

# **LAPORAN TUGAS BESAR PEMROGRAMAN MOBILE**

**DISUSUN OLEH :**

**MUHAMMAD MAKHFUD REZKY THAMRIN**

**1809075028**



**PROGRAM STUDI TEKNIK ELEKTRO  
FAKULTAS TEKNIK  
UNIVERSITAS MULAWARMAN**

**SAMARINDA  
2021**

# **BAB I**

## **PENDAHULUAN**

### **1.1 Latar Belakang**

Pada saat sekarang ini seiring dengan perkembangan teknologi, masyarakat telah menganggap teknologi bukan sebagai sesuatu yang mewah lagi tetapi sudah sebagai suatu kebutuhan. Perkembangan teknologi ini tidak terlepas dari perkembangan teknologi smartphone. Masyarakat yang ada di dunia saat ini hampir semua telah mengenal smartphone dan sudah hampir tidak ada lagi yang menggunakan handphone. Hal ini tidak terlepas dari perkembangan pesat sistem operasi Android. Sejak hadir pada tahun 2008, saat ini hampir semua vendor dan produsen smartphone telah menggunakan sistem operasi Android. Terkecuali dua produsen besar lainnya yaitu Microsoft dan Apple. Kehadiran sistem operasi Android berdampak besar terhadap perkembangan teknologi smartphone yang kita rasakan saat sekarang ini.

Android sendiri adalah sistem operasi mobile yang dikembangkan oleh Google. Google mengembangkan Android sebagai sistem operasi yang bersifat open source, artinya sistem operasi ini dapat dikembangkan oleh siapa saja yang ingin mengembangkannya karena itulah sebagai besar vendor dan produsen smartphone di dunia saat ini menggunakan Android. Mereka bebas untuk mengubah dan mendesain versi Android mereka masing – masing tanpa menghilangkan inti dari Android yang dikembangkan oleh Google. Hal ini juga berdampak langsung terhadap para pengembang aplikasi Android. Google juga menyediakan perangkat lunak ( software ) bagi setiap orang yang ingin mengembangkan aplikasi Android sehingga akan memudahkan seseorang dalam mengembangkan suatu aplikasi Android.

Telah banyak berbagai aplikasi Android yang sudah dikembangkan oleh para pengembang diantaranya aplikasi berjenis chatting, social media, game, pengolah kata ( word processing ), pemutar musik dan video serta berbagai jenis aplikasi lainnya. Diantara berbagai macam jenis aplikasi yang telah hadir pada Android, salah satu aplikasi sederhana yang memiliki banyak fungsi dan manfaat adalah

aplikasi penyimpan catatan. Aplikasi penyimpan catatan ini sendiri sesuai dengan namanya berfungsi untuk menyimpan segala catatan yang dibuat oleh pengguna yang nantinya akan tersimpan dan dapat dilihat kembali saat diperlukan oleh pengguna. Karena hal itulah Penulis ingin mengembangkan aplikasi penyimpan catatan ini yang sederhana tetapi akan sangat dirasakan oleh setiap orang fungsi dan manfaatnya.

## **1.2 Rumusan Masalah**

Berdasarkan permasalahan yang sudah dijelaskan pada latar belakang maka rumusan masalah yang diajukan adalah :

- a. Bagaimana merancang dan mendesain aplikasi penyimpan catatan ini?
- b. Bagaimana merancang *database* serta kode program agar aplikasi penyimpan catatan ini berjalan dengan baik?

## **1.3 Tujuan**

Mengacu pada latar belakang, rumusan masalah dan batasan masalah maka tujuan dari pengembangan aplikasi penyimpan catatan ini adalah :

- a. Membangun aplikasi catatan yang dapat memudahkan pengguna untuk menyimpan catatan.
- b. Aplikasi catatan ini dapat membuat catatan baru, mengubah catatan yang telah ada dan juga menghapus catatan yang sudah tidak diperlukan sesuai dengan kebutuhan pengguna.

## **1.4 Manfaat**

Setelah dijelaskan tujuan dari pengembangan aplikasi catatan ini maka manfaat yang dapat diambil adalah :

- a. Pengguna aplikasi catatan ini nantinya akan mudah untuk membuat catatan – catatan yang diperlukannya.
- b. Pengguna tidak akan mengalami kendala seperti kehilangan catatan dan juga lupa saat menyimpan catatan yang dia perlukan.

## **BAB II**

### **LANDASAN TEORI**

#### **2.1 Android**

Menurut Juhara (2016), Android adalah sistem operasi berbasis Linux yang dimodifikasi untuk perangkat bergerak ( *mobile device* ) yang terdiri dari sistem operasi, *middleware* dan aplikasi- aplikasi utama. Android dikembangkan oleh Google dimana Android didesain untuk perangkat *mobile* dengan layar sentuh. Tidak hanya pada perangkat *mobile* kini Android juga telah ada pada televisi yaitu Android TV, pada kendaraan yaitu Android Auto, pada jam tangan yaitu Android Wear dan Android juga terdapat pada berbagai perangkat elektronik lainnya seperti pada *notebook*, konsol game, kamera digital dan lainnya.

Kode program Android dirilis oleh Google dibawah lisensi *open source* sehingga berbagai perusahaan yang memproduksi *smartphone* dapat merilis sistem operasi Android mereka sendiri tanpa menghilangkan inti dari Android yang dikembangkan oleh Google. Tidak hanya itu karena sifatnya yang *open source* membuat banyak pengembang dan para antusias dapat mengembangkan Android mereka dengan berbagai fitur – fitur baru dan kemudian dibagikan kepada orang lain agar dapat digunakan.

#### **2.2 Android Studio**

Android Studio merupakan software integrated development environment ( IDE ) yang resmi untuk membangun aplikasi Android. Android Studio dibangun berdasarkan IntelliJ IDEA yang merupakan software integrated development enviroment untuk membangun aplikasi atau software dengan bahasa pemrograman Java/ Seluruh fitur yang ada pada InteliJ IDEA juga terdapat pada Android Studio yang kemudian ditambahkan lagi fitur – fitur lainnya agar dapat meningkatkan produktivitas para pengembang aplikasi Android yaitu seperti :

1. System build Gradle yang fleksibel.
2. Emulator Android yang kaya dengan fitur.

3. Dukungan untuk membangun aplikasi Android untuk perangkat apapun baik itu pada smartphone, Android TV, Android Wear dan perangkat Android lainnya.

### 2.3 SQLite

SQLite merupakan sistem manajemen basis data relasional ( *relational database management system* ) yang dibangun menggunakan bahasa pemrograman C. Berbeda dengan sistem manajemen basis data yang lain, SQLite bukan merupakan *database* yang bersifat *client – server*. Tetapi SQLite merupakan *database* yang disisipkan pada sesuatu perangkat lunak, aplikasi dan juga sistem operasi. Android merupakan salah satu sistem operasi yang didalamnya disisipkan *database* SQLite.

