LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 5

ANDROID UI DESIGN

Oleh:

Hari Octavian Delrossi NIM. 2210817210033

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS LAMBUNG MANGKURAT
JUNI 2024

LEMBAR PENGESAHAN
LAPORAN PRAKTIKUM PEMROGRAMAN MOBILE
MODUL 5

Laporan Praktikum Pemrograman Mobile Modul 5: Android UI Design ini disusun sebagai syarat lulus mata kuliah Praktikum Pemrograman Mobile. Laporan Praktikum ini dikerjakan oleh:

Nama Praktikan : Hari Octavian Delrossi
NIM : 2210817210033

Menyetujui, Asisten Praktikum	Mengetahui, Dosen Penanggung Jawab Praktikum
Muhammad Asandy Putra NIM. 2110817310002	Muti'a Maulida, S.Kom., M.T.I. NIP. 198810272019032013

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

SOAL PRAKTIKUM

Buat sebuah aplikasi e-commerce menggunakan Material Design Components (MDC). Aplikasi ini akan menampilkan daftar produk, rincian produk, dan keranjang belanja. User interface untuk halaman utama yang menampilkan daftar produk menggunakan RecyclerView sesuai dengan prinsip Material Design yang akan dibuat. Navigasi antar layar akan ditambahkan menggunakan Navigation Component untuk mengatur navigasi antara halaman utama dan halaman rincian produk. Model data untuk produk, adapter untuk RecyclerView, serta layout untuk item produk juga akan dibuat. Terakhir, ViewModel untuk menyimpan data keranjang belanja akan dibuat dan fungsionalitas untuk menambahkan produk ke keranjang dari halaman rincian produk akan diimplementasikan.

A. Source Code

Tabel 1. Source Code Hasil Jawaban MainActivity.kt

```
1  package com.google.codelabs.mdc.kotlin.shrine
2
3  import android.os.Bundle
4  import androidx.appcompat.app.AppCompatActivity
5  import androidx.fragment.app.Fragment
6
7  class MainActivity : AppCompatActivity(), NavigationHost
   {
8
9      override fun onCreate(savedInstanceState: Bundle?) {
10         super.onCreate(savedInstanceState)
11         setContentView(R.layout.shr_main_activity)
12
13         if (savedInstanceState == null) {
14             supportFragmentManager
15                     .beginTransaction()
16                     .add(R.id.container, LoginFragment())
17                     .commit()
18         }
19     }
20
21     /**
22      * Navigate to the given fragment.
23      *
24      * @param fragment       Fragment to navigate to.
```

```
25      * @param addToBackstack Whether or not the current
    fragment should be added to the backstack.
26      */
27     override    fun    navigateTo(fragment:    Fragment,
    addToBackstack: Boolean) {
28         val transaction = supportFragmentManager
29                 .beginTransaction()
30                 .replace(R.id.container, fragment)
31
32         if (addToBackstack) {
33             transaction.addToBackStack(null)
34         }
35
36         transaction.commit()
37     }
38 }
```

