Modul Hari Kelima

Basic Android Application Development

Hanya untuk penggunaan pribadi dan proses pembelajaran di kelas Basic Android Application Development Codepolitan.com

1. CRUD Sederhana dengan SQLite

Database merupakan salah satu topik yang paling sering digunakan disebuah aplikasi Android. Apa jadinya jika sebuah aplikasi tidak bisa menyimpan sebuah data yang diperlukan oleh pengguna?

Solusi database yang telah didukung langsung tanpa perlu menggunakan library pihak ketiga adalah dengan menggunakan `SQLiteOpenHelper`. Kelas ini membantu developer Android saat akan menggunakan sistem database SQLite.

2. Project Baru

Mari kita buat project baru untuk belajar menggunakan SQLite database di Android. Kita beri nama project yang baru ini `BelajarDatabase` dengan `Basic Activity` sebagai template awalnya.

Tambahkan baris berikut ke `build.gradle` karena kita akan menggunakan RecyclerView dan CardView.

```
implementation 'com.android.support:recyclerview-v7:26.1.0'
implementation 'com.android.support:cardview-v7:26.1.0'
```

3. RecyclerView

Sebelum masuk ke pembahasan database, mari kita buat terlebih RecyclerView untuk aplikasi ini.

3.1. Model

Pertama buat dulu kelas data/model, berinama `Item.java`.

```
public class Product
public class Item {
```

```
                id;
            ig name;
            riority;

             ) name, int priority) {
            iame;
           .cy = priority;


        getId() {
       .n id;


    c void setId(int id) {
    this.id = id;


  ublic String getName() {
      return namd;
  }

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

  public int getPriority() {
      return priority;
  }

  public void setPriority(int priority) {
      this.priority = priority;
  }

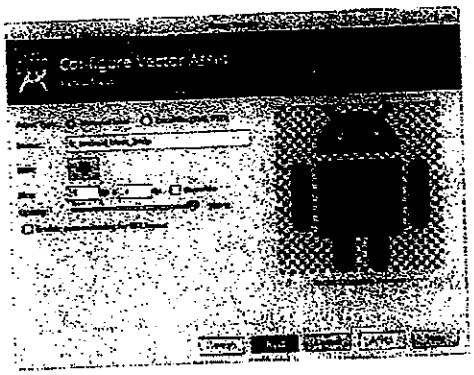
  public Item() {

  }

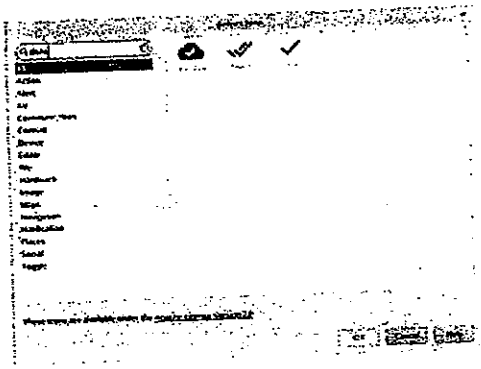
  public Item(int id, String name, int priority) {
      this.id = id;
      this.name = name;
      this.priority = priority;
  }
}
```

3.2. Import Gambar Ikon

Impor dua gambar ikon yang akan kita butuhkan di aplikasi ini. Klik kanan di *package name* aplikasi kita lalu pilih **New > Vector Asset**. Pada jendela yang muncul, klik tombol yang ada di sebelah kanan tulisan **Icon**.



Setelah itu, lanjutkan dengan mencari ikon `edit`, `add` dan `done`. Klik **Next** dan **Finish** untuk melanjutkan.



3.3. Layout Item RecyclerView

Setelah gambar ikon yang dibutuhkan telah diimpor, sekarang mari kita buat layout baru untuk item RecyclerView. Buat file baru bernama `item.xml` di direktori `layout`.