Perkembangan SQLite diawali oleh penciptanya yaitu D. Richard Hipp pada tahun 2000 dimana Hipp bekerja untuk Angkatan Laut Amerika Serikat. Kemudian SQLite dirilis pada bulan Agustus 2000 dan perkembangannya terus dilakukan sampai saat sekarang ini dengan versi terakhirnya saat ini adalah SQLite 3.18.0.

### 2.4 Kotlin

Kotlin adalah sebuah bahasa pemrograman dengan pengetikan statis yang berjalan pada Mesin Virtual Java ataupun menggunakan kompiler LLVM yang dapat pula dikompilasikan kedalam bentuk kode sumber JavaScript. Pengembang utamanya berasal dari tim programer dari JetBrains yang bermarkas di Rusia.[2] Meskipun sintaksisnya tidak kompatibel dengan bahasa Java, Kotlin didesain untuk dapat bekerja sama dengan kode bahasa Java dan bergantung kepada kode bahasa Java dari Kelas Pustaka Java yang ada, seperti berbagai framework Java yang ada. Tim Pengembang memutuskan menamakannya Kotlin dengan mengambil nama dari sebuah pulau di Rusia, sebagaimana Java yang mengambil nama dari pulau Jawa di Indonesia.[3] Setelah Google mengumumkan bahwa Kotlin menjadi bahasa kelas satu bagi Android, maka bersama Java dan C++, Kotlin menjadi bahasa resmi untuk pengembangan aplikasi-aplikasi Android .

## BAB III

### PERANCANGAN APLIKASI

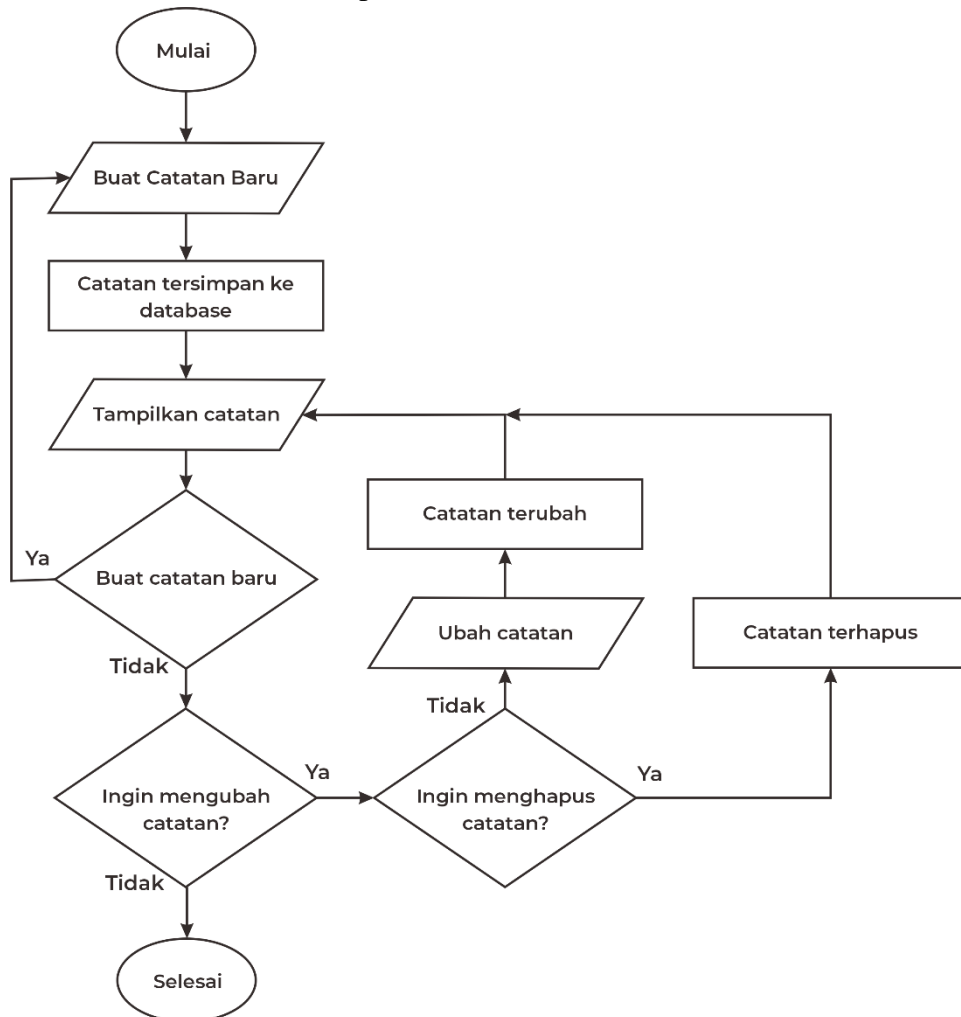
#### 3.1 Perancangan Sistem

##### 3.1.1 Spesifikasi Minimum

Aplikasi NoteKu ini dapat berjalan dengan spesifikasi minimum Android 5.0 Lolipop (Api level 21).

##### 3.1.2 Flowchart

Untuk memahami proses kerja aplikasi catatan ini maka dibutuhkan flowchart (diagram alir) yang akan menunjukkan bagaimana proses kerja aplikasi catatan, berikut adalah flowchart dari aplikasi catatan :



**Gambar 1.** Flowchart aplikasi NoteKu

## 3.2 Perancangan UI/UX

### 3.2.1 Components

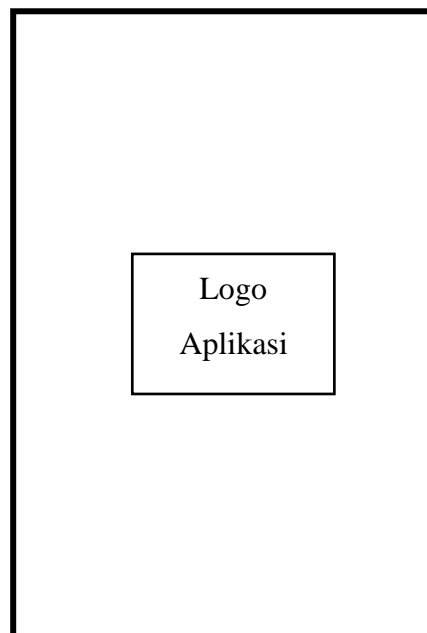
Component pada aplikasi NoteKu ini, tersusun dari:

- |                           |                       |
|---------------------------|-----------------------|
| a. App Bar Layout         | l. Linear Layout      |
| b. Bottom App Bar         | m. Menu               |
| c. Button                 | n. Nested Scroll View |
| d. CardView               | o. Progress Bar       |
| e. ConstraintLayout       | p. RecyclerView       |
| f. Coordinator Layout     | q. Relative Layout    |
| g. Edit Text              | r. Rounded ImageView  |
| h. Floating Action Button | s. Text View          |
| i. FrameLayout            | t. Toolbar            |
| j. Image View             | u. View               |
| k. Item                   |                       |