Tabel 2. Source Code Hasil Jawaban LoginFragment.kt

```
1  package com.google.codelabs.mdc.kotlin.shrine
2
3  import android.os.Bundle
4  import android.text.Editable
5  import android.view.LayoutInflater
6  import android.view.View
7  import android.view.ViewGroup
8  import androidx.fragment.app.Fragment
9  import
   kotlinx.android.synthetic.main.shr_login_fragment.
   password_edit_text
10 import
   kotlinx.android.synthetic.main.shr_login_fragment.
   password_text_input
11 import kotlinx.android.synthetic.main.shr_login_
   fragment.view.next_button
12 import
   kotlinx.android.synthetic.main.shr_login_fragment.
   view.password_edit_text
13
14 /**
15  * Fragment representing the login screen for Shrine.
16  */
```

```
17 class LoginFragment : Fragment() {
18
19     override fun onCreateView(
20             inflater:     LayoutInflater,     container:
   ViewGroup?, savedInstanceState: Bundle?): View? {
21         // Inflate the layout for this fragment.
22         val                    view                    =
   inflater.inflate(R.layout.shr_login_fragment, container,
   false)
23
24         // Set an error if the password is less than 8
   characters.
25         view.next_button.setOnClickListener({
26             if
   (!isPasswordValid(password_edit_text.text!!)) {
27                 password_text_input.error              =
   getString(R.string.shr_error_password)
28             } else {
29                 // Clear the error.
30                 password_text_input.error = null
31                 // Navigate to the next Fragment.
32                 (activity                              as
   NavigationHost).navigateTo(ProductGridFragment(), false)
33             }
34         })
35
36         // Clear the error once more than 8 characters
   are typed.
37         view.password_edit_text.setOnKeyListener({ _, _,
   _ ->
38             if
   (isPasswordValid(password_edit_text.text!!)) {
39                 // Clear the error.
40                 password_text_input.error = null
41             }
42             false
43         })
44
45         return view
46     }
47
```

```
48
49      // "isPasswordValid" from "Navigate to the next
    Fragment" section method goes here
50      private fun isPasswordValid(text: Editable?): Boolean
    {
51          return text != null && text.length >= 8
52      }
53  }
```

Tabel 3. Source Code Hasil Jawaban NavigationIconClickListener.kt

```
1   package com.google.codelabs.mdc.kotlin.shrine
2
3   import android.animation.AnimatorSet
4   import android.animation.ObjectAnimator
5   import android.app.Activity
6   import android.content.Context
7   import android.graphics.drawable.Drawable
8   import android.util.DisplayMetrics
9   import android.view.View
10  import android.view.animation.Interpolator
11  import android.widget.ImageView
12
13  /**
14   * [android.view.View.OnClickListener] used to
    translate the product grid sheet downward on
15   * the Y-axis when the navigation icon in the
    toolbar is pressed.
16   */
17  class NavigationIconClickListener @JvmOverloads
    internal constructor(
18          private val context: Context, private val
    sheet: View, private val interpolator: Interpolator? =
    null,
19          private val openIcon: Drawable? = null,
    private val closeIcon: Drawable? = null) :
    View.OnClickListener {
20
21      private val animatorSet = AnimatorSet()
22      private val height: Int
23      private var backdropShown = false
24
```

```
25      init {
26          val displayMetrics = DisplayMetrics()
27          (context                                          as
    Activity).windowManager.defaultDisplay.
    getMetrics(displayMetrics)
28          height = displayMetrics.heightPixels
29      }
30
31      override fun onClick(view: View) {
32          backdropShown = !backdropShown
33
34          // Cancel the existing animations
35          animatorSet.removeAllListeners()
36          animatorSet.end()
37          animatorSet.cancel()
38
39          updateIcon(view)
40
41          val translateY =
     height                                                -
    context.resources.getDimensionPixelSize(R.dimen.
    shr_product_grid_reveal_height)
42
43          val  animator  =  ObjectAnimator.ofFloat(sheet,
    "translationY",  (if  (backdropShown)  translateY  else
    0).toFloat())
44          animator.duration = 500
45          if (interpolator != null) {
46              animator.interpolator = interpolator
47          }
48          animatorSet.play(animator)
49          animator.start()
50      }
51
52      private fun updateIcon(view: View) {
53          if (openIcon != null && closeIcon != null) {
54              if (view !is ImageView) {
                    throw
    IllegalArgumentException("updateIcon() must be called on
55  an ImageView")
                }
```

```
            if (backdropShown) {
56              view.setImageDrawable(closeIcon)
57          } else {
58              view.setImageDrawable(openIcon)
59          }
60      }
61  }
62 }
63
64
```

