aktivitas saat ini ditutup menggunakan `finish()`, dan pengguna dikembalikan ke layar sebelumnya tanpa menyimpan data.

6. Layout Item_pelanggan.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:orientation="vertical"
    android:layout_marginTop="8dp"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">
    <androidx.cardview.widget.CardView
        android:id="@+id/cvItemPel"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:cardCornerRadius="4dp"
    >
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:padding="16dp"
            android:orientation="vertical">
            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:weightSum="2"
                android:orientation="horizontal">
                <LinearLayout
                    android:layout_weight="1"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:orientation="vertical">
                    <TextView
                        android:id="@+id/tvItemPelNama"
                        android:textSize="15sp"
                        android:textStyle="bold"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:text="FULAN"
                        android:textColor="#52FF75"/>
                    <TextView
                        android:id="@+id/tvItemPelHp"
                        android:textSize="12sp"
                        android:layout_marginTop="4dp"
                        android:textColor="@color/black"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:text="0853xxxx"/>
                </LinearLayout>
            </LinearLayout>
        </LinearLayout>
    </androidx.cardview.widget.CardView>
</LinearLayout>
```

Output Tampilan :



Penjelasan :

Layout XML ini mendefinisikan struktur tampilan untuk menampilkan item pelanggan di dalam aplikasi. Root layout menggunakan `LinearLayout` dengan orientasi vertikal dan memiliki background putih serta margin di bagian atas. Layout ini berisi sebuah `CardView` yang digunakan untuk membungkus elemen-elemen dalam tampilan, memberikan efek sudut membulat melalui atribut `cardCornerRadius`.

`CardView` ini digunakan untuk menciptakan tampilan yang rapi dan tertata untuk setiap item pelanggan. Di dalam `CardView`, terdapat `LinearLayout` dengan orientasi vertikal, yang berfungsi untuk menampung komponen-komponen tampilan seperti nama pelanggan dan nomor telepon.

Layout ini juga memiliki padding yang mengambil nilai dari resources agar memberikan jarak yang sesuai antara komponen-komponen di dalamnya. Pada `LinearLayout` berikutnya yang memiliki orientasi horizontal dan `weightSum 2`, item-item di dalamnya akan dibagi secara merata menggunakan `layout_weight`, memastikan setiap komponen menyesuaikan ukuran layar dengan baik. Untuk menampilkan data pelanggan, digunakan dua `TextView`. Yang pertama, `TextView` dengan ID `tvItemPelangganName` bertugas menampilkan nama pelanggan dengan ukuran teks 16sp dan warna kuning. `TextView` kedua dengan ID `tvItemPelangganTelp` digunakan untuk menampilkan nomor telepon pelanggan,

dengan ukuran teks yang lebih kecil (12sp) dan berwarna hitam. Keduanya ditempatkan di dalam LinearLayout vertikal untuk memastikan tampilan yang teratur dan mudah dibaca oleh pengguna aplikasi.

7. Layout activity_pelanggan.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:background="@color/white"
    tools:context=".pelanggan.PelangganActivity">

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

        <Button
            android:id="@+id/btnPelAdd"
            android:backgroundTint="#4EF47D"
            android:padding="16dp"
            android:textSize="12sp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginBottom="12dp"
            android:text="Tambah Pelanggan"
            android:textColor="@color/black"
            app:icon="@drawable/ic_add" />

        <androidx.recyclerview.widget.RecyclerView
            android:id="@+id/rvPelanggan"
            android:layout_width="match_parent"
            android:layout_height="wrap_content" />

    </LinearLayout>

</LinearLayout>
```

Output Tampilan :



Penjelasan :

Kode XML di atas adalah struktur tata letak (layout) dalam Android untuk aktivitas bernama `PelangganActivity`. Layout ini menggunakan komponen `LinearLayout` dengan orientasi vertikal dan latar belakang berwarna putih. Di dalamnya terdapat dua komponen utama:

pertama, sebuah `Button` yang digunakan untuk menambahkan pelanggan dengan teks "Tambah Pelanggan",

dan kedua, sebuah `RecyclerView` yang digunakan untuk menampilkan daftar pelanggan. Komponen `RecyclerView` ditempatkan di bawah tombol dan akan memuat data dalam bentuk daftar yang dapat digulir secara dinamis.

Tombol dengan ID `btnPLAdd` memiliki atribut seperti `padding`, ukuran teks `12sp`, serta ikon dengan sumber dari `drawable ic_add`. Atribut `backgroundTint` berfungsi untuk mengatur warna latar belakang tombol menjadi kuning muda (`#F4D04E`).

Selanjutnya, komponen `RecyclerView` yang memiliki ID `rvPelanggan` akan digunakan untuk menampilkan data pelanggan secara efisien dalam bentuk daftar atau grid menggunakan adapter dan view holder yang akan diprogram lebih lanjut di bagian kode Java atau Kotlin dalam aplikasi.

