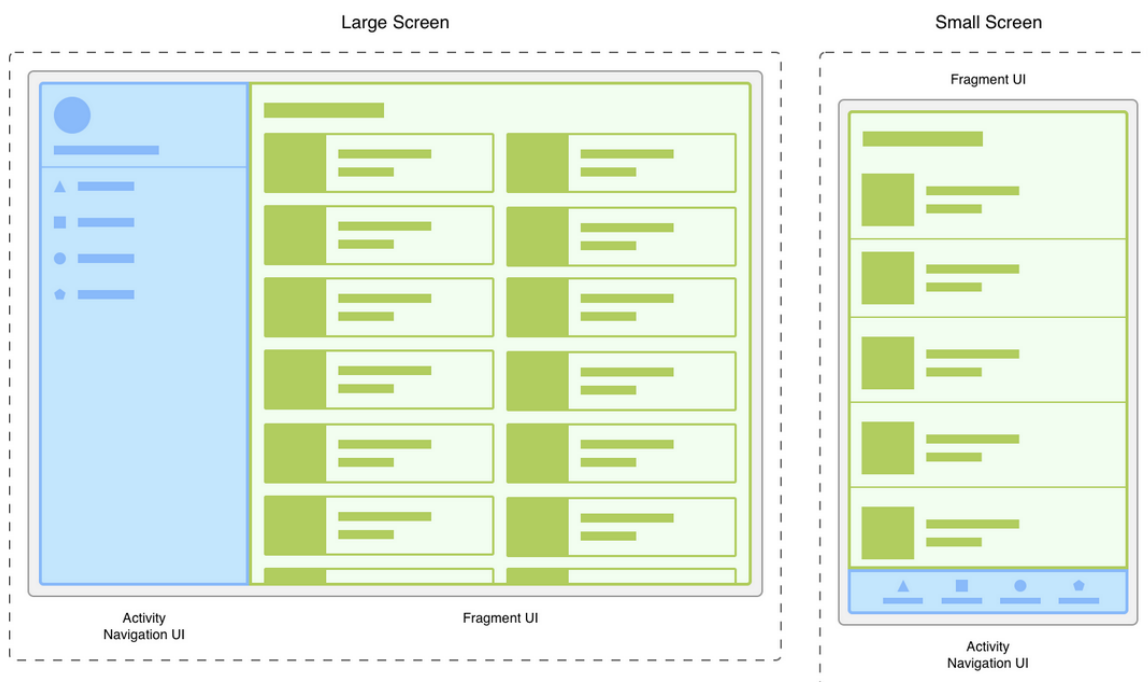


## Pertemuan 5

### Fragment (Tab Layout)

Sebuah fragment pada android mewakili bagian UI yang dapat digunakan kembali dari sebuah aplikasi. Sebuah fragment itu mengelola sendiri *layout* nya, memiliki *lifecycle*, dan menangani *input* secara langsung. Meski begitu, sebuah fragment tidak dapat berdiri sendiri, sehingga harus melekat pada *activity* atau fragment lainnya. Oleh sebab itu, secara hirarki, sebuah fragment adalah bagian dari tampilan induknya. Dengan Fragment, maka pengembang dimudahkan dengan adanya system modularitas dan penggunaan kembali UI dengan membagi menjadi potongan-potongan terpisah (fragmen). Jika sebuah *activity* merupakan fungsi yang baik untuk mengatur halaman UI dengan elemen global, maka fragmen lebih baik digunakan untuk mengatur spesifik sebuah atau sebagian layar UI.

Sebagai contoh ketika aplikasi yang akan dikembangkan harus mampu untuk beradaptasi pada semua ukuran layar perangkat. Jika menggunakan sebuah *activity*, maka proses pengelolaan seluruh perbedaan seperti *Gambar 1* menjadi berat. Memisahkan elemen navigasi dengan konten membuat proses kelola menjadi lebih mudah. Activity berperan dalam pengelolaan navigasi UI yang tepat, sementara proses menampilkan konten dan pengaturannya akan dikelola oleh fragmen.



Gambar 1. Dua ukuran layar berbeda pada perangkat (\*sumber : <https://developer.android.com/guide/fragments>)

Membagi UI menjadi beberapa fragmen memudahkan untuk mengubah tampilan aktivitas pada saat runtime. Saat *activity* sedang dalam tahap mulai, maka fragmen dapat ditambahkan, diganti, atau dihapus. Kita dapat menyimpan catatan perubahan ini di back-stack yang dikelola oleh *activity*, yang memungkinkan perubahan tersebut dikembalikan seperti semula. Pengembang dapat menggunakan beberapa instance dari kelas fragmen yang sama dalam *activity* yang sama, dalam beberapa *activity*,

atau bahkan sebagai turunan dari fragmen lain. Dengan begitu, sebaiknya kita membuat sebuah fragmen dengan *logic* yang spesifik untuk mengelola UI nya sendiri.

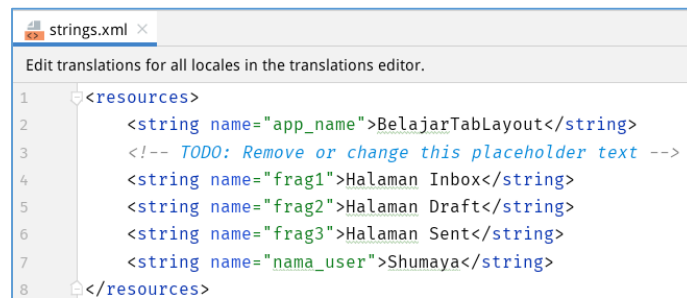
Pada contoh kali ini kita akan menggunakan **Fragment** dengan implementasi **PagerAdapter**. Implementasi dari PagerAdapter tersebut merepresentasikan setiap halaman sebagai sebuah Fragment yang dikelola oleh **Fragment Manager**. Kejadian ini akan selalu dikelola oleh Manager selama pengguna menggunakan atau kembali ke halaman tersebut. Fragment pager jenis ini lebih baik digunakan jika kita sudah mengetahui jumlah halaman yang ingin digunakan (statis). Oleh sebab itu lebih cocok dipakai ketika ingin membuat tab. Sebab, fragment yang dikunjungi oleh pengguna akan disimpan ke dalam memori. Sementara untuk bagian/tampilan yang tidak terlihat akan dimusnahkan. Hal ini akan berdampak kepada penggunaan memori menjadi lebih besar karena fragment dapat menyimpan status yang berubah-ubah. Untuk kondisi tersebut sebaiknya gunakan **FragmentStatePagerAdapter**. (\*sumber : <https://developer.android.com/guide/fragments>)

Pada percobaan kali ini kita akan mencoba membuat tampilan menu Tab. Menu akan terletak di sebuah halaman dimana proses navigasi akan menggunakan Tab yang dapat di klik dan menampilkan halaman yang berbeda. Tampilan seperti ini dapat membuat aplikasi terasa lebih ringkas karena tidak perlu menyediakan ruang yang besar untuk menampilkan menu. Silahkan buat sebuah proyek android baru, kemudian beri nama **"BelajarTabLayout"**. Setelah proyek terbentuk dengan sempurna, silahkan buka file **build.gradle(Module:\_\_\_)** untuk menambahkan beberapa librari kedalam proyek terbut. Silahkan tambahkan beberapa dependencies yang belum ada seperti pada tampilan di bawah ini. Setelahnya, silahkan lakukan proses sinkronisasi terlebih dahulu sebelum melanjutkan pembuatan proyek.

```
24 buildFeatures{
25     viewBinding true
26 }
27
28
29 dependencies {
30     implementation fileTree(dir: "libs", include: ["*.jar"])
31     implementation "org.jetbrains.kotlin:kotlin-stdlib:1.7.10"
32     implementation 'androidx.core:core-ktx:1.9.0'
33     implementation 'androidx.appcompat:appcompat:1.5.1'
34     implementation 'androidx.constraintlayout:constraintlayout:2.1.4'
35     implementation 'com.google.android.material:material:1.6.1'
36     implementation 'androidx.legacy:legacy-support-v4:1.0.0'
37     implementation 'androidx.navigation:navigation-fragment-ktx:2.5.2'
38     implementation 'androidx.navigation:navigation-ui-ktx:2.5.2'
39     implementation 'de.hdodenhof:circleimageview:3.1.0'
40     testImplementation 'junit:junit:4.13.2'
41     androidTestImplementation 'androidx.test.ext:junit:1.1.3'
42     androidTestImplementation 'androidx.test.espresso:espresso-core:3.4.0'
43 }
```