### 3.2.2 Layout

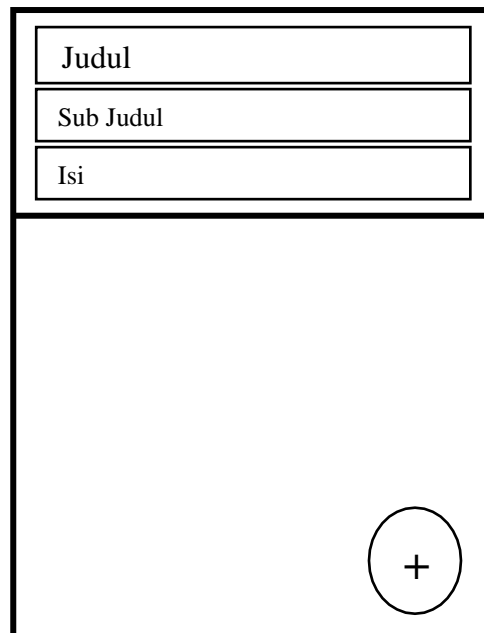
Berikut adalah rancangan tampilan aplikasi NoteKu berdasarkan masing – masing activity-nya :

- a. Activity Splash Screen



**Gambar 2.** Rancangan layout Splash Screen

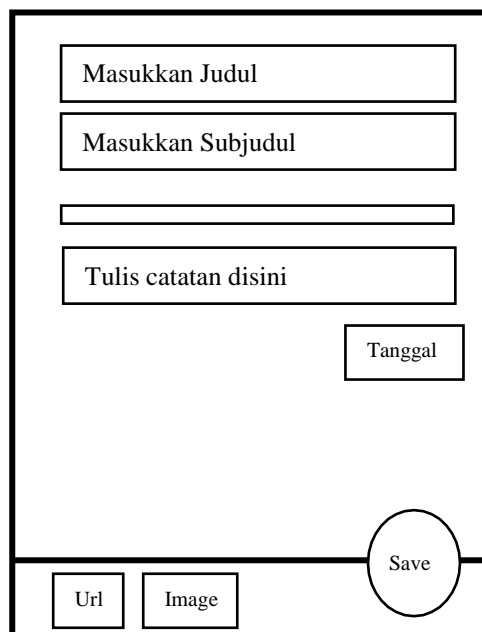
b. Activity Main



The diagram shows a vertical rectangular container. At the top, there are three stacked rectangular input fields. The first field is labeled 'Judul', the second 'Sub Judul', and the third 'Isi'. Below these fields is a large, empty rectangular area. In the bottom right corner of this large area, there is a circular button containing a plus sign (+).

**Gambar 3.** Rancangan layout Activity Main

c. Activity Create Note



The diagram shows a vertical rectangular container. At the top, there are three stacked rectangular input fields. The first field is labeled 'Masukkan Judul', the second 'Masukkan Subjudul', and the third is empty. Below the third field is a rectangular input field labeled 'Tulis catatan disini'. To the right of this field, there is a smaller rectangular input field labeled 'Tanggal'. At the bottom of the container, there are two small rectangular input fields labeled 'Url' and 'Image'. To the right of these fields is a circular button labeled 'Save'.

**Gambar 4.** Rancangan layout Activity Create Note



### 3.2.3 Style

- a. Tema Noteku menggunakan Theme Material Components Day Night No Action Bar
- b. App Theme App Bar Overlay menggunakan Theme Overlay Material Components Dark Action Bar
- c. Warna utama menggunakan warna teal 700
- d. Warna gelap utama menggunakan warna teal 700
- e. Warna aksen menggunakan kode warna #01221F

### 3.2.4 Animation

Animasi yang digunakan pada aplikasi NoteKu ini hanya ada pada activity splash screen. Hanya berupa progress bar.



**Gambar 5.** Progress bar

## 3.3 Coding

### 3.3.1 CreateNoteActivity

```
package com.pm.noteku

import android.Manifest
import android.annotation.SuppressLint
import android.app.Dialog
import android.content.Intent
import android.content.pm.PackageManager
import android.graphics.BitmapFactory
import android.graphics.drawable.ColorDrawable
import android.net.Uri
import android.os.AsyncTask
import android.os.Bundle
import android.provider.MediaStore
import android.util.Patterns
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
```

```

import android.widget.*
import androidx.appcompat.app.AlertDialog
import androidx.appcompat.app.AppCompatActivity
import androidx.core.app.ActivityCompat
import androidx.core.content.ContextCompat
import kotlinx.android.synthetic.main.activity_create_note.*
import kotlinx.android.synthetic.main.layout_delete.*
import kotlinx.android.synthetic.main.layout_url.*
import kotlinx.android.synthetic.main.layout_url.view.*
import java.text.SimpleDateFormat
import java.util.*

class CreateNoteActivity : AppCompatActivity() {

    var alertDialog: AlertDialog? = null
    var selectImagePath: String? = null
    var modelNoteExtra: ModelNote? = null

    @SuppressWarnings("SetTextI18n", "RestrictedApi")
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_create_note)

        tvDateTime.setText("Terakhir diubah : " + SimpleDateFormat("dd
MMMM yyyy", Locale.getDefault()).format(Date()))

        selectImagePath = ""

        if (intent.getBooleanExtra("EXTRA", false)) {
            modelNoteExtra = intent.getSerializableExtra("EXTRA_NOTE")
as ModelNote
            setViewOrUpdateNote()
        }

        if (modelNoteExtra != null) {
            linearDelete.visibility = View.VISIBLE
            btnDelete.setOnClickListener {
                showDeleteDialog()
            }
        }

        btnHapusUrl.setOnClickListener {
            tvUrlNote.setText(null)
            tvUrlNote.setVisibility(View.GONE)
            btnHapusUrl.setVisibility(View.GONE)
        }

        btnAddUrl.setOnClickListener {
            showDialogUrl()
        }

        btnAddImage.setOnClickListener {
            if (ContextCompat.checkSelfPermission(applicationContext,
                Manifest.permission.READ_EXTERNAL_STORAGE) !=
PackageManager.PERMISSION_GRANTED) {
                ActivityCompat.requestPermissions(this@CreateNoteActivity,