Tabel 4. Source Code Hasil Jawaban NavigationHost.kt

```
1  package com.google.codelabs.mdc.kotlin.shrine
2
3  import androidx.fragment.app.Fragment
4
5
6  /**
7   * A host (typically an `Activity`} that can display
   fragments and knows how to respond to
8   * navigation events.
9   */
10 interface NavigationHost {
11     /**
12      * Trigger a navigation to the specified fragment,
   optionally adding a transaction to the back
13      * stack to make this navigation reversible.
14      */
15     fun navigateTo(fragment: Fragment, addToBackstack:
   Boolean)
16 }
```

Tabel 5. Source Code Hasil Jawaban ProductCardRecyclerViewAdapter.kt

```
1  package com.google.codelabs.mdc.kotlin.shrine
2
3  import android.view.LayoutInflater
4  import android.view.ViewGroup
5  import androidx.recyclerview.widget.RecyclerView
6  import com.google.codelabs.mdc.kotlin.shrine.network.
   ImageRequester
7  import com.google.codelabs.mdc.kotlin.shrine.network.
```

```
    ProductEntry
8
9   /**
10   * Adapter used to show a simple grid of products.
11   */
12  class ProductCardRecyclerViewAdapter(private val
    productList:            List<ProductEntry>)            :
    RecyclerView.Adapter<ProductCardViewHolder>() {
13
14      override fun onCreateViewHolder(parent: ViewGroup,
    viewType: Int): ProductCardViewHolder {
15          val                  layoutView                  =
    LayoutInflater.from(parent.context).inflate(R.layout.
    shr_product_card, parent, false)
16          return ProductCardViewHolder(layoutView)
17      }
18
19      override         fun         onBindViewHolder(holder:
    ProductCardViewHolder, position: Int) {
20          if (position < productList.size) {
21              val product = productList[position]
22              holder.productTitle.text = product.title
23              holder.productPrice.text = product.price
24              ImageRequester.setImageFromUrl(holder.
    productImage, product.url)
25          }
26      }
27
28      override fun getItemCount(): Int {
29          return productList.size
30      }
31  }
```

Tabel 6. Source Code Hasil Jawaban ProductCardViewHolder.kt

```
1   package com.google.codelabs.mdc.kotlin.shrine
2
3   import android.view.View
4   import android.widget.TextView
5   import androidx.recyclerview.widget.RecyclerView
6   import com.android.volley.toolbox.NetworkImageView
7
```

```
8  class ProductCardViewHolder(itemView: View) :
   RecyclerView.ViewHolder(itemView) {
9
10     var productImage: NetworkImageView =
   itemView.findViewById(R.id.product_image)
11     var productTitle: TextView =
   itemView.findViewById(R.id.product_title)
12     var productPrice: TextView =
   itemView.findViewById(R.id.product_price)
13 }
```