8. Layout activity_pelanggan_add.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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.PelangganAddActivity">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:layout_margin="16dp">
        <TextView
            android:textColor="@color/black"
            android:textStyle="bold"
            android:layout_marginBottom="16dp"
            android:textSize="23sp"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="PELANGGAN"/>
        <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/edPelNama"
    android:layout_marginTop="8dp"
    android:padding="12dp"
    android:layout_width="match_parent"
    android:layout_height="48dp"
/>
</LinearLayout>
<LinearLayout
    android:layout_marginTop="12dp"
    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="Email"
        android:textColor="@android:color/black"/>
    <EditText
        android:inputType="textEmailAddress"
        android:id="@+id/edPelEmail"
        android:layout_marginTop="8dp"
        android:padding="12dp"
        android:drawablePadding="8dp"
        android:drawableLeft="@drawable/ic_email"
        android:layout_width="match_parent"
        android:layout_height="48dp"
        />
    </LinearLayout>
    <LinearLayout
        android:layout_marginTop="12dp"
        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"/>
        <EditText
            android:drawablePadding="8dp"
            android:id="@+id/edPelAddHp"
            android:inputType="number"
            android:layout_marginTop="8dp"

```

```

        android:padding="12dp"
        android:drawableLeft="@drawable/ic_hp"
        android:layout_width="match_parent"
        android:layout_height="48dp"
    />
</LinearLayout>
<LinearLayout
    android:layout_marginTop="32dp"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:weightSum="2">
    <Button
        android:id="@+id/btnPelAddSimpan"
        android:layout_marginRight="4dp"
        android:background="#3DF24C"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Simpan"/>
    <Button
        android:id="@+id/btnPelAddBatal"
        android:layout_marginLeft="4dp"
        android:textColor="@color/white"
        android:background="@color/black"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"

```

```

        android:layout_weight="1"
        android:text="Batal"/>
</LinearLayout>
</LinearLayout>
</LinearLayout>

```

Output tampilan :

The screenshot shows a mobile application interface with a title bar at the top containing the text "PELANGGAN". Below the title bar, there are three input fields, each with a label and an icon to its left: "Nama" with a person icon, "Email" with an envelope icon, and "HP" with a telephone handset icon. Each input field is represented by a horizontal line. At the bottom of the form, there are two buttons: "Simpan" (Save) and "Batal" (Cancel), both with a yellow background and black text. The "Batal" button has a small red close icon in its top right corner.

Penjelasan :

Kode XML di atas menggambarkan tata letak untuk aktivitas Android bernama `PelangganActivity` yang menggunakan `ConstraintLayout` sebagai root layout-nya. Layout ini dirancang untuk menambahkan atau mengedit informasi pelanggan, dengan beberapa komponen input seperti `TextView` dan `EditText` yang digunakan untuk mengumpulkan data pengguna, yaitu nama, email, dan nomor HP. Atribut seperti `padding`, `background`, dan `textColor` digunakan untuk mengatur estetika visual komponen.

Bagian pertama layout adalah `TextView` berukuran besar dengan teks "Pelanggan", yang berfungsi sebagai judul halaman. Kemudian, ada tiga set `LinearLayout` yang masing-masing mengandung sebuah `TextView` untuk label (seperti "Nama", "Email", dan "HP") serta `EditText` untuk memasukkan data terkait. Komponen `EditText` ini dilengkapi dengan ikon di sebelah kiri yang menandakan jenis data yang akan diinput, seperti ikon pengguna untuk nama, ikon email, dan ikon ponsel untuk nomor HP. Atribut `background` digunakan untuk memberi batas (border) hijau di sekitar area input.

Bagian terakhir adalah dua tombol, yaitu tombol "Simpan" dan "Batal", yang ditempatkan berdampingan dalam sebuah `LinearLayout` horizontal dengan `weightSum` sebesar 2. Tombol- tombol ini memiliki fungsi masing-masing untuk menyimpan atau membatalkan perubahan data pelanggan. Tombol "Simpan" diberi warna latar belakang kuning (`#FFC107`), sedangkan tombol "Batal" menggunakan warna kuning yang lebih terang (`#FFE33B`). Layout ini dirancang untuk memastikan bahwa setiap elemen ditempatkan secara rapi dan mudah digunakan dalam tampilan aplikasi.

B. LAYANAN

1. Class SQLiteHelper2.java

```
package com.example.laundryapps.database;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

import com.example.laundryapps.model.ModelLayanan;

import java.util.ArrayList;
import java.util.List;

public class SQLiteHelper2 extends SQLiteOpenHelper {

    public static final String DATABASE_NAME = "my_service.db";
    public static final int DATABASE_VERSION = 1;
    public static final String TABLE_SERVICE = "service";

    public static final String KEY_SERVICE_ID = "service_id"; // ID layanan
    public static final String KEY_SERVICE_NAME = "name"; // Nama layanan
    public static final String KEY_SERVICE_PRICE = "price"; // Harga layanan