```

```

    arrayOf(Manifest.permission.READ_EXTERNAL_STORAGE),
    REQUEST_PERMISSION)
        } else {
            selectImage()
        }
    }

    fabDeleteImage.setOnClickListener {
        imageNote.setImageBitmap(null)
        imageNote.setVisibility(View.GONE)
        fabDeleteImage.setVisibility(View.GONE)
        selectImagePath = ""
    }

    fabSaveNote.setOnClickListener(View.OnClickListener {
        if (editTextTitle.getText().toString().isEmpty()) {
            Toast.makeText(this@CreateNoteActivity, "Judul Tidak
Boleh Kosong", Toast.LENGTH_SHORT).show()
            return@OnClickListener
        } else if (editTextSubTitle.getText().toString().isEmpty()
&& editTextDesc.getText().toString().isEmpty()) {
            Toast.makeText(this@CreateNoteActivity, "Catatan Tidak
Boleh Kosong", Toast.LENGTH_SHORT).show()
            return@OnClickListener
        }

        val modelNote = ModelNote()
        modelNote.title = editTextTitle.getText().toString()
        modelNote.subTitle = editTextSubTitle.getText().toString()
        modelNote.noteText = editTextDesc.getText().toString()
        modelNote.dateTime = tvDateTime.getText().toString()
        modelNote.imagePath = selectImagePath

        if (tvUrlNote.getVisibility() == View.VISIBLE) {
            modelNote.url = tvUrlNote.getText().toString()
            btnHapusUrl.visibility = View.VISIBLE
        }

        if (modelNoteExtra != null) {
            modelNote.id = modelNoteExtra!!.id
        }

        class saveNoteAsyncTask : AsyncTask<Void?, Void?, Void?>()
        {
            override fun doInBackground(vararg p0: Void?): Void? {

                NoteDatabase.getInstance(applicationContext)?.noteDao()?.insert(modelN
ote)

                return null
            }

            override fun onPostExecute(aVoid: Void?) {
                super.onPostExecute(aVoid)
                val intent = Intent()
                setResult(RESULT_OK, intent)
                finish()
            }
        }
    })

```

```

        }
    }
    saveNoteAsyncTask().execute()
})
}

@SuppressLint("RestrictedApi")
private fun setViewOrUpdateNote() {
    editTextTitle.setText(modelNoteExtra?.title)
    editTextSubTitle.setText(modelNoteExtra?.subTitle)
    editTextDesc.setText(modelNoteExtra?.noteText)

    if (modelNoteExtra?.imagePath != null &&
        modelNoteExtra?.imagePath?.trim()?.isEmpty()!!) {

imageNote.setImageBitmap(BitmapFactory.decodeFile(modelNoteExtra?.imagePath))

        imageNote.visibility = View.VISIBLE
        selectImagePath = modelNoteExtra?.imagePath
        fabDeleteImage.visibility = View.VISIBLE
    }

    if (modelNoteExtra?.url != null &&
        modelNoteExtra?.url?.trim()?.isEmpty()!!) {
        tvUrlNote.text = modelNoteExtra?.url
        tvUrlNote.visibility = View.VISIBLE
        btnHapusUrl.visibility = View.VISIBLE
    }
}

@SuppressLint("QueryPermissionsNeeded")
private fun selectImage() {
    val intent = Intent(Intent.ACTION_PICK,
        MediaStore.Images.Media.EXTERNAL_CONTENT_URI)
    if (intent.resolveActivity(packageManager) != null) {
        startActivityForResult(intent, REQUEST_SELECT)
    }
}

override fun onRequestPermissionsResult(requestCode: Int,
    permissions: Array<String>, grantResults: IntArray) {
    super.onRequestPermissionsResult(requestCode, permissions,
        grantResults)
    if (requestCode == REQUEST_PERMISSION && grantResults.size >
        0) {
        if (grantResults[0] == PackageManager.PERMISSION_GRANTED)
        {
            selectImage()
        } else {
            Toast.makeText(this, "Permission Denied",
                Toast.LENGTH_SHORT).show()
        }
    }
}

@SuppressLint("RestrictedApi")
override fun onActivityResult(requestCode: Int, resultCode: Int,

```

```

data: Intent?) {
    super.onActivityResult(requestCode, resultCode, data)
    if (requestCode == REQUEST_SELECT && resultCode == RESULT_OK)
    {
        if (data != null) {
            val selectImgUri = data.data
            if (selectImgUri != null) {
                try {
                    val inputStream =
contentResolver.openInputStream(selectImgUri)
                    val bitmap =
BitmapFactory.decodeStream(inputStream)
                    imageNote.setImageBitmap(bitmap)
                    imageNote.visibility = View.VISIBLE
                    fabDeleteImage.visibility = View.VISIBLE
                    selectImagePath = getPathFromUri(selectImgUri)
                } catch (e: Exception) {
                    e.printStackTrace()
                    Toast.makeText(this, e.message,
Toast.LENGTH_SHORT).show()
                }
            }
        }
    }

    private fun getPathFromUri(contentUri: Uri): String? {
        val filePath: String?
        val cursor = contentResolver.query(contentUri, null, null,
null, null)
        if (cursor == null) {
            filePath = contentUri.path
        } else {
            cursor.moveToFirst()
            val index = cursor.getColumnIndex("_data")
            filePath = cursor.getString(index)
            cursor.close()
        }
        return filePath
    }

    private fun showDeleteDialog() {
        val dialog = Dialog(this@CreateNoteActivity)
        dialog setContentView(R.layout.layout_delete)
        dialog.tvHapusCatatan.setOnClickListener {

            class HapusNoteAsyncTask : AsyncTask<Void?, Void?,
Void?>() {
                override fun doInBackground(vararg p0: Void?): Void? {

                    NoteDatabase.getInstance(applicationContext)?.noteDao()?.delete(modelNoteExtra)

                    return null
                }

                override fun onPostExecute(aVoid: Void?) {
                    super.onPostExecute(aVoid)

```

```

        val intent = Intent()
        intent.putExtra("NoteDelete", true)
        setResult(RESULT_OK, intent)
        finish()
    }
}
HapusNoteAsyncTask().execute()
}
dialog.tvBatalHapus.setOnClickListener {
    dialog.dismiss()
}
dialog.show()
}

private fun showDialogUrl() {
    if (alertDialog == null) {
        val builder = AlertDialog.Builder(this@CreateNoteActivity)
        val view =
            LayoutInflater.from(this).inflate(R.layout.layout_url,
            findViewById(R.id.layoutUrl) as? ViewGroup)
        builder.setView(view)

        alertDialog = builder.create()
        if (alertDialog?.window != null) {

alertDialog?.window?.setBackgroundDrawable(ColorDrawable(0))
        }

        val etUrl = view.editTextAddUrl
        etUrl.requestFocus()

        view.tvOk.setOnClickListener {
            if (etUrl.text.toString().trim().isEmpty()) {
                Toast.makeText(this@CreateNoteActivity, "Masukan
                Url", Toast.LENGTH_SHORT).show()
            } else if
            (!Patterns.WEB_URL.matcher(etUrl.text.toString()).matches()) {
                Toast.makeText(this@CreateNoteActivity, "Url Anda
                Tidak Benar", Toast.LENGTH_SHORT).show()
            } else {
                tvUrlNote.text = etUrl.text.toString()
                tvUrlNote.visibility = View.VISIBLE
                btnHapusUrl.visibility = View.VISIBLE
                alertDialog?.dismiss()
            }
        }
        view.tvBatal.setOnClickListener {
            alertDialog?.dismiss()
        }
    }
    alertDialog?.show()
}

companion object {
    private const val REQUEST_PERMISSION = 1
    private const val REQUEST_SELECT = 2
}
}

