Perancangan Aplikasi Qeuangans

logbook

Faris Muhammad Ihsan
1.18.4.099
Syabriena Putri Veriane
1.18.4.094

Informatics Research Center
Applied Bachelor Program of Informatics Engineering

Bandung
2019

Contents

Chapter 1

Pertemuan 1

1.1 Issues #1

Pada *issues #1* (*Hardcoded String*) *Hardcoded string* sebenarnya bukan merupakan *error*, namun hanya sebagai peringatan. Peringatan ini terjadi karena menyimpan *hard code string* pada file *layout*.

```
android:text="Tabungan anda"
```

Pemecahan dari Hardcoded String tersebut adalah dengan menuliskan*string* pada file terpisah yang telah disediakan yaitu String.XML. Penulisan *Hardcode String* yang terpisah pada file String.XML ini dapat memudahkan developer pada saat akan melakukan pengubahan nama. Developer hanya perlu mengubah pada file String.XML.

Solusinya, Pada file String.XML dituliskan baris kode seperti:

```
<resources>
<string name="app_name">Qeuangans</string>
<string name="Tabungan_anda">Tabungan Anda</string>
</resources>
```

Kemudian, untuk Memanggil *Hardcoded String* mengunakan perintah:

```
android:text="@string/Tabungan_anda
```

Perintah tersebut dituliskan pada MainActivity.XML

1.2 Issues #2

Pada *issues #2* (*Error: Tag start is Not Closed*) berisi *Error* pada file XML. Masalah ini disebabkan karena pada file XML tersebut terdapat baris perintah tanpa *tag* penutup. XML merupakan file (*eXtensible Markup Language*) dan merupakan bahasa *Markup* layaknya bahasa HTML. Pada bahasa XML ini diperlukan tag pembuka dan penutup(contoh: ¡/¿), jika ada salah satu tag (baik pembuka maupun penutup) yang tidak ditulis, maka akan terbentuk error dan baris perintah tidak akan di eksekusi.

```
<TextView android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignParentTop="true"
android:layout_marginTop="18dp"
android:textColor="#000000"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintLeft_toLeftOf="parent"
app:layout_constraintRight_toRightOf="parent"
app:layout_constraintTop_toTopOf="parent"
android:text="@string/Tabungan_anda"
```

Pada *error* ini memperbaikinya dengan cara menambahkan tag pada akhir baris perintah.

```
<TextView android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignParentTop="true"
android:layout_marginTop="18dp"
android:textColor="#000000"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintLeft_toLeftOf="parent"
app:layout_constraintRight_toRightOf="parent"
app:layout_constraintTop_toTopOf="parent"
android:text="@string/Tabungan_anda" />
```

1.3 Issues #3

Pada *issues #3* (*Unusend Import Statement*) yaitu ketika kita akan mengimport suatu *statement* maka akan menuliskannya pada awal baris kode. Mengimport disini

berarti mengambil method yang ada pada kelas lain. Namun, *Unused Import Statement* akan muncul ketika ada baris *import* yang tidak digunakan. Hal ini hanya berupa peringatan dan bukan error.

```
package com.example.qeuangans;

import androidx.appcompat.app.AppCompatActivity;

import android.database.Cursor;
import android.graphics.Color;
import android.graphics.drawable.ColorDrawable;
import android.os.Bundle;
import android.util.Log;
import android.view.View; //Unused Import Statement: Issues \#6
import android.widget.AdapterView; //Unused Import Statement: Issues \#6
import android.widget.ArrayAdapter;
import android.widget.Toast;
```

Solusinya adalah dengan menghapus import yang tidak digunakan karena methodnya tidak dibuat, maka perintah import tersebut dihapus.

```
package com.example.qeuangans;

import androidx.appcompat.app.AppCompatActivity;

import android.database.Cursor;
import android.graphics.Color;
import android.graphics.drawable.ColorDrawable;
import android.os.Bundle;
import android.util.Log;
import android.widget.ArrayAdapter;
import android.widget.Toast;
```

1.4 Issues #4

Pada *issues #4* (*Error: Cannot Resolve Method*) disini merupakan jenis *error* yang disebabkan karena pemanggilan method yang salah atau method yang belum dibuat. Namun ketika baris kode ini dijalankan tidak ada nama method yang sesuai. Pada

kasus ini, terdapat method yang belum dibuat namun sudah dipanggil. Kode Untuk memanggilnya menggunakan.

```
jmlSaldo();
```

Pada pemanggilan tersebut terdapat method yang belum dibuat.