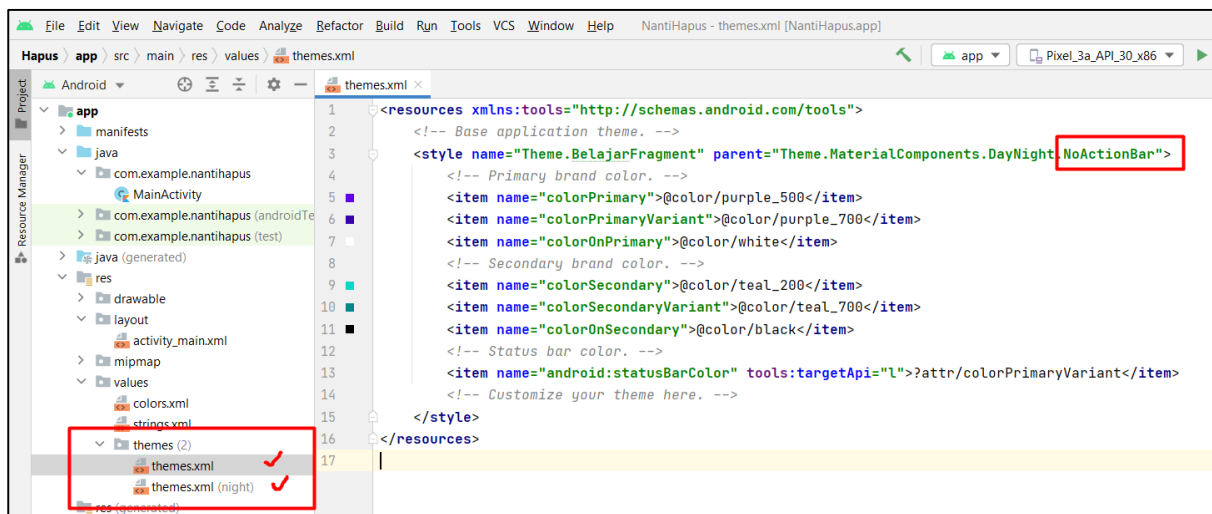
Gambar 2. Dependencies pada build.gradle

Langkah selanjutnya adalah mendaftarkan beberapa jenis teks yang akan kita gunakan pada aplikasi di file **strings.xml** yang terletak di dalam packages **values**.



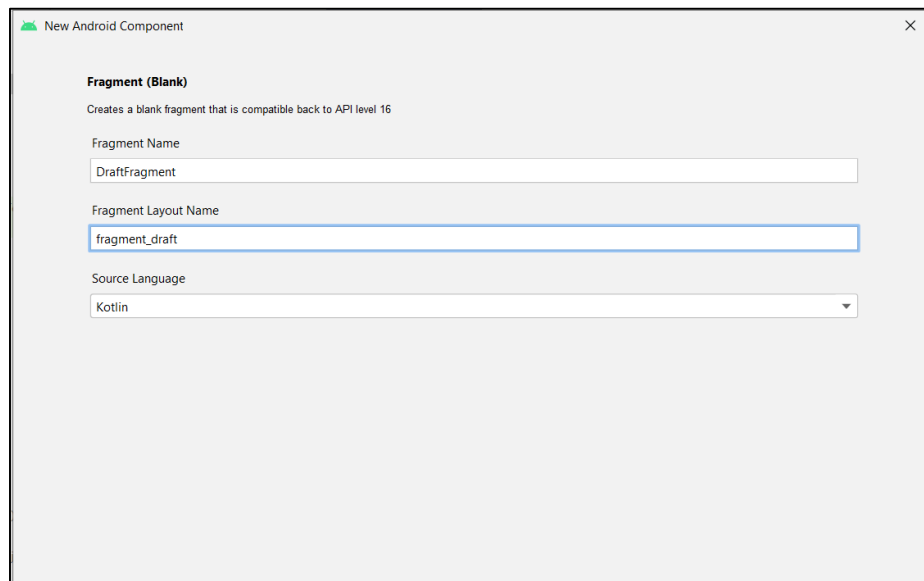
Gambar 3. Halaman string.xml

Kemudian silahkan ubah sedikit themes seperti pada *screenshot* di bawah ini.



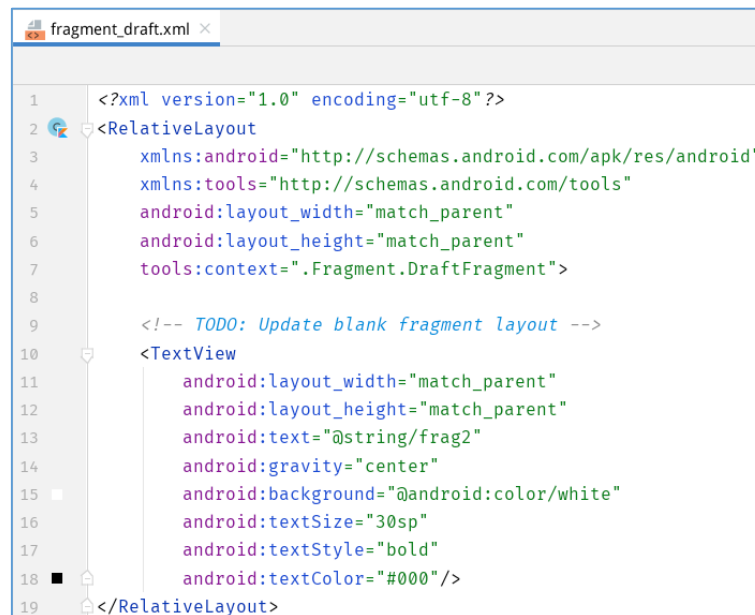
Gambar 4. Halaman themes.xml

Setelah itu, kita akan membuat tiga buah fragment baru. Ketika kita membuat fragment, maka kita tidak perlu membuat file kotlin secara terpisah. Caranya adalah dengan klik **kanan** pada **packages Layout** → **New** → **Fragment** → **Fragment (Blank)**. Lakukan hal ini sebanyak tiga kali untuk membuat tiga buah fragment. Beri nama masing-masing fragmentnya sebagai berikut **DraftFragment**, **InboxFragment**, **SentFragment**.



Gambar 5. Masukkan nama Fragment

Setelah anda berhasil membuat tiga buah file fragment, maka pada hirarki projek akan terlihat **tiga class kotlin** didalam package **Java** dan **tiga buah layout fragment** pada package **Layout**. Silahkan buka setiap file layout tersebut, kemudian tambahkan sebuah **TextView** untuk membedakan setiap halaman. Misal : **fragment\_draft.xml** berisi TextView bertuliskan "**Halaman Draft**". Lihat contoh dibawah. Lakukan hal tersebut pada setiap halaman fragment pada package Layout.



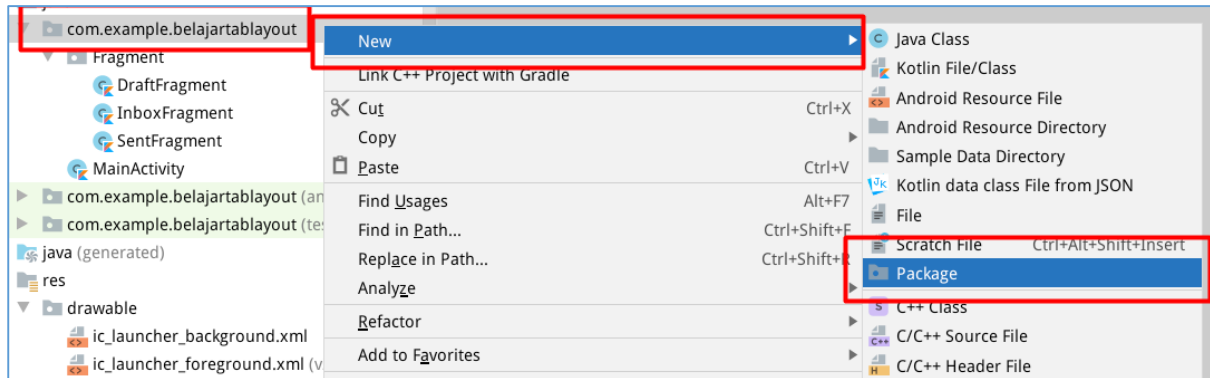
Gambar 6. Halaman fragment\_draft.xml

Setelah selesai, silahkan cari **gambar atau icon** profil user dari internet. Kemudian pastikan nama file tersebut tidak mengandung spasi. Lalu silahkan masukkan gambar atau icon tersebut ke dalam package **Drawable**. Seperti di contoh, beri nama file "**profile**" (baris kode 28). Kemudian buka file **activity\_main.xml** lalu buat kode program seperti berikut ini.