    // Query untuk membuat tabel service
    private static final String CREATE_TABLE_SERVICE = "CREATE TABLE " +
        TABLE_SERVICE + " (" +
        + KEY_SERVICE_ID + " TEXT PRIMARY KEY, " // Ubah dari INTEGER menjadi TEXT untuk UUI
        + KEY_SERVICE_NAME + " TEXT, "
        + KEY_SERVICE_PRICE + " REAL)";

    // Konstruktor
    public SQLiteHelper2(Context context) {
```

```
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        // Membuat tabel service
        db.execSQL(CREATE_TABLE_SERVICE);
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        // Hapus tabel lama jika ada
        db.execSQL("DROP TABLE IF EXISTS " + TABLE_SERVICE);
        // Buat tabel baru
        onCreate(db);
    }

    // Method untuk menambah data layanan
    public boolean insertService(String id, String name, double price) {
        SQLiteDatabase db = this.getWritableDatabase();
        ContentValues values = new ContentValues();
        values.put(KEY_SERVICE_ID, id); // Menyimpan ID layanan
        values.put(KEY_SERVICE_NAME, name);
        values.put(KEY_SERVICE_PRICE, price);

        long rowId = db.insert(TABLE_SERVICE, null, values);
        db.close();
        return rowId != -1; // Mengembalikan true jika insert berhasil
    }
}
```



```
// Method untuk mengambil semua data layanan
public List<Modellayanan> getAllServices() { //usage
    List<Modellayanan> services = new ArrayList<>();
    String selectQuery = "SELECT * FROM " + TABLE_SERVICE;

    SQLiteDatabase db = this.getReadableDatabase();
    Cursor cursor = db.rawQuery(selectQuery, selectionArgs: null);

    if (cursor.moveToFirst()) {
        do {
            Modellayanan service = new Modellayanan();
            service.setId(cursor.getString(0)); // Mengambil id layanan (sekarang String)
            service.setName(cursor.getString(1)); // Mengambil nama layanan
            service.setPrice(cursor.getInt(2)); // Mengambil harga layanan
            services.add(service);
        } while (cursor.moveToNext());
    }

    cursor.close();
    db.close();
    return services;
}
}
```

Penjelasan :

- Kelas SQLiteHelper2:

- Kelas ini adalah subclass dari SQLiteOpenHelper yang digunakan untuk mengelola database SQLite di aplikasi Android.

- Digunakan untuk membuat, meng-upgrade, dan mengoperasikan tabel layanan (service) dalam database.

- Konstanta:

- DATABASE_NAME: Nama database adalah "my_service.db".

- DATABASE_VERSION: Versi database saat ini adalah 1.

- TABLE_SERVICE: Nama tabel adalah "service".

- KEY_SERVICE_ID: Kolom untuk ID layanan yang berupa teks.

- KEY_SERVICE_NAME: Kolom untuk nama layanan.

- KEY_SERVICE_PRICE: Kolom untuk harga layanan.

- Konstruktor SQLiteHelper2:

- Menginisialisasi database SQLite dengan nama dan versi yang telah ditentukan.

- Metode onCreate:

- Digunakan untuk membuat tabel "service" dengan kolom service_id, name, dan price.

- Metode onUpgrade:

- Menghapus tabel yang ada jika versi database berubah.

- Setelah tabel dihapus, tabel baru akan dibuat dengan memanggil onCreate.

- Metode insertService:

- Menyimpan data layanan ke dalam tabel "service".
- Parameter: id (UUID layanan), name (nama layanan), dan price (harga layanan).
- Menggunakan ContentValues untuk menyimpan data ke dalam tabel dan mengembalikan true jika proses berhasil.
- Metode getAllServices:
- Mengambil semua data layanan dari tabel "service".
- Membuat list ModelLayanan dari hasil query yang menggunakan Cursor.
- Mengembalikan daftar objek ModelLayanan berisi ID, nama, dan harga layanan yang tersimpan di database.

2. Class AdapterLayanan.java

```
package com.example.laundryapps.adapter;

import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;

import com.example.laundryapps.R;
import com.example.laundryapps.model.ModelLayanan;

import java.util.List;

public class AdapterLayanan extends RecyclerView.Adapter<AdapterLayanan.LayananViewHolder> {
    private Context context;
    private List<ModelLayanan> layananList;

    // Konstruktor AdapterLayanan
    public AdapterLayanan(Context context, List<ModelLayanan> layananList) {
        this.context = context;
        this.layananList = layananList;
    }

    @NonNull
    @Override
    public LayananViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        // Inflate layout untuk item layanan
        View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.item_layanan, parent, attachToRoot: false);
        return new LayananViewHolder(view);
    }
}
```

```

@Override
public void onBindViewHolder(@NonNull LayananViewHolder holder, int position) {
    // Menghubungkan data layanan dengan tampilan
    ModelLayanan layanan = layananList.get(position);
    holder.txtName.setText(layanan.getName());
    holder.txtPrice.setText(String.valueOf(layanan.getPrice()));
}