```

### 3.3.2 MainActivity

```
package com.pm.noteku

import android.annotation.SuppressLint
import android.content.Intent
import android.os.AsyncTask
import android.os.Bundle
import android.view.Menu
import android.view.MenuItem
import androidx.appcompat.app.AppCompatActivity
import androidx.recyclerview.widget.LinearLayoutManager
import androidx.recyclerview.widget.StaggeredGridLayoutManager
import kotlinx.android.synthetic.main.activity_create_note.*
import kotlinx.android.synthetic.main.activity_main.*
import kotlinx.android.synthetic.main.activity_main.toolbar
import java.util.*
import kotlin.collections.ArrayList

class MainActivity : AppCompatActivity(), onClickItemListener {

    private val modelNoteList: MutableList<ModelNote> = ArrayList()
    private var noteAdapter: NoteAdapter? = null
    private var onClickPosition = -1

    @SuppressLint("Assert")
    override fun onCreate( savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        setSupportActionBar(toolbar)
        assert(supportActionBar != null)

        fabCreateNote.setOnClickListener {
            startActivityForResult(Intent(this@MainActivity,
                CreateNoteActivity::class.java), REQUEST_ADD)
        }

        noteAdapter = NoteAdapter(modelNoteList, this)
        rvListNote.setAdapter(noteAdapter)

        modeGrid()

        getNote(REQUEST_SHOW, false)
    }

    private fun modeGrid() {
        rvListNote.layoutManager = StaggeredGridLayoutManager(2,
            StaggeredGridLayoutManager.VERTICAL)
    }

    private fun modeList() {
```

```

        rvListNote.LayoutManager = LinearLayoutManager(this)
    }

    private fun getNote(requestCode: Int, deleteNote: Boolean) {

        @Suppress("UNCHECKED_CAST")
        class GetNoteAsyncTask : AsyncTask<Void?, Void?,
List<ModelNote>>() {
            override fun doInBackground(vararg p0: Void?):
List<ModelNote>? {
                return
                NoteDatabase.getInstance(this@MainActivity)?.noteDao()?.allNote as
List<ModelNote>?
            }

            override fun onPostExecute(notes: List<ModelNote>) {
                super.onPostExecute(notes)
                if (requestCode == REQUEST_SHOW) {
                    modelNoteList.addAll(notes)
                    noteAdapter?.notifyDataSetChanged()
                } else if (requestCode == REQUEST_ADD) {
                    modelNoteList.add(0, notes[0])
                    noteAdapter?.notifyItemInserted(0)
                    rvListNote.smoothScrollToPosition(0)
                } else if (requestCode == REQUEST_UPDATE) {
                    modelNoteList.removeAt(onClickPosition)
                    if (deleteNote) {

noteAdapter?.notifyItemRemoved(onClickPosition)
                    } else {
                        modelNoteList.add(onClickPosition,
notes[onClickPosition])

noteAdapter?.notifyItemChanged(onClickPosition)
                    }
                }
            }
        }
        GetNoteAsyncTask().execute()
    }

    public override fun onActivityResult(requestCode: Int, resultCode:
Int, data: Intent?) {
        super.onActivityResult(requestCode, resultCode, data)
        if (requestCode == REQUEST_ADD && resultCode == RESULT_OK) {
            getNote(REQUEST_ADD, false)

        } else if (requestCode == REQUEST_UPDATE && resultCode ==
RESULT_OK) {
            if (data != null) {
                getNote(REQUEST_UPDATE,
data.getBooleanExtra("NoteDelete", false))
            }
        }
    }
}

```



```

        override fun onClick(modelNote: ModelNote, position: Int) {
            onClickPosition = position
            val intent = Intent(this, CreateNoteActivity::class.java)
            intent.putExtra("EXTRA", true)
            intent.putExtra("EXTRA_NOTE", modelNote)
            startActivityForResult(intent, REQUEST_UPDATE)
        }

        companion object {
            private const val REQUEST_ADD = 1
            private const val REQUEST_UPDATE = 2
            private const val REQUEST_SHOW = 3
        }

        override fun onCreateOptionsMenu(menu: Menu): Boolean {
            menuInflater.inflate(R.menu.menu_main, menu)
            return true
        }

        override fun onOptionsItemSelected(item: MenuItem): Boolean {
            when (item.itemId) {
                R.id.listView -> modelList()
                R.id.gridView -> modeGrid()
            }
            return super.onOptionsItemSelected(item)
        }
    }
}

```

### 3.3.3 ModelNote

```

package com.pm.noteku

import androidx.room.ColumnInfo
import androidx.room.Entity
import androidx.room.PrimaryKey
import java.io.Serializable

@Entity(tableName = "notes")
class ModelNote : Serializable {
    @PrimaryKey(autoGenerate = true)
    var id = 0

    @ColumnInfo(name = "title")
    var title: String? = null

    @ColumnInfo(name = "date_time")
    var dateTime: String? = null

    @ColumnInfo(name = "sub_title")
    var subTitle: String? = null

    @ColumnInfo(name = "note_text")
    var noteText: String? = null

    @ColumnInfo(name = "image_path")

```

```

        var imagePath: String? = null

        @ColumnInfo(name = "color")
        var color: String? = null

        @ColumnInfo(name = "web_url")
        var url: String? = null
        override fun toString(): String {
            return "$title : $dateTime"
        }
    }
}

```

### 3.3.4 NoteAdapter

```

package com.pm.noteku

import android.graphics.BitmapFactory
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.TextView
import androidx.cardview.widget.CardView
import androidx.recyclerview.widget.RecyclerView
import com.makeramen.roundedimageview.RoundedImageView
import com.pm.noteku.NoteAdapter.NoteViewHolder
import kotlinx.android.synthetic.main.list_item_note.view.*

class NoteAdapter(private val modelNoteListFilter: List<ModelNote>,
                  private val onClickItem: onClickItemListener) :
    RecyclerView.Adapter<NoteViewHolder>() {

    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int):
        NoteViewHolder {
        val view =
            LayoutInflater.from(parent.context).inflate(R.layout.list_item_note,
                parent, false)
        return NoteViewHolder(view)
    }

    override fun onBindViewHolder(holder: NoteViewHolder, position:
        Int) {
        val modelNote = modelNoteListFilter[position]

        holder.title.text = modelNote.title
        holder.text.text = modelNote.noteText
        holder.timeDate.text = modelNote.dateTime

        if (modelNote.imagePath != null) {
            holder.roundedImageView.setImageBitmap(BitmapFactory.decodeFile(modelN
                ote.imagePath))
            holder.roundedImageView.visibility = View.VISIBLE
        } else {
            holder.roundedImageView.visibility = View.GONE
        }
    }
}