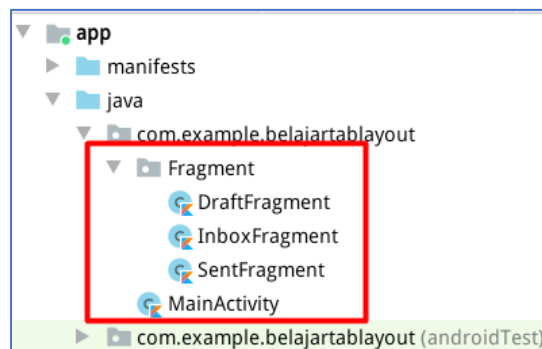
```
activity_main.xml x
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:app="http://schemas.android.com/apk/res-auto"
5      xmlns:tools="http://schemas.android.com/tools"
6      android:layout_width="match_parent"
7      android:layout_height="match_parent"
8      android:orientation="vertical"
9      tools:context=".MainActivity">
10
11      <com.google.android.material.appbar.AppBarLayout
12          android:layout_width="match_parent"
13          android:layout_height="wrap_content">
14
15          <androidx.appcompat.widget.Toolbar
16              android:id="@+id/toolbar_awa1"
17              android:layout_width="match_parent"
18              android:layout_height="wrap_content"
19              android:background="@color/design_default_color_on_secondary"
20              app:popupTheme="@style/Theme.BelajarFragment">
21
22              <de.hdodenhof.circleimageview.CircleImageView
23                  android:id="@+id/foto_profile"
24                  android:layout_width="50dp"
25                  android:layout_height="50dp"
26                  android:layout_marginTop="8dp"
27                  android:layout_marginBottom="8dp"
28                  android:src="@drawable/profile"
29                  android:tint="@android:color/white" />
30
31              <TextView
32                  android:id="@+id/pengguna"
33                  android:layout_width="wrap_content"
34                  android:layout_height="wrap_content"
35                  android:layout_marginStart="24dp"
36                  android:text="Shumaya"
37                  android:textColor="@android:color/white"
38                  android:textSize="15sp" />
39
40          </androidx.appcompat.widget.Toolbar>
41
42          <com.google.android.material.tabs.TabLayout
43              android:id="@+id/tab_layout"
44              android:layout_width="match_parent"
45              android:layout_height="wrap_content"
46              android:background="@android:color/ho1o_blue_dark"
47              app:tabIndicatorColor="@android:color/white" />
48
49      </com.google.android.material.appbar.AppBarLayout>
50
51      <androidx.viewpager.widget.ViewPager
52          android:id="@+id/view_pager"
53          android:layout_width="match_parent"
54          android:layout_height="match_parent" />
55
56  </LinearLayout>
```

Gambar 7. Halaman activity\_main.xml

Kita selesai melakukan modifikasi pada halaman Layout. Selanjutnya, agar susunan file Kotlin semakin rapi, maka silahkan buat sebuah package baru bernama "**Fragment**". Kemudian, klik dan tarik tiga buah file Fragment yang telah terbentuk untuk dimasukkan ke dalam package Fragment tersebut. Silahkan perhatikan susunan hirarki proyek pada *Gambar 9* dibawah ini.

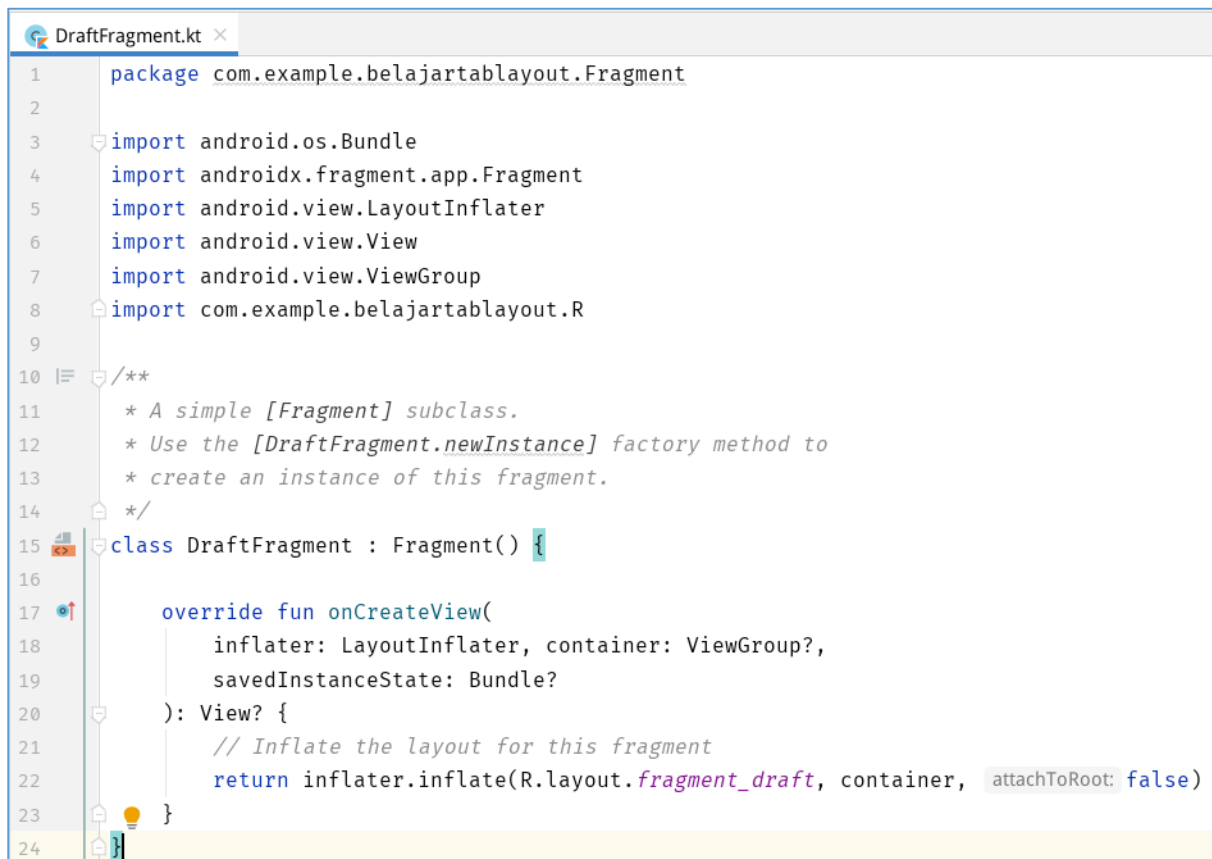


Gambar 8. Membuat sebuah package "Fragment"



Gambar 9. Hirarki folder dari proyek BelajarTabLayout

Selanjutnya buka halaman **DraftFragment.kt**, kemudian silahkan hapus kode program yang tidak terpakai. Berikut kode program yang digunakan dalam proyek Tab Layout. Lakukan hal yang sama pada file **InboxFragment** dan **SentFragment**.



```
1 package com.example.belajartablayout.Fragment
2
3 import android.os.Bundle
4 import androidx.fragment.app.Fragment
5 import android.view.LayoutInflater
6 import android.view.View
7 import android.view.ViewGroup
8 import com.example.belajartablayout.R
9
10 /**
11  * A simple [Fragment] subclass.
12  * Use the [DraftFragment.newInstance] factory method to
13  * create an instance of this fragment.
14  */
15 class DraftFragment : Fragment() {
16
17     override fun onCreateView(
18         inflater: LayoutInflater, container: ViewGroup?,
19         savedInstanceState: Bundle?
20     ): View? {
21         // Inflate the layout for this fragment
22         return inflater.inflate(R.layout.fragment_draft, container, attachToRoot: false)
23     }
24 }
```