@Override
public int getItemCount() {
    // Mengembalikan jumlah item dalam daftar layanan
    return layananList.size();
}

// ViewHolder untuk AdapterLayanan
public class LayananViewHolder extends RecyclerView.ViewHolder { 4 usages
    TextView txtName, txtPrice; 2 usages

    public LayananViewHolder(@NonNull View itemView) { 1 usage
        super(itemView);
        txtName = itemView.findViewById(R.id.tvItemLayananName);
        txtPrice = itemView.findViewById(R.id.tvItemHarga);
    }
}
}

```

Penjelasan :

Kelas AdapterLayanan:

- Menggunakan RecyclerView.Adapter untuk menampilkan data layanan di dalam tampilan daftar (RecyclerView).
- Mengelola data layanan dalam bentuk list (List<ModelLayanan> layananList).
- Context digunakan untuk menyimpan informasi tentang konteks aplikasi, yang nantinya bisa digunakan untuk inflating layout.

Konstruktor:

- AdapterLayanan(Context context, List<ModelLayanan> layananList) adalah konstruktor yang menerima parameter konteks dan daftar layanan.
- Inisialisasi variabel context dan layananList dilakukan di sini.
- onCreateViewHolder:
 - Metode ini digunakan untuk meng-inflate layout item layanan (R.layout.item_layanan) dan membuat objek LayananViewHolder.
 - Mengembalikan objek LayananViewHolder yang akan memegang tampilan item individual dalam RecyclerView.
- onBindViewHolder:
 - Menghubungkan data layanan dengan tampilan item (ViewHolder).

- Mengambil objek ModelLayanan dari daftar berdasarkan posisi, lalu mengatur teks di txtName dan txtPrice sesuai dengan data layanan (layanan.getName() dan layanan.getPrice()).
- getItemCount:
- Mengembalikan jumlah total item dalam daftar layanan (layananList.size()).
- Kelas LayananViewHolder:
- Menyimpan referensi ke tampilan yang ada di item layanan (TextView untuk nama layanan dan harga).
- txtName untuk nama layanan dan txtPrice untuk harga layanan, yang diambil dari layout item (findViewById(R.id.tvItemLayananName) dan findViewById(R.id.tvItemHarga)).

3. Class ModelLayanan.java

```
package com.example.laundryapps.model;

public class ModelLayanan { 15 usages
    private String id; // ID layanan 3 usages
    private String name; // Nama layanan 3 usages
    private int price; // Mengubah harga dari double ke int 3 usages

    public ModelLayanan() {} 2 usages

    public ModelLayanan(String id, String name, int price) { // Constructor yang diperbarui no usages
        this.id = id;
        this.name = name;
        this.price = price;
    }

    public String getId() {
        return id;
    }

    public void setId(String id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public int getPrice() { 2 usages
        return price; // Mengembalikan harga sebagai integer
    }

    public void setPrice(int price) { 2 usages
        this.price = price;
    }
}
```

Penjelasan :

Atribut:

- String id: Menyimpan ID unik layanan.
- String name: Menyimpan nama layanan.
- int price: Menyimpan harga layanan, yang diubah dari tipe double ke int.

Konstruktor:

- Konstruktor kosong ModelLayanan(): Konstruktor default tanpa parameter.
- Konstruktor dengan parameter ModelLayanan(String id, String name, int price): Menginisialisasi objek layanan dengan ID, nama, dan harga.

Getter dan Setter:

- getId() dan setId(String id): Mengambil dan mengatur ID layanan.
- getName() dan setName(String name): Mengambil dan mengatur nama layanan.
- getPrice() dan setPrice(int price): Mengambil dan mengatur harga layanan dalam bentuk integer.

4. Class LayananActivity.java

```
package com.example.laundryapps.layanan;

import android.annotation.SuppressLint;
import android.app.ProgressDialog;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.animation.AlphaAnimation;
import android.widget.Button;
import android.widget.Toast;

import androidx.activity.EdgeToEdge;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.graphics.Insets;
import androidx.core.view.ViewCompat;
import androidx.core.view.WindowInsetsCompat;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import com.example.laundryapps.R;
import com.example.laundryapps.adapter.AdapterLayanan;
import com.example.laundryapps.database.SQLiteHelper2;
import com.example.laundryapps.model.ModelLayanan;

import java.util.ArrayList;
import java.util.List;

public class LayananActivity extends AppCompatActivity {

    private SQLiteHelper2 db; // Database helper untuk layanan 2 usages
    private Button btnAddLayanan; // Tombol untuk menambah layanan 2 usages
    private RecyclerView rvLayanan; // RecyclerView untuk menampilkan data layanan 4 usages
```

