

MAKALAH - MOBILE PROGRAMMING

UJIAN AKHIR SEMESTER



OLEH:

AHMAD DHANI – (1914311048)

NISA RISIKA AMALIA – (1914311061)

NUR ACHMAD BUDY KURNIAWAN – (1914311005)

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS BHAYANGKARA SURABAYA

2020

KATA PENGANTAR

Puji dan syukur kami panjatkan kehadirat Allah SWT, atas segala limpahan kekuatan serta dengan segala keterbatasan waktu, tenaga dan pikiran yang dimiliki, akhirnya kami bisa menyelesaikan makalah ini dengan baik dan tepat pada waktunya.

Kami menyadari bahwa Laporan ini belum sempurna, masih banyak kesalahan dan kekurangan. Oleh sebab itu, kami mengharapkan kritik dan saran yang membangun demi kesempurnaan laporan ini. Akhirnya, semoga segala bantuan yang telah diberikan semua pihak di atas menjadi amalan yang bermanfaat dan mendapatkan balasan dari Allah SWT dan laporan ini menjadi informasi bermanfaat bagi pembaca atau pihak lain yang membutuhkannya.

Surabaya, 10 Januari 2021

DAFTAR ISI

KATA PENGANTAR	2
DAFTAR ISI	3
BAB I : PENDAHULUAN	4
1.1 Latar Belakang	4
1.2 Rumusan Masalah	5
1.3 Tujuan Makalah	5
BAB II : IMPLEMENTASI CRUD	6
2.1 Penggunaan SQLite Database	6
2.2 Membuat Folder Awal dan Mengatur Build Gradle	7
2.3 Membuat Kelas SQLite Helper dan Fungsi CRUD	9
BAB III : USER INTERFACE	17
3.1 Membuat User Interface	17
3.2 Menambahkan Form Tambah dan Edit	19
3.3 Membuat Halaman Splash Screen	24
3.4 Membuat Halaman Register & Login	27
BAB IV : PENUTUP	34
4.1 Kesimpulan	34
DAFTAR PUSTAKA	35

BAB I

PENDAHULUAN

1.1 - Latar Belakang

Teknologi informasi saat ini telah berkembang sangat jauh dan telah merevolusi cara hidup kita, baik terhadap cara berkomunikasi, cara belajar, cara bekerja, dan lain sebagainya. Era informasi memberikan ruang lingkup yang sangat besar untuk mengorganisasikan segala kegiatan melalui cara baru, inovatif, instan, akurat, dan tepat waktu serta memberikan kenyamanan yang lebih dalam mengelola dan menikmati kehidupan.

Budaya Menggunakan Smartphone telah menjadi tren gaya hidup manusia, khususnya pada ponsel dengan sistem berbasis android. Android adalah sebuah sistem operasi mobile yang dikhususkan untuk smartphone dan tablet. Android menjadi terkenal dan menjadikannya sistem operasi mobile nomor satu di industri *mobile device* (smartphone dan tablet).

Saat ini android banyak di gunakan untuk membantu kemudahan manusia di beberapa bidang dalam kehidupan. Sebagai sarana hiburan, jejaring sosial ataupun sebagai media untuk mendapatkan dan mengolah data informasi. Kemudahan pemakaian serta portabilitas yang ditawarkan smartphone inilah yang membuatnya menjadi alat untuk mendukung aktivitas dan produktifitas beberapa kalangan. Termasuk dalam media pembelajaran.

1.2 - Rumusan Masalah

Berdasarkan latar belakang diatas, maka rumusan masalah yang akan dibahas adalah kami akan membuat program sederhana **CRUD** (*Create, Read, Update & Delete*) menggunakan Android Studio dengan memanfaatkan komponen SQLite Database.

1.3 - Tujuan Makalah

Tujuan dari laporan ini adalah mengetahui bagaimana langkah-langkah pembuatan Aplikasi Catatan Sederhana dengan SQLite sebagai penyimpanan database.

BAB II

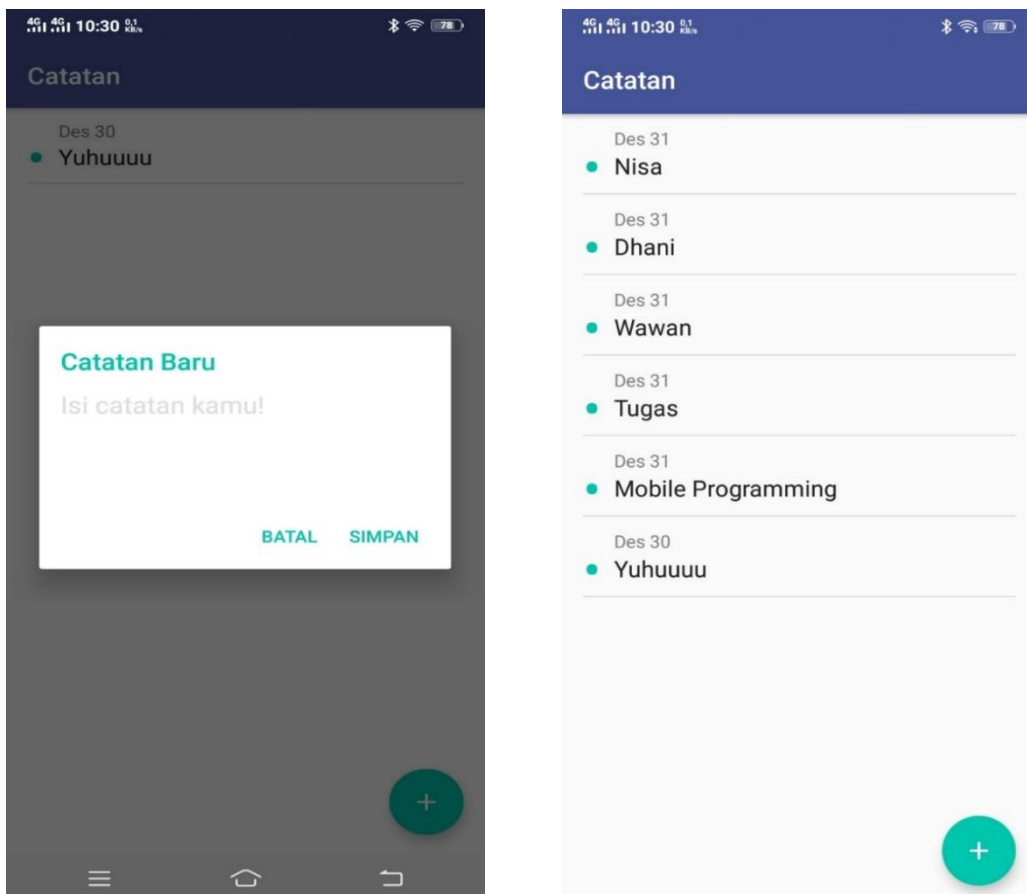
IMPLEMENTASI CRUD

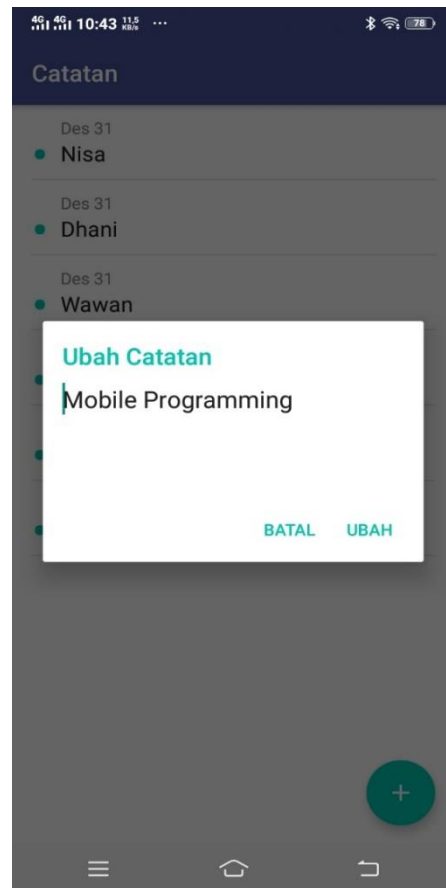
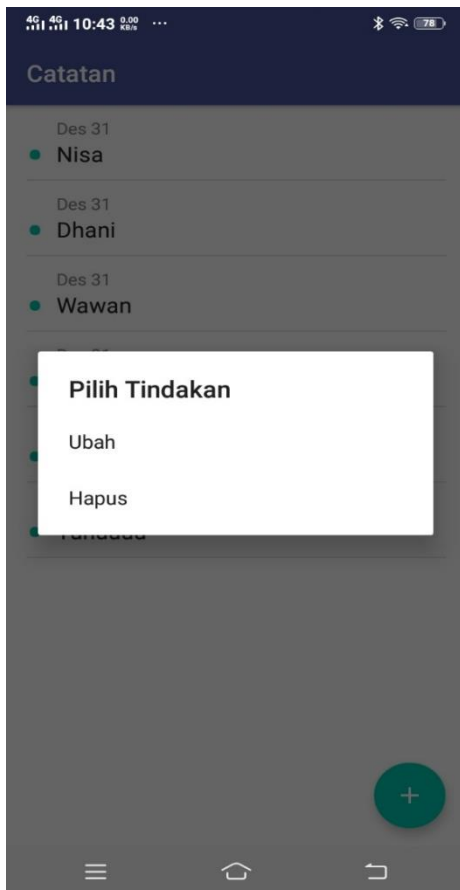
2.1 - Penggunaan SQLite Database

Di Android, ada beberapa cara untuk menyimpan data persisten. SQLite adalah salah satu cara menyimpan data aplikasi. Ini adalah database yang sangat ringan yang dilengkapi dengan OS Android.