Gambar 10. Halaman DraftFragment.kt

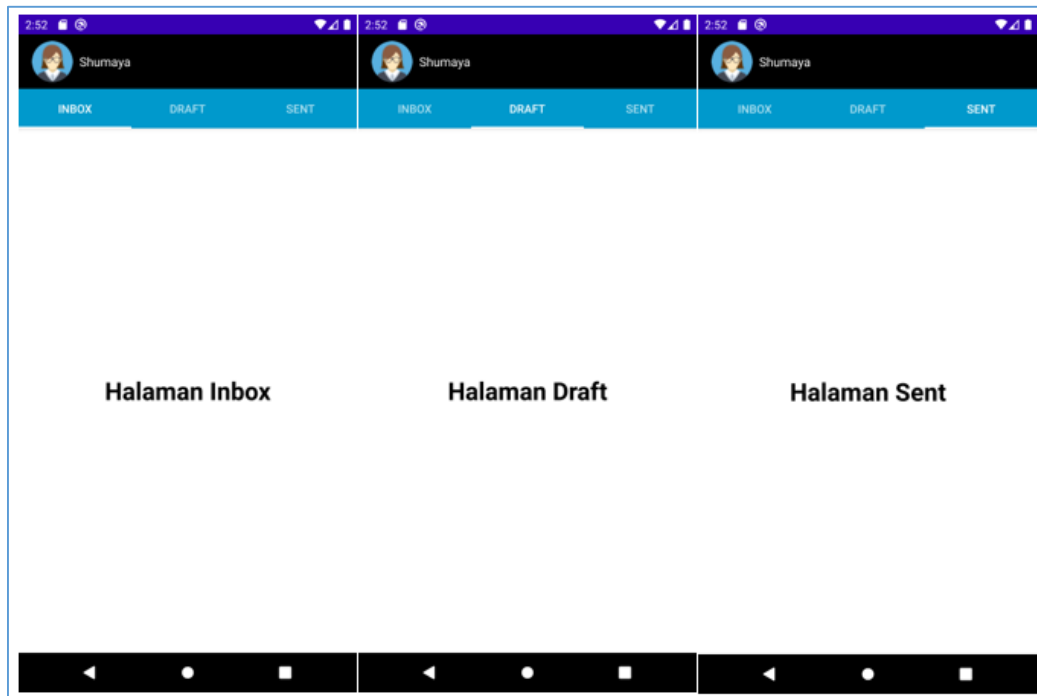
Langkah terakhir adalah memanggil ketiga file Fragment tersebut melalui class MainActivity.kt.

```
MainActivity.kt
3  import androidx.appcompat.app.AppCompatActivity
4  import android.os.Bundle
5  import androidx.appcompat.widget.Toolbar
6  import androidx.fragment.app.Fragment
7  import androidx.fragment.app.FragmentManager
8  import androidx.fragment.app.FragmentPagerAdapter
9  import androidx.viewpager.widget.ViewPager
10 import com.example.belajartablayout.Fragment.DraftFragment
11 import com.example.belajartablayout.Fragment.InboxFragment
12 import com.example.belajartablayout.Fragment.SentFragment
13 import com.example.belajartablayout.databinding.ActivityMainBinding
14 import com.google.android.material.tabs.TabLayout
15
16 class MainActivity : AppCompatActivity() {
17
18     private lateinit var binding: ActivityMainBinding
19
20     override fun onCreate(savedInstanceState: Bundle?) {
21         super.onCreate(savedInstanceState)
22         binding = ActivityMainBinding.inflate(layoutInflater)
23         setContentView(binding.root)
24
25         val toolbar: Toolbar = binding.toolbarAwal
26         setSupportActionBar(toolbar)
27         supportActionBar!!.title = ""
28
29         val tabLayout: TabLayout = binding.tabLayout
30         val viewPager: ViewPager = binding.viewPager
31         val viewPagerAdapter = ViewPagerAdapter(supportFragmentManager)
32
33         viewPagerAdapter.tambahFragment(InboxFragment(), judul: "Inbox")
34         viewPagerAdapter.tambahFragment(DraftFragment(), judul: "Draft")
35         viewPagerAdapter.tambahFragment(SentFragment(), judul: "Sent")
36
37         viewPager.adapter = viewPagerAdapter
38         tabLayout.setupWithViewPager(viewPager)
39     }
40
41     internal class ViewPagerAdapter(fragmentManager: FragmentManager): FragmentPagerAdapter(
42         fragmentManager, BEHAVIOR_RESUME_ONLY_CURRENT_FRAGMENT){
43
44         private val fragments: ArrayList<Fragment>
45         private val judul: ArrayList<String>
46
47         init {
48             fragments = ArrayList()
49             judul = ArrayList()
50         }
51
52         override fun getCount(): Int {
53             return fragments.size
54         }
55
56         override fun getItem(position: Int): Fragment {
57             return fragments[position]
58         }
59
60         fun tambahFragment(fragment: Fragment, judul: String){
61             fragments.add(fragment)
62             judul.add(judul)
63         }
64
65         override fun getPageTitle(position: Int): CharSequence? {
66             return judul[position]
67         }
68     }
69 }
```



Gambar 11. Halaman MainActivity.kt

Kemudian untuk melihat hasilnya, silahkan jalankan aplikasi tersebut pada perangkat anda atau dengan menggunakan emulator. Jika berhasil, maka tampilannya akan seperti di bawah ini.



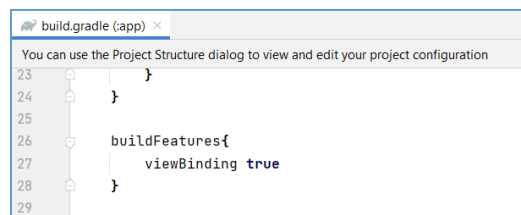
Gambar 12. Hasil proyek BelajarTabLayout

**Tugas Analisa :**

1. Apa perbedaan membuat file kotlin secara otomatis ketika menambahkan layout fragment dengan menambahkan satu persatu class nya?
2. Apa yang dimaksud dengan ViewPager? Jelaskan.

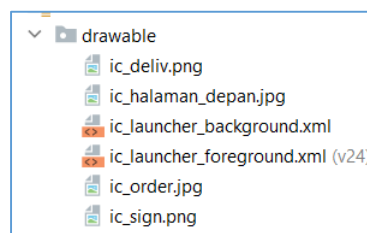
## Fragment (ViewPager2)

Seperti yang telah dilihat pada percobaan sebelumnya bahwa FragmentPagerAdapter sudah tidak lagi didukung oleh Google untuk kedepannya. Google menyarankan untuk menggunakan ViewPager 2 untuk menggantikan FragmentPagerAdapter. Untuk percobaan kedua ini kita akan coba membuat sebuah aplikasi yang mengimplementasikan ViewPager2. Aplikasi ini akan menjadi pengganti Tab Layout, dimana untuk berpindah halaman akan menggunakan fungsi *slide*. Buat sebuah proyek baru bernama „**BelajarViewPager2**“. Buka halaman **build.gradle** untuk menambahkan fitur viewBinding.



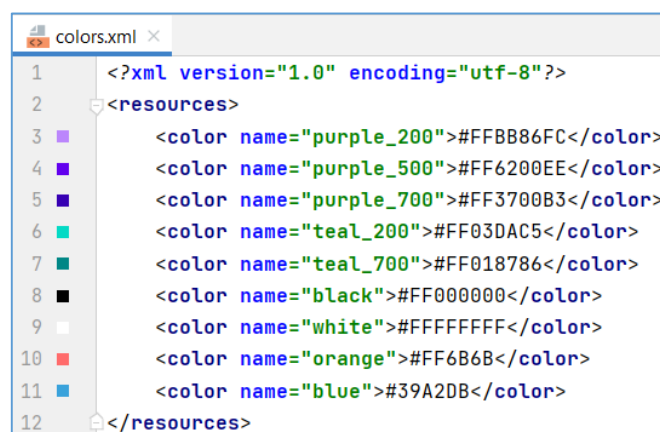
Gambar 13. Halaman build.gradle (:app)

Kemudian silahkan cari dan tambahkan 4 buah gambar yang akan digunakan pada setiap halaman.