```

private AdapterLayanan adapterLayanan; // Adapter untuk RecyclerView 3 usages
private ArrayList<ModelLayanan> list; // List untuk menyimpan data layanan 4 usages
private ProgressDialog progressDialog; // Untuk menampilkan loading dialog 8 usages
private AlphaAnimation btnAnimasi = new AlphaAnimation(1F, 0.5F); // Animasi tombol 10 usages

@SuppressLint("MissingInflatedId")
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    EdgeToEdge.enable(this);
    setContentView(R.layout.activity_layanan); // Pastikan ini sesuai dengan layout yang benar

    // Set window insets untuk system bars
    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;
    });

    // Inisialisasi view dan event handling
    setContentView();
    eventHandling();
    getData();
}

private void eventHandling() { 1 usage
    btnAddLayanan.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            startActivity(new Intent(getApplicationContext(), LayananActivity.this, LayananAddActivity.class));
        }
    });
}

private void getData() { 1 usage
    list.clear(); // Bersihkan list sebelum mengambil data baru
    showMsg(); // Tampilkan loading dialog

    try {
        List<ModelLayanan> p = db.getAllServices(); // Ambil data dari database
        if (p != null && p.size() > 0) {
            list.addAll(p); // Tambahkan data ke list
            adapterLayanan.notifyDataSetChanged(); // Beri tahu adapter bahwa ada data baru
        } else {
            Toast.makeText(context, this, text: "Data tidak ditemukan", Toast.LENGTH_SHORT).show();
        }
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        progressDialog.dismiss(); // Sembunyikan loading dialog
    }
}

private void setContentView() { 1 usage
    db = new SQLiteHelper2(context, this); // Inisialisasi SQLiteHelper untuk layanan
    progressDialog = new ProgressDialog(context, this); // Inisialisasi ProgressDialog
    btnAddLayanan = findViewById(R.id.btnAdd); // Temukan tombol tambah layanan dari layout
    rvLayanan = findViewById(R.id.rvLayanan); // Temukan RecyclerView dari layout
    list = new ArrayList<>(); // Inisialisasi list untuk menampung data layanan

    // Inisialisasi LinearLayoutManager untuk RecyclerView
    LinearLayoutManager llm = new LinearLayoutManager(context, this);
    rvLayanan.setLayoutManager(llm); // Set layout manager ke RecyclerView

    rvLayanan.setHasFixedSize(true); // Optimalkan ukuran RecyclerView

    // Inisialisasi adapter untuk RecyclerView
    adapterLayanan = new AdapterLayanan(context, this, list);
    rvLayanan.setAdapter(adapterLayanan); // Set adapter ke RecyclerView
}

private void showMsg() { 1 usage
    if (progressDialog == null) {
        progressDialog = new ProgressDialog(context, this);
        progressDialog.setTitle("Informasi");
        progressDialog.setMessage("Loading Data...");
        progressDialog.setCancelable(false); // Tidak bisa dibatalkan dengan menekan di luar
    }
    progressDialog.show(); // Tampilkan dialog
}
}

```

Penjelasan :

Kelas LayananActivity:

- Kelas ini adalah Activity yang digunakan untuk menampilkan daftar layanan dalam aplikasi laundry.
- Menggunakan RecyclerView untuk menampilkan data layanan, dan ProgressDialog untuk menampilkan loading saat mengambil data.

Variabel Utama:

- SQLiteHelper2 db: Objek database helper untuk operasi database layanan.
- Button btnAddLayanan: Tombol untuk menambahkan layanan baru.
- RecyclerView rvLayanan: RecyclerView untuk menampilkan daftar layanan.
- AdapterLayanan adapterLayanan: Adapter untuk menghubungkan data layanan ke RecyclerView.
- ArrayList<ModelLayanan> list: List untuk menyimpan data layanan yang akan ditampilkan.

Metode Utama:

- onCreate: Menginisialisasi komponen UI dan menyiapkan RecyclerView serta event handling untuk tombol tambah layanan.
- eventHandling: Mengatur klik tombol tambah layanan untuk membuka LayananAddActivity.
- getData: Mengambil data layanan dari database menggunakan SQLiteHelper2 dan menampilkannya di RecyclerView.
- setView: Inisialisasi tampilan dan layout, serta menghubungkan adapter dengan RecyclerView.
- showMsg: Menampilkan ProgressDialog saat data sedang diambil.

5. Class LayananAddActivity.java

```
package com.example.laundryapps.layanan;

import android.annotation.SuppressLint;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import com.example.laundryapps.R;
import com.example.laundryapps.database.SQLiteHelper2;
import com.example.laundryapps.model.ModelLayanan;

import java.util.UUID;

public class LayananAddActivity extends AppCompatActivity {

    EditText edtLayananName, edtLayananPrice; // Variabel untuk nama dan harga layanan 2 usages
    Button btnAddLayanan, btnCancel; 2 usages
    SQLiteHelper2 db; 2 usages

    @SuppressWarnings("MissingInflatedId")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_layanan_add);