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v7.widget.CardView xmlns:android="http://
schemas.android.com/apk/res/android"
    xmlns:card_view="http://schemas.android.com/apk/res-auto"
```

```
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginBottom="1dp"
    card_view:cardElevation="2dp"
    card_view:cardUseCompatPadding="true">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:padding="12dp">
        <TextView
            android:gravity="center_vertical"
            android:id="@+id/product_name"
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="4"
            android:text="Todo"
            android:textSize="12dp"
            android:textStyle="bold" />
        <ImageView
            android:id="@+id/edit_todo"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_gravity="center_vertical"
            android:layout_weight="1"
            android:src="@drawable/ic_edit_black_24dp" />
        <ImageView
            android:id="@+id/done_todo"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:layout_weight="1"
            android:src="@drawable/ic_done_black_24dp" />
    </LinearLayout>
</android.support.v7.widget.CardView>
```

3.4. Adapter RecyclerView

Buat Adapter untuk RecyclerView kita, berinama `TodoListAdapter.java`.

```
public class TodoListAdapter extends
 RecyclerView.Adapter<TodoListAdapter.TodoViewHolder> {
```

```
    ArrayList<Item> itemArrayList;
    Context context;

    public TodoListAdapter(ArrayList<Item> itemArrayList, Context
context) {
        this.itemArrayList = itemArrayList;
        this.context = context;
    }

    @Override
    public TodoViewHolder onCreateViewHolder(ViewGroup parent, int
viewType) {
        View view = LayoutInflater.from(context).inflate(R.layout.item,
parent, false);
        return new TodoViewHolder(view);
    }

    @Override
    public void onBindViewHolder(TodoViewHolder holder, int position) {
        Item item = itemArrayList.get(position);
        holder.todoText.setText(item.getName());
    }

    @Override
    public int getItemCount() {
        return itemArrayList.size();
    }

    public class TodoViewHolder extends RecyclerView.ViewHolder{
        TextView todoText;
        ImageView editImageView;
        ImageView doneImageView;

        public TodoViewHolder(View itemView) {
            super(itemView);

            todoText = itemView.findViewById(R.id.todo_text);
            editImageView = itemView.findViewById(R.id.edit_todo);
            doneImageView = itemView.findViewById(R.id.done_todo);
        }
    }
}
```

3.5. Menampilkan Data Dummy di RecyclerView

Buka file `content.xml` lalu ubah isinya menjadi:

```
<RelativeLayout 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:layout_height="match_parent"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context="com.codepolitan.belajardatabase.MainActivity"
    tools:showIn="@layout/activity_main">

    <android.support.v7.widget.RecyclerView
        android:id="@+id/recyclerView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"/>

</RelativeLayout>
```

Lalu, tambahkan ketiga variabel berikut di `MainActivity.java`:

```
public class MainActivity extends AppCompatActivity {

    RecyclerView recyclerView;
    TodoListAdapter adapter;

    ArrayList<Item> todoList = new ArrayList<>();

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

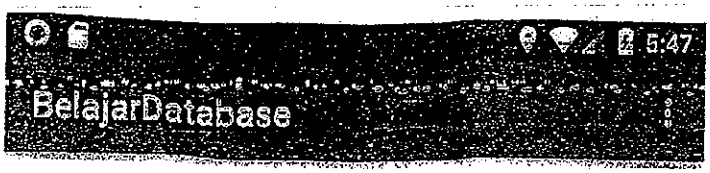
        FloatingActionButton fab = (FloatingActionButton)
 findViewById(R.id.fab);
        fab.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
```

```
                Snackbar.make(view, "Replace with your own action",
Snackbar.LENGTH_LONG)
                        .setAction("Action", null).show();
            }
        });

        setDummyData();
        recyclerView = findViewById(R.id.recyclerView);
        adapter = new TodoListAdapter(todoList, MainActivity.this);
        recyclerView.setAdapter(adapter);
        recyclerView.setLayoutManager(new
LinearLayoutManager(MainActivity.this));
    }

    private void setDummyData() {
        todoList.add(new Item("Buat Model", 3));
        todoList.add(new Item("Buat Layout Item", 1));
        todoList.add(new Item("Buat Adapter", 2));
        todoList.add(new Item("Iniasalisasi RecyclerView", 4));
    }
}
```

Jalankan project ini dan pastikan aplikasinya sudah menampilkan data dummy seperti pada gambar berikut:



Buat Model

Buat Layout Item

Buat Adapter

Inisalisasi RecyclerView



4. SQLiteOpenHelper

Untuk membuat aplikasi kita bekerja dengan database, mari buat file baru bernama `MyDatabase.java` dengan isi sebagai berikut:

```
public class MyDatabase extends SQLiteOpenHelper {

    private static final int DATABASE_VERSION = 1;
    private static final String DATABASE_NAME = "todo_db";
    private static final String TABLE_TODO = "todo_table";
    private static final String COLUMN_ID = "_id";
    private static final String COLUMN_TODO_NAME = "name";
    private static final String COLUMN_TODO_PRIORITY = "priority";

    public MyDatabase(Context context) {
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        String CREATE_PRODUCTS_TABLE = "CREATE    TABLE " +
                TABLE_TODO +
                "(" + COLUMN_ID +
                " INTEGER PRIMARY KEY," +
                COLUMN_TODO_NAME +
                " TEXT," +
                COLUMN_TODO_PRIORITY +
                " INTEGER" + ")";
        db.execSQL(CREATE_PRODUCTS_TABLE);
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int
newVersion) {
        db.execSQL("DROP TABLE IF EXISTS " + TABLE_TODO);
        onCreate(db);
    }

    public void addItem(Item item){
        ContentValues values = new ContentValues();
        values.put(COLUMN_TODO_NAME, item.getName());
        values.put(COLUMN_TODO_PRIORITY, item.getPriority());
        SQLiteDatabase db = this.getWritableDatabase();
        db.insert(TABLE_TODO, null, values);
    }

    public List<Item> getAll(){
        String sql = "select * from " + TABLE_TODO;
        SQLiteDatabase db = this.getReadableDatabase();
```

```
        List<Item> storeProducts = new ArrayList<>();
        Cursor cursor = db.rawQuery(sql, null);
        if(cursor.moveToFirst()){
            do{
                int id = Integer.parseInt(cursor.getString(0));
                String name = cursor.getString(1);
                int priority = cursor.getInt(2);
                storeProducts.add(new Item(id, name, priority));
            }while (cursor.moveToNext());
        }
        cursor.close();
        return storeProducts;
    }

    public void updateItem(Item item){
        ContentValues values = new ContentValues();
        values.put(COLUMN_TODO_NAME, item.getName());
        values.put(COLUMN_TODO_PRIORITY, item.getPriority());
        SQLiteDatabase db = this.getWritableDatabase();
        db.update(TABLE_TODO, values, COLUMN_ID    + "    = ?", new
String[] { String.valueOf(item.getId())});
    }

    public void deleteItem(int id){
        SQLiteDatabase db = this.getWritableDatabase();
        db.delete(TABLE_TODO, COLUMN_ID    + "    = ?", new String[]
{ String.valueOf(id)});
    }
}
```

Mari kita bahas kode di atas satu persatu.

4.1. Variabel-variabel konstan

Perhatikan potongan kode berikut:

```
private static final int DATABASE_VERSION = 1;
private static final String DATABASE_NAME = "todo_db";
private static final String TABLE_TODO = "todo_table";
private static final String COLUMN_ID = "_id";
private static final String COLUMN_TODO_NAME = "name";
private static final String COLUMN_TODO_PRIORITY = "priority";
```

Kita baru saja membuat variabel-variabel konstan yang nantinya akan kita gunakan dibeberapa tempat.

- `DATABASE_VERSION` merupakan variabel yang isinya harus kita ubah apabila struktur tabel ada yang berubah. Misalnya, saat kita menambah kolom baru, menghapus kolom, mengubah nama kolom, atau mengubah nama tabel, maka nilai variabel ini harus kita naikkan. Isi angka 1 saat pertama kali dibuat.
- `DATABASE_NAME` merupakan variabel yang menyimpan nama database kita. SQLite merupakan sistem database yang seluruh datanya disimpan kedalam sebuah file. Pada contoh di atas artinya kita ingin membuat sebuah database dengan nama `todo_db.db`.
- `TABLE_TODO` merupakan variabel yang menyimpan nama tabel. Pada contoh di atas artinya kita ingin membuat sebuah tabel di dalam database `todo_db.db` bernama `todo_table`.
- `COLUMN_ID`, `COLUMN_TODO_NAME`, dan `COLUMN_TODO_PRIORITY` adalah variabel-variabel yang menyimpan nama kolom-kolom. Pada contoh di atas artinya kita ingin membuat kolom bernama `_id`, `name`, dan `priority` di dalam tabel `todo_table`.

4.2. Konstruktor