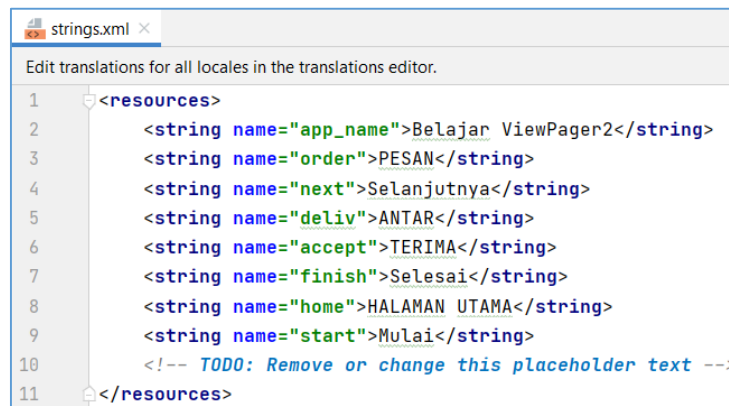


Gambar 14. File dalam folder Drawable

Silahkan daftarkan beberapa jenis warna dan teks yang akan kita gunakan berulang kali pada aplikasi. Daftarkan hal tersebut pada halaman **colors.xml** dan halaman **strings.xml**.



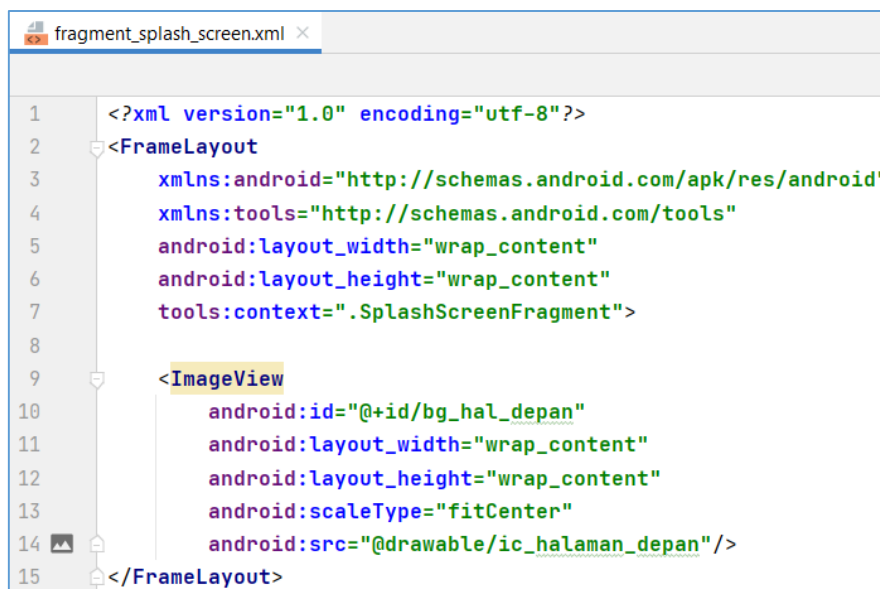
Gambar 15. Halaman color.xml



```
1 <resources>
2   <string name="app_name">Belajar ViewPager2</string>
3   <string name="order">PESAN</string>
4   <string name="next">Selanjutnya</string>
5   <string name="deliv">ANTAR</string>
6   <string name="accept">TERIMA</string>
7   <string name="finish">Selesai</string>
8   <string name="home">HALAMAN UTAMA</string>
9   <string name="start">Mulai</string>
10  <!-- TODO: Remove or change this placeholder text -->
11 </resources>
```

Gambar 16. Halaman strings.xml

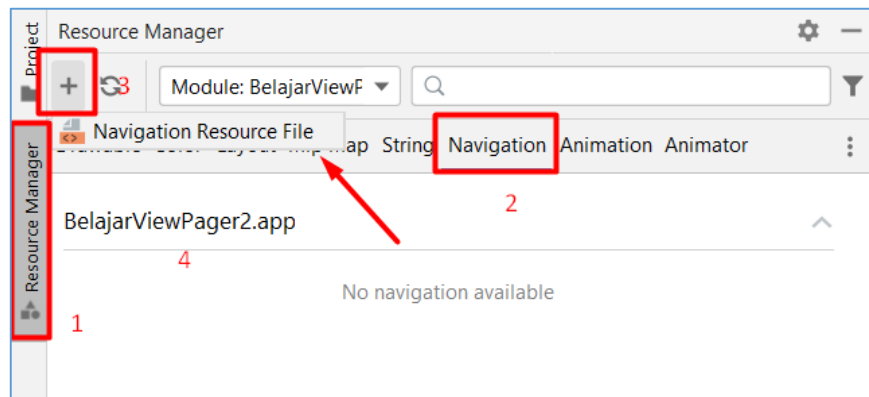
Setelah itu, tambahkan sebuah fragment baru, beri nama „**SplashScreenFragment**”. Kemudian buka layout **fragment\_splash\_screen.xml** dan ketikkan kode xml berikut untuk membuat tampilan UI halaman pembuka. Sesuaikan nama gambar dengan nama yang disimpan pada folder drawable.



```
1 <?xml version="1.0" encoding="utf-8"?>
2 <FrameLayout
3     xmlns:android="http://schemas.android.com/apk/res/android"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:layout_width="wrap_content"
6     android:layout_height="wrap_content"
7     tools:context=".SplashScreenFragment">
8
9     <ImageView
10         android:id="@+id/bg_hal_depan"
11         android:layout_width="wrap_content"
12         android:layout_height="wrap_content"
13         android:scaleType="fitCenter"
14         android:src="@drawable/ic_halaman_depan"/>
15 </FrameLayout>
```

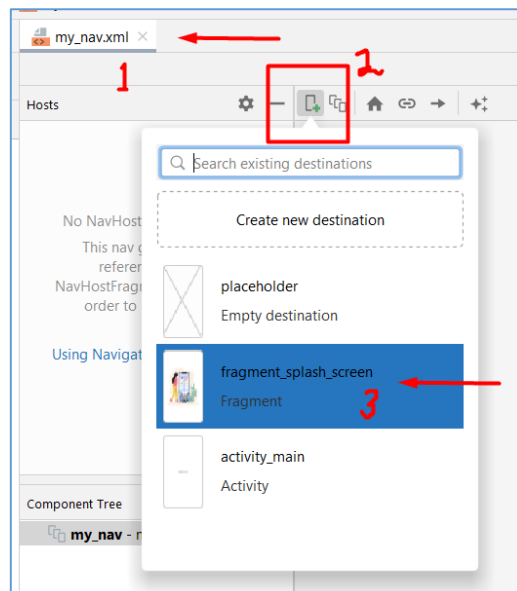
Gambar 17. Halaman fragment\_splash\_screen.xml

Kemudian kita akan membuat *navigation graph* untuk memudahkan kita membuat navigasi dari sebuah halaman ke halaman lainnya. *Navigation graph* ini merupakan fitur yang belum lama dikenalkan oleh google untuk memudahkan pengembang. Caranya seperti pada Gambar 18, klik **Resource Manager** -> kemudian arahkan ke tab “**Navigation**” -> Klik tanda + -> Klik **Navigation Resource File** -> beri nama “**my\_nav**” -> jika ada error message, klik “**Yes**”.