Solusi yang harus dilakukan adalah dengan menambahkan method dengan nama jmlSaldo(); beserta isi dari method yang dipanggil tersebut.

```
private void jmlSaldo() {
        Cursor cursor = db.saldo();
        while (cursor.moveToNext()) {
        textsaldo.append("Rp " + cursor.getString(0) + ",00");
        }
}
```

1.5 Issues #5

Pada *issues #5* (*Error: Cannot Resolve Symbol*) Merupakan jenis error ketika ada objek yang dipanggil namun id nya tidak sama atau tidak ada (*typo*), yang membuat baris kode tersebut tidak terbaca dan error sehingga objek untuk menampilkan saldo pada tampilah aplikasi tidak akan muncul.

```
<TextView
android:layout_width="match_parent"
android:layout_height="57dp"
android:layout_alignParentTop="true"
android:layout_marginTop="58dp"
android:hint="@string/Rp00"
android:textColor="#000000"
android:textSize="20sp"
android:id="@+id/textado" <!-- Cannot Resolve Symbol: Issues #3--> />
```

Solusinya adalah memberi id sesuai dengan yang dipanggil sehingga tampilan saldo akan muncul.

```
<TextView
        android:layout_width="match_parent"
        android:layout_height="57dp"
```

```
        android:layout_alignParentTop="true"
        android:layout_marginTop="58dp"
        android:hint="@string/Rp00"
        android:textColor="#000000"
        android:textSize="20sp"
        android:id="@+id/textsaldo" <!-- Cannot Resolve Symbol: Issues #3--> />
```

1.6 Issues #6

Pada *issues #6* (*intent Pindah()*) ini digunakan sebagai perintah untuk pindah dari *Activity* yang satu ke yang lainnya. Intent merupakan objek yang menyediakan fungsi untuk pindah dari satu *activity* ke *activity* lainnya. Penggunaan intent dari *method* pindah sebagai berikut:

```
public void Pindah(View view) {
        Intent intent = new Intent(MainActivity.this, InputData.class);
        startActivity(intent);
    }
```

Intent pada method ini digunakan untuk pindah dari *Activity* MainActivity.class ke *Activity* InputData.class. Intent disini diaktifkan dengan perintah startActivity(intent);

1.7 Issues #7

Pada *issues #7* (*Error: Semicolon Expected*) *Semicolon Expected* adalah *error* yang terjadi ketika sebuah baris program yang sudah ditulis kekurangan tanda *semicolon*(;) pada akhir dari program, sehingga ketika baris program di eksekusi akan memunculkan peringatan *error Semicolon Expected*. Contoh error:

```
startActivity(intent)
```

Solusinya adalah dengan menambahkan *semicolon*(;) pada akhir baris program.

```
startActivity(intent);
```

1.8 Issues #8

Pada *issues #8* (*Error: Cannot resolve symbol*) *Cannot Resolve Symbol* disini adalah ketika sebuah method yang menggunakan method dari kelas lainnya tapi tidak melakukan impor method maka akan muncul error ini. Solusinya adalah dengan cara menambahkan import method pada awal kode program.

1.9 Issues #9

Pada *issues #9* (*Error:* } *Expected*) merupakan error yang disebabkan karena kurangnya tanda (}) pada akhir kode program. Sama halnya seperti file XML yang memmerlukan tag pembuka dan penutup, sebuah method pada java juga harus menggunakan tanda pembuka({) dan tanda penutupnya(}). Contoh *Error:* } *Expected*:

```
public void Pindah(View view) {
        Intent intent = new Intent(MainActivity.this, InputData.class);
        startActivity(intent);
```

Solusinya adalah dengan menambahkan tanda penutup} pada akhir method

```
public void Pindah(View view) {
        Intent intent = new Intent(MainActivity.this, InputData.class);
        startActivity(intent);
}
```

1.10 Issues #10

Pada *issues #10* (*Error: Missing Parent*) *Missing Parent* adalah *error* yang disebabkan ketika kita ingin mendefinisikan objek namun belum mengimport library nya.

