

LAPORAN PRAKTIKUM PENGEMBANGAN APLIKASI BERGERAK

Android Beginner 3 - WEEK 5



Disusun oleh:

Nama : Alfath Roziq Widhayaka

Nim : L0122012

Kelas : A

PROGRAM STUDI INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI DAN SAINS DATA

UNIVERSITAS SEBELAS MARET

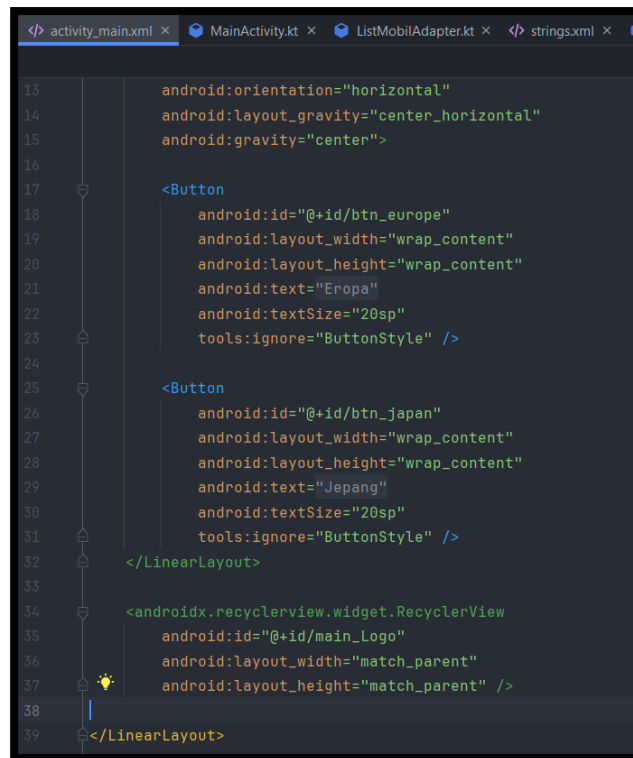
2024

1. Screenshot Source Code

Source code dibawah ini merupakan source code praktikum PAB kelima yang menggunakan **RecyclerView** yang diterapkan di Android Studio. **RecyclerView** hanya membuat objek sesuai dengan ukuran layar. Aplikasi yang menggunakan **RecyclerView** menjadi efisien dan lebih menghemat memori. Saya disini membuat list item layout dengan tema “**Daftar Merk Mobil**”.

A. Langkah-langkah menampilkan data menggunakan RecyclerView

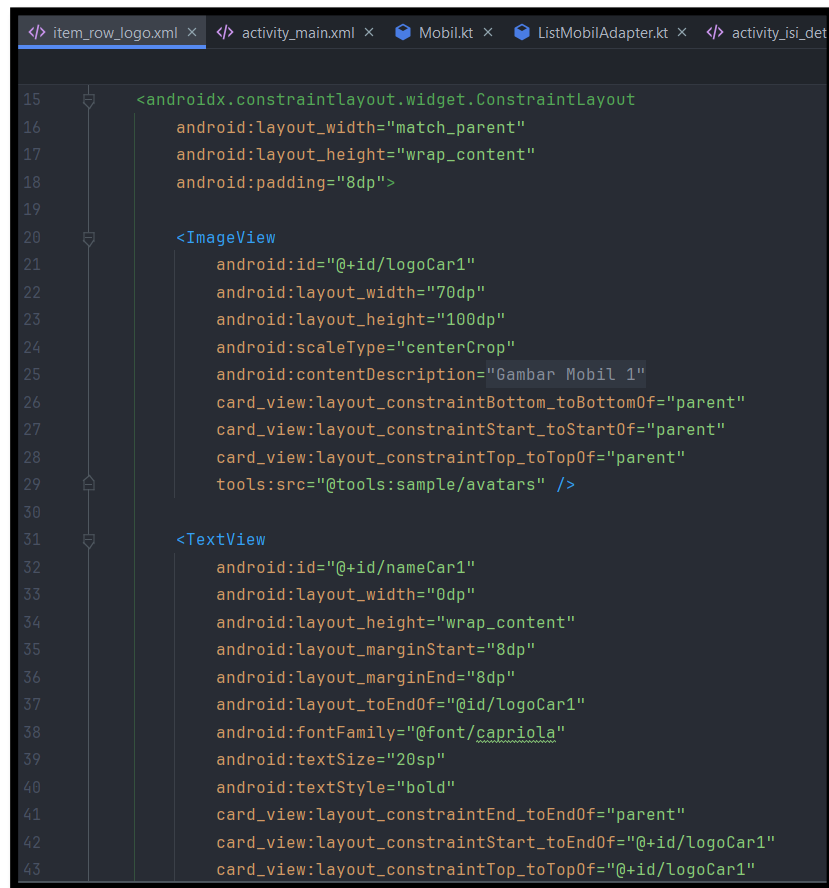
1. Menambahkan RecyclerView ke XML ActivityMain dan menambahkan 2 button sebagai pencarian masing-masing jenis mobil eropa dan mobil jepang.