Kami akan membuat Aplikasi Catatan sederhana dengan SQLite sebagai penyimpanan database.

Berikut ini adalah screenshot dari Aplikasi “**CatatanKu**” yang telah kami buat :





2.2 - Membuat Folder Awal & Mengatur Build Gradle

1. Buat proyek baru di Android Studio dari **File** ⇒ **New Project** dan pilih Aktivitas Dasar dari template.
2. Buka **build.gradle** pada direktori aplikasi dan tambahkan dependensi **RecyclerView** . **RecyclerView** akan digunakan untuk menampilkan Catatan.

```
dependencies {
    implementation 'com.android.support:appcompat-v7:26.1.0'
    implementation 'com.android.support.constraint:constraint-layout:1.0.2'
    implementation 'com.android.support:design:26.1.0'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'com.android.support.test:runner:1.0.1'
    androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.1'

    //Tambahkan dependencies dibawah seperti ini
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation 'com.android.support:recyclerview-v7:26.1.0'
}
```

3. Tambahkan codingan di bawah ini ke dalam file *colors.xml* , *dimens.xml* dan *strings.xml*.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="colorPrimary">#455399</color>
    <color name="colorPrimaryDark">#455399</color>
    <color name="colorAccent">#00c6ae</color>
    <color name="msg_no_notes">#999</color>
    <color name="hint_enter_note">#89c3c3c3</color>
    <color name="timestamp">#858585</color>
    <color name="note_list_text">#232323</color>
</resources>
```

[colors.xml]

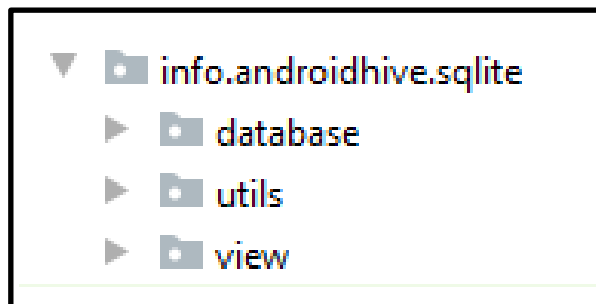
```
<resources>
    <dimen name="fab_margin">16dp</dimen>
    <dimen name="activity_margin">16dp</dimen>
    <dimen name="dot_margin_right">10dp</dimen>
    <dimen name="msg_no_notes">26sp</dimen>
    <dimen name="margin_top_no_notes">120dp</dimen>
    <dimen name="lbl_new_note_title">20sp</dimen>
    <dimen name="dimen_10">10dp</dimen>
    <dimen name="input_new_note">20sp</dimen>
    <dimen name="dot_height">30dp</dimen>
    <dimen name="dot_text_size">40sp</dimen>
    <dimen name="timestamp">14sp</dimen>
    <dimen name="note_list_text">18sp</dimen>
</resources>
```

[dimens.xml]

```
<resources>
    <string name="app_name">CatatanKu</string>
    <string name="action_settings">Settings</string>
    <string name="activity_title_home">Notes</string>
    <string name="msg_no_notes">Tidak ada catatan!</string>
    <string name="lbl_new_note_title">Catatan Baru</string>
    <string name="lbl_edit_note_title">Ubah Catatan</string>
    <string name="hint_enter_note">Isi catatan kamu!</string>
</resources>
```

[strings.xml]

4. Buat beberapa folder bernama **database / model** , **utils** , dan **view** . Di bawah ini adalah struktur dan folder yang diperlukan.



2.3 - Membuat Kelas SQLite Helper dan Fungsi CRUD

5. Pada folder **database / model** , buatlah kelas bernama **Note.java** . Di kelas ini kita akan mendefinisikan *Nama Tabel SQLite* , *Nama Kolom* dan membuat *Tabel Query SQL* bersama dengan metode-nya.

```
1 package info.androidhive.sqlite.database.model;
2
3 public class Note {
4     public static final String TABLE_NAME = "notes";
5
6     public static final String COLUMN_ID = "id";
7     public static final String COLUMN_NOTE = "note";
8     public static final String COLUMN_TIMESTAMP = "timestamp";
9
10    private int id;
11    private String note;
12    private String timestamp;
13
14    // Membuat tabel sql Query
15    public static final String CREATE_TABLE =
16        "CREATE TABLE " + TABLE_NAME + "("
17        + COLUMN_ID + " INTEGER PRIMARY KEY AUTOINCREMENT,"
18        + COLUMN_NOTE + " TEXT,"
19        + COLUMN_TIMESTAMP + " DATETIME DEFAULT CURRENT_TIMESTAMP"
20        + ")";
21
22    public Note() {
23    }
24
25    public Note(int id, String note, String timestamp) {
26        this.id = id;
27        this.note = note;
28        this.timestamp = timestamp;
```

```

29     }
30
31     public int getId() { return id; }
32
33     public String getNote() { return note; }
34
35     public void setNote(String note) { this.note = note; }
36
37     public String getTimestamp() { return timestamp; }
38
39     public void setId(int id) {
40         this.id = id;
41     }
42
43     public void setTimestamp(String timestamp) {
44         this.timestamp = timestamp;
45     }
46 }

```

Tabel `notes` membutuhkan tiga kolom yaitu `id`, `note` dan `timestamp`.

- Kolom `id` didefinisikan sebagai **Primary Key** dan **Auto Increment** yang berarti setiap catatan akan diidentifikasi secara unik oleh `id` tersebut.
- Kolom `catatan` menyimpan teks catatan yang akan kita masukkan.
- Kolom `timestamp` menyimpan tanggal dan waktu catatan dibuat.

6. Pada folder **database**, kita akan membuat kelas bernama **DatabaseHelper.java** dan membutuhkan kelas dari **SQLiteOpenHelper**. Kelas ini memegang metode terkait database untuk melakukan operasi **CRUD**.

- **onCreate()** hanya akan dipanggil sekali saat aplikasi dipasang. Pada metode ini, kita akan mengeksekusi pernyataan "Buat Tabel SQL" untuk membuat tabel yang diperlukan.
- **onUpgrade()** dipanggil saat pembaruan catatan. kita perlu mengubah **DATABASE_VERSION** untuk menjalankan metode ini.

```

31      // Membuat Tabel
32      @Override
33      public void onCreate(SQLiteDatabase db) {
34
35          // create notes table
36          db.execSQL(Note.CREATE_TABLE);
37      }
38
39      // Memperbarui Database
40      @Override
41      public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
42          // Drop older table if existed
43          db.execSQL("DROP TABLE IF EXISTS " + Note.TABLE_NAME);
44
45          // Create tables again
46          onCreate(db);
47      }

```

[DatabaseHelper.java] – onCreate & onUpgrade

Berikut ini adalah Metode pada Class **DatabaseHelper** yang akan ditambahkan :

a) Menambahkan Catatan

- **ContentValues ()** digunakan untuk menentukan nama kolom dan data yang akan disimpan. Di sini, kita hanya menetapkan nilai catatan dan mengabaikan ``id`` dan ``timestamp``. dua kolom tersebut akan dimasukkan secara otomatis.
- Setiap kali koneksi database, harus ditutup setelah selesai dengan akses database. Fungsi **db.close ()** akan menutup koneksi.
- Setelah catatan dimasukkan, ``id`` dari catatan yang baru dimasukkan akan dikembalikan.

b) Membaca Catatan

- **getNote ()** mengambil catatan ``id`` yang sudah ada dan mengambil objek catatan.
- **getAllNotes ()** mengambil semua catatan dalam urutan menurun menurut waktu.
- **getNotesCount ()** mengembalikan jumlah catatan yang disimpan dalam database.

c) Memperbarui Catatan

- Memperbarui data membutuhkan akses yang dapat ditulis . Catatan akan diperbarui dengan `id`.

d) Menghapus Catatan

- Menghapus data juga membutuhkan akses yang dapat ditulis . Metode menghapus catatan akan digunakan dengan cara menemukan `id` .

Setelah menambahkan semua metode diatas, kelas **DatabaseHelper.java** akan terisi seperti ini :