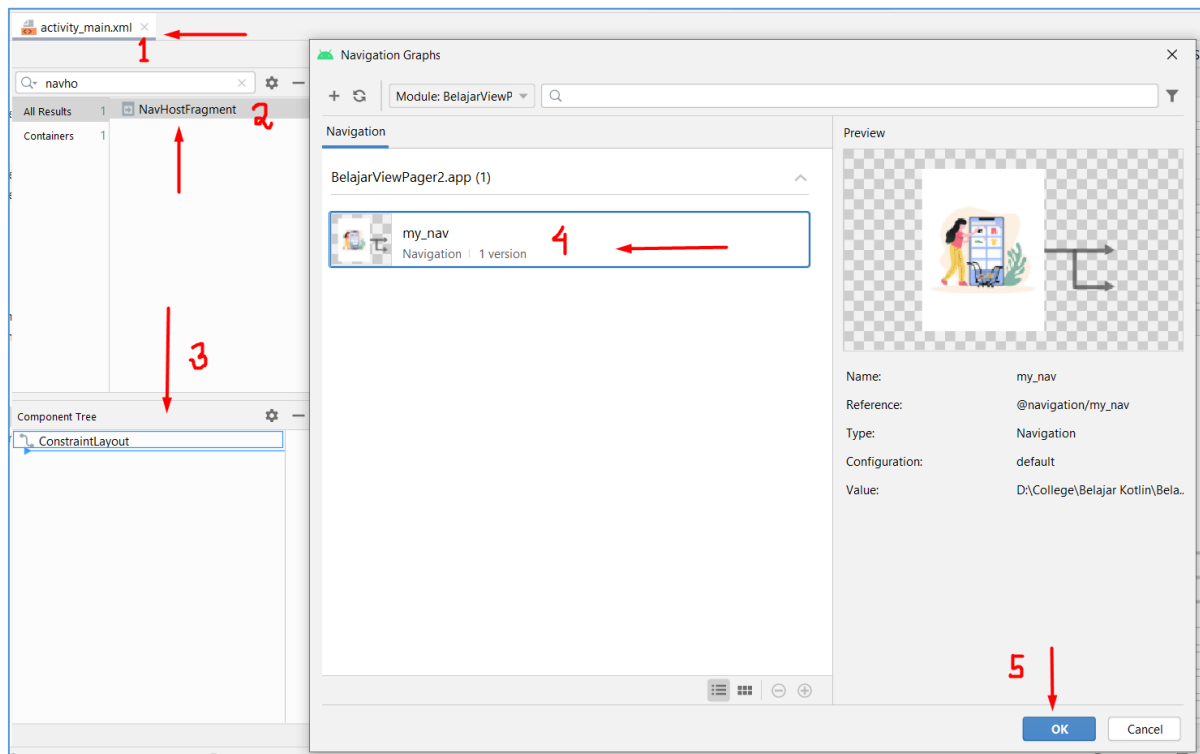
Gambar 18. Langkah membuat *navigation graph*

Setelah halaman `my_nav.xml` terbentuk, maka kita dapat membuat alur navigasi. Kita dapat menambahkan halaman dengan cara klik **“New Destination”** seperti pada Gambar 19. Kemudian pilih halaman **fragment\_splash\_screen** yang telah kita buat sebelumnya.



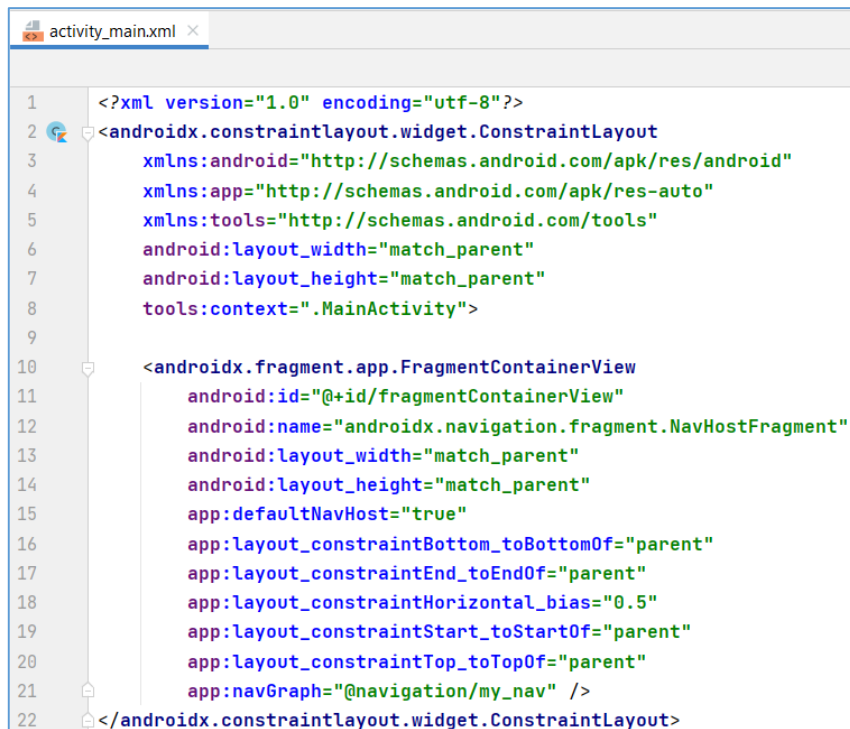
Gambar 19. Menambahkan fragment splash screen

Setelah itu, kita ingin memastikan bahwa halaman splash screen ini akan muncul di awal setiap kali aplikasi ini dijalankan. Maka kita akan membuat navigation host pada halaman **activity\_main.xml**. Buka halaman **activity\_main.xml**, klik tab **“Design”**, kemudian pilih objek **NavHostFragment** -> drag ke **Component Tree** dibawah **ConstraintLayout**. Setelahnya akan terbuka halaman **Navigation Graph** -> klik **my\_nav** -> **Ok** (lihat Gambar 20).



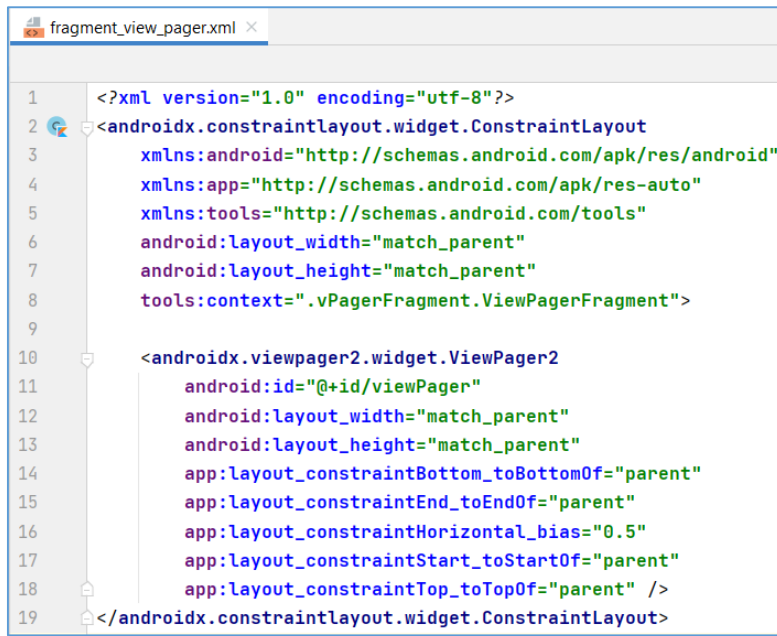
Gambar 20. Langkah - langkah membuat NavHostFragment

Setelah itu, untuk membuat ukuran dan posisi gambar menjadi lebih rapi, maka silahkan tambahkan baris kode xml untuk pengaturan posisi Fragment Container seperti dibawah ini di halaman **activity\_main.xml** pada tab **"Code"**. Sesuaikan mana baris kode yang belum ada, lalu tambahkan.