```
public MyDatabase(Context context) {
    super(context, DATABASE_NAME, null, DATABASE_VERSION);
}
```

Pada potongan kode di atas kita membuat konstruktor yang nantinya akan selalu dieksekusi setiap kali melakukan inisiasi kelas `MyDatabase`.

4.3. onCreate

```
@Override
public void onCreate(SQLiteDatabase db) {
    String CREATE_PRODUCTS_TABLE = "CREATE    TABLE " +
            TABLE_TODO +
            "(" + COLUMN_ID +
            " INTEGER PRIMARY KEY," +
            COLUMN_TODO_NAME +
            " TEXT," +
```

```
            COLUMN_TODO_PRIORITY +
            " INTEGER" + ")";
    db.execSQL(CREATE_PRODUCTS_TABLE);
}
```

Method `onCreate` akan dipanggil saat kelas `MyDatabase` di inisiasi untuk pertama kali kecuali jika nanti dipanggil secara eksplisit.

Pada contoh di atas, kita membuat sebuah tabel baru dengan query SQL `CREATE TABLE todo_table( _id INTEGER PRIMARY KEY, name TEXT, priority INTEGER)`.

4.4. onUpgrade

```
@Override
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion)
 {
    db.execSQL("DROP TABLE IF EXISTS " + TABLE_TODO);
    onCreate(db);
}
```

Method `onUpgrade` akan dipanggil saat kelas `MyDatabase` di inisiasi dan angka variabel `DATABASE_VERSION` berbeda dengan inisiasi sebelumnya (artinya ada yang berubah dari struktur database kita sehingga perlu dilakukan sesuatu untuk meng-*handle*-nya).

Pada contoh kode di atas, saat terjadi perubahan struktur data, maka kita ingin menghapus tabel yang sudah ada lalu memanggil method `onCreate` untuk membuat ulang tabel tersebut.

4.5. addItem

```
public void addItem(Item item){
    ContentValues values = new ContentValues();
    values.put(COLUMN_TODO_NAME, item.getName());
    values.put(COLUMN_TODO_PRIORITY, item.getPriority());
    SQLiteDatabase db = this.getWritableDatabase();
    db.insert(TABLE_TODO, null, values);
}
```

Method `addItem` kita gunakan untuk menambahkan data ke dalam database. Di dalam method ini kita menerima sebuah objek Item.

Lalu, kita juga membuat sebuah objek `values` dengan tipe `ContentValues`. Objek bertipe `ContentValues` dapat menyimpan data dengan struktur `key-value` sama seperti Bundle (yang kita gunakan untuk mengirim data melalui intent). Objek `item` yang sudah terima kita ambil data `name` dan `priority`-nya untuk dimasukkan ke dalam objek `values`.

Baris `db.insert(TABLE_TODO, null, values);` akan mengisi tabel `todo_table` dengan data yang ada di dalam objek `values`.

4.6. getAll

```
public List<Item> getAll(){
    String sql = "select * from " + TABLE_TODO;
    SQLiteDatabase db = this.getReadableDatabase();
    List<Item> storeProducts = new ArrayList<>();
    Cursor cursor = db.rawQuery(sql, null);
    if(cursor.moveToFirst()){
        do{
            int id = Integer.parseInt(cursor.getString(0));
            String name = cursor.getString(1);
            int priority = cursor.getInt(2);
            storeProducts.add(new Item(id, name, priority));
        }while (cursor.moveToNext());
    }
    cursor.close();
    return storeProducts;
}
```

Method `getAll` kita pakai untuk mengambil seluruh isi tabel `todo_table`. Objek bertipe `Cursor` dapat kita pakai untuk membaca hasil query database. Setiap selesai membaca hasil query, jangan lupa untuk memanggil `cursor.close()`.

4.7. updateItem