Tabel 7. Source Code Hasil Jawaban ProductGridFragment.kt

```
1  package com.google.codelabs.mdc.kotlin.shrine
2
3  import android.os.Build
4  import android.os.Bundle
5  import android.view.LayoutInflater
6  import android.view.Menu
7  import android.view.MenuInflater
8  import android.view.View
9  import android.view.ViewGroup
10 import android.view.animation.
   AccelerateDecelerateInterpolator
11 import androidx.appcompat.app.AppCompatActivity
12 import androidx.core.content.ContextCompat
13 import androidx.fragment.app.Fragment
14 import androidx.recyclerview.widget.
   GridLayoutManager
15 import androidx.recyclerview.widget.RecyclerView
16 import com.google.codelabs.mdc.kotlin.
   shrine.network.ProductEntry
17 import com.google.codelabs.mdc.kotlin.shrine.
   staggeredgridlayout.
   StaggeredProductCardRecyclerViewAdapter
18 import kotlinx.android.synthetic.main.shr_
   product_grid_fragment.view.*
19
20 class ProductGridFragment : Fragment() {
21
22     override fun onCreate(savedInstanceState: Bundle?) {
23         super.onCreate(savedInstanceState)
```

```
24          setHasOptionsMenu(true)
25      }
26
27      override fun onCreateView(
28              inflater: LayoutInflater, container:
    ViewGroup?, savedInstanceState: Bundle?): View? {
29          // Inflate the layout for this fragment with
    the ProductGrid theme
30          val view = inflater.inflate(R.layout.shr_product_
    grid_fragment, container, false)
31
32          // Set up the toolbar.
33          (activity as AppCompatActivity).
    setSupportActionBar(view.app_bar)
34          view.app_bar.setNavigationOnClickListener(
    NavigationIconClickListener(
35              activity!!,
36              view.product_grid,
37              AccelerateDecelerateInterpolator(),
38              ContextCompat.getDrawable(context!!,
    R.drawable.shr_branded_menu), // Menu open icon
39              ContextCompat.getDrawable(context!!,
    R.drawable.shr_close_menu))) // Menu close icon
40
41          // Set up the RecyclerView
42          view.recycler_view.setHasFixedSize(true)
43          val gridLayoutManager =
    GridLayoutManager(context, 2, RecyclerView.
    HORIZONTAL, false)
44          gridLayoutManager.spanSizeLookup =
    object : GridLayoutManager.SpanSizeLookup() {
45              override fun getSpanSize(position:
    Int): Int {
46                  return if (position % 3 == 2)
    2 else 1
47              }
48          }
49          view.recycler_view.layoutManager
    = gridLayoutManager
50          val adapter =
    StaggeredProductCardRecyclerViewAdapter(
```

```
51              ProductEntry.initProductEntryList
(resources))
52          view.recycler_view.adapter = adapter
53          val largePadding = resources.
getDimensionPixelSize(R.dimen.shr_staggered_product_
grid_spacing_large)
54          val                  smallPadding                =
resources.getDimensionPixelSize(R.dimen.shr_staggered_
product_grid_spacing_small)
55          view.recycler_view.addItemDecoration
(ProductGridItemDecoration(largePadding,
smallPadding))
56
57          // Set cut corner background for API 23+
58          if (Build.VERSION.SDK_INT >= Build.
VERSION_CODES.M) {
59              view.product_grid.background               =
context?.getDrawable(R.drawable.shr_product_
grid_background_shape)
60          }
61
62          return view;
63      }
64
65      override    fun    onCreateOptionsMenu(menu:    Menu,
menuInflater: MenuInflater) {
66          menuInflater.inflate(R.menu.shr_toolbar_menu,
menu)
67          super.onCreateOptionsMenu(menu, menuInflater)
68      }
69  }
```

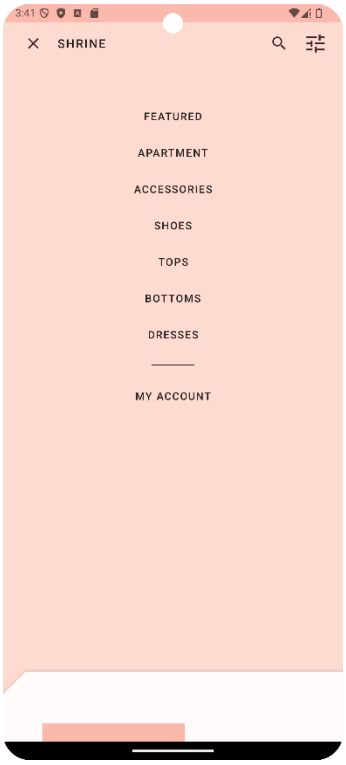
Tabel 8. Source Code Hasil Jawaban activity_main.xml

```
1   package com.google.codelabs.mdc.kotlin.shrine
2
    import android.graphics.Rect
3   import android.view.View
4   import androidx.recyclerview.widget.RecyclerView
5
6   /**
7
```

```
 * Custom item decoration for a vertical
8 [ProductGridFragment] [RecyclerView]. Adds a
 * small amount of padding to the left of grid items, and
9 a large amount of padding to the right.
10  */
11 class ProductGridItemDecoration(private val largePadding:
   Int, private val smallPadding: Int) :
12 RecyclerView.ItemDecoration() {
13
14     override fun getItemOffsets(outRect: Rect, view: View,
                                   parent: RecyclerView,
15 state: RecyclerView.State) {
16         outRect.left = smallPadding
17         outRect.right = smallPadding
18     }
   }
```