Gambar 1 sc recyclerview dan 2 button filter

Source code di atas merupakan representasi tata letak XML untuk antarmuka pengguna dalam aplikasi Android pada **activity_main.xml**. Ini menggunakan tata letak linear (LinearLayout) dengan orientasi vertikal untuk menempatkan elemen-elemen di dalamnya secara berurutan dari atas ke bawah. RecyclerView ditambahkan pada kelas **activity_main.xml** yang diidentifikasi oleh ID **@+id/main_Logo**, yang akan menampilkan daftar item dalam tampilan scrollable. RecyclerView ini memiliki lebar dan tinggi yang sama dengan tata letak utama (match_parent), sehingga mengisi seluruh ruang yang tersedia di tata letak linear utama. Pada source code diatas juga menunjukkan penambahan 2 button utama dengan masing-masing id yang berbeda yang digunakan sebagai pencarian mobil Eropa dan Jepang.

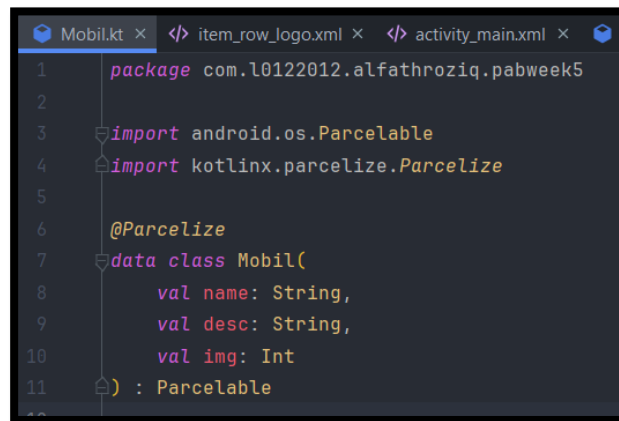
2. Membuat layout XML untuk layout item pada **item_row_logo.xml**.



Gambar 2 sc layout item_row_logo.xml

Source code XML milik **item_row_logo.xml** di atas mendefinisikan tata letak untuk sebuah tampilan kartu (CardView) dalam sebuah aplikasi Android. Kartu tersebut berisi **gambar mobil** (diberi ID logoCar1) yang diletakkan di sebelah kiri atas dengan lebar 70dp dan tinggi 100dp, diikuti oleh **nama mobil** (diberi ID nameCar1) yang diletakkan di sebelah kanan gambar mobil dengan jenis huruf tebal dan ukuran 20sp, dan **deskripsi mobil** (diberi ID tv_item_description) yang diletakkan di bawah nama mobil dengan maksimal 5 baris teks. Semua tampilan tersebut dikelompokkan dalam sebuah layout **ConstraintLayout** yang diatur di dalam CardView dengan sudut melengkung sebesar 4dp. Tampilan tersebut diatur secara responsif dengan lebar yang cocok dengan parent dan tinggi yang mengikuti konten, serta memiliki margin yang ditentukan untuk tata letak yang lebih baik.

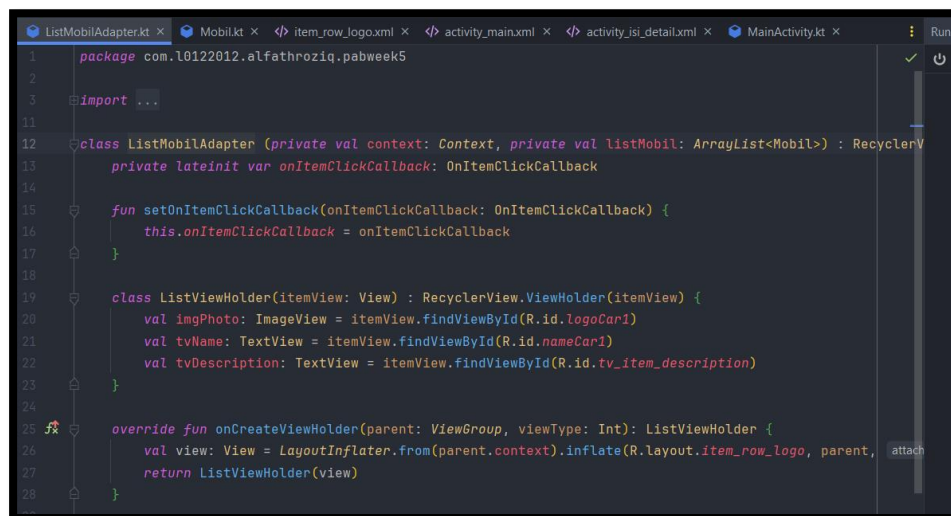
3. Menambah class baru dan menambahkan pewarisan **RecyclerView.Adapter**.



```
1 package com.l0122012.alfathroziq.pabweek5
2
3 import android.os.Parcelable
4 import kotlinx.parcelize.Parcelize
5
6 @Parcelize
7 data class Mobil(
8     val name: String,
9     val desc: String,
10    val img: Int
11 ) : Parcelable
```