```

```

        holder.cvNote.setOnClickListener {
            onItemClick.onClick(modelNote, position)
        }
    }

    override fun getItemCount(): Int {
        return modelNoteListFilter.size
    }

    override fun getItemViewType(position: Int): Int {
        return position
    }

    inner class NoteViewHolder internal constructor(itemView: View) :
RecyclerView.ViewHolder(itemView) {
        var title: TextView
        var text: TextView
        var timeDate: TextView
        var cvNote: CardView
        var roundedImageView: RoundedImageView

        init {
            title = itemView.tvTitle
            text = itemView.tvText
            timeDate = itemView.tvTime
            roundedImageView = itemView.roundedImage
            cvNote = itemView.cvNote
        }
    }
}

```

### 3.3.5 NoteDao

```

package com.pm.noteku

import androidx.room.*

@Dao
interface NoteDao {
    @get:Query("SELECT * FROM notes ORDER BY id DESC")
    val allNote: List<ModelNote?>

    @Insert(onConflict = OnConflictStrategy.REPLACE)
    fun insert(modelNote: ModelNote?)

    @Delete
    fun delete(modelNote: ModelNote?)
}

```

### 3.3.6 NoteDatabase

```

package com.pm.noteku

import android.content.Context
import androidx.room.Database
import androidx.room.Room

```

```

import androidx.room.RoomDatabase

@Database(entities = [ModelNote::class], version = 1, exportSchema =
false)
abstract class NoteDatabase : RoomDatabase() {
    abstract fun noteDao(): NoteDao?
    companion object {
        private var noteDatabase: NoteDatabase? = null
        @Synchronized
        fun getInstance(context: Context): NoteDatabase? {
            if (noteDatabase == null) {
                noteDatabase = Room.databaseBuilder(context,
NoteDatabase::class.java, "Notedb").build()
            }
            return noteDatabase
        }
    }
}

```

### 3.3.7 onClickItemListener

```

package com.pm.noteku

interface onClickItemListener {
    fun onClick(modelNote: ModelNote, position: Int)
}

```

### 3.3.8 SplashScreen

```

package com.pm.noteku

import android.content.Intent
import android.os.Bundle
import android.os.Handler
import androidx.appcompat.app.AppCompatActivity

class SplashScreen : AppCompatActivity() {
    private val SPLASH_TIME_OUT : Long = 3000
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_splash_screen)
        Handler().postDelayed({
            startActivity(Intent(this, MainActivity::class.java))
            finish()
        }, SPLASH_TIME_OUT)
    }
}

```

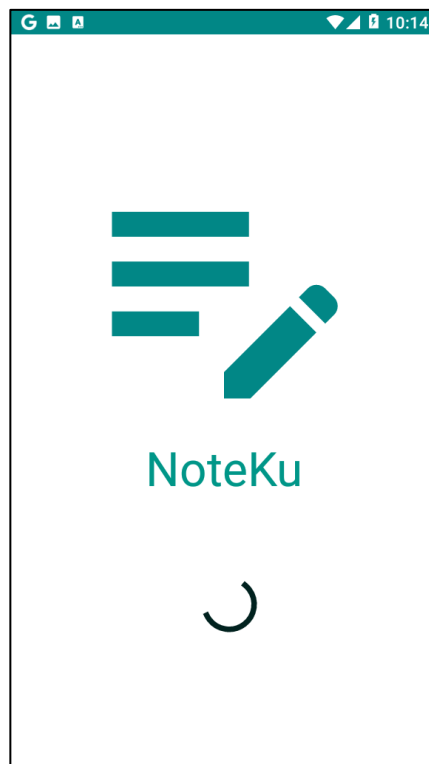
## 3.4 Testing

Pada bagian ini akan dijelaskan bagaimana hasil dari pengujian terhadap aplikasi catatan yang sebelumnya telah dirancang. Hasil pengujian ini akan dijelaskan

berdasarkan setiap *activity* atau menu yang ada pada aplikasi catatan, adapun hasil pengujian aplikasi catatan ini sebagai berikut :

#### 3.4.1 Splash Screen

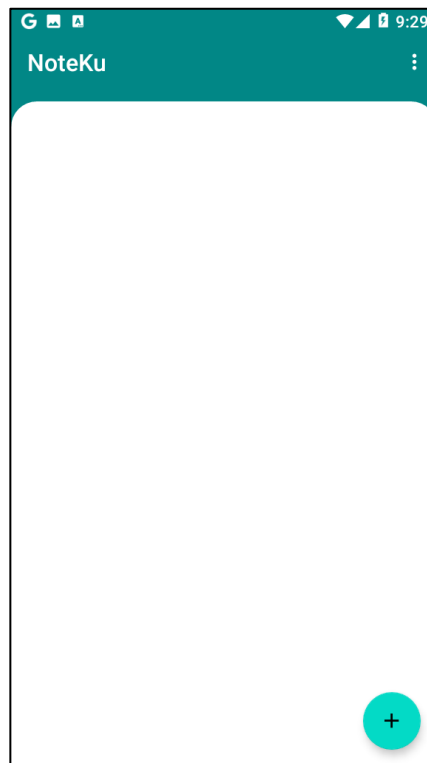
Splash Screen aplikasi ini merupakan tampilan awal yang pertama kali ditampilkan oleh aplikasi saat pengguna menjalankan aplikasi catatan. Splash Screen ini berguna sebagai tampilan awal yang akan menampilkan logo dan juga nama aplikasi catatan.



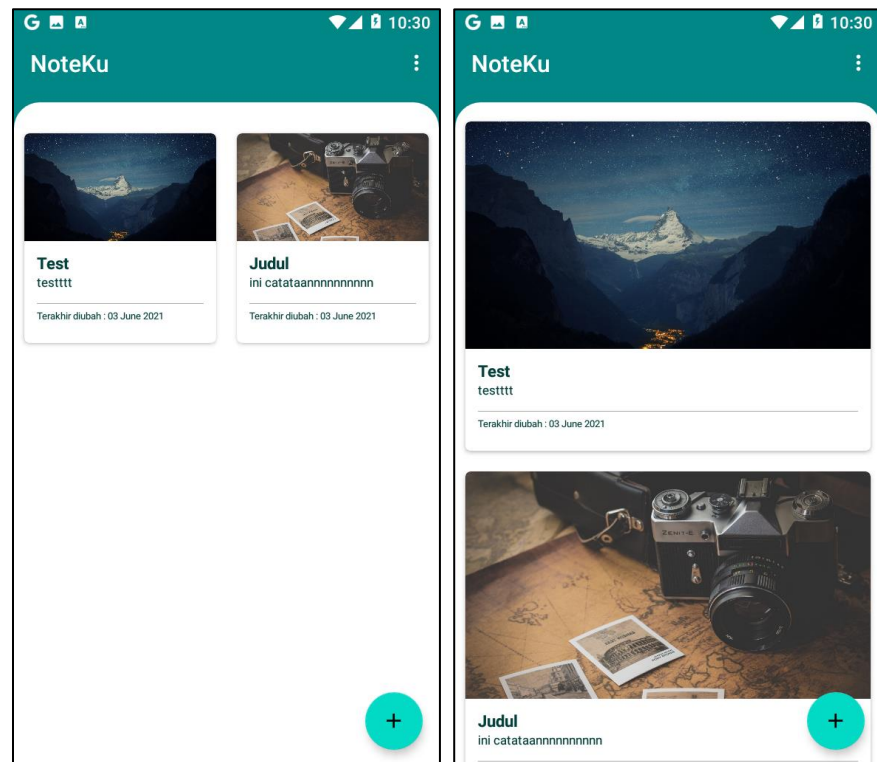
**Gambar 6.** Tampilan splash screen

#### 3.4.2 Activity Main

Activity main akan ditampilkan setelah splash screen aplikasi sehingga activity main menjadi menu utama dari aplikasi catatan ini. Pada bagian atas kanan terdapat menu option yang berisikan beberapa pilihan view yang dapat digunakan pada aplikasi dan pada bagian bawah kanan terdapat button ( tombol ) yang berfungsi untuk menampilkan activity create note dimana pengguna dapat membuat catatan baru.