        // Inisialisasi Views
        edtLayananName = findViewById(R.id.edLayananName);
        edtLayananPrice = findViewById(R.id.edLayananPrice);
        btnAddLayanan = findViewById(R.id.btnPeLAddSimpan);
        btnCancel = findViewById(R.id.btnPeLAddBatal);

        // Inisialisasi SQLiteHelper
        db = new SQLiteHelper2(this);

        // Set action untuk tombol Simpan
        btnAddLayanan.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                ModelLayanan ml = new ModelLayanan();
                String uuid = UUID.randomUUID().toString(); // Generate unique ID
                ml.setId(uuid); // Set ID sebagai String UUID
                ml.setName(edtLayananName.getText().toString());

                // Parsing harga ke integer
                String priceText = edtLayananPrice.getText().toString();
                int price;

                try {
                    price = Integer.parseInt(priceText); // Mengonversi input menjadi integer
                } catch (NumberFormatException e) {
                    Toast.makeText(this, "Harga tidak valid", Toast.LENGTH_SHORT).show();
                    return; // Menghentikan eksekusi jika harga tidak valid
                }

                ml.setPrice(price); // Mengatur harga ke model layanan

                // Menyimpan ke database dan memeriksa apakah berhasil
                boolean cek = db.insertService(ml.getId(), ml.getName(), ml.getPrice());
                if (cek) {
                    Toast.makeText(this, "Data berhasil disimpan", Toast.LENGTH_SHORT).show();
                    startActivity(new Intent(this, LayananActivity.class));
                    finish();
                } else {
                    Toast.makeText(this, "Data gagal disimpan", Toast.LENGTH_SHORT).show();
                }
            }
        });

        // Fungsi untuk tombol Batal
        btnCancel.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish(); // Menutup activity jika tombol dibatalkan
            }
        });
    }
}
```

Penjelasan:

Variabel Utama:

- o EditText edtLayananName, edtLayananPrice: Input untuk nama dan harga layanan.
- o Button btnAddLayanan, btnCancel: Tombol untuk menyimpan layanan dan membatalkan.
- o SQLiteHelper2 db: Objek helper untuk mengelola operasi database layanan.
- Pada metode onCreate, variabel-variabel diinisialisasi dengan findViewById untuk menghubungkan dengan elemen di layout XML. Inisialisasi database juga dilakukan di sini melalui SQLiteHelper2.

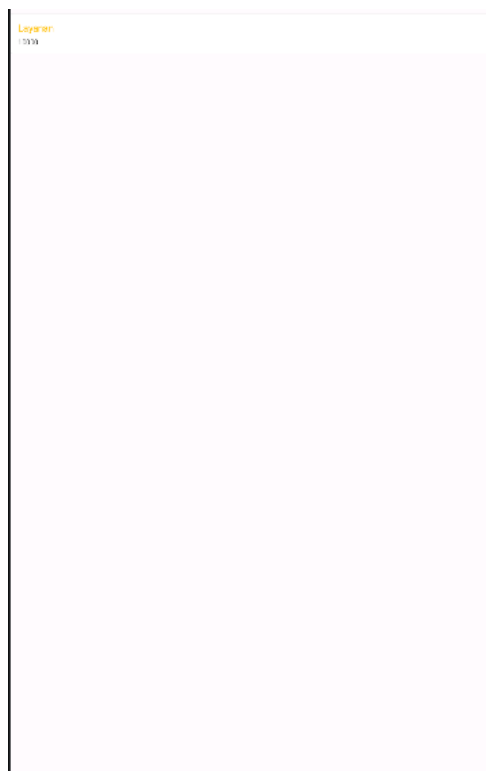
```

<TextView
    android:id="@+id/tvItemLayananName"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="layanan"
    android:textColor="#FFC107"
    android:textSize="16sp"/>