Gambar 3 sc class Mobil.kt

Source code di atas adalah definisi dari data class Kotlin bernama **Mobil.kt** yang diimplementasikan dengan parcelable menggunakan anotasi **@Parcelize**. Kelas Mobil memiliki tiga properti yaitu **name** untuk nama mobil, **desc** untuk deskripsi mobil, dan **img** untuk menyimpan referensi gambar mobil dalam bentuk Integer. Menggunakan **Parcelable** memungkinkan objek kelas Mobil dapat dikirimkan antar komponen aplikasi Android dengan mudah dan efisien.



```
1 package com.l0122012.alfathroziq.pabweek5
2
3 import ...
4
11
12 class ListMobilAdapter (private val context: Context, private val listMobil: ArrayList<Mobil>) : RecyclerView
13     private lateinit var onItemClickCallback: OnItemClickListener
14
15 fun setOnItemClickListener(onItemClickListener: OnItemClickListener) {
16     this.onItemClickCallback = onItemClickCallback
17 }
18
19 class ViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView) {
20     val imgPhoto: ImageView = itemView.findViewById(R.id.logoCar1)
21     val tvName: TextView = itemView.findViewById(R.id.nameCar1)
22     val tvDescription: TextView = itemView.findViewById(R.id.tv_item_description)
23 }
24
25 override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ViewHolder {
26     val view: View = LayoutInflater.from(parent.context).inflate(R.layout.item_row_logo, parent,
27     return ViewHolder(view)
28 }
```

Gambar 4 sc class ListMobilAdapter.kt

Source code di atas adalah adapter untuk **RecyclerView** yang bertugas menghubungkan data **listMobil** dengan tampilan item pada RecyclerView. Kelas **ListMobilAdapter.kt** menerima **ArrayList<Mobil>** sebagai parameter konstruktor untuk menampilkan daftar mobil. Setiap item pada RecyclerView ditampilkan menggunakan layout **item_row_logo** tadi. Pada metode **onBindViewHolder**, data dari setiap item dalam listMobil diikat ke tampilan ViewHolder, dan juga menetapkan **OnClickListener** untuk setiap item. Ketika item diklik, Intent dibuat untuk menavigasi ke halaman detail (IsiDetail) dan data mobil yang sesuai dikirimkan melalui Intent. Interface **OnItemClickListener** digunakan untuk menangani klik pada item RecyclerView jika diperlukan.

4. Menambahkan empty activity view untuk fungsi **intent onClick**.

```
11 class IsiDetail : AppCompatActivity() {
12     companion object {
13         const val EXTRA_NAME = "extra_name"
14         const val EXTRA_DESC = "extra_desc"
15         const val EXTRA_IMG = "extra_img"
16     }
17     override fun onCreate(savedInstanceState: Bundle?) {
18         super.onCreate(savedInstanceState)
19         setContentView(R.layout.activity_isi_detail)
20         enableEdgeToEdge()
21
22         ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) { v, insets ->
23             val systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars())
24             v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom)
25             insets ^setOnApplyWindowInsetsListener
26         }
27
28         val imgMenu : ImageView = findViewById(R.id.picture)
29         val tvMenuName : TextView = findViewById(R.id.namePicture)
30         val tvMenuDesc : TextView = findViewById(R.id.descPicture)
31
32         val img = intent.getIntExtra(EXTRA_IMG, defaultValue: 0)
33         val name = intent.getStringExtra(EXTRA_NAME)
34         val desc = intent.getStringExtra(EXTRA_DESC)
35
36         imgMenu.setImageResource(img)
37         tvMenuName.text = name
38         tvMenuDesc.text = desc
39     }
40 }
```