B. Output Program

Gambar 1. Screenshot Halaman Utama



Gambar 2. Screenshot Backdrop

C. Pembahasan

- MainActivity.kt

Mengimplementasikan NavigationHost untuk mengelola navigasi antar fragmen. Dalam metode onCreate, layout shr_main_activity diatur sebagai tampilan konten, dan jika savedInstanceState adalah null, LoginFragment ditambahkan ke R.id.container menggunakan supportFragmentManager. Metode navigateTo diimplementasikan untuk memungkinkan navigasi ke fragmen yang ditentukan, dengan pilihan untuk menambahkan transaksi ke backstack. Jika addToBackstack bernilai true, fragmen saat ini ditambahkan ke backstack sebelum mengganti fragmen di R.id.container dengan fragmen baru, dan kemudian transaksi tersebut dikomit.

- LoginFragment.kt

Merupakan fragmen yang mewakili layar login untuk aplikasi Shrine. Dalam metode onCreateView, layout shr_login_fragment di-inflate sebagai tampilan fragmen. Ketika tombol next_button diklik, sistem memeriksa apakah password yang dimasukkan valid (minimal 8 karakter) menggunakan metode isPasswordValid. Jika password tidak valid, sebuah error ditampilkan pada password_text_input; jika valid, error dihapus dan aplikasi

menavigasi ke ProductGridFragment menggunakan metode navigateTo dari NavigationHost. Selain itu, jika pengguna mengetik lebih dari 8 karakter dalam password_edit_text, error akan dihapus secara otomatis.

- NavigationIconClickListener.kt

Mengimplementasikan View.OnClickListener untuk mengelola interaksi saat ikon navigasi di toolbar ditekan, yang akan menggerakkan "product grid sheet" ke bawah pada sumbu Y. Kelas ini mengambil konteks, tampilan lembaran (sheet), interpolator opsional, dan ikon untuk tampilan terbuka dan tertutup sebagai parameter. Pada inisialisasi, tinggi layar diambil untuk menghitung posisi terjemahan Y.

Ketika metode onClick dipanggil, status backdropShown dibalik, animasi yang ada dibatalkan, dan ikon diperbarui. Objek ObjectAnimator digunakan untuk menggerakkan lembaran (sheet) ke posisi yang sesuai, berdasarkan apakah lembaran tersebut ditampilkan atau disembunyikan. Animasi berlangsung selama 500 milidetik, dan interpolator diterapkan jika disediakan. Metode updateIcon digunakan untuk mengganti ikon pada ImageView berdasarkan status backdropShown.

Secara keseluruhan, kelas ini mengelola animasi transisi untuk lembar produk dan ikon navigasi, memberikan pengalaman pengguna yang dinamis dan responsif.

- NavigationHost.kt

NavigationHost diimplementasikan oleh sebuah Activity untuk menampilkan fragmen dan menangani event navigasi. Antarmuka ini memiliki satu metode, navigateTo, yang menerima dua parameter: fragment dari tipe Fragment yang menentukan fragmen tujuan navigasi, dan addToBackstack dari tipe Boolean yang menentukan apakah transaksi navigasi tersebut harus ditambahkan ke backstack agar dapat dibalik. Dengan kata lain, antarmuka ini digunakan untuk mengabstraksi logika navigasi antar fragmen di dalam aplikasi, memungkinkan Activity yang mengimplementasikannya untuk menavigasi ke fragmen tertentu dan mengelola backstack secara opsional.