```
public void updateItem(Item item){
    ContentValues values = new ContentValues();
    values.put(COLUMN_TODO_NAME, item.getName());
    values.put(COLUMN_TODO_PRIORITY, item.getPriority());
    SQLiteDatabase db = this.getWritableDatabase();
    db.update(TABLE_TODO, values, COLUMN_ID    + "    = ?", new String[]
{ String.valueOf(item.getId())});
```

```
}
```

Isi dari method `updateItem` yang akan kita gunakan untuk mengubah isi database mirip dengan isi dari method `addItem`. Perbedaannya ada di baris terakhir dimana kita memanggil `db.update()`. Parameter pertama method `update` adalah nama tabel, diikuti oleh nama kolom yang akan dikenai perintah `WHERE` dan array yang berisi data nilai dari perintah `WHERE`.

4.8. deleteItem

```
public void deleteItem(int id){
    SQLiteDatabase db = this.getWritableDatabase();
    db.delete(TABLE_TODO, COLUMN_ID    + "    = ?", new String[]
  { String.valueOf(id)});
}
```

Isi dari method `deleteItem` yang akan kita gunakan untuk menghapus isi database menggunakan `db.delete()`. Parameter pertama method `delete` adalah nama tabel, diikuti oleh nama kolom yang akan dikenai perintah `WHERE` dan array yang berisi data nilai dari perintah `WHERE`.

5. Activity Tambah Data

Mari kita buat Activity baru untuk menambahkan data ke database. Beri nama Activity ini `AddEditActivity` (Activity yang sama akan kita gunakan untuk melakukan edit nanti) dan pilih template Basic Activity. Setelah loading Android Studio selesai, buka file `content_add_edit.xml` tambahkan kode berikut:

```
<?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:layout_height="match_parent"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context="com.codepolitan.belajardatabase.AddEditActivity"
    tools:showIn="@layout/activity_add_edit"
    android:orientation="vertical"
    >
```

```
    <EditText
        android:id="@+id/enter_name"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Masukkan teks..."
        android:textSize="12sp"
        android:inputType="text"
        android:layout_marginRight="8dp"
        android:layout_marginEnd="8dp"
        android:padding="12dp"/>
    <EditText
        android:id="@+id/enter_priority"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Masukkan prioritas"
        android:layout_marginTop="8dp"
        android:textSize="12sp"
        android:inputType="number"
        android:maxLines="1"
        android:layout_marginRight="8dp"
        android:layout_marginEnd="8dp"
        android:padding="12dp"/>

</LinearLayout>
```

Lalu tambahkan gambar ikon `save` dan ubah kode untuk `FloatingActionButton` di file `activity_add_edit.xml` menjadi:

```
<android.support.design.widget.FloatingActionButton
    android:id="@+id/fab"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="bottom|end"
    android:layout_margin="@dimen/fab_margin"
    android:src="@drawable/ic_save_black_24dp" />
```

Agar warna ikon menjadi putih, ubah isi file `ic_save_black_24dp.xml`:

```
<android.support.design.widget.FloatingActionButton
    android:id="@+id/fab"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="bottom|end"
```

```
    android:layout_margin="@dimen/fab_margin"
    android:src="@drawable/ic_save_black_24dp" />
```

Terakhir ubah kode `AddEditActivity` menjadi:

```
public class AddEditActivity extends AppCompatActivity {

    TextView textViewName, textViewPriority;
    MyDatabase database;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_add_edit);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

        textViewName = findViewById(R.id.enter_name);
        textViewPriority = findViewById(R.id.enter_priority);

        database = new MyDatabase(AddEditActivity.this);

        FloatingActionButton fab = (FloatingActionButton)
findViewById(R.id.fab);
        fab.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                String name = textViewName.getText().toString();
                int prio =
Integer.parseInt(textViewPriority.getText().toString());
                Item item = new Item(name, prio);
                database.addItem(item);

                Intent intent = new Intent(AddEditActivity.this,
MainActivity.class);
                startActivity(intent);
                finish();
            }
        });
        getSupportActionBar().setDisplayHomeAsUpEnabled(true);
    }