Gambar 5 sc class IsiDetail.kt

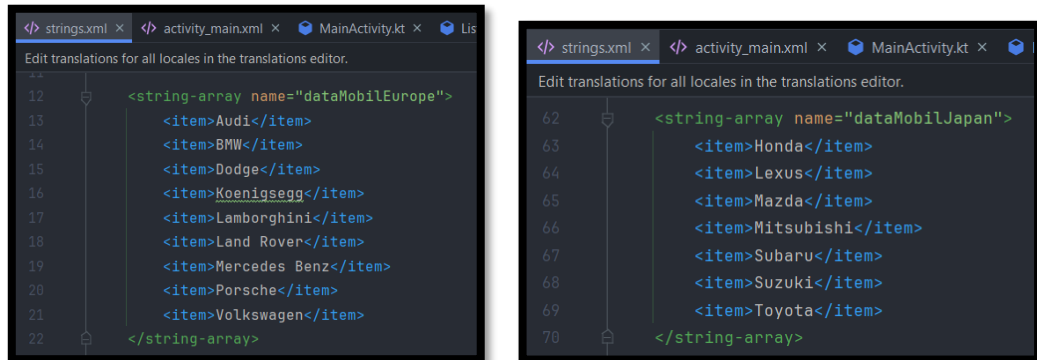
Source code di atas merupakan kelas **IsiDetail.kt** yang berfungsi untuk menampilkan detail dari sebuah item logo mobil yang dipilih oleh pengguna. Pada saat activity dibuat **onCreate()**, layout **activity_isi_detail** diatur sebagai tata letak tampilan activity tersebut. Kegunaan **ImageView** dan **TextView** untuk menampilkan gambar, nama, dan deskripsi mobil diinisialisasi. Data mobil yang dikirimkan melalui Intent dari activity sebelumnya **ListMobilAdapter.kt** diambil dan ditampilkan sesuai dengan komponen tampilan yang sesuai.

```
11 <ImageView
12     android:id="@+id/picture"
13     android:layout_width="match_parent"
14     android:layout_height="350dp"
15     app:layout_constraintTop_toTopOf="parent"
16     app:layout_constraintStart_toStartOf="parent"
17     app:layout_constraintEnd_toEndOf="parent"
18     android:contentDescription="Gambar Mobil"
19     android:layout_margin="8dp" />
20
21 <TextView
22     android:id="@+id/namePicture"
23     android:layout_width="wrap_content"
24     android:layout_height="wrap_content"
25     android:fontFamily="@font/capriola"
26     android:textStyle="bold"
27     android:textSize="30sp"
28     app:layout_constraintStart_toStartOf="@id/picture"
29     app:layout_constraintTop_toBottomOf="@id/picture" />
30
31 <TextView
32     android:id="@+id/descPicture"
33     android:layout_width="wrap_content"
34     android:layout_height="wrap_content"
35     android:fontFamily="@font/poppins"
36     android:textSize="16sp"
37     app:layout_constraintStart_toStartOf="@id/picture"
38     app:layout_constraintTop_toBottomOf="@id/namePicture" />
```

Gambar 6 sc layout activity_isi_detail.xml

Source code diatas merupakan source code **activity_isi_detail.xml** yang digunakan sebagai tata letak komponen yang nantinya akan ditampilkan saat menggunakan fungsi **intent onClick**. Semuanya diatur sedemikian rupa agar konten nya rapi.

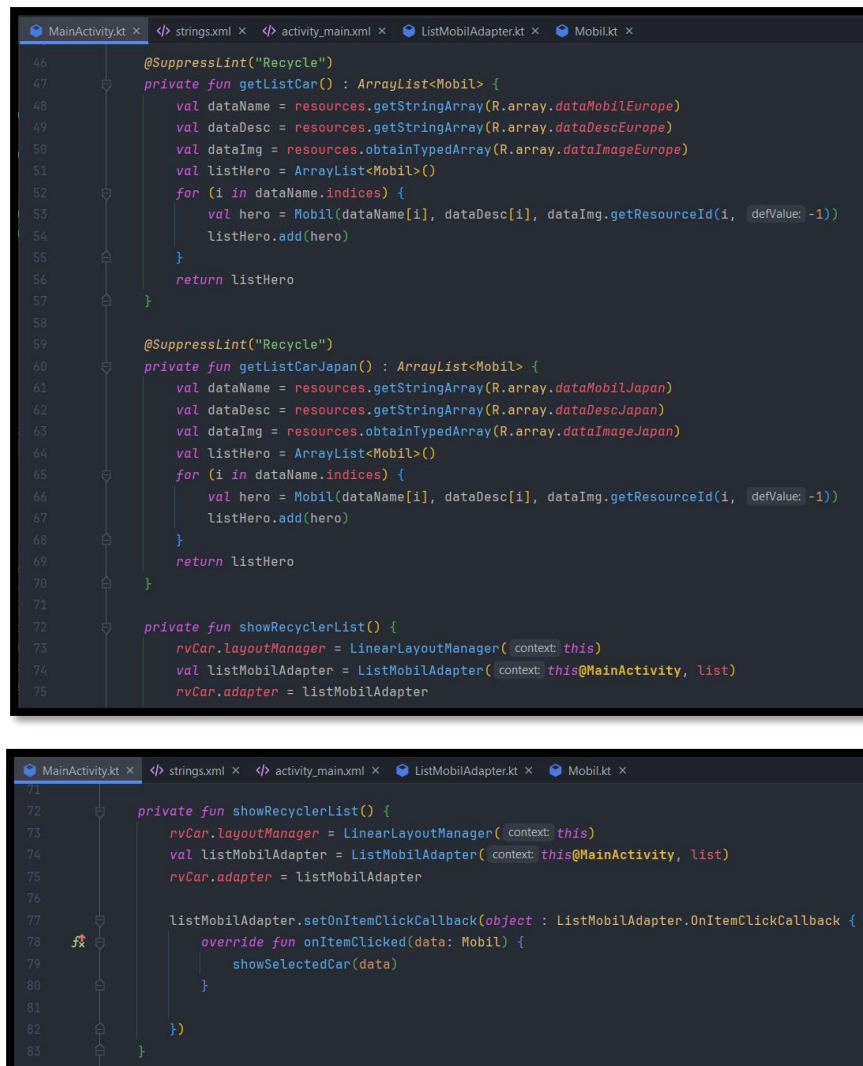
5. Memasukkan data string masing-masing mobil Eropa dan mobil Jepang pada **strings.xml**.