```
1 package info.androidhive.sqlite.database;
2
3 import android.content.ContentValues;
4 import android.content.Context;
5 import android.database.Cursor;
6 import android.database.sqlite.SQLiteDatabase;
7 import android.database.sqlite.SQLiteOpenHelper;
8
9 import java.util.ArrayList;
10 import java.util.List;
11
12 import info.androidhive.sqlite.database.model.Note;
13
14 public class DatabaseHelper extends SQLiteOpenHelper {
15
16     // Versi Database
17     private static final int DATABASE_VERSION = 1;
18
19     // Deklarasi Nama Database
20     private static final String DATABASE_NAME = "notes_db";
21
22
23     public DatabaseHelper(Context context) {
24         super(context, DATABASE_NAME, null, DATABASE_VERSION);
25     }
26
27     // Membuat Tabel
28     @Override
29     public void onCreate(SQLiteDatabase db) {
30
31         // create notes table
32         db.execSQL(Note.CREATE_TABLE);
33     }
34
35     // Memperbarui Database
36     @Override
37     public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
38         // Drop older table if existed
39         db.execSQL("DROP TABLE IF EXISTS " + Note.TABLE_NAME);
40
41         // Create tables again
42         onCreate(db);
43     }
44 }
```

```

45 public long insertNote(String note) {
46     SQLiteDatabase db = this.getWritableDatabase();
47
48     ContentValues values = new ContentValues();
49     // `id` dan `timestamp` akan masuk secara otomatis
50     // tidak perlu untuk menambahkannya
51     values.put(Note.COLUMN_NOTE, note);
52
53     // Masukkan baris
54     long id = db.insert(Note.TABLE_NAME, nullColumnHack: null, values);
55
56     // Menutup koneksi database
57     db.close();
58
59     // Mengembalikan baris ID yang baru dimasukkan
60     return id;
61 }
62
63 public Note getNote(long id) {
64
65     SQLiteDatabase db = this.getReadableDatabase();
66
67     Cursor cursor = db.query(Note.TABLE_NAME,
68         new String[]{Note.COLUMN_ID, Note.COLUMN_NOTE, Note.COLUMN_TIMESTAMP},
69         selection: Note.COLUMN_ID + "=?",
70         new String[]{String.valueOf(id)},
71         null,
72         null,
73         null,
74         null,
75         null,
76         null);
77
78     if (cursor != null)
79         cursor.moveToFirst();
80
81     Note note = new Note(
82         cursor.getInt(cursor.getColumnIndex(Note.COLUMN_ID)),
83         cursor.getString(cursor.getColumnIndex(Note.COLUMN_NOTE)),
84         cursor.getString(cursor.getColumnIndex(Note.COLUMN_TIMESTAMP)));
85     cursor.close();
86
87     return note;
88 }
89
90 public List<Note> getAllNotes() {
91     List<Note> notes = new ArrayList<>();
92
93     // Memilih semua Query
94     String selectQuery = "SELECT * FROM " + Note.TABLE_NAME + " ORDER BY " +
95         Note.COLUMN_TIMESTAMP + " DESC";
96
97     SQLiteDatabase db = this.getWritableDatabase();
98     Cursor cursor = db.rawQuery(selectQuery, selectionArgs: null);
99
100     if (cursor.moveToFirst()) {
101         do {
102             Note note = new Note();
103             note.setId(cursor.getInt(cursor.getColumnIndex(Note.COLUMN_ID)));
104             note.setNote(cursor.getString(cursor.getColumnIndex(Note.COLUMN_NOTE)));
105             note.setTimestamp(cursor.getString(cursor.getColumnIndex(Note.COLUMN_TIMESTAMP)));
106
107             notes.add(note);
108         } while (cursor.moveToNext());
109     }
110
111     db.close();
112     return notes;
113 }

```

```

109     public int getNotesCount() {
110         String countQuery = "SELECT * FROM " + Note.TABLE_NAME;
111         SQLiteDatabase db = this.getReadableDatabase();
112         Cursor cursor = db.rawQuery(countQuery, selectionArgs: null);
113
114         int count = cursor.getCount();
115         cursor.close();
116
117
118         return count;
119     }
120
121     @ public int updateNote(Note note) {
122         SQLiteDatabase db = this.getWritableDatabase();
123
124         ContentValues values = new ContentValues();
125         values.put(Note.COLUMN_NOTE, note.getNote());
126
127         // Update Baris
128         return db.update(Note.TABLE_NAME, values, whereClause: Note.COLUMN_ID + " = ?",
129             new String[]{String.valueOf(note.getId())});
130     }
131
132     @ public void deleteNote(Note note) {
133         // Menghapus Catatan
134         SQLiteDatabase db = this.getWritableDatabase();
135         db.delete(Note.TABLE_NAME, whereClause: Note.COLUMN_ID + " = ?",
136             new String[]{String.valueOf(note.getId())});
137         db.close();
138     }
139 }
140
141 //Selesai (line terakhir)

```

[DatabaseHelper.java] – Full Source

7. Di bawah paket **utils** , buat dua kelas

bernama **RecyclerTouchListener.java** dan **MyDividerItemDecoration.java**

- Kelas **RecyclerTouchListener** berfungsi untuk menambahkan Aksi Touch ke baris **RecyclerView**.

```

1 package info.androidhive.sqlite.utils;
2
3 import android.content.Context;
4 import android.support.v7.widget.RecyclerView;
5 import android.view.GestureDetector;
6 import android.view.MotionEvent;
7 import android.view.View;
8
9 public class RecyclerTouchListener implements RecyclerView.OnItemTouchListener {
10
11     private ClickListener clicklistener;
12     private GestureDetector gestureDetector;
13
14     public RecyclerTouchListener(Context context, final RecyclerView recycleView, final ClickListener clicklistener) {
15
16         this.clicklistener = clicklistener;
17         gestureDetector = new GestureDetector(context, new GestureDetector.SimpleOnGestureListener() {
18             @Override
19             public boolean onSingleTapUp(MotionEvent e) { return true; }
20
21             @Override
22             public void onLongPress(MotionEvent e) {
23                 View child = recycleView.findChildViewUnder(e.getX(), e.getY());
24                 if (child != null && clicklistener != null) {
25                     clicklistener.onLongClick(child, recycleView.getChildAdapterPosition(child));
26                 }
27             }
28         });
29     }
30 }
31

```

```

33     @Override
34     public boolean onInterceptTouchEvent(RecyclerView rv, MotionEvent e) {
35         View child = rv.findChildViewUnder(e.getX(), e.getY());
36         if (child != null && clicklistener != null && gestureDetector.onTouchEvent(e)) {
37             clicklistener.onClick(child, rv.getChildAdapterPosition(child));
38         }
39
40         return false;
41     }
42
43     @Override
44     public void onTouchEvent(RecyclerView rv, MotionEvent e) {
45
46     }
47
48     @Override
49     public void onRequestDisallowInterceptTouchEvent(boolean disallowIntercept) {
50
51     }
52
53     public interface Clicklistener {
54         void onClick(View view, int position);
55
56         void onLongClick(View view, int position);
57     }
58 }

```

- Kelas **MyDividerItemDecoration** berfungsi untuk menambahkan garis pemisah antar baris catatan.

```

1  package info.androidhive.sqlite.utils;
2
3  import android.content.Context;
4  import android.content.res.Resources;
5  import android.content.res.TypedArray;
6  import android.graphics.Canvas;
7  import android.graphics.Rect;
8  import android.graphics.drawable.Drawable;
9  import android.support.v7.widget.LinearLayoutManager;
10 import android.support.v7.widget.RecyclerView;
11 import android.util.TypedValue;
12 import android.view.View;
13
14
15 public class MyDividerItemDecoration extends RecyclerView.ItemDecoration {
16
17     private static final int[] ATTRS = new int[]{
18         android.R.attr.listDivider
19     };
20
21     public static final int HORIZONTAL_LIST = LinearLayoutManager.HORIZONTAL;
22     public static final int VERTICAL_LIST = LinearLayoutManager.VERTICAL;
23
24     private Drawable mDivider;
25     private int mOrientation;
26     private Context context;
27     private int margin;

```