Chapter 2

Pertemuan 2

2.1 Issues #11

Pada *issues #11* (*Cek mainActivity*) *mainActivity* Merupakan activity yang utama ketika program dijalankan. Pada saat cek mainActivity, Program sudah bisa berjalan tanpa error.

2.2 Issues #12

Pada *issues #12* (Penambahan Fitur Laporan) merupakan penambahan laporan.java pada aplikasi, penambahan fitur laporan ini menggunakan arraylist

```
import androidx.appcompat.app.AppCompatActivity;

import android.database.Cursor;
import android.os.Bundle;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.Toast;

import java.util.ArrayList;
public class laporan extends AppCompatActivity {
DatabaseHelper db;
ListView listLaporanK;
@Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
```

```
        setContentView(R.layout.activity_laporan);

        listLaporanK = findViewById(R.id.listLaporanK);

        LK = new ArrayList<>();

        db = new DatabaseHelper(this);

        datalaporankeluar();
    }

    private void datalaporankeluar() {
        Cursor cursor = db.datalaporankeluar();
        if (cursor.getCount() == 0) {
            Toast.makeText(this, "TIDAK ADA DATA", Toast.LENGTH_SHORT).show();
        } else {
            while (cursor.moveToNext()) {
                LK.add("\n" + cursor.getString(0) + "\n" + "Tanggal : "
                + cursor.getString(5) + "\n" + "\n" + "Pemasukan : "
                + cursor.getString(1)
                        + "\n" + "Jumlah : Rp " + cursor.getString(2)
                        + ",00" + "\n" + "\n" + "Pengeluaran : "
                        + cursor.getString(3)
                        + "\n" + "Jumlah : Rp" + cursor.getString(4)
                        + ",00" + "\n");
            }
        }
    }
}
```

2.3 Issues #13

Pada *issues #13* (Tombol laporan tidak muncul pada aplikasi). Pada issues ini, tombol laporan tidak muncul pada aplikasi karena tombol tersebut belum dipanggil untuk dimunculkan. untuk memanggilnya menggunakan

```
tombolLaporan = findViewById(R.id.tombolLaporan);
```

disini berarti tombolLaporan ini dihubungkan dengan id dari tombol yang ada di mainActivity.XML dengan id tombolLaporan.

2.4 Issues #14

Pada *issues #14* (*Error: Undefined Method*) *Undefined Method* merupakan jenis *error* yang disebaban karena ketika method dipanggil tetapi method yang dipanggil tidak ada, maka akan muncul *Undefined Method*. Solusi yang digunakan adalah dengan membuat method yang dipanggil.
Pemanggilan method:

```
Laporan();
```

Method yang dipanggil:

```
public void Laporan(View view) {
    Intent intent = new Intent(MainActivity.this, laporan.class);
    startActivity(intent);
}
```

2.5 Issues #15

Pada *issues #15* (*Error: ID is not Defined Anywhere*) *ID is not Defined Anywhere* merupakan sebuah *error*. *Error* ini terjadi ketika sebuah id tombol pada activity_main.XML (android:id="Laporan") tidak memiliki ID yang benar, maka solusinya adalah memberikan id yang benar dengan menambahkan @+id pada awal kata "Laporan" contoh:

```
android:id="@+id/tombolLaporan"
```

penggunaan @+id disini agar sesuai dengan penulisan yang digunakan pada android.

2.6 Issues #16

Pada *issues #16* (*Error: Missing Constraints*) *Missing Constraints* adalah *error* yang terjadi ketika sebuah tampilan yang ada tidak ditentukan posisinya secara pasti. Contoh tampilan tanpa cnstraint pada activity_main.XML:

```
<TextView android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignParentTop="true"
android:layout_marginTop="18dp"
android:textColor="#000000"
android:text="@string/Tabungan_anda"
```

Jika posisi tidak ditentukan, maka yang terjadi adalah, tampilan tersebut akan otomatis berada pada (0,0) bagian layar atau di posisi kiri atas ketika aplikasi dijalankan. Solusi error tersebut adalah dengan menentukan posisinya.

```
<TextView android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignParentTop="true"
android:layout_marginTop="18dp"
android:textColor="#000000"
android:text="@string/Tabungan_anda"
app:layout_constraintTop_toBottomOf="@+id/TombolLaporankeluar"
```

layout_constraintTop_toBottomOf merupakan penentuan posisi pada bagian bawah yang dihubungkan dengan tampilan yang memiliki id "TombolLaporankeluar" yang ada pada activity_main.XML.