Gambar 7 sc strings.xml

Source code diatas merupakan strings yang digunakan untuk sumber daya dalam aplikasi ini seperti nama mobil, deskripsi mobil, dan gambar mobil. Terdapat dua set array string yang berisi data mobil dari dua wilayah berbeda, yaitu **dataMobilEurope** dan **dataMobilJapan**. Setiap array berisi nama-nama mobil dan deskripsi singkat tentang mobil tersebut. Terdapat dua set array integer yang berisi referensi gambar untuk masing-masing mobil dalam setiap wilayah yaitu **dataImageEurope** dan **dataImageJapan**.

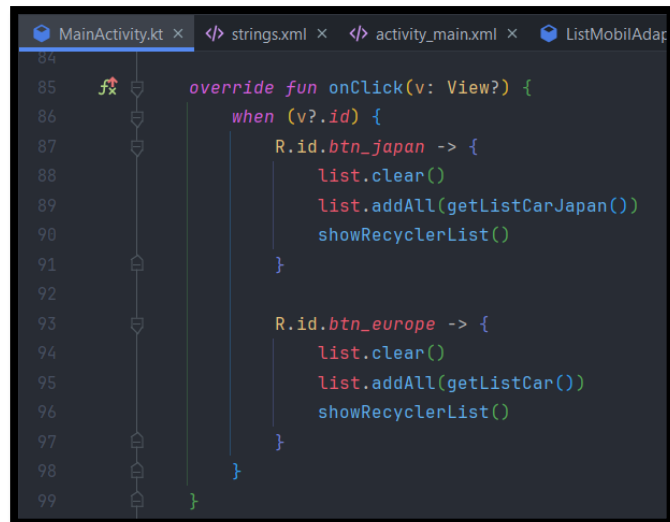
6. Menambahkan RecyclerView dengan adapter dan layoutManager pada **MainActivity.kt** dan menambahkan fungsi **button** agar dapat melakukan aksinya filter masing-masing Eropa dan Jepang.



```
46 @SuppressLint("Recycle")
47 private fun getListCar() : ArrayList<Mobil> {
48     val dataName = resources.getStringArray(R.array.dataMobilEurope)
49     val dataDesc = resources.getStringArray(R.array.dataDescEurope)
50     val dataImg = resources.obtainTypedArray(R.array.dataImageEurope)
51     val listHero = ArrayList<Mobil>()
52     for (i in dataName.indices) {
53         val hero = Mobil(dataName[i], dataDesc[i], dataImg.getResourceId(i, defValue: -1))
54         listHero.add(hero)
55     }
56     return listHero
57 }
58
59 @SuppressLint("Recycle")
60 private fun getListCarJapan() : ArrayList<Mobil> {
61     val dataName = resources.getStringArray(R.array.dataMobilJapan)
62     val dataDesc = resources.getStringArray(R.array.dataDescJapan)
63     val dataImg = resources.obtainTypedArray(R.array.dataImageJapan)
64     val listHero = ArrayList<Mobil>()
65     for (i in dataName.indices) {
66         val hero = Mobil(dataName[i], dataDesc[i], dataImg.getResourceId(i, defValue: -1))
67         listHero.add(hero)
68     }
69     return listHero
70 }
71
72 private fun showRecyclerList() {
73     rvCar.layoutManager = LinearLayoutManager(context: this)
74     val listMobilAdapter = ListMobilAdapter(context: this@MainActivity, list)
75     rvCar.adapter = listMobilAdapter
76
77     listMobilAdapter.setOnItemClickListener(object : ListMobilAdapter.OnItemClickListener {
78         override fun onItemClick(data: Mobil) {
79             showSelectedCar(data)
80         }
81     })
82 }
83 }
```