    <TextView
        android:id="@+id/tvItemHarga"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="4dp"
        android:text="10000"
        android:textColor="@color/black"
        android:textSize="12sp"/>
</LinearLayout>
</LinearLayout>
</LinearLayout>
</androidx.cardview.widget.CardView>
</LinearLayout>

```

Output Tampilan :



Penjelasan :

Root Layout (LinearLayout):

- `xmlns:android="http://schemas.android.com/apk/res/android"`: Namespace untuk atribut Android.
- `xmlns:app="http://schemas.android.com/apk/res-auto"`: Namespace untuk atribut dari library tambahan seperti CardView.

- `android:orientation="vertical"`: Arah susunan elemen di dalam layout ini adalah vertikal.
- `android:layout_width="match_parent"`: Lebar mengikuti lebar layar.
- `android:layout_height="wrap_content"`: Tinggi mengikuti tinggi konten.
- `android:layout_marginTop="@dimen/sm"`: Memberikan margin di bagian atas yang ditentukan dari dimensi bernama sm.
- `android:background="@color/white"`: Latar belakang berwarna putih.

CardView (CardView):

- `android:id="@+id/cvItemPelanggan"`: ID untuk referensi elemen ini dalam kode Java/Kotlin.
- `android:layout_width="match_parent"`: Lebar mengikuti lebar layar.
- `android:layout_height="wrap_content"`: Tinggi mengikuti konten di dalamnya.
- `app:cardCornerRadius="@dimen/xs"`: Memberikan radius sudut pada CardView, dengan nilai diambil dari dimensi xs.

Inner LinearLayout (Parent):

- `android:layout_width="match_parent"`: Lebar mengikuti lebar layar.
- `android:layout_height="wrap_content"`: Tinggi mengikuti konten di dalamnya.
- `android:padding="@dimen/md"`: Padding di sekitar elemen dalam layout, dengan nilai dari dimensi md.
- `android:orientation="vertical"`: Elemen di dalamnya disusun secara vertikal.

LinearLayout (Info Layanan):

- `android:layout_width="match_parent"`: Lebar mengikuti layar.
- `android:layout_height="wrap_content"`: Tinggi mengikuti konten.
- `android:weightSum="2"`: Menetapkan total bobot untuk elemen di dalamnya.
- `android:orientation="horizontal"`: Elemen di dalamnya disusun secara horizontal.

LinearLayout (Kolom Informasi Layanan):

- `android:layout_width="0dp"`: Mengatur lebar mengikuti bobot (`layout_weight`).
- `android:layout_weight="1"`: Memberikan bobot 1, sehingga elemen ini mengambil separuh ruang.
- `android:layout_height="wrap_content"`: Tinggi mengikuti konten.
- `android:orientation="vertical"`: Elemen di dalamnya disusun secara vertikal.

TextView (Nama Layanan):

- `android:id="@+id/tvItemLayananName"`: ID untuk referensi dalam kode Java/Kotlin.
- `android:layout_width="wrap_content"`: Lebar menyesuaikan teks.
- `android:layout_height="wrap_content"`: Tinggi mengikuti konten.
- `android:text="Layanan"`: Menampilkan teks "Layanan" sebagai contoh.
- `android:textColor="#FFC107"`: Teks berwarna kuning (#FFC107).
- `android:textSize="16sp"`: Ukuran teks sebesar 16sp.

TextView (Harga Layanan):

- `android:id="@+id/tvItemHarga"`: ID untuk referensi dalam kode Java/Kotlin.
- `android:layout_width="wrap_content"`: Lebar menyesuaikan konten teks.
- `android:layout_height="wrap_content"`: Tinggi mengikuti konten.
- `android:layout_marginTop="@dimen/xs"`: Memberikan margin atas kecil, dengan nilai dari dimensi xs.
- `android:text="10000"`: Teks yang ditampilkan adalah harga (contoh: 10000).
- `android:textColor="@color/black"`: Teks berwarna hitam.
- `android:textSize="12sp"`: Ukuran teks sebesar 12sp.

7. Layout activity_layanan.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:layout_width="match_parent"
    android:id="@+id/main"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="@color/white"
    tools:context=".Layanan.LayananActivity">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:layout_margin="16dp">
        <Button
            android:id="@+id/btnPLAdd"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginBottom="12dp"
            android:backgroundTint="#45FF73"
            android:padding="16dp"
            android:text="Tambah Layanan"
            android:textSize="12sp"
            app:icon="@drawable/ic_add"/>
        </Button>
    </LinearLayout>

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/rvLayanan"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>
</LinearLayout>
```

Output tampilan :



Penjelasan

Root Layout (LinearLayout):

- `xmlns:android`, `xmlns:app`, `xmlns:tools`: Deklarasi namespace untuk atribut Android library tambahan, dan fitur alat.
- `android:layout_width="match_parent"`: Menyesuaikan lebar root layout dengan lebar layar.
- `android:layout_height="match_parent"`: Menyesuaikan tinggi root layout dengan tinggi layar.
- `android:orientation="vertical"`: Menyusun elemen secara vertikal.
- `android:background="@color/white"`: Menetapkan latar belakang layout berwarna putih.

Sub-Layout Linear (Untuk Tombol):

- `android:layout_margin="16dp"`: Memberikan margin 16dp di sekeliling layout.
- Di dalamnya terdapat sebuah tombol.

Button:

- `android:id="@+id/btnPLAdd"`: ID tombol "Tambah Layanan".
- `android:layout_width="wrap_content"` dan `android:layout_height="wrap_content"`: Mengatur ukuran tombol agar sesuai dengan konten.

- `android:layout_marginBottom="12dp"`: Memberi margin bawah sebesar 12dp.
- `android:backgroundTint="#F4D04E"`: Menentukan warna latar belakang tombol.
- `android:padding="16dp"`: Menambahkan padding di dalam tombol.
- `android:text="Tambah Layanan"`: Teks yang ditampilkan pada tombol.
- `app:icon="@drawable/ic_add"`: Menampilkan ikon "add" di dalam tombol (harus memiliki resource gambar `ic_add`).