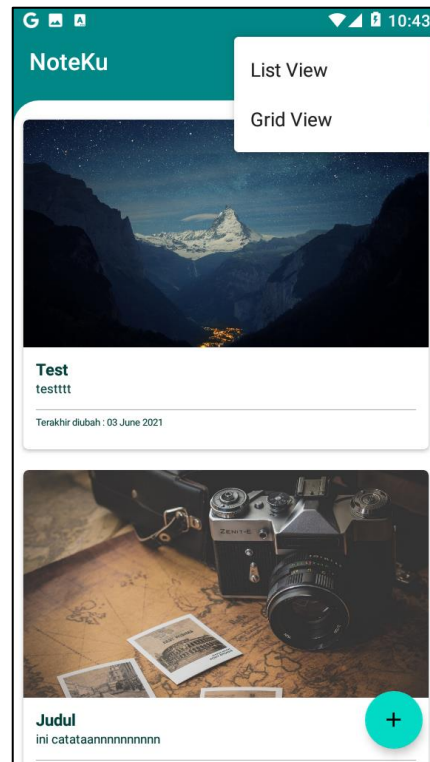
**Gambar 7.** Tampilan activity main saat tidak ada catatan



**Gambar 8.** Tampilan activity main saat terdapat catatan dengan list view dan grid view

### 3.4.3 Menu

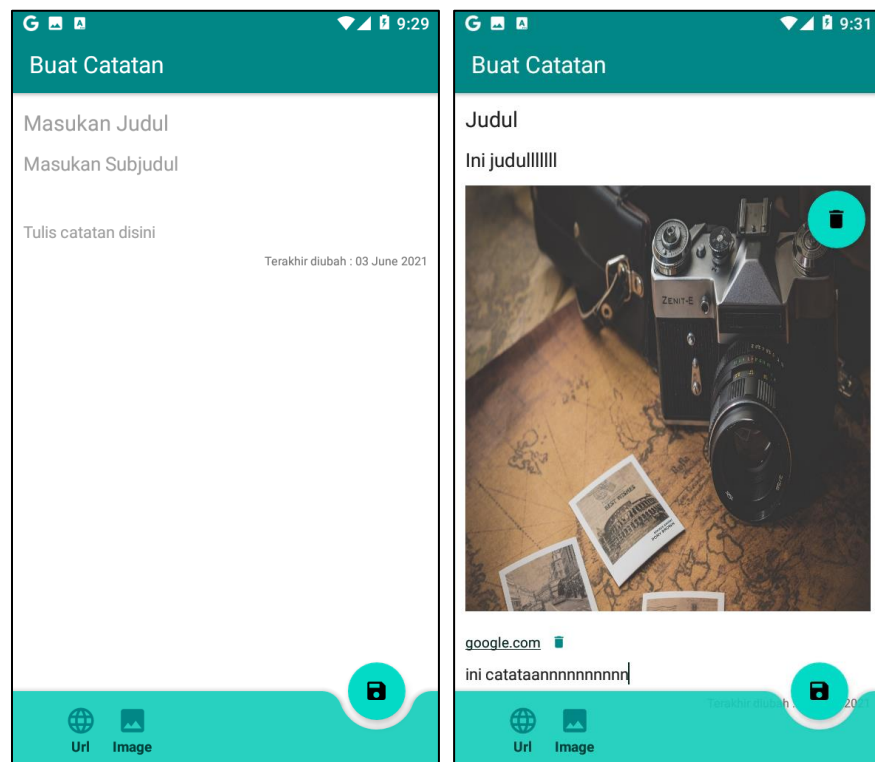
Menu merupakan menu yang akan tampil saat pengguna mengetuk *button* menu di sebelah kanan atas layar saat berada pada *activity* main. Pada menu ini terdapat pilihan list view dan grid view untuk tampilan daftar catatan.



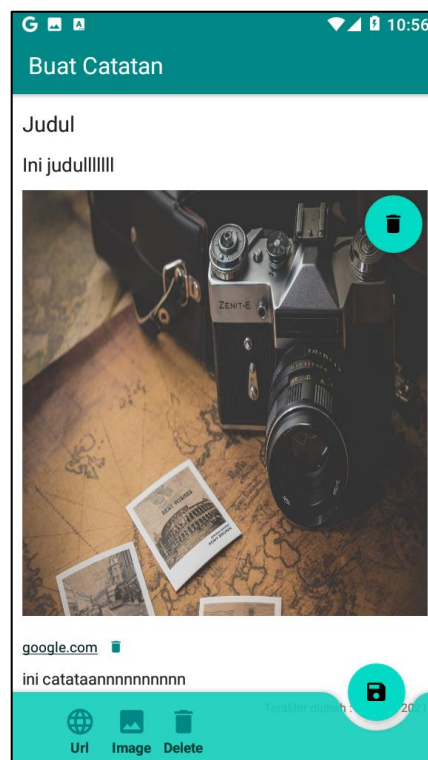
**Gambar 9.** Tampilan menu

### 3.4.4 Activity Create Note

Activity ini merupakan menu saat pengguna akan membuat catatan baru dan mengedit catatan. Di bagian bawah kiri terdapat button untuk memasukkan url dan foto. Dan juga terdapat button delete yang akan muncul jika pengguna mengedit catatan. Di bagian bawah kanan button untuk menyimpan catatan. Gambar dan url yang telah dimasukkan akan ditampilkan di atas isi catatan. Di bawah isi catatan terdapat tanggal terakhir diubah.