Gambar 8 sc class MainActivity.kt recyclerview

Source code di atas adalah kelas **MainActivity.kt**. **RecyclerView** ditambahkan pada kelas ini untuk menampilkan daftar mobil yang diinisialisasi dan diatur dengan kelas **ListMobilAdapter.kt**. Daftar mobil diambil dari string array (dataMobil, dataDesc) dan array of image resources (dataImage) menggunakan metode **getListCar()** dan **getListCarJapan** dari **string.xml**. Metode **showRecyclerList()** ditambahkan dan digunakan untuk menampilkan daftar mobil dalam bentuk daftar, dengan setiap item memiliki fungsi klik. Pada method **onCreate()**, RecyclerView diinisialisasi dengan adapter dan **layoutManager** menggunakan pernyataan **rvCar.adapter = listMobilAdapter** dan **rvCar.layoutManager = LinearLayoutManager(this)** untuk menampilkan daftar mobil dalam bentuk daftar atau grid, serta menetapkan fungsi **onClick** untuk setiap item menuju halaman intent melalui **setOnItemClickListener()**.



```
84  
85  override fun onClick(v: View?) {  
86      when (v?.id) {  
87          R.id.btn_japan -> {  
88              list.clear()  
89              list.addAll(getListCarJapan())  
90              showRecyclerList()  
91          }  
92  
93          R.id.btn_europe -> {  
94              list.clear()  
95              list.addAll(getListCar())  
96              showRecyclerList()  
97          }  
98      }  
99  }
```

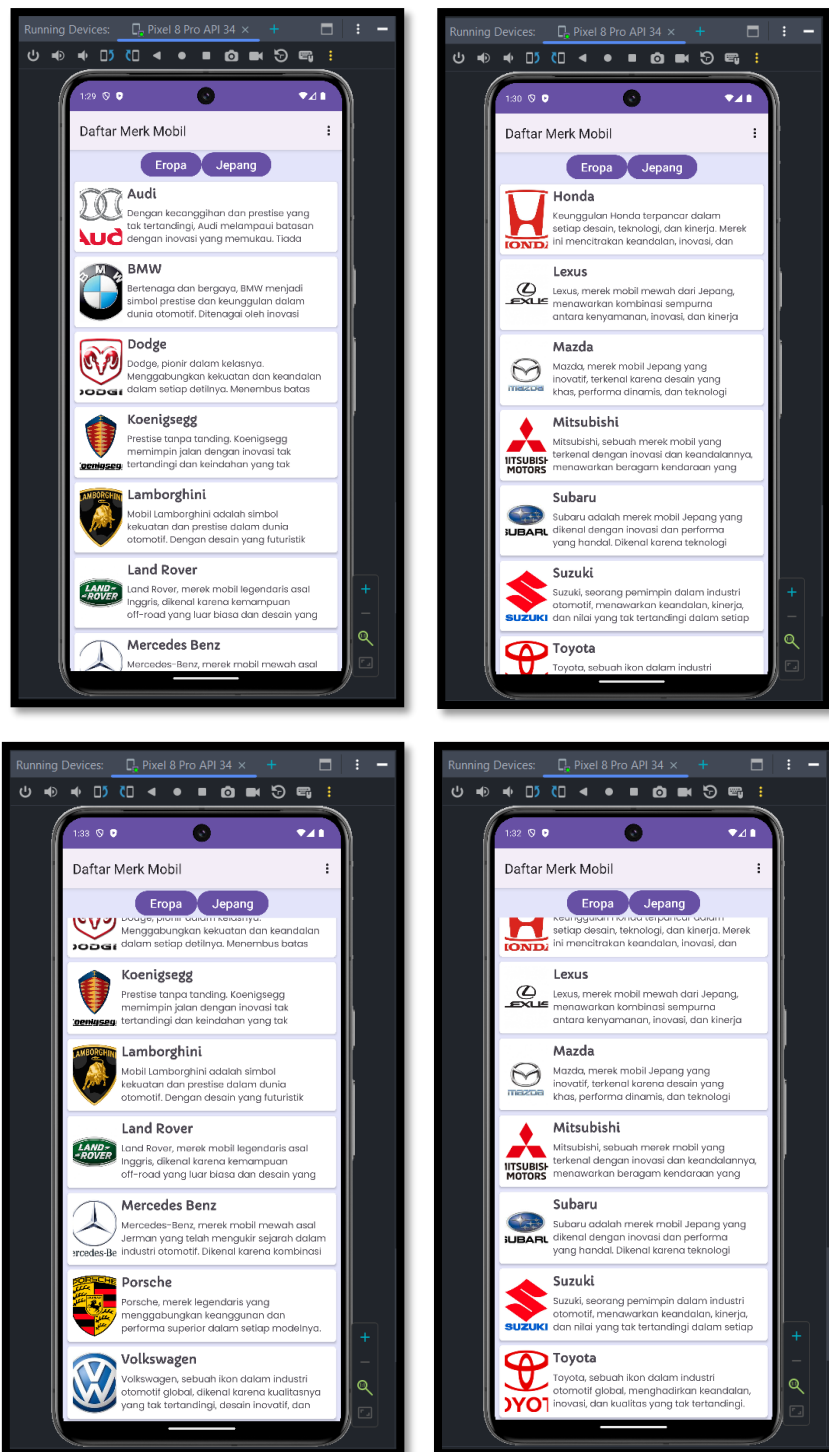
Gambar 9 sc class MainActivity.kt fungsi button