}
```

Untuk bisa menambahkan data baru menggunakan Activity ini, mari kita tambahkan baris berikut di dalam `MainActivity.java`:

```
public class MainActivity extends AppCompatActivity {

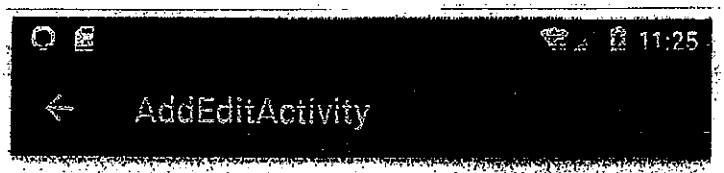
    // ...

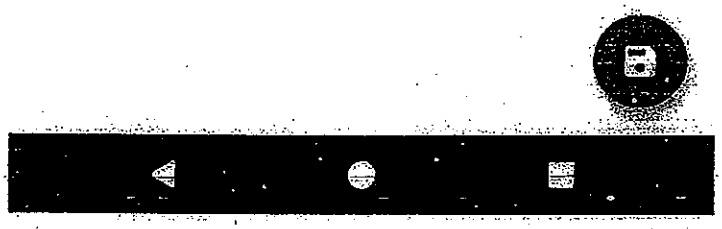
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        // ...

        FloatingActionButton fab = (FloatingActionButton)
findViewById(R.id.fab);
        fab.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent intent = new Intent(MainActivity.this,
AddEditActivity.class);
                startActivity(intent);
            }
        });

        // ...
    }
}
```

Run aplikasi ke emulator atau perangkat lalu coba klik tombol Add. Pastikan setelah tombol Add di klik kita di bawah ke `AddEditActivity` untuk menambahkan data. Isi data yang diminta dan klik tombol **Save**. `AddEditActivity` akan ditutup dan kita akan di bawa ke `MainActivity` kembali untuk menampilkan data yang telah disimpan.





Saat kembali ke `MainActivity` perhatikan bahwa kita sudah data yang kita simpan belum muncul. Hal ini karena kita belum menambahkan kode untuk menarik data dari database.

6. Membaca Data dari Database

Data yang ditampilkan oleh RecyclerView masih menggunakan data dummy. Maka sekarang, mari kita tambahkan data untuk membaca isi database.

Ubah isi method `onCreate` menjadi:

```
public class MainActivity extends AppCompatActivity {

    RecyclerView recyclerView;
    TodoListAdapter adapter;

    ArrayList<Item> todoList = new ArrayList<>();

    MyDatabase database;

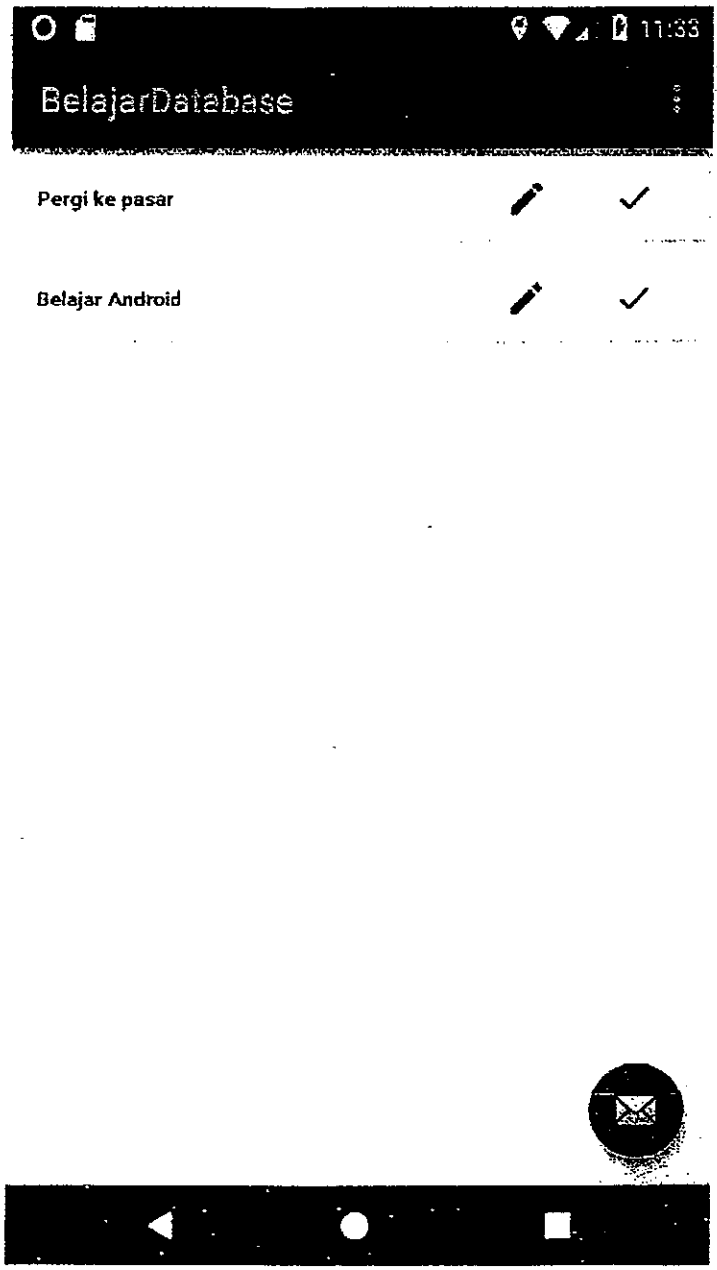
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