Gambar 21. Halaman activity\_main.xml

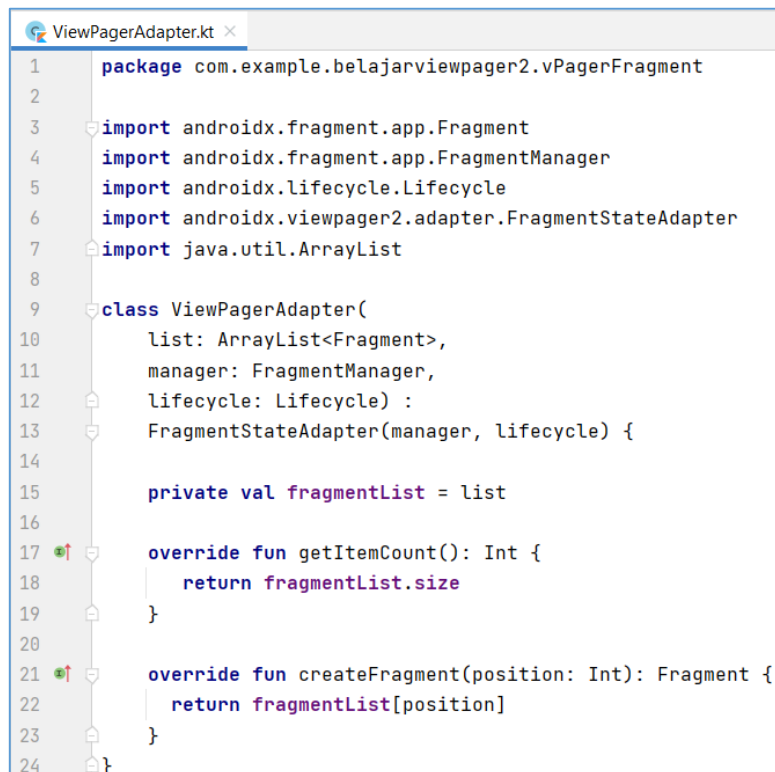
Selanjutnya kita membuat sebuah package baru pada proyek bernama **"vPagerFragment"**. Package ini akan menyimpan dua buah file bernama **ViewPagerFragment** dan **ViewPagerAdapter**. Buat sebuah **Fragment** baru, pilih **Blank Fragment**, beri nama **ViewPagerFragment**. Lalu buka halaman **fragment\_view\_pager.xml** dan ketik kode xml berikut.



```
1 <?xml version="1.0" encoding="utf-8"?>
2 <androidx.constraintlayout.widget.ConstraintLayout
3     xmlns:android="http://schemas.android.com/apk/res/android"
4     xmlns:app="http://schemas.android.com/apk/res-auto"
5     xmlns:tools="http://schemas.android.com/tools"
6     android:layout_width="match_parent"
7     android:layout_height="match_parent"
8     tools:context=".vPagerFragment.ViewPagerFragment">
9
10    <androidx.viewpager2.widget.ViewPager2
11        android:id="@+id/viewPager"
12        android:layout_width="match_parent"
13        android:layout_height="match_parent"
14        app:layout_constraintBottom_toBottomOf="parent"
15        app:layout_constraintEnd_toEndOf="parent"
16        app:layout_constraintHorizontal_bias="0.5"
17        app:layout_constraintStart_toStartOf="parent"
18        app:layout_constraintTop_toTopOf="parent" />
19 </androidx.constraintlayout.widget.ConstraintLayout>
```

Gambar 22. Halaman fragment\_view\_pager.xml

Setelah itu, tambahkan sebuah file kotlin baru pada package **vPagerFragment** bernama **"ViewPagerAdapter.kt"**. Kemudian ketikkan kode program berikut ini.



```
1 package com.example.belajarviewpager2.vPagerFragment
2
3 import androidx.fragment.app.Fragment
4 import androidx.fragment.app.FragmentManager
5 import androidx.lifecycle.Lifecycle
6 import androidx.viewpager2.adapter.FragmentStateAdapter
7 import java.util.ArrayList
8
9 class ViewPagerAdapter(
10     list: ArrayList<Fragment>,
11     manager: FragmentManager,
12     lifecycle: Lifecycle) :
13     FragmentStateAdapter(manager, lifecycle) {
14
15     private val fragmentList = list
16
17     override fun getItemCount(): Int {
18         return fragmentList.size
19     }
20
21     override fun createFragment(position: Int): Fragment {
22         return fragmentList[position]
23     }
24 }
```

Gambar 23. Halaman ViewPagerAdapter.kt

Kemudian buat sebuah **package** baru lagi di dalam package “**vPagerFragment**” yang bernama “**screen**” agar tampilan hirarki proyek menjadi lebih rapi. Package ini akan menyimpan tiga buah halaman yang akan kita buat nantinya. Kemudian tambahkan sebuah fragment baru didalam package “**screen**”, kemudian beri nama “**HalamanPertama**”, “**HalamanKedua**”, “**HalamanKetiga**”. Kemudian buka halaman **fragment\_halaman\_pertama.xml** dan tambahkan kode xml berikut.

```
fragment_halaman_pertama.xml
1  <?xml version="1.0" encoding="utf-8"?>
2  <androidx.constraintlayout.widget.ConstraintLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:app="http://schemas.android.com/apk/res-auto"
5      xmlns:tools="http://schemas.android.com/tools"
6      android:layout_width="match_parent"
7      android:layout_height="match_parent"
8      tools:context=".vPagerFragment.screens.HalamanPertama">
9
10     <ImageView
11         android:id="@+id/bg_hal_satu"
12         android:layout_width="300dp"
13         android:layout_height="300dp"
14         android:src="@drawable/ic_order"
15         android:layout_marginTop="150dp"
16         app:layout_constraintTop_toTopOf="parent"
17         app:layout_constraintEnd_toEndOf="parent"
18         app:layout_constraintStart_toStartOf="parent" />
19
20     <TextView
21         android:id="@+id/txt_hal_satu"
22         android:layout_width="wrap_content"
23         android:layout_height="wrap_content"
24         android:text="@string/order"
25         android:textSize="55sp"
26         android:textStyle="bold"
27         app:layout_constraintEnd_toEndOf="parent"
28         app:layout_constraintStart_toStartOf="parent"
29         app:layout_constraintTop_toBottomOf="@id/bg_hal_satu" />
30
31     <TextView
32         android:id="@+id/txt_hal_next"
33         android:layout_width="wrap_content"
34         android:layout_height="wrap_content"
35         android:text="@string/next"
36         android:textSize="25sp"
37         android:textStyle="bold"
38         android:textColor="@color/orange"
39         android:layout_marginEnd="30dp"
40         android:layout_marginBottom="50dp"
41         app:layout_constraintBottom_toBottomOf="parent"
42         app:layout_constraintEnd_toEndOf="parent" />
43 </androidx.constraintlayout.widget.ConstraintLayout>
```

Gambar 24. Halaman fragment\_halaman\_utama.xml

Lakukan hal serupa pada **fragment\_halaman\_kedua.xml** dan **fragment\_halaman\_ketiga.xml** dengan **mengubah gambar** yang ditampilkan, serta **teks** yang digunakan. Pada halaman kedua menggunakan teks **ANTAR** dan halaman ketiga menggunakan teks **TERIMA**. Setelah itu pada package „vPagerFragment“, buat sebuah file kotlin baru bernama „ViewPagerFragment“. Silahkan tambahkan kode berikut.



```
ViewPagerFragment.kt
1 package com.example.belajarviewpager2.vPagerFragment
2 import android.os.Bundle
3 import androidx.fragment.app.Fragment
4 import android.view.LayoutInflater
5 import android.view.View
6 import android.view.ViewGroup
7 import com.example.belajarviewpager2.R
8 import com.example.belajarviewpager2.databinding.FragmentViewPagerBinding
9 import com.example.belajarviewpager2.vPagerFragment.screens.HalamanKedua
10 import com.example.belajarviewpager2.vPagerFragment.screens.HalamanKetiga
11 import com.example.belajarviewpager2.vPagerFragment.screens.HalamanPertama
12
13 class ViewPagerFragment : Fragment() {
14     lateinit var binding: FragmentViewPagerBinding
15     override fun onCreateView(
16         inflater: LayoutInflater, container: ViewGroup?,
17         savedInstanceState: Bundle?
18     ): View? {
19         // Inflate the layout for this fragment
20         val view = inflater.inflate(R.layout.fragment_view_pager,
21             container, attachToRoot: false)
22         binding = FragmentViewPagerBinding.bind(view)
23         val fragmentList = arrayListOf(
24             HalamanPertama(),
25             HalamanKedua(),
26             HalamanKetiga()
27         )
28         val adapter = ViewPagerAdapter(fragmentList,
29             requireActivity().supportFragmentManager, lifecycle)
30         binding.viewPager.adapter = adapter
31         return view
32     }
33 }
```

Gambar 25. Halaman ViewPagerFragment.kt

Setelah itu silahkan buka halaman **“HalamanPertama.kt”**, kemudian tambahkan kode *logic* berikut ini.

```
HalamanPertama.kt
1 package com.example.belajarviewpager2.vPagerFragment.screens
2
3 import android.os.Bundle
4 import androidx.fragment.app.Fragment
5 import android.view.LayoutInflater
6 import android.view.View
7 import android.view.ViewGroup
8 import androidx.viewpager2.widget.ViewPager2
9 import com.example.belajarviewpager2.R
10 import com.example.belajarviewpager2.databinding.FragmentHalamanPertamaBinding
11
12 class HalamanPertama : Fragment() {
13     private lateinit var binding: FragmentHalamanPertamaBinding
14
15     override fun onCreateView(
16         inflater: LayoutInflater, container: ViewGroup?,
17         savedInstanceState: Bundle?
18     ): View? {
19         // Inflate the layout for this fragment
20         val view = inflater.inflate(R.layout.fragment_halaman_pertama,
21             container, attachToRoot: false)
22         binding = FragmentHalamanPertamaBinding.bind(view)
23         val viewPager = activity?.findViewById<ViewPager2>(R.id.viewPager)
24
25         binding.txtHalNext.setOnClickListener { it: View!
26             viewPager?.currentItem = 1
27         }
28         return view
29     }
30 }
```