Source code di atas merupakan bagian dari sebuah fungsi **onClick** yang digunakan ketika pengguna melakukan klik pada suatu View. Jika View yang diklik memiliki ID yang sesuai dengan **R.id.btn_japan**, maka yang akan dilakukan adalah menghapus semua elemen yang ada dalam list, menambahkan semua elemen dari list mobil Jepang ke dalam list dan memanggil fungsi **showRecyclerList()** untuk menampilkan daftar mobil dalam bentuk list. Begitu juga sebaliknya pada **R.id.btn_europe**.

2. Screenshot Emulator

Berikut merupakan hasil dari praktikum PAB kelima yang menerapkan materi Recycler View dan fungsi intent onClick untuk membuat interface menjadi lebih menarik. Berikut merupakan hasil pada pekerjaan milik saya dengan tema "Daftar Merk Mobil".

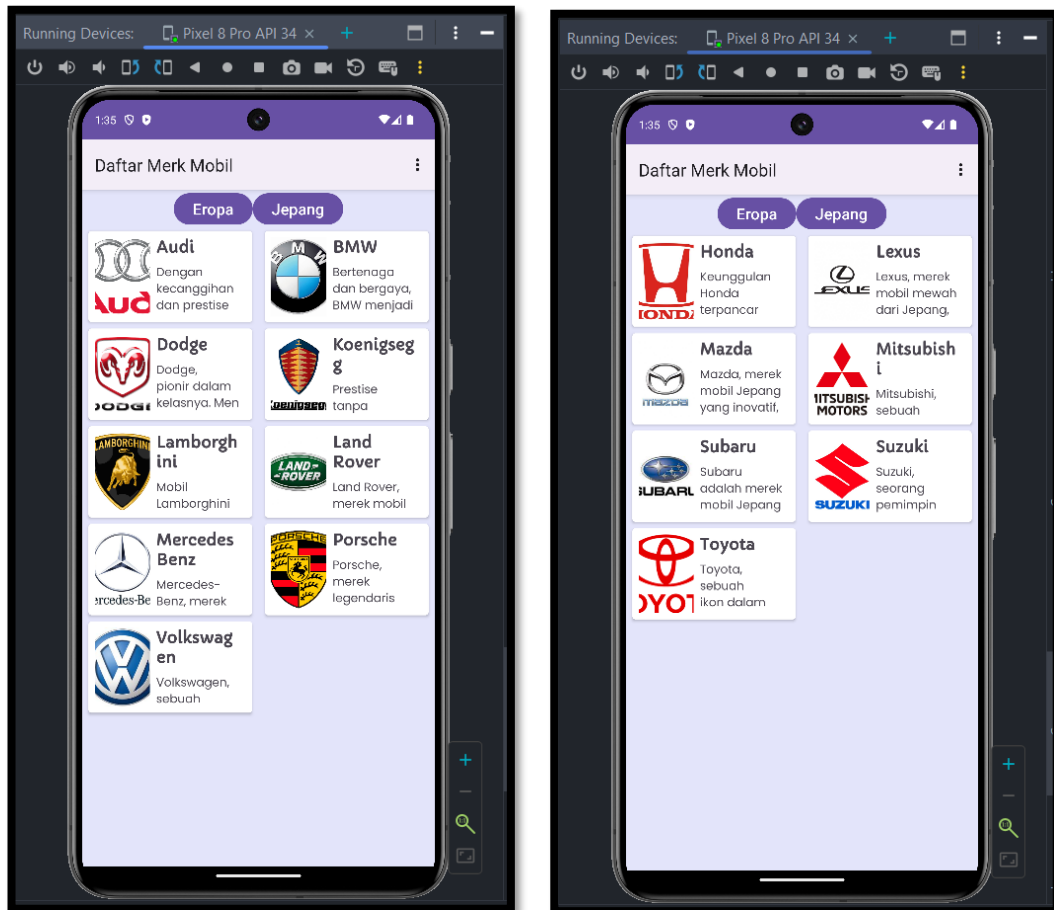
A. List



Gambar 1 hasil list yang di scroll (kiri gambar filter eropa, kanan gambar filter jepang)
Gambar di atas menampilkan hasil dari emulator setelah aplikasi dijalankan. Untuk item mobil eropa terdapat 9 item dan mobil jepang terdapat 6 pada aplikasi, sehingga pengguna

dapat melakukan scrolling ke bawah karena menggunakan fungsi **RecyclerView**. Tata letak setiap item tersusun rapi dan teratur, tidak ada yang terlihat rusak atau terganggu dalam penampilannya. Gambar diatas merupakan susunan menggunakan sistem **List**.