RecyclerView:

- `android:id="@+id/rvLayanan"`: ID untuk RecyclerView yang akan digunakan untuk menampilkan daftar layanan.
- `android:layout_width="match_parent"`: Lebar RecyclerView mengikuti lebar layar.
- `android:layout_height="wrap_content"`: Tinggi RecyclerView disesuaikan dengan konten yang ditampilkan.

8. Layout `activity_layanan_add.xml`

```
<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:padding="16dp"
    android:background="@color/white"
    tools:context=".Layanan.LayananAddActivity">

    <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="Tambah Layanan"/>
        <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 Layanan"
    android:textColor="@android:color/black"/>
<EditText
    android:drawablePadding="8dp"
    android:id="@+id/edLayananName"
    android:padding="8dp"
    android:layout_marginTop="8dp"
    android:layout_width="match_parent"
    android:layout_height="48dp"
/>
</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="Harga"
        android:textColor="@android:color/black"/>
    <EditText

```

```

        android:inputType="numberDecimal"
        android:id="@+id/edLayananPrice"
        android:drawablePadding="8dp"
        android:layout_marginTop="8dp"
        android:padding="16dp"
        android:layout_width="match_parent"
        android:layout_height="48dp"
    />
</LinearLayout>

```

```

<LinearLayout
    android:layout_marginTop="32dp"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:weightSum="2">
    <Button
        android:id="@+id/btnPelAddSimpan"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginRight="4dp"
        android:layout_weight="1"
        android:background="#FFC107"
        android:text="Simpan"
        android:textColor="@android:color/white"/>
    <Button

```

```

        <Button
            android:id="@+id/btnPeLAddBatal"
            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>
</LinearLayout>
</androidx.constraintlayout.widget.ConstraintLayout>

```

Output tampilan :

Tambah Layanan

Nama Layanan

Harga

Simpan

Batal

Penjelasan :

Root Layout (ConstraintLayout):

- `xmlns:android`, `xmlns:app`, `xmlns:tools`: Deklarasi namespace untuk atribut Android, library tambahan, dan alat bantu.
- `android:id="@+id/main"`: ID untuk root layout.
- `android:layout_width="match_parent"`: Lebar mengikuti lebar layar.
- `android:layout_height="match_parent"`: Tinggi mengikuti tinggi layar.
- `android:padding="16dp"`: Padding di sekeliling layout sebesar 16dp.
- `android:background="@color/white"`: Latar belakang layout berwarna putih.
- `tools:context=".ayanan.LayananAddActivity"`: Menetapkan activity yang menggunakan layout ini.
- **LinearLayout (Kontainer Utama):**
 - `android:layout_width="match_parent"`: Lebar menyesuaikan lebar layar.
 - `android:layout_height="wrap_content"`: Tinggi mengikuti konten.
 - `android:orientation="vertical"`: Elemen di dalamnya disusun secara vertikal.
 - `android:layout_margin="16dp"`: Memberi margin di sekeliling layout.

TextView (Judul):

- `android:text="Tambah Layanan"`: Teks untuk judul halaman.
- `android:textColor="@color/black"`: Teks berwarna hitam.
- `android:textStyle="bold"`: Teks dicetak tebal.
- `android:textSize="24sp"`: Ukuran teks sebesar 24sp.
- `android:paddingBottom="16dp"`: Padding bawah sebesar 16dp untuk memberi jarak dengan elemen berikutnya.

LinearLayout (Input Nama Layanan):

- `android:layout_width="match_parent"`: Lebar menyesuaikan layar.
- `android:layout_height="wrap_content"`: Tinggi sesuai dengan konten.
- `android:orientation="vertical"`: Elemen disusun vertikal.

Di dalamnya terdapat:

- o **TextView**: Menampilkan label "Nama Layanan".
- o **EditText**: Input untuk nama layanan, dengan padding dan background yang ditentukan.

LinearLayout (Input Harga Layanan):

- `android:layout_width="match_parent"`: Lebar sesuai layar.
- `android:layout_height="wrap_content"`: Tinggi mengikuti konten.

- `android:orientation="vertical"`: Menyusun elemen secara vertikal.

Di dalamnya terdapat:

- o `TextView`: Menampilkan label "Harga".
- o `EditText`: Input harga layanan dengan tipe input `numberDecimal`.
- `LinearLayout` (Tombol Simpan dan Batal):
 - `android:layout_width="match_parent"`: Lebar sesuai layar.
 - `android:layout_height="wrap_content"`: Tinggi sesuai konten.
 - `android:orientation="horizontal"`: Elemen disusun secara horizontal.
 - `android:weightSum="2"`: Total bobot untuk kedua tombol.

Di dalamnya terdapat dua tombol:

- o `Button` (Simpan): Tombol untuk menyimpan layanan, dengan warna latar kuning dan teks putih.
- o `Button` (Batal): Tombol untuk membatalkan aksi, dengan warna latar kuning terang dan teks putih