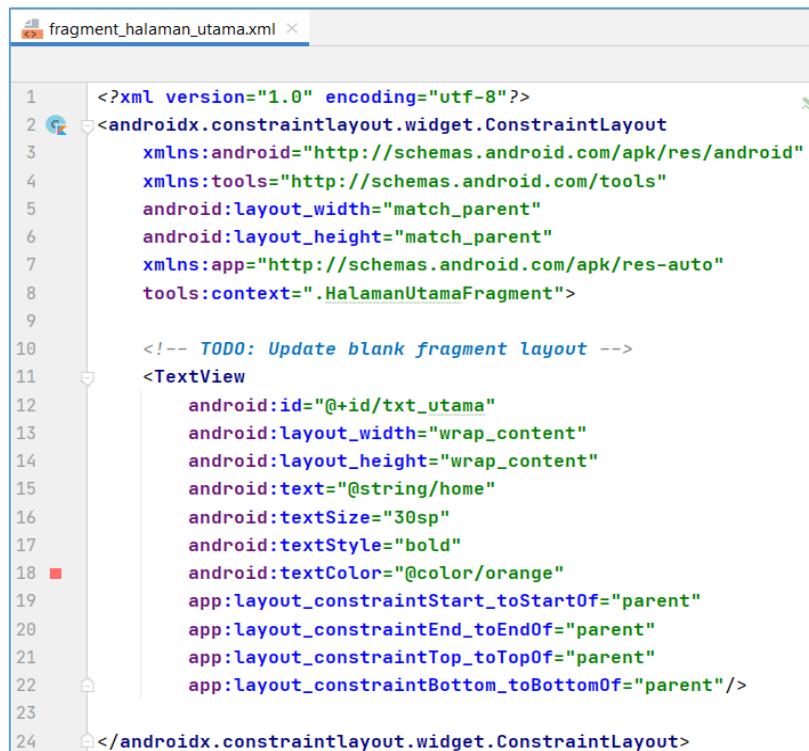
Gambar 26. HalamanPertama.kt

```
HalamanKedua.kt
1 package com.example.belajarviewpager2.vPagerFragment.screens
2
3 import android.os.Bundle
4 import androidx.fragment.app.Fragment
5 import android.view.LayoutInflater
6 import android.view.View
7 import android.view.ViewGroup
8 import androidx.viewpager2.widget.ViewPager2
9 import com.example.belajarviewpager2.R
10 import com.example.belajarviewpager2.databinding.FragmentHalamanKeduaBinding
11
12 class HalamanKedua : Fragment() {
13     private lateinit var binding: FragmentHalamanKeduaBinding
14
15     override fun onCreateView(
16         inflater: LayoutInflater, container: ViewGroup?,
17         savedInstanceState: Bundle?
18     ): View? {
19         // Inflate the layout for this fragment
20         val view = inflater.inflate(R.layout.fragment_halaman_kedua,
21             container, attachToRoot: false)
22         binding = FragmentHalamanKeduaBinding.bind(view)
23         val viewPager = activity?.findViewById<ViewPager2>(R.id.viewPager)
24
25         binding.txtHalNext2.setOnClickListener { it: View!
26             viewPager?.currentItem = 2
27         }
28         return view
29     }
30 }
31 }
```

Gambar 27. HalamanKedua.kt

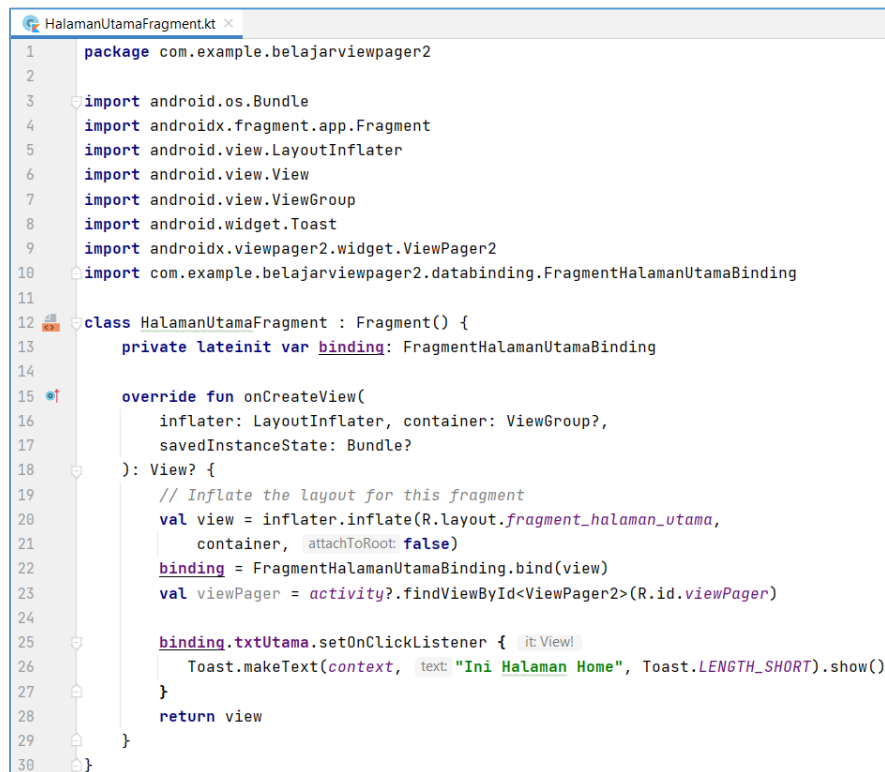
Kemudian kita akan membuat sebuah fragment baru bernama „HalamanUtamaFragment“, dimana ketika nanti kita sudah sampai dihalaman akhir, maka ketika tombol “selesai” di tekan akan

mengarah ke halaman utama. Buka halaman **fragment\_halaman\_utama.xml** dan tambahkan kode program berikut ini.



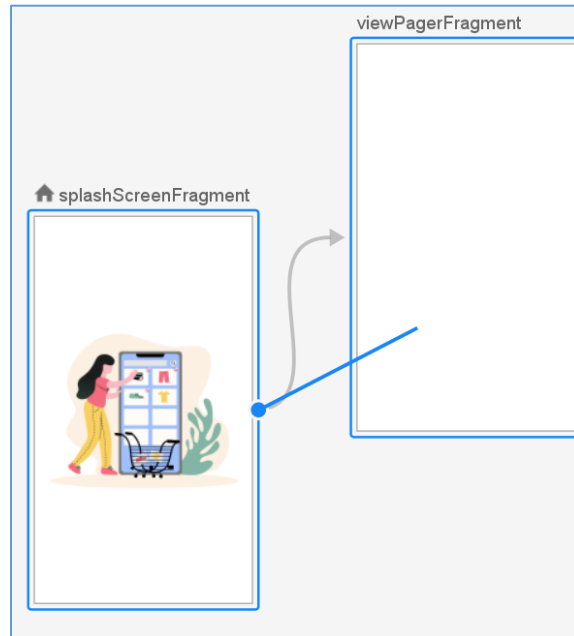
Gambar 28. Halaman fragment\_halaman\_utama.xml

Kemudian buka „**HalamanUtamaFragment.kt**“ kemudian tambahkan kode program kotlin berikut ini.



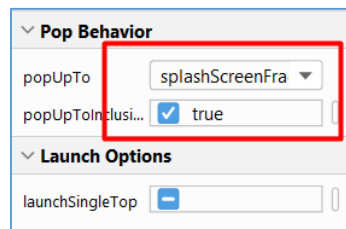
Gambar 29. HalamanUtamaFragment.kt

Kemudian silahkan buka kembali halaman navigasi yang telah kita buat untuk menghubungkan beberapa halaman. Buka halaman „**my\_nav.xml**“, kemudian lihat Gambar 32. Tambahkan halaman **viewPagerFragment** dan **HalamanUtama** ke dalam canvas. Kemudian Tarik garis dari **splashScreenFragment** ke **viewPagerFragment**.