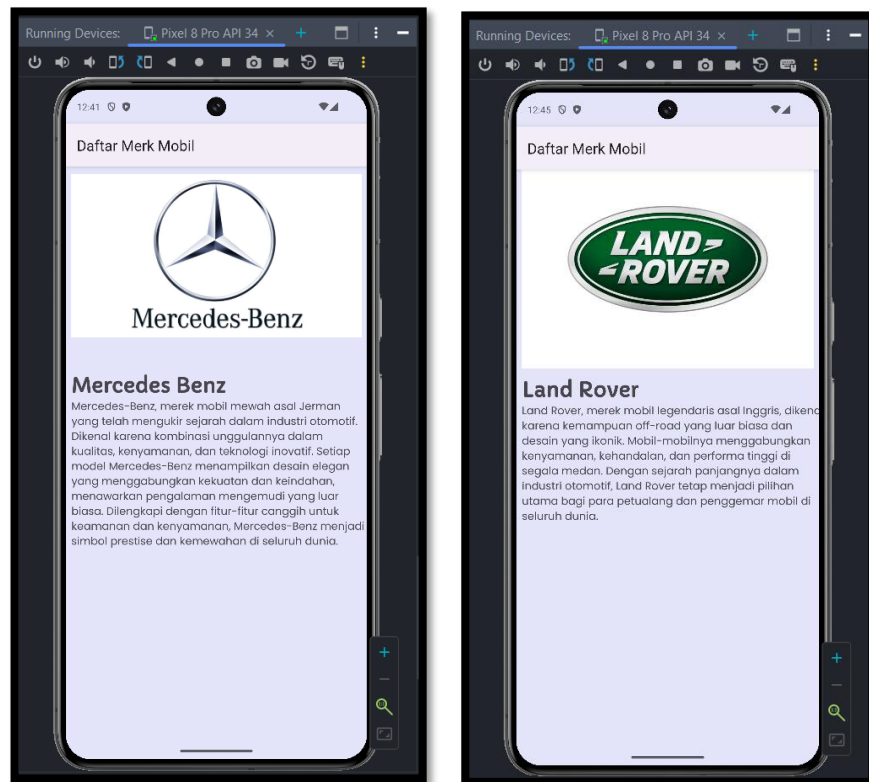
B. Grid



Gambar 2 hasil grid yang di scroll (kiri gambar eropa, kanan gambar jepang)

Tampilan emulator diatas merupakan tampilan setelah menjalankan aplikasi, untuk item mobil eropa terdapat 9 item dan mobil jepang terdapat 6 pada aplikasi. **Fungsi scrolling ke bawah tetap bisa digunakan pada aplikasi ini menggunakan RecyclerView**, namun karena ini mode grid pada masing-masing jenis mobil dan layar emulator pada Pixel 8 pro yang sangat lebar membuat tampilan nya menjadi lebih kecil dengan menampung banyak merk mobil. Setiap elemen disusun secara rapi dan teratur tanpa cacat visual, menunjukkan tata letak yang baik. Penyusunan elemen dilakukan dalam format **grid**, memastikan keteraturan dalam presentasi konten.

C. Ketika di Klik



Gambar 3 hasil onClick pada salah dua item merk mobil

Tampilan emulator diatas merupakan tampilan pada salah dua item untuk membuktikan apakah fungsi **onClick** berjalan dengan sempurna. Terlihat fungsi **onClick** dapat berjalan dengan menggunakan source code diatas.

KESIMPULAN :

Praktikum week 5 ini mengimplementasikan **RecyclerView** dalam pengembangan aplikasi Android untuk menampilkan daftar merk mobil dengan efisien dan hemat memori. Dengan menggunakan RecyclerView, aplikasi dapat menangani daftar item yang besar dengan baik, memungkinkan pengguna untuk melakukan scrolling dengan lancar. Penggunaan **fungsi intent onClick** juga meningkatkan interaktivitas aplikasi dengan memungkinkan pengguna untuk melihat detail item mobil yang dipilih. Selain itu, adanya **fitur filter untuk mobil Eropa dan Jepang** memberikan pengalaman yang lebih personal dan spesifik bagi pengguna dalam menjelajahi daftar mobil. Keseluruhan, praktikum ini **berhasil menggabungkan konsep RecyclerView, fungsi intent onClick, dan fitur filter** dengan baik untuk menciptakan antarmuka pengguna yang menarik dan responsif. Penerapan tata letak yang rapi dan teratur dalam daftar mobil, baik dalam mode list maupun grid, menunjukkan kualitas desain yang baik dalam pengembangan aplikasi. Penggunaan CardView memberikan sentuhan visual yang menarik, sementara fungsi onClick berhasil menghubungkan pengguna langsung ke halaman detail item mobil yang dipilih.