```

29 @ public MyDividerItemDecoration(Context context, int orientation, int margin) {
30     this.context = context;
31     this.margin = margin;
32     final TypedArray a = context.obtainStyledAttributes(ATTRS);
33     mDivider = a.getDrawable(index: 0);
34     a.recycle();
35     setOrientation(orientation);
36 }
37
38 public void setOrientation(int orientation) {
39     if (orientation != HORIZONTAL_LIST && orientation != VERTICAL_LIST) {
40         throw new IllegalArgumentException("invalid orientation");
41     }
42     mOrientation = orientation;
43 }
44
45 @Override
46 public void onDrawOver(Canvas c, RecyclerView parent, RecyclerView.State state) {
47     if (mOrientation == VERTICAL_LIST) {
48         drawVertical(c, parent);
49     } else {
50         drawHorizontal(c, parent);
51     }
52 }
53
54 @ public void drawVertical(Canvas c, RecyclerView parent) {
55     final int left = parent.getPaddingLeft();
56     final int right = parent.getWidth() - parent.getPaddingRight();
57
58     final int childCount = parent.getChildCount();
59     for (int i = 0; i < childCount; i++) {
60         final View child = parent.getChildAt(i);
61         final RecyclerView.LayoutParams params = (RecyclerView.LayoutParams) child
62             .getLayoutParams();
63         final int top = child.getBottom() + params.bottomMargin;
64         final int bottom = top + mDivider.getIntrinsicHeight();
65         mDivider.setBounds(left: left + dpToPx(margin), top, right: right - dpToPx(margin), bottom);
66         mDivider.draw(c);
67     }
68 }
69
70 @ public void drawHorizontal(Canvas c, RecyclerView parent) {
71     final int top = parent.getPaddingTop();
72     final int bottom = parent.getHeight() - parent.getPaddingBottom();
73
74     final int childCount = parent.getChildCount();
75     for (int i = 0; i < childCount; i++) {
76         final View child = parent.getChildAt(i);
77         final RecyclerView.LayoutParams params = (RecyclerView.LayoutParams) child
78             .getLayoutParams();
79         final int left = child.getRight() + params.rightMargin;
80         final int right = left + mDivider.getIntrinsicHeight();
81         mDivider.setBounds(left, top: top + dpToPx(margin), right, bottom: bottom - dpToPx(margin));
82         mDivider.draw(c);
83     }
84 }
85
86 @Override
87 public void getItemOffsets(Rect outRect, View view, RecyclerView parent, RecyclerView.State state) {
88     if (mOrientation == VERTICAL_LIST) {
89         outRect.set(left: 0, top: 0, right: 0, mDivider.getIntrinsicHeight());
90     } else {
91         outRect.set(left: 0, top: 0, mDivider.getIntrinsicWidth(), bottom: 0);
92     }
93 }
94
95 private int dpToPx(int dp) {
96     Resources r = context.getResources();
97     return Math.round(TypedValue.applyDimension(TypedValue.COMPLEX_UNIT_DIP, dp, r.getDisplayMetrics()));
98 }
99
100 }

```


BAB III

USER INTERFACE

3.1 - Membuat User Interface

8. Buat tata letak xml baru bernama **note_list_row.xml** . Tata letak ini berfungsi untuk membuat desain item catatan tunggal dalam daftar.

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      android:layout_width="match_parent"
4      android:layout_height="wrap_content"
5      android:clickable="true"
6      android:foreground="?attr/selectableItemBackground"
7      android:paddingBottom="10dp"
8      android:paddingLeft="16dp"
9      android:paddingRight="16dp"
10     android:paddingTop="10dp">
11
12     <TextView
13         android:id="@+id/dot"
14         android:layout_width="wrap_content"
15         android:layout_height="30dp"
16         android:layout_marginRight="10dp"
17         android:layout_marginTop="10dp"
18         android:includeFontPadding="false"
19         android:textColor="@color/colorAccent"
20         android:lineSpacingExtra="0dp"
21         android:textSize="40sp" />
22
23     <TextView
24         android:id="@+id/timestamp"
25         android:layout_width="match_parent"
26         android:layout_height="wrap_content"
27         android:layout_toRightOf="@id/dot"
28         android:textColor="@color/timestamp"
29         android:textSize="14sp" />
30
31     <TextView
32         android:id="@+id/note"
33         android:layout_width="match_parent"
34         android:layout_height="wrap_content"
35         android:layout_below="@id/timestamp"
36         android:layout_toRightOf="@id/dot"
37         android:textColor="@color/note_list_text"
38         android:textSize="18sp" />
39
40 </RelativeLayout>
```

9. Di bawah view package, buat kelas bernama **NotesAdapter.java** . Kelas adapter ini akan merender **RecyclerView** dengan tata letak dan kumpulan data yang ditentukan.

```
1 package info.androidhive.sqlite.view;
2
3 import android.content.Context;
4 import android.support.v7.widget.RecyclerView;
5 import android.text.Html;
6 import android.util.Log;
7 import android.view.LayoutInflater;
8 import android.view.View;
9 import android.view.ViewGroup;
10 import android.widget.TextView;
11
12 import java.text.ParseException;
13 import java.text.SimpleDateFormat;
14 import java.util.Date;
15 import java.util.List;
16
17 import info.androidhive.sqlite.R;
18 import info.androidhive.sqlite.database.model.Note;
19
20 public class NotesAdapter extends RecyclerView.Adapter<NotesAdapter.MyViewHolder> {
21
22     private Context context;
23     private List<Note> notesList;
24
25     public class MyViewHolder extends RecyclerView.ViewHolder {
26         public TextView note;
27         public TextView dot;
28         public TextView timestamp;
29
30         public MyViewHolder(View view) {
31             super(view);
32             note = view.findViewById(R.id.note);
33             dot = view.findViewById(R.id.dot);
34             timestamp = view.findViewById(R.id.timestamp);
35         }
36     }
37
38     public NotesAdapter(Context context, List<Note> notesList) {
39         this.context = context;
40         this.notesList = notesList;
41     }
42
43     @Override
44     public MyViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
45         View itemView = LayoutInflater.from(parent.getContext())
46             .inflate(R.layout.note_list_row, parent, attachToRoot: false);
47
48         return new MyViewHolder(itemView);
49     }
50 }
```

```

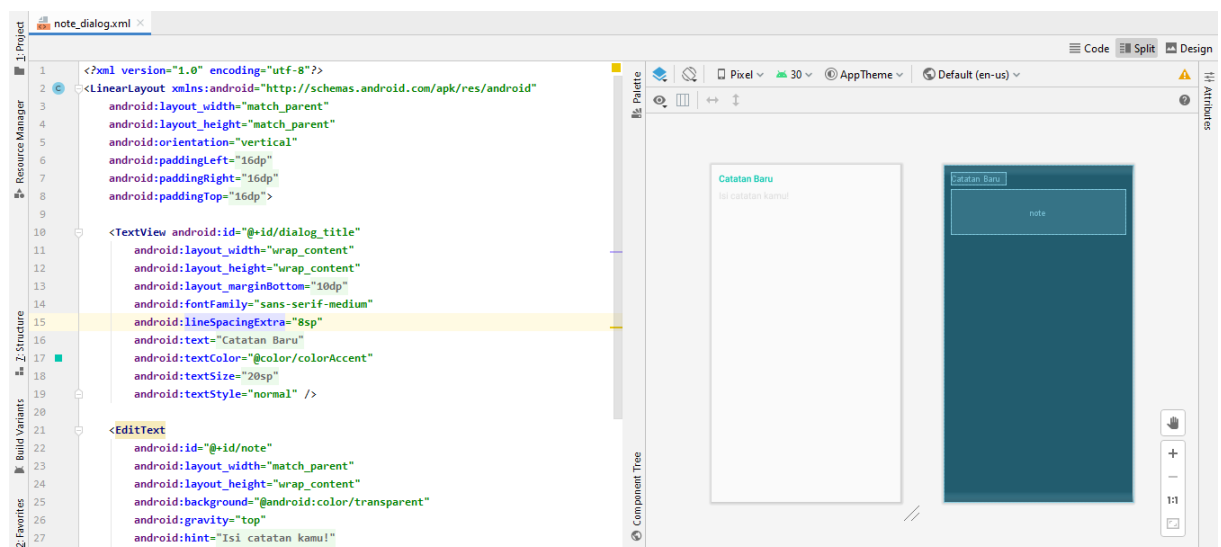
52         @Override
53         public void onBindViewHolder(MyViewHolder holder, int position) {
54             Note note = notesList.get(position);
55
56             holder.note.setText(note.getNote());
57             holder.dot.setText(Html.fromHtml( source: "&#8226;"));
58
59             // Menampilkan timestamp
60             holder.timestamp.setText(formatDate(note.getTimestamp()));
61         }
62
63         @Override
64         public int getItemCount() { return notesList.size(); }
65
66         /**
67          * Formatting timestamp to `MMM d` format
68          */
69         private String formatDate(String dateStr) {
70             try {
71                 SimpleDateFormat fmt = new SimpleDateFormat( pattern: "yyyy-MM-dd HH:mm:ss");
72                 Date date = fmt.parse(dateStr);
73                 SimpleDateFormat fmtOut = new SimpleDateFormat( pattern: "MMM d");
74                 return fmtOut.format(date);
75             } catch (ParseException e) {
76
77             }
78
79             return "";
80         }
81     }
82 }
83
84