- ProductCardRecyclerViewAdapter.kt

Sebuah adapter untuk RecyclerView yang menampilkan grid produk. Adapter ini menerima daftar produk (productList) dan menggunakan ProductCardViewHolder untuk mengelola tampilan setiap item dalam daftar. Metode onCreateViewHolder digunakan untuk meng-

inflate layout shr_product_card dan mengembalikan instance ProductCardViewHolder, sementara metode onBindViewHolder mengikat data dari ProductEntry ke view holder berdasarkan posisi, mengatur judul, harga, dan gambar produk menggunakan ImageRequester. Metode getItemCount mengembalikan jumlah total item dalam daftar produk, memungkinkan RecyclerView mengetahui berapa banyak item yang harus ditampilkan. Dengan demikian, ProductCardRecyclerViewAdapter mengatur data produk dan menghubungkannya dengan tampilan yang sesuai dalam grid.

- ProductCardViewHolder.kt

Sebuah kelas yang memperluas RecyclerView.ViewHolder dan digunakan untuk mengelola tampilan item dalam RecyclerView yang menampilkan produk-produk. Kelas ini memiliki tiga properti: productImage, productTitle, dan productPrice. Properti productImage adalah NetworkImageView yang digunakan untuk menampilkan gambar produk yang diambil dari jaringan, sedangkan productTitle dan productPrice adalah TextView yang digunakan untuk menampilkan judul dan harga produk. Pada inisialisasi, findViewById digunakan untuk menghubungkan tampilan dari itemView dengan properti kelas ini, memastikan bahwa setiap item dalam RecyclerView memiliki referensi ke elemen UI yang sesuai untuk menampilkan data produk.

- ProductGridFragment.kt

Sebuah fragmen yang menampilkan grid produk menggunakan RecyclerView. Pada metode onCreate, fragmen disiapkan untuk memiliki menu opsional. Pada metode onCreateView, layout shr_product_grid_fragment di-inflate dan toolbar diatur menggunakan setSupportActionBar. NavigationIconClickListener dihubungkan dengan app_bar untuk mengatur ikon navigasi. RecyclerView diatur dengan GridLayoutManager untuk menampilkan item dalam dua kolom secara horizontal, dengan penyesuaian untuk ukuran span berdasarkan posisi item. Adapter StaggeredProductCardRecyclerViewAdapter dihubungkan ke RecyclerView, yang diisi dengan daftar produk yang diinisialisasi dari sumber daya. Padding besar dan kecil ditambahkan sebagai dekorasi item dalam RecyclerView. Untuk perangkat dengan API 23 atau lebih tinggi, latar belakang sudut dipotong diterapkan pada grid produk. Metode onCreateOptionsMenu meng-inflate menu toolbar dari shr_toolbar_menu. Secara keseluruhan, fragmen ini mengelola tampilan grid produk yang kaya dengan penyesuaian tata letak dan interaksi pengguna.

- ProductGridItemDecoration.kt

merupakan custom item decoration untuk RecyclerView dalam ProductGridFragment. Kelas ini memperluas RecyclerView.ItemDecoration dan digunakan untuk menambahkan padding khusus ke item-item dalam grid.

Konstruktor kelas ini menerima dua parameter: largePadding dan smallPadding, yang merupakan jumlah padding besar dan kecil yang akan diterapkan pada item-item dalam grid.

Metode getItemOffsets di-override untuk mengatur padding pada item-item dalam RecyclerView. Parameter outRect adalah objek Rect yang digunakan untuk mengatur offset padding untuk item. Padding kecil (smallPadding) diterapkan ke kiri dan kanan setiap item di grid.

Dengan demikian, ProductGridItemDecoration menyediakan cara untuk menambahkan ruang (padding) di sekitar item-item dalam grid produk, membuat tata letak lebih rapi dan teratur.

TAUTAN GIT

Praktikum-Pemrograman-Mobile/Praktikum Modul 5 at main · harioct/Praktikum-Pemrograman-Mobile (github.com)