        FloatingActionButton fab = (FloatingActionButton)
findViewById(R.id.fab);
        fab.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent intent = new Intent(MainActivity.this,
AddEditActivity.class);
                startActivity(intent);
            }
        });

        // setDummyData();
        database = new MyDatabase(MainActivity.this);
        todoList.addAll(database.getAll());

        recyclerView = findViewById(R.id.recyclerView);
        adapter = new TodoListAdapter(todoList, MainActivity.this);
        recyclerView.setAdapter(adapter);
        recyclerView.setLayoutManager(new
LinearLayoutManager(MainActivity.this));
    }
```

Perhatikan bahwa kita membuat objek baru dari kelas `MyDatabase`, melakukan inisiasi, dan memanggil method `database.getAll()` yang akan menarik data dari database dan memasukkannya ke dalam ArrayList `todoList`. Sekarang RecyclerView kita sudah menampilkan data yang disimpan ke dalam database.



7. Update Isi Database

Setelah berhasil menambahkan data ke dalam database lalu membaca dan menampilkannya ke RecyclerView, sekarang mari kita tambahkan fitur untuk mengupdate datanya.

Kita akan memanfaatkan `AddEditActivity` yang sudah ada. Pertama, ubah dulu method `onBindViewHolder` yang ada di dalam `TodoListAdapter` menjadi:

```
@Override
public void onBindViewHolder(TodoViewHolder holder, int position) {
    final Item item = itemArrayList.get(position);
    holder.todoText.setText(item.getName());

    holder.editImageView.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Intent intent = new Intent(context, AddEditActivity.class);
            intent.putExtra("ITEM", item);
            context.startActivity(intent);
        }
    });
}
```

Kita menambahkan sebuah `onClickListener` pada ImageView yang menampilkan gambar pensil (edit) karena kita ingin membuka halaman update saat menyentuh gambar tersebut. Di dalamnya kita membuat sebuah objek Intent dan menitipkan objek item yang sebelumnya sudah kita buat. Perhatikan penambahkan `final` didepan deklarasi objek item. `Final` harus kita tambahkan karena objek item diakses dari dalam inner class.

Karena kita ingin menitipkan objek item bersama dengan intent dan mengirimkannya ke `AddEditActivity`, maka kita juga harus memperbarui kode `Item.java` agar meng-implement Parcelable (masih ingat cara menambahkan Parcelable ke kelas POJO?). sefelah Item.java ke AddEditActivity

```
public class AddEditActivity extends AppCompatActivity {

    TextView textViewName, textViewPriority;
    MyDatabase database;

    Boolean edit = false;

    Item itemIntent;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
```