```

3.2 - Menambahkan Form Tambah dan Edit

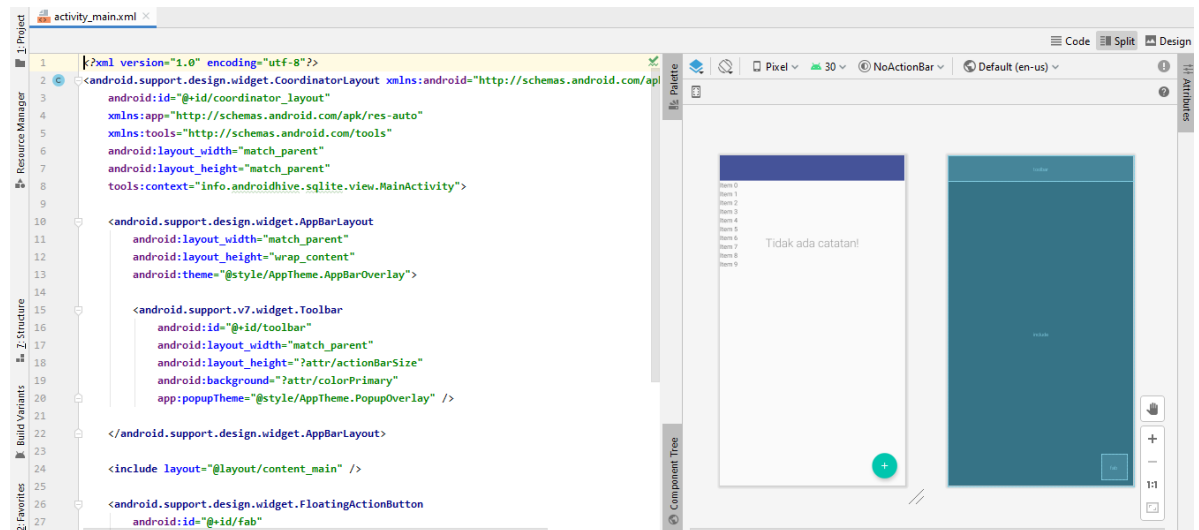
Jika kita mengamati desain aplikasi, sebuah catatan biasanya dibuat atau diperbarui menggunakan Dialog. Jadi, kita perlu membuat tata letak khusus dengan input **EditText** dan membuat di **AlertDialog** .

10. Buat tata letak lain bernama **note_dialog.xml**

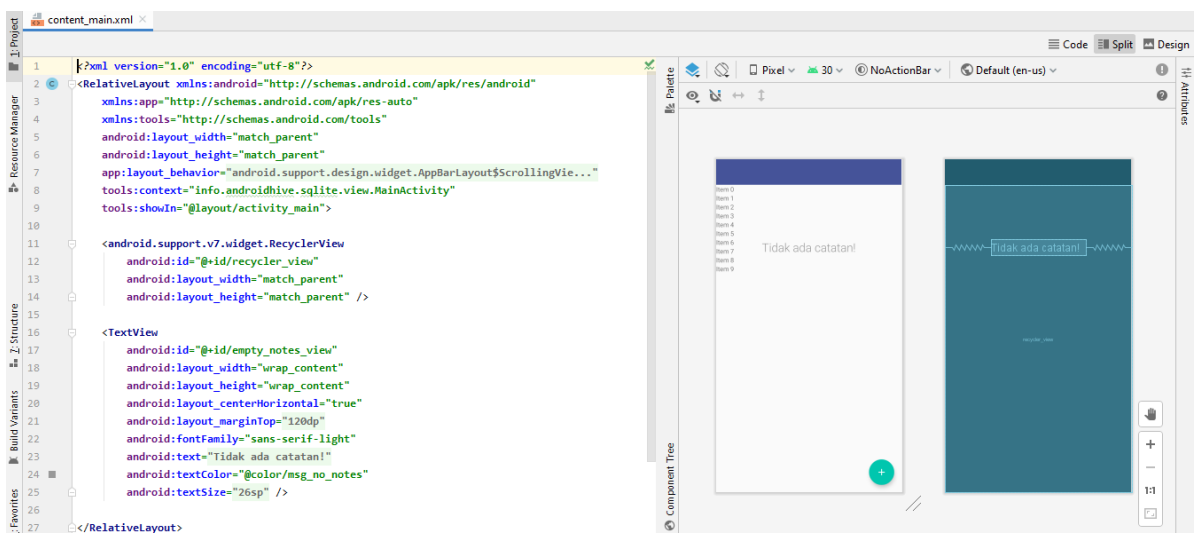


11. Buka file tata letak aktivitas utama (**activity_main.xml** dan **content_main.xml**) dan tambahkan widget **RecyclerView** . Kita juga dapat mengubah ikon **FAB** (*Floating Action Button*) di sini.

[activity_main.xml]



[content_main.xml]



12. Terakhir buka **MainActivity.java** dan lakukan perubahan di bawah ini :

- **showNoteDialog ()** buka dialog untuk membuat catatan baru. Dialog ini akan ditampilkan dengan cara mengklik **FAB** (*Floating Action Button*) .
- **createNote ()** membuat catatan baru dalam database dan menambahkan catatan yang baru disisipkan dalam daftar **RecyclerView**.
- **showActionsDialog ()** menampilkan dialog dengan opsi Edit dan Hapus . Dialog ini dapat dijalankan dengan menekan lama catatan dalam daftar..
- **deleteNote ()** menghapus catatan dari database. Catatan yang dihapus kembali dihapus dari daftar dengan memanggil **notifyItemRemoved ()** pada adapter.
- **toggleEmptyNotes ()** mengubah visibilitas catatan dan tampilan catatan kosong bergantung pada jumlah (**db.getNotesCount () > 0**) catatan..

[**MainActivity.java**]

```
88      /** Menambahkan catatan baru kedalam database ...*/
92      private void createNote(String note) {
93          // inserting note in db and getting
94          // newly inserted note id
95          long id = db.insertNote(note);
96
97          // get the newly inserted note from db
98          Note n = db.getNote(id);
99
100         if (n != null) {
101             // adding new note to array list at 0 position
102             notesList.add(0, n);
103
104             // refreshing the list
105             mAdapter.notifyDataSetChanged();
106
107             toggleEmptyNotes();
108         }
109     }
```

Menambahkan Catatan Baru kedalam Database

```

111  /**
112   * Memperbarui catatan di dalam databasedan mengatur
113   * posisi catatan berdasarkan urutan
114   */
115  private void updateNote(String note, int position) {
116      Note n = notesList.get(position);
117      // updating note text
118      n.setNote(note);
119
120      // updating note in db
121      db.updateNote(n);
122
123      // refreshing the list
124      notesList.set(position, n);
125      mAdapter.notifyDataSetChanged(position);
126
127      toggleEmptyNotes();
128  }

```

Memperbarui Catatan didalam Database

```

130  /**
131   * Menghapus catatan dari SQLite
132   */
133  private void deleteNote(int position) {
134      // Menghapus catatan dari database
135      db.deleteNote(notesList.get(position));
136
137      // Menghapus catatan dari daftar catatan
138      notesList.remove(position);
139      mAdapter.notifyItemRemoved(position);
140
141      toggleEmptyNotes();
142  }
143

```

Menghapus Catatan dari SQLite

```

145  /**
146   * Membuka form pilihan dengan indikator ubah dan hapus
147   */
148  private void showActionsDialog(final int position) {
149      CharSequence colors[] = new CharSequence[]{"Ubah", "Hapus"};
150
151      AlertDialog.Builder builder = new AlertDialog.Builder( context: this);
152      builder.setTitle("Pilih Tindakan");
153      builder.setItems(colors, new DialogInterface.OnClickListener() {
154          @Override
155          public void onClick(DialogInterface dialog, int which) {
156              if (which == 0) {
157                  showNoteDialog( shouldUpdate: true, notesList.get(position), position);
158              } else {
159                  deleteNote(position);
160              }
161          }
162      });
163      builder.show();
164  }

```

Membuka Form Pilihan dengan Indikator 'Ubah' dan 'Hapus'

```

166 // Membuka dialog seluruh catatan
167 private void showNoteDialog(final boolean shouldUpdate, final Note note, final int position) {
168     LayoutInflater inflaterAndroid = LayoutInflater.from(getApplicationContext());
169     View view = inflaterAndroid.inflate(R.layout.note_dialog, root: null);
170
171     AlertDialog.Builder alertDialogBuilderUserInput = new AlertDialog.Builder(context: MainActivity.this);
172     alertDialogBuilderUserInput.setView(view);
173
174     final EditText inputNote = view.findViewById(R.id.note);
175     TextView dialogTitle = view.findViewById(R.id.dialog_title);
176     dialogTitle.setText(!shouldUpdate ? getString(R.string.lbl_new_note_title) : getString(R.string.lbl_edit_note_title));
177
178     if (shouldUpdate && note != null) {
179         inputNote.setText(note.getNote());
180     }
181     alertDialogBuilderUserInput
182         .setCancelable(false)
183         .setPositiveButton(shouldUpdate ? "Ubah" : "Simpan", new DialogInterface.OnClickListener() {
184             public void onClick(DialogInterface dialogBox, int id) {
185
186             }
187         })
188         .setNegativeButton(text: "Batal",
189             new DialogInterface.OnClickListener() {
190             public void onClick(DialogInterface dialogBox, int id) {