2.7 Issues #17

Pada *issues #17* (*Error: Cannot Resolve Method add(java.lang.string)*) *Cannot Resolve Method add(java.lang.string* adalah ketika method tidak bisa menambahkan value string pada arraylist yang akan ditampilkan.

2.8 Issues #18

Pada *issues #18* (Mengkoneksikan Database) Untuk mengkoneksikan database dapat menggunakan perintah:

```
db = new DatabaseHelper(this);
```

Perintah ini dimasukan pada file MainActivity.java

2.9 Issues #19

Pada *issues #19* (*Error: Cannot return value from void method.*) *Error: Cannot return value from void method* merupakan sebuah *error* yang disebabkan karena mencoba menggunakan perintah return(mengembalikan nilai) pada method void. Contoh pada DatabaseHelper.java:

```
public void datalaporankeluar(){
        SQLiteDatabase db = this.getReadableDatabase();
        Cursor saldo;
        saldo = db.rawQuery("SELECT SUM(" +COL3+")-SUM(" +COL5+") FROM "
        +TBLNAME, null);
        return saldo;
    }
```

Sedangkan void sendiri merukpakan method yang tidak mengembalikan nilai sehingga return tidak bisa digunakan pada method void. Untuk solusinya yaitu dengan membuat methud tersebut menjadi function. Contoh:

```
public Cursor datalaporankeluar(){
    SQLiteDatabase db = this.getReadableDatabase();
    Cursor saldo;
    saldo = db.rawQuery("SELECT SUM(" +COL3+")-SUM(" +COL5+") FROM "
    +TBLNAME, null);
    return saldo;
}
```

Void disini diubah menjadi "cursor" karena digunakan untuk menganmbil value suatu data pada database yang di eksekusi dengan perintah "return saldo;"

2.10 Issues #20

Pada *issues #20* (Koneksi ke Database) Untuk membuat koneksi ke database mengunakan perintah:

```
db = new DatabaseHelper(this);
```

Chapter 3

Pertemuan 2

3.1 Issues #21

Pada *issues #21* (*Error: Cannot Resolve Symbol*) *Cannot Resolve symbol*

3.2 Issues #22

Pada *issues #22* (Test Notifikasi) issues ini berisi test dari notifikasi yang sudah berjalan. Notifikasi ini menggunakan method Notiftest yang diletakan pada mainActivity.java.

```
public void Notiftest(View view) {
NotificationCompat.Builder builder = new NotificationCompat.Builder(this)
        .setSmallIcon(R.mipmap.ic_launcher)
           .setContentTitle("Issues Notif")
        .setAutoCancel(true)
        .setContentText("Berhasil Pak");
    NotificationManager notificationManager =
    (NotificationManager)getSystemService(Context.NOTIFICATION_SERVICE);
    notificationManager.notify(NOTIFICATION_ID,builder.build());
}
```

3.3 Issues #23

Pada *issues #23* (Membuat layout pemasukan dan pengeluaran)layout dibuat dalam file xml dengan memasukan tombol-tombol dan inputan text. Dalam pembuatan layout ini menggunakan cara *Drag and Drop* melalui opsi yang sudah disediakan oleh android studio dan menghasilkan tampilan seprti:

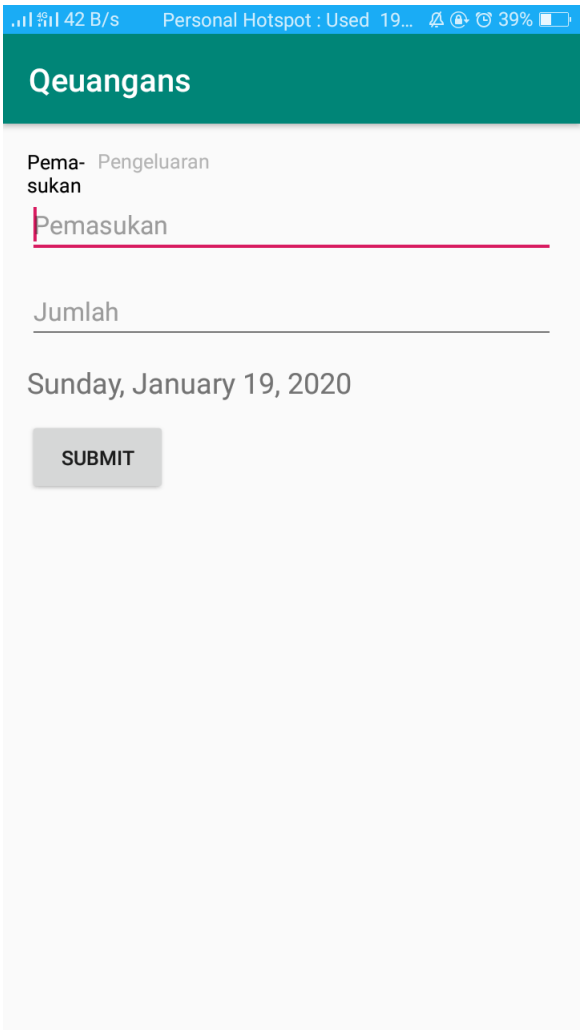

Figure 3.1: Gambar Layout Pemasukan

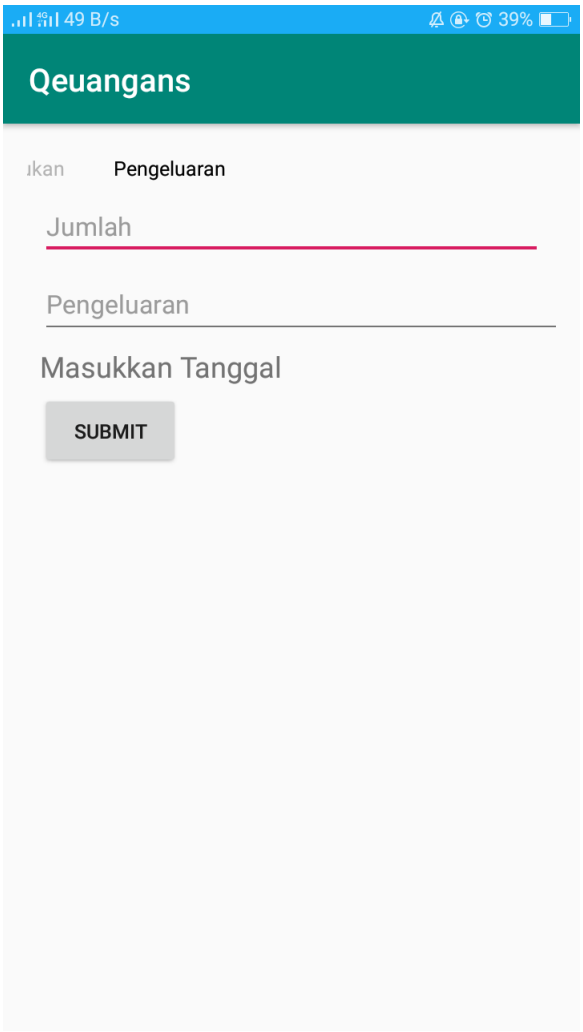

Figure 3.2: Gambar Layout Pengeluaran

3.4 Issues #24

Pada *issues #24* (*Datepicker Dialog* pada input pengeluaran). *Datepicker* merupakan fasilitas yang disediakan android untuk memilih tanggal. Untuk menggunakan *datepicker* menggunakan baris kode seperti berikut:

```
final int year = calendar.get(Calendar.YEAR);
final int month = calendar.get(Calendar.MONTH);
final int day = calendar.get(Calendar.DAY_OF_MONTH);

tglpengeluaran.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        DatePickerDialog datepickerdialog =
        new DatePickerDialog(InputData2.this,
                android.R.style.Theme_Holo_Light_Dialog_MinWidth,
                setListener, year,month,day);
        datepickerdialog.getWindow().setBackgroundDrawable
        (new ColorDrawable(Color.TRANSPARENT));
        datepickerdialog.show();
    }
});
setListener = new DatePickerDialog.OnDateSetListener() {
    @Override
    public void onDateSet(DatePicker datePicker, int year, int month,
    int dayOfMont) {
        month = month+1;
        String date = day+"/"+month+"/"+year;
        tglpengeluaran.setText(date);
    }
};
```

3.5 Issues #25

Pada *issues #25* Menambahkan getar pada notifikasi. Bars kode ini dimasukan pada method notiftest pada mainActivity.java

```
    .setContentText("Berhasil Pak");
```