**Gambar 10.** Tampilan Activity Create Note saat belum di isi dan setelah di isi

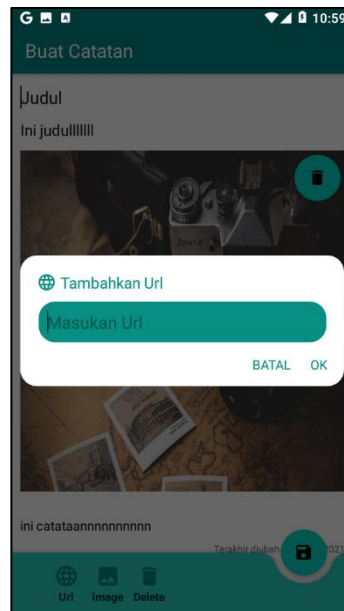


**Gambar 11.** Tampilan Activity Create Note saat mengedit catatan



### 3.4.5 Layout Url

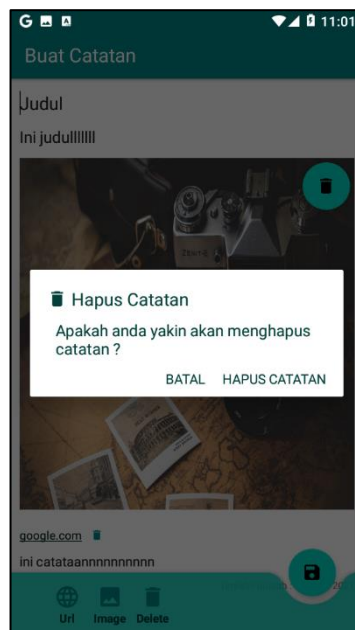
Tampilan ini akan muncul jika mengklik button url pada activity create note. Yang mana untuk memasukkan url ke dalam catatan.



**Gambar 12.** Tampilan layout url

### 3.4.6 Layout Delete

Tampilan ini akan muncul jika mengklik button delete pada activity create note. Yang mana untuk menghapus catatan.



**Gambar 13.** Tampilan layout delete

## 3.5 Debuging

### 3.5.1 Terdouble App

Jika aplikasi telah terinstal maka akan terinstal dua aplikasi, yang mana juga isinya tidak semua masuk aktivitasnya. Hanya satu app saja yang lengkap. Masalahnya ternyata ada di AndroidManifest.xml

```
<activity android:name=".SplashScreen">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
<activity android:name=".MainActivity">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
<activity android:name=".CreateNoteActivity" />
```

Untuk intent filternya terdoubl pada dua activity. Hanya memakai satu intent filter yang di gunakan pada activity yang pertama kali tampil saja. Jadi hanya di letakan pada activity splash screen saja seperti pada koding dibawah.

```
<activity android:name=".SplashScreen">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
<activity android:name=".MainActivity"/>
<activity android:name=".CreateNoteActivity" />
```

### 3.5.2 Gambar dan url tidak tertampil saat edit

Saat mengedit catatan pada activity, url dan gambar yang telah di inputkan tidak tertampil, tapi masih ada. Masalahnya ada pada koding dibawah

```
if (modelNoteExtra?.imagePath != null &&
modelNoteExtra?.imagePath?.trim()?.isEmpty()!!) {

imageNote.setImageBitmap(BitmapFactory.decodeFile(modelNoteExtra?.imagePath))
    imageNote.visibility = View.VISIBLE
    selectImagePath = modelNoteExtra?.imagePath
    fabDeleteImage.visibility = View.VISIBLE
}
if (modelNoteExtra?.url != null &&
modelNoteExtra?.url?.trim()?.isEmpty()!!) {
    tvUrlNote.text = modelNoteExtra?.url
    tvUrlNote.visibility = View.VISIBLE
    btnHapusUrl.visibility = View.VISIBLE
}
```

Nah pada koding diatas terdapat kesalahan yang mana, jika tidak ada data gambar / url, maka akan menampilkan gambar/url. Ini terbalik seharusnya jika ada data gambar/url maka gambar/url akan ditampilkan. Seperti pada koding dibawah.

```
if (modelNoteExtra?.imagePath != null &&
modelNoteExtra?.imagePath?.trim()?.isEmpty()!!) {

imageNote.setImageBitmap(BitmapFactory.decodeFile(modelNoteExtra?.imagePath))
    imageNote.visibility = View.VISIBLE
    selectImagePath = modelNoteExtra?.imagePath
    fabDeleteImage.visibility = View.VISIBLE
}

if (modelNoteExtra?.url != null &&
modelNoteExtra?.url?.trim()?.isEmpty()!!) {
    tvUrlNote.text = modelNoteExtra?.url
    tvUrlNote.visibility = View.VISIBLE
    btnHapusUrl.visibility = View.VISIBLE
}
```

### 3.5.3 Aplikasi tekeluar

Saat aplikasi dibuild tidak ada masalah, dan diinstal ternyata pas membuka app tekeluar. Saat cek pada logcat. Ternyata terdapat masalah pada activity create note, yang mana saya cek ternyata gara” tema bar bawah yang tidak cocok. Jadi saya mencoba mengganti” dan berhasil dengan tema seperti dibawah.

```
<style name="Theme.NoteKu"
parent="Theme.MaterialComponents.DayNight.NoActionBar">
    <item name="colorPrimary">@color/teal_700</item>
    <item name="colorPrimaryDark">@color/teal_700</item>
    <item name="colorAccent">#01221F</item>
</style>

<style name="AppTheme.AppBarOverlay"
parent="ThemeOverlay.MaterialComponents.Dark.ActionBar" />
```

### 3.5.4 Errorr

Sebelumnya terdapat banyak errorr, seperti:

- Banyak errorr pada activity dikarenakan belum memasukkan Dependencies
- Errorr pada activity seperti private class dan overade
- Dan beberapa errorr kecil yang di selesaikan dengan bantuan errorr pada android studio

## **BAB IV**

### **PENUTUP**

#### **4.1 Kesimpulan**

Dari hasil uji coba aplikasi catatan yang telah dilakukan pada emulator Android yang dijalankan pada emulator dapat disimpulkan bahwa :

- a. Aplikasi catatan dapat berjalan dengan baik pada perangkat dengan sistem operasi Android.
- b. Aplikasi catatan dapat digunakan dengan mudah serta efektif bagi pengguna yang ingin menyimpan catatan mereka secara teratur.
- c. Pengguna dapat mengatur catatan mereka sesuai dengan keinginan karena pengguna sendiri yang akan mengatur dalam membuat catatan baru, mengubah catatan yang lama serta menghapus catatan yang telah dibuatnya.

#### **4.2 Saran**

Saya menyadari bahwa aplikasi catatan ini masih banyak terdapat kekurangan dan berbagai hal yang harus diperbaiki agar kedepannya pengguna akan lebih mudah dalam menggunakan aplikasi catatan ini serta bertambahnya fitur– fitur lain yang akan menambah manfaat untuk pengguna dalam menyimpan catatan mereka.

## **LAMPIRAN**

Referensi aplikasi ini berasal dari link dibawah

<https://rivaldi48.blogspot.com/2019/04/tutorial-membuat-aplikasi-catatan.html>.

Referensi splash screen aplikasi ini berasal dari link dibawah

<https://badoystudio.com/membuat-splash-screen-dengan-mudah-di-android-studio/>.

Dan referensi untuk mengatasi erorr pada aplikasi berasal dari situs dibawah

<https://stackoverflow.com/>.

Dan sisanya berasal dari modul praktikum