```

Membuka Dialog Seluruh Catatan

```

221 /**
222  * Beralih ke tampilan list kosong jika tidak ada Catatan yang dibuat
223  */
224 private void toggleEmptyNotes() {
225
226     if (db.getNotesCount() > 0) {
227         noNotesView.setVisibility(View.GONE);
228     } else {
229         noNotesView.setVisibility(View.VISIBLE);
230     }
231 }
232 }

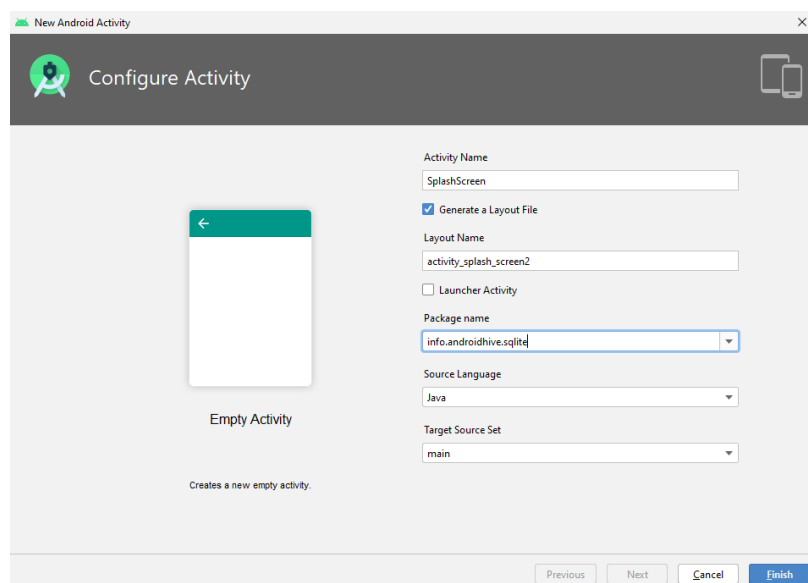
```

Beralih ke Tampilan List Kosong jika Tidak ada Catatan

3.3 - Membuat Halaman Splash Screen

Pertama, kita akan membuat layout baru yang akan diisi text atau image yang akan menjadi tampilan di layout Splash Screen.

Klik kanan pada [**App** → **New** → **Activity** → **Empty Activity**]



- Isi Activity Name dengan tulisan **SplashScreen** seperti pada gambar diatas

Pada *activity_splash_screen.xml* kita isikan dengan kode berikut :

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <RelativeLayout
3     xmlns:android="http://schemas.android.com/apk/res/android"
4     xmlns:app="http://schemas.android.com/apk/res-auto"
5     xmlns:tools="http://schemas.android.com/tools"
6     android:layout_width="match_parent"
7     android:layout_height="match_parent"
8     tools:context=".view.splashScreen"
9     android:id="@+id/splash_screen">
10
11     <ImageView
12         android:layout_width="231dp"
13         android:layout_height="235dp"
14         android:layout_alignParentStart="true"
15         android:layout_alignParentLeft="true"
16         android:layout_alignParentEnd="true"
17         android:layout_alignParentRight="true"
18         android:layout_marginStart="91dp"
19         android:layout_marginLeft="91dp"
20         android:layout_marginEnd="90dp"
21         android:layout_marginRight="90dp"
22         android:layout_marginTop="100dp"
23         android:backgroundTint="#FFFFFF"
24         android:scaleType="centerCrop"
25         android:src="@drawable/logoapps" />
```




- Jangan lupa untuk memasukkan gambar yang akan ditampilkan di layout splash screen dalam folder **drawable** dan beri nama “**logo**”.

Pada *SplashScreen.java* isi dengan kode berikut ini :

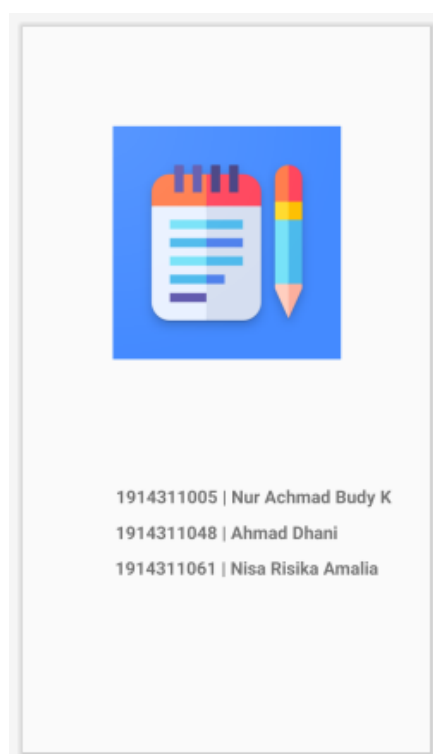


Edit pada *AndroidManifest.xml* rubah menjadi seperti berikut :

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3     package="info.androidhive.sqlite">
4
5     <application
6         android:allowBackup="true"
7         android:icon="@mipmap/ic_launcher"
8         android:label="CatatanKu"
9         android:roundIcon="@mipmap/ic_launcher_round"
10        android:supportRtl="true"
11        android:theme="@style/AppTheme">
12        <activity android:name=".view.RegisActivity"
13            android:parentActivityName=".view.LoginActivity"></activity>
14        <activity android:name=".view.LoginActivity" />
15        <activity android:name=".view.splashScreen">
16            <intent-filter>
17                <action android:name="android.intent.action.MAIN" />
18
19                <category android:name="android.intent.category.LAUNCHER" />
20            </intent-filter>
21        </activity>
22        <activity
23            android:name=".view.MainActivity"
24            android:label="CatatanKu"
25            android:theme="@style/AppTheme.NoActionBar"></activity>
26    </application>
27
```

Selesai.

Berikut adalah tampilan dari Splash Screen yang telah dibuat :

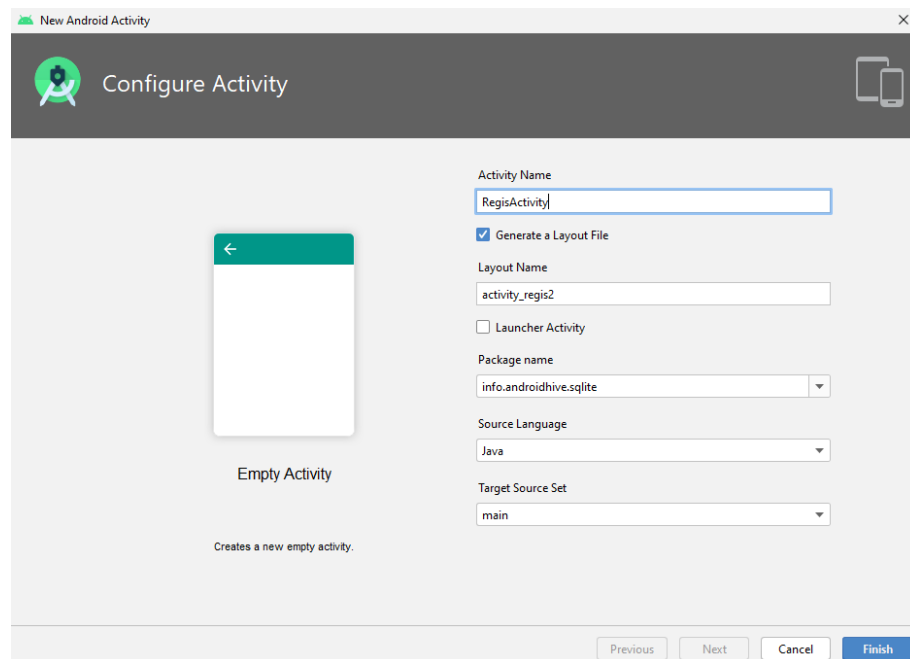


3.4 - Membuat Halaman Register dan Login

A. Halaman Register

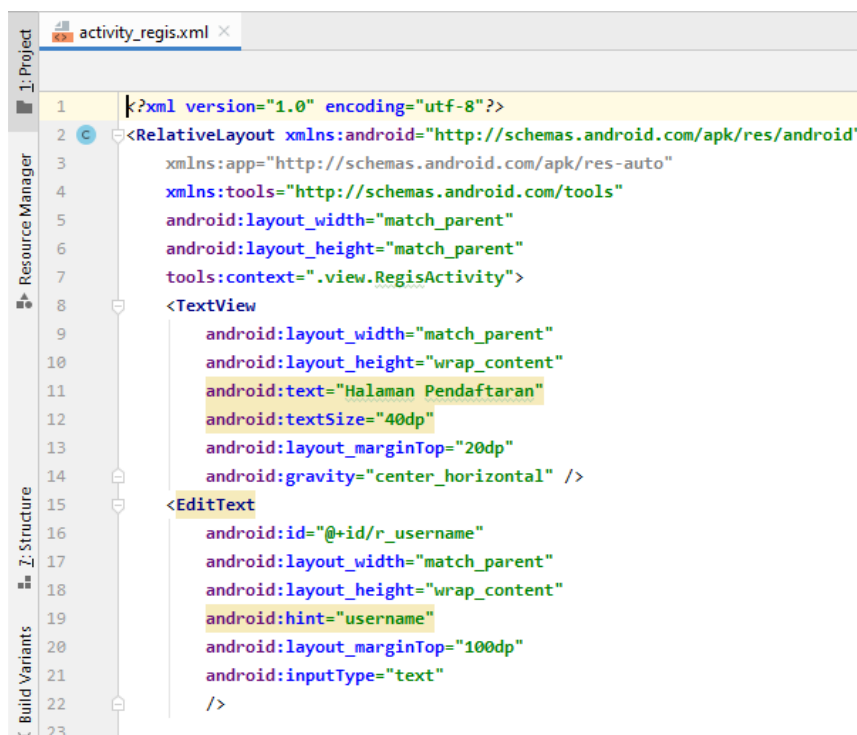
Langkah pertama, kita akan membuat Activity Baru bernama RegisActivity