3.6 Issues #26

Pada *issues #26* Menambahkan *DatePickerDialog* pada InputData.java

```
tglpemasukan.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            DatePickerDialog datePickerDialog = new DatePickerDialog(
                    InputData.this,android.R.style
                    .Theme_Holo_Light_Dialog_MinWidth,
                    setListener,year,month,day);
            datePickerDialog.getWindow().setBackgroundDrawable
            (new ColorDrawable(Color.TRANSPARENT));
            datePickerDialog.show();
        }
    });

    setListener = new DatePickerDialog.OnDateSetListener() {
        @Override
        public void onDateSet(DatePicker datePicker, int year,
        int month, int dayOfMonth) {
            month = month+1;
            String date = day+"/"+month+"/"+year;
            tglpemasukan.setText(date);
        }
    };
```

3.7 Issues #27

Pada *issues #27* menambahkan suara pada notifikasi dengan memasukan

```
.setSound(RingtoneManager.getDefaultUri
(RingtoneManager.TYPE_NOTIFICATION));
```

pada Notiftest() pada mainActivity.java

3.8 Issues #28

Pada *issues #28* (Mengubah fungsi Pada Pemasukan). Pengubahan fungsi ini dilakukan pada InputData.java karena layout pemasukan dan pengeluaran yang dibagi menjadi dua.

3.9 Issues #29

Pada *issues #29* (*Error: Cannot Resolve Symbol 'jenispengeluaran'*)

```
private void AddData() {
    inputmasuk.setOnClickListener(
            new View.OnClickListener() {
                @Override
                public void onClick(View view) {
                        boolean isInseted = myDB.insertData
                        (jenispemasukan.getText().toString(),
                                jmlpemasukan.getText().toString(),
                                **jenispengeluaran** .getText().toString(),
                                jmlpengeluaran .getText().toString(),
                                tglpemasukan.getText().toString());
                        if (isInseted = true)
                            Toast.makeText(InputData.this, "INPUT BERHASIL",
                            Toast.LENGTH_LONG).show();
                        else
                            Toast.makeText(InputData.this, "INPUT GAGAL",
                            Toast.LENGTH_LONG).show();

                }
            }
```

Error ini terjadi karena method *insertData* pada *DatabaseHelper.java* tidak ada perintah untuk melakukan inputan jenispengeluaran sehingga pada method AddData() diatas terjadi *error cannot resolve symbol*. Solusinya adalah dengan menghapus jenispengeluaran pada method AddData.

```
private void AddData() {
    inputmasuk.setOnClickListener(
```

```
new View.OnClickListener() {
    @Override
    public void onClick(View view) {
            boolean isInseted = myDB.insertData(jenispemasukan
            .getText().toString(),
                    jmlpemasukan.getText().toString(), tglpemasukan
                    .getText().toString());
            if (isInseted = true)
                Toast.makeText(InputData.this, "INPUT BERHASIL",
                Toast.LENGTH_LONG).show();
            else
                Toast.makeText(InputData.this, "INPUT GAGAL",
                Toast.LENGTH_LONG).show();

    }
}
```

3.10 Issues #30

Pada *issues #30* (Menambahkan *Current Date* pada Input Pemasukan (InputData.java)). Current Date disini merupakan tanggal pada hari ini, cara menambahkannya dengan menambahkan baris perintah

```
Calendar calendar = Calendar.getInstance();
String currentDate = DateFormat.getDateInstance(DateFormat.FULL)
.format(calendar.getTime());
```

pada InputData.java

Chapter 4

Pertemuan 4

4.1 Issues #31

Pada *issues #31* (Menambahkan *Casting* pada InputData2.java) casting ini dilakukan untuk mensingkronisasi view pada file activity_InputData2.XML dengan InputData2.java.

```
jenispengeluaran = findViewById(R.id.jenispengeluaran);
```

jenis pengeluaran disini merupakan *variable* yang dapat dipanggil. Dalam *variable* tersebut terdapat *findViewById* yang digunakan untuk mencari view pada activity_InputData2.XML dan *R.id.jenispengeluaran* merupakan id dari view yang akan di sinkronisasi pada activity_InputData2.XML