Klik kanan pada [**App** -> **New** -> **Activity** -> **Empty Activity**]



- Isi Activity Name dengan tulisan **RegisActivity** seperti pada gambar diatas

Pada *activity_regis.xml*, kita isikan kode seperti berikut :



```

24 <EditText
25     android:id="@+id/r_password"
26     android:layout_width="match_parent"
27     android:layout_height="wrap_content"
28     android:layout_below="@+id/r_username"
29     android:layout_marginTop="4dp"
30     android:hint="password"
31     android:inputType="textPassword"/>
32
33 <Button
34     android:id="@+id/btn_regis"
35     android:layout_width="wrap_content"
36     android:layout_height="wrap_content"
37     android:text="Daftar"
38     android:layout_below="@id/r_password"
39 />
40
41 </RelativeLayout>

```

Kemudian, pada bagian *RegisActivity.java* kita isikan kode seperti berikut :


```

1 package info.androidhive.sqlite.view;
2
3 import android.support.v7.app.AppCompatActivity;
4 import android.os.Bundle;
5 import android.view.View;
6 import android.widget.TextView;
7 import android.widget.Toast;
8
9 import info.androidhive.sqlite.R;
10 import info.androidhive.sqlite.database.DatabaseHelper;
11 import info.androidhive.sqlite.database.model.Note;
12 import info.androidhive.sqlite.database.model.NoteUser;
13
14 public class RegisActivity extends AppCompatActivity {
15     public TextView user,pw,btnRegis;
16     private DatabaseHelper db;
17     @Override
18     protected void onCreate(Bundle savedInstanceState) {
19         super.onCreate(savedInstanceState);
20         setContentView(R.layout.activity_regis);
21         user = findViewById(R.id.r_username);
22         pw = findViewById(R.id.r_password);
23         btnRegis = findViewById(R.id.btn_regis);
24
25         btnRegis.setOnClickListener((view) -> {
26             String x =user.getText().toString();
27             String y =pw.getText().toString();
28             buatUser(x,y);
29         });
30
31     }
32
33     public void buatUser(String user, String pw){
34         db = new DatabaseHelper( context: this);
35
36         NoteUser n = db.getUser(user);
37         if (n == null) {
38             long userDb = db.insertUser(user,pw);
39             Toast.makeText( context: RegisActivity.this, text: user+" Berhasil di Registrasi", Toast.LENGTH_SHORT).show();
40         }else{
41             Toast.makeText( context: this, text: "Username Udah Ada", Toast.LENGTH_SHORT).show();
42         }
43     }
44 }

```

Selanjutnya, buat Class baru pada folder **database/model**, bernama *NoteUser.java*

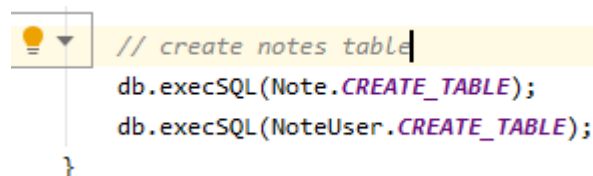
Isi kodenya seperti berikut ini :



```
1 package info.androidhive.sqlite.database.model;
2
3 public class NoteUser {
4     public static final String TABLE_NAME = "user_table";
5
6
7     public static final String COLUMN_USER = "user";
8     public static final String COLUMN_PW = "password";
9
10    private String user;
11    private String password;
12
13    // Membuat tabel pada sql Query
14    public static final String CREATE_TABLE =
15        "CREATE TABLE " + TABLE_NAME + "("
16        + COLUMN_USER + " TEXT PRIMARY KEY,"
17        + COLUMN_PW + " TEXT)";
18
19
20    public NoteUser(String user, String password) {
21
22        this.user = user;
23        this.password = password;
24    }
25
26
27    public String getUser() { return user; }
28
29
30
31    public void setUser(String user) { this.user = user; }
32
33
34
35    public String getPassword() { return password; }
36
37
38
39    public void setPassword(String password) { this.password = password; }
40
41
42 }
```

Class **NoteUser** ini berfungsi untuk pembuatan Nama Tabel dan Kolom user.

Terakhir, kita tambahkan beberapa fungsi di baris kode pada Class **DatabaseHelper**



```
// create notes table
db.execSQL(Note.CREATE_TABLE);
db.execSQL(NoteUser.CREATE_TABLE);
}
```

- Pembuatan Tabel pada Catatan

```

52 public long insertUser(String user, String pw){
53     SQLiteDatabase db = this.getWritableDatabase();
54
55     ContentValues values = new ContentValues();
56
57     values.put(NoteUser.COLUMN_USER,user);
58     values.put(NoteUser.COLUMN_PW,pw);
59
60     long users = db.insert(NoteUser.TABLE_NAME, nullColumnHack: null,values);
61
62     db.close();
63
64     return users;
65 }

```

- Fungsi **insertUser** untuk menambahkan kolom username dan pw saat registrasi

```

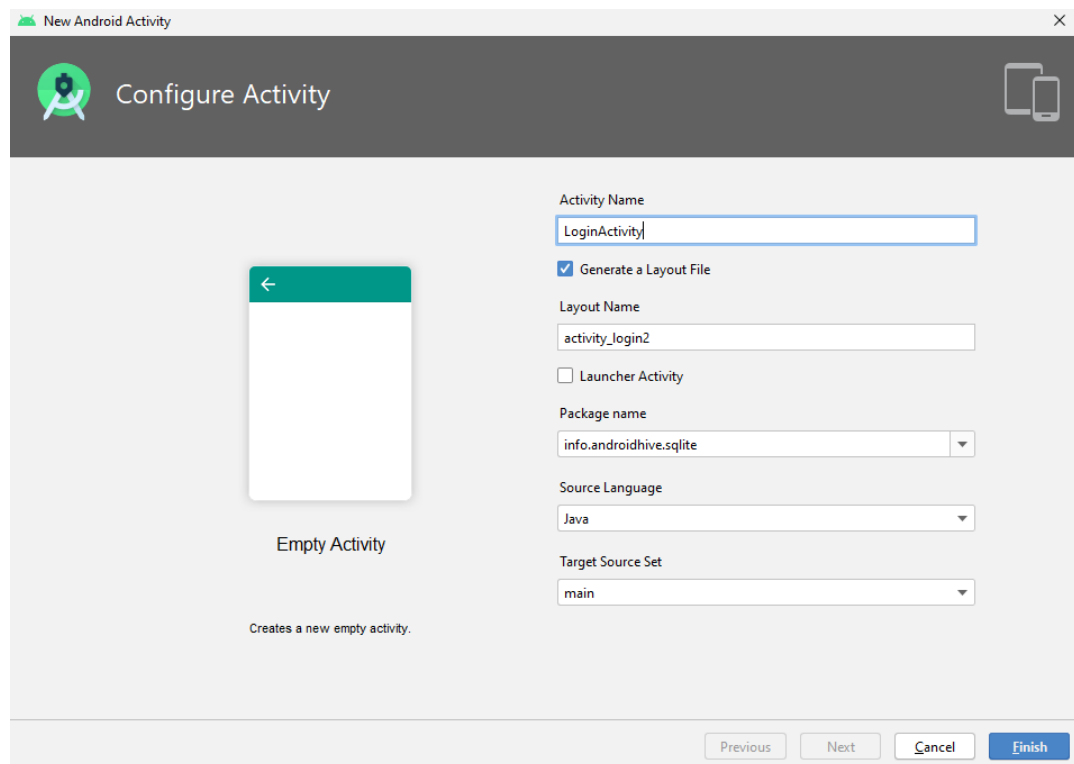
85 public NoteUser getUser(String user){
86     SQLiteDatabase db = this.getReadableDatabase();
87     Cursor cursor = db.query(NoteUser.TABLE_NAME,
88         new String[]{NoteUser.COLUMN_USER,NoteUser.COLUMN_PW},
89         selection: NoteUser.COLUMN_USER + "=?",
90         new String[]{user},
91         null,
92         null,
93         null,
94         null,
95         null,
96         null,
97         null,
98         null,
99         null);
100     if ( cursor != null){
101         cursor.moveToFirst();
102         if(cursor.getCount()>0){
103             NoteUser note = new NoteUser(
104                 cursor.getString(cursor.getColumnIndex(NoteUser.COLUMN_USER)),
105                 cursor.getString(cursor.getColumnIndex(NoteUser.COLUMN_PW)));
106             cursor.close();
107             return note;
108         }
109     }
110     return null;
111 }

```

- Fungsi **getUser** untuk mendapatkan username dan password saat user input dan memasukkannya ke Database SQLite.

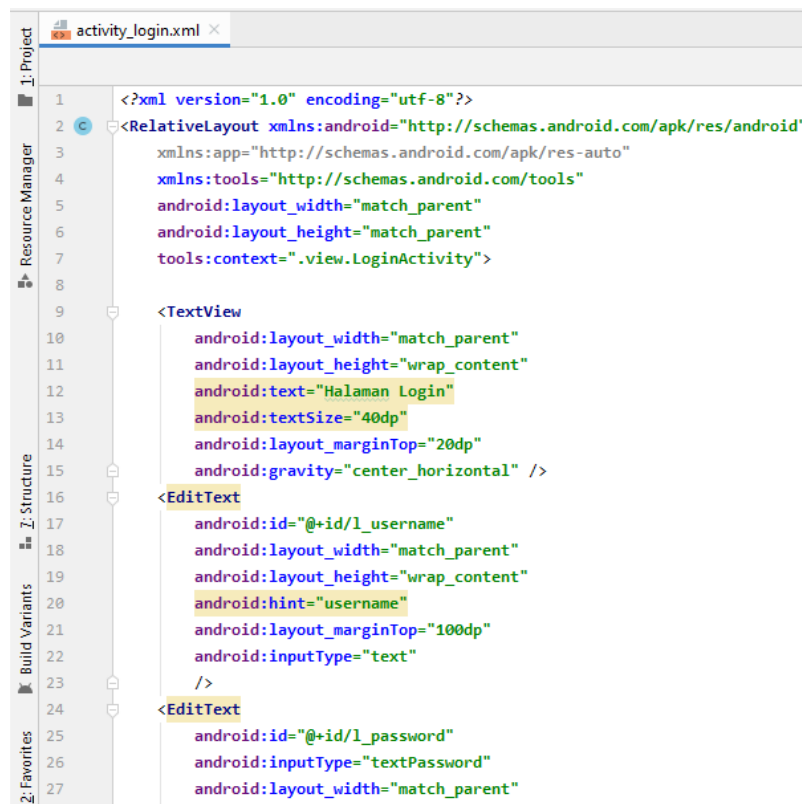
B. Halaman Login

Langkah pertama, kita akan membuat Activity Baru bernama LoginActivity



- Isi Activity Name dengan tulisan **LoginActivity** seperti pada gambar diatas

Pada *activity_login.xml*, kita isikan kode seperti berikut :



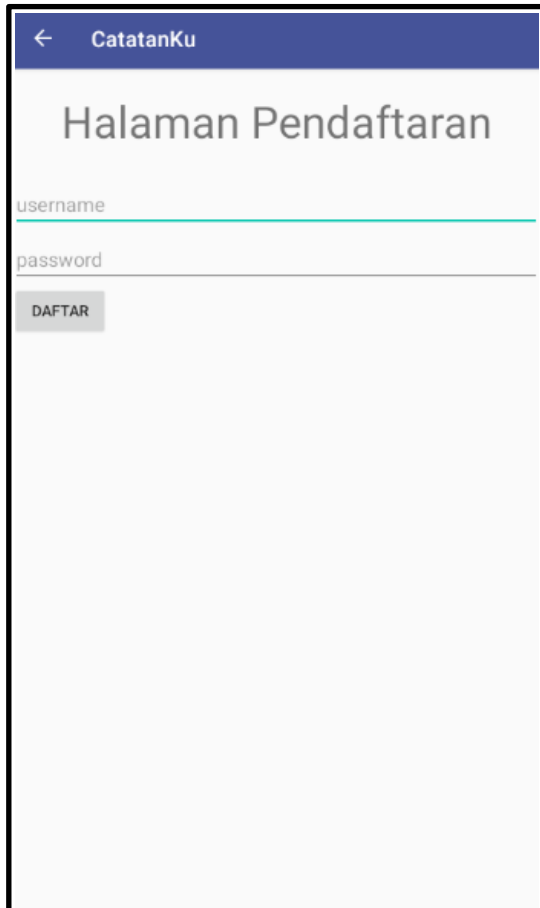
Kemudian, pada **LoginActivity.java** kita isikan kode seperti berikut ini :

```
1 package info.androidhive.sqlite.view;
2
3 import android.content.Intent;
4 import android.support.v7.app.AppCompatActivity;
5 import android.os.Bundle;
6 import android.view.View;
7 import android.widget.TextView;
8 import android.widget.Toast;
9
10 import info.androidhive.sqlite.R;
11 import info.androidhive.sqlite.database.DatabaseHelper;
12 import info.androidhive.sqlite.database.model.NoteUser;
13
14 public class LoginActivity extends AppCompatActivity {
15     private TextView textRegis;
16     public TextView user,pw,btnlogin;
17     private DatabaseHelper db;
18     @Override
19     protected void onCreate(Bundle savedInstanceState) {
20         super.onCreate(savedInstanceState);
21         setContentView(R.layout.activity_login);
22
23         textRegis = findViewById(R.id.ke_regis);
24         user = findViewById(R.id.l_username);
25         pw = findViewById(R.id.l_password);
26         btnlogin = findViewById(R.id.btn_masuk);
27
28         btnlogin.setOnClickListener((view) -> {
29             cariUser(user.getText().toString().trim(),pw.getText().toString().trim());
30         });
31
32         textRegis.setOnClickListener((view) -> {
33             startActivity(new Intent(getApplicationContext(), RegisActivity.class));
34         });
35     }
36
37     public void cariUser(String user, String pw){
38         db = new DatabaseHelper( context: this);
39         NoteUser n = db.getUser(user);
40         if (n == null) {
41             Toast.makeText( context: this, text: "Username Anda Salah", Toast.LENGTH_SHORT).show();
42         }else{
43             if(n.getPassword().equals(pw)){
44                 startActivity(new Intent(getApplicationContext(), MainActivity.class));
45             }else{
46                 Toast.makeText( context: this, text: "Password Anda Salah", Toast.LENGTH_SHORT).show();
47             }
48         }
49     }
50 }
```

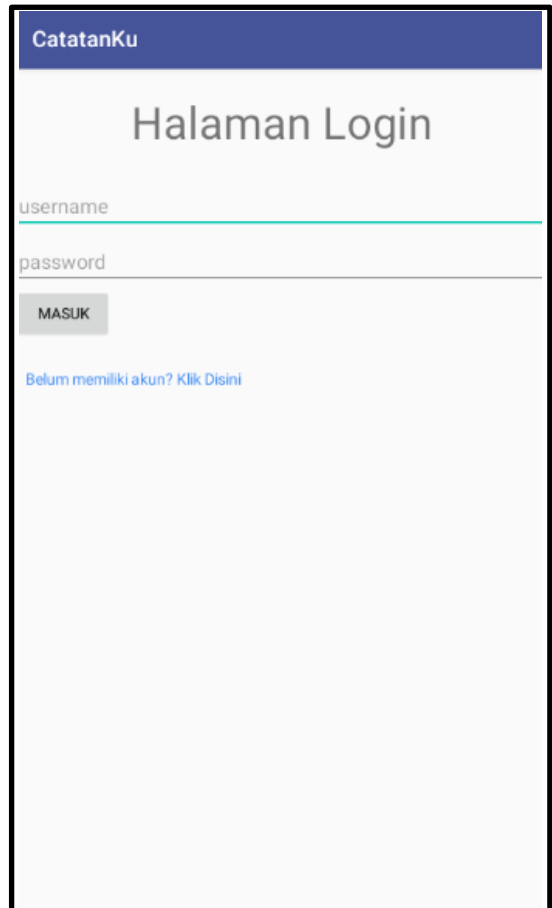
- function **cariUser** berfungsi untuk mencari **user** pada database. Jika user ada maka login berhasil, jika tidak ada, maka akan menampilkan “Username Anda Salah”

C. Tampilan Halaman Register dan Login

Berikut adalah tampilan pada Halaman Register dan Login yang telah dibuat diatas.



Register



Login

BAB IV

PENUTUP

4.1 - Kesimpulan

Berdasarkan uji coba dan analisis yang telah dilakukan maka dapat diambil kesimpulan bahwa SQLite Helper mudah dan dapat digunakan untuk melakukan implementasi CRUD pada aplikasi berbasis android.

DAFTAR PUSAKA

[1] - Vogel, L. (2010). Android sqlite database and contentprovider-tutorial. *Java, Eclipse, Android and Web programming tutorials*, 8.

[2] – Android SQLite Database CRUD with Example Application ---

(<https://medium.com/@bhawanthagunawardana/android-sqlite-database-crud-s-with-example-application-4f5a841da8f6>)

[3] – Save Data using SQLite ---

(<https://developer.android.com/training/data-storage/sqlite>)

[4] – Android Studio Documentation ---

(<https://developer.android.com/studio/build>)