4.2 Issues #32

Pada *issues #32* (penambahan perintah SQL) Perintah SQL ini ditambahkan untuk melakukan input data ke database yang diinputkan pada DatabaseHelper.java

```
db.execSQL("create table "+ TBLNAME +"(ID INTEGER PRIMARY KEY AUTOINCREMENT, J
            "PEMASUKAN NUMBER, JENIS_PENGELUARAN STRING, PENGELUARAN NUMBER, T
            "PEMASUKAN NUMBER, JENIS_PENGELUARAN STRING, PENGELUARAN NUMBER, '
            "TANGGAL_PEMASUKAN STRING, TANGGAL_PENGELUARAN STRING)");
```

4.3 Issues #33

Pada *issues #33* (*Error: setVibrate(long[]) in Builder cannot be applied to ()*) disebabkan karena perintah .setVibrate(); hanya bisa menerima tipe data Array. Maka melakukan perubahan kode seperti ini:

```
long[] PolaGetar = {100, 100};
.setVibrate(PolaGetar);
```

4.4 Issues #34

Pada *issues #34*(mengubah gambar pada notifikasi). Pengubahan atau penambahan gambar pada notifikasi dengan menambahkan baris kode seperti dibawah pada method notifikasi.

```
.setSmallIcon(R.drawable.ic_cash)
```

4.5 Issues #35

Pada *issues #35* (*Cannot resolve method 'setContentTitle(java.lang.string)'*) Penyebabnya adalah Kekurangan tanda kurung tutup pada

```
.setSmallIcon(R.drawable.ic_cash
.setContentTitle("Issues Notif")
```

solusinya dengan menambahkan kurung tutup pada

```
.setSmallIcon(R.drawable.ic_cash)
```

4.6 Issues #36

Pada *issues #36* (*Error:) expected*). Penyebabnya adalah kurangnya tanda) pada

```
notificationManager.notify(NOTIFICATION_ID,builder.build();
```

Solusinya menambahkan kurung tutup pada baris kode tersebut seperti:

```
notificationManager.notify(NOTIFICATION_ID,builder.build());
```

4.7 Issues #37

Pada *issues #37*(*Array initializer is not allowed here*).

```
long PolaGetar = {100, 100};
```

Karena pada baris tersebut tidak ada tanda [] setelah long yang menandakan bahwa tipe data tersebuat adalah array.
Solusinya adalah dengan menambahkan [] setelah long untuk menandakan bahwa data tersebut merupakan array.

```
long [] PolaGetar = {100, 100};
```

4.8 Issues #38

Pada *issues #38* (*Error: when INSERT INTO tabungan*). Error ini terjadi ketika akan memasukan data input pemasukan ke dalam tabel tabungan pada field Jenis_pemasukan, Jumlah_pemasukan, tanggal. Hal ini terjadi karena field tanggal tidak ada. Solusi, menambahkan field tanggal_pemasukan pada DatabaseHelper

```
public static final String COL6 = "TANGGAL_PEMASUKAN";
```

4.9 Issues #39

Pada *issues #39*(Mengubah Format Penulisan Tanggal). Untuk mengubah format penulisan tanggal menggunakan perintah yang dimasukan pada input data.java

```
final Calendar calendar = Calendar.getInstance();
String currentDate = DateFormat.getDateInstance(DateFormat.FULL)
.format(calendar.getTime());

myDB = new DatabaseHelper(this);


calendar.set(Calendar.YEAR, year);
            calendar.set(Calendar.MONTH, month);
            calendar.set(Calendar.DAY_OF_MONTH, dayOfMonth);
            String date = DateFormat.getDateInstance(DateFormat.FULL)
            .format(calendar.getTime());
```

4.10 Issues #40

Pada *issues #40*(Menambahkan perintah SQL pembuatan tabel). Perintah SQL ini ditulis dalam method public void onCreate(SQLiteDatabase db) pada DatabaseHelper.java

```
public void onCreate(SQLiteDatabase db) {
    db.execSQL("create table "+ TBLNAME +
    "(ID_PEMASUKAN INTEGER PRIMARY KEY AUTOINCREMENT," +
    "JENIS_PEMASUKAN STRING, PEMASUKAN NUMBER," +
     "TANGGAL_PEMASUKAN STRING)");

    db.execSQL("create table "+ TBLNAME2 +
    "(ID_PENGELUARAN INTEGER PRIMARY KEY AUTOINCREMENT," +
    "JENIS_PENGELUARAN STRING, PENGELUARAN NUMBER," +
    "TANGGAL_PENGELUARAN STRING)");
}
```

Chapter 5

Pertemuan 5

5.1 Issues #41

Pada *issues #41*(Menambahkan perintah pada onUpgrade database helper). Penambahan perintah yang dilakukan ini agar database di-replace ketika sudah ada filenya pada perangkat yang digunakan. PErintah ini ditulis pada DatabaseHelper

```
@Override
public void onUpgrade(SQLiteDatabase db, int i, int i1) {
    db.execSQL("DROP TABLE IF EXISTS "+TBLNAME);
    db.execSQL("DROP TABLE IF EXISTS "+TBLNAME2);
    onCreate(db);
}
```

5.2 Issues #42

Pada *issues #42*(Mengubah list.java). Pengubahan pada list ini digunakan untuk meisahkan antara list pemasukan dan pengeluaran

```
if (cursor.getCount() == 0) {
    Toast.makeText(this, "TIDAK ADA DATA", Toast.LENGTH_SHORT)
    .show();
} else {
    while (cursor.moveToNext()) {
        listitem.add("\n" + "Tanggal : " + cursor.getString(3)
                    + "\n" + "\n" + "Jenis Pemasukan : "
                    + cursor.getString(1)
                    + "\n" + "Jumlah : Rp " + cursor.getString(2)
```

```
                + ",00" + "\n");
        }
        final ArrayAdapter adapter = new ArrayAdapter<>
        (this, android.R.layout.simple_list_item_1, listitem);
        listView.setAdapter(adapter);
```

5.3 Issues #43

Pada *issues #43*(Menambahkan List2.java) Penambahan ini dilakukan melalui *new* pada aplikasi android studio. New – activity – empty activity

5.4 Issues #44

Pada *issues #44*(Menambahkan ListView pada activity_list2.xml). Untuk menambahkan listView ini, dengan cara *drag n drop* melalui palletes yang ada pada desain activity_list2.xml Namun Bisa juga dengan melakukannya secara manual dengan menginputkan baris kode ini:

```
<com.baoyz.swipemenulistview.SwipeMenuListView
android:id="@+id/listView2"
android:layout_width="match_parent"
android:layout_height="match_parent"/>
```

5.5 Issues #45

Pada *issues #45*(Menambahkan perintah untuk menampilkan data pada List2.java). Penambahan perintah untuk menampilkan data pada arraylist yang diambil dari database.

5.6 Issues #46

Pada *issues #46*(*Error: Cannot Resolve Method tampilkanDataKeluar()*). Error ini terjadi karena method yang dipanggil tidak ada, maka kita harus membuat methodnya pada DatabaseHelper.java

```
public Cursor tampilkanDataKeluar() {
    SQLiteDatabase db = this.getReadableDatabase();
    Cursor res;
```

```
        res = db.rawQuery("SELECT * FROM " + TBLNAME2, null);
        return res;
    }
```

5.7 Issues #47

Pada *issues #47*(*Error: ; expected*). Error ini terjadi karena kekurangan tanda ; pada akhir baris kode.

```
public Cursor tampilkanDataKeluar() {
SQLiteDatabase db = this.getReadableDatabase();
Cursor res;
res = db.rawQuery("SELECT * FROM "+TBLNAME2,null);
return
```

Solusi: Menambahkan tanda ; pada akhir baris kode setelah perintah *retutn*

```
return;
```

5.8 Issues #48

Pada *issues #48*(*Error: '}' expected*). Error ini terjadi karena kekurangan tanda kurung tutup (}) pada akhir method atau prosedur.

```
public Cursor tampilkanDataKeluar() {
    SQLiteDatabase db = this.getReadableDatabase();
    Cursor res;
    res = db.rawQuery("SELECT * FROM " + TBLNAME2, null);
    return res;
```

Solusi: Menambahkan } pada akhir method

```
public Cursor tampilkanDataKeluar() {
    SQLiteDatabase db = this.getReadableDatabase();
    Cursor res;
    res = db.rawQuery("SELECT * FROM " + TBLNAME2, null);
    return res;
}
```

5.9 Issues #49

Pada *issues #49*(Set Tanggal Otomatis Hari Ini). Untuk set tanggal otomatis hari ini dengan menuliskan

```
tglpengeluaran.setText(currentDate);
```

Pada inputData2.java

5.10 Issues #50

Pada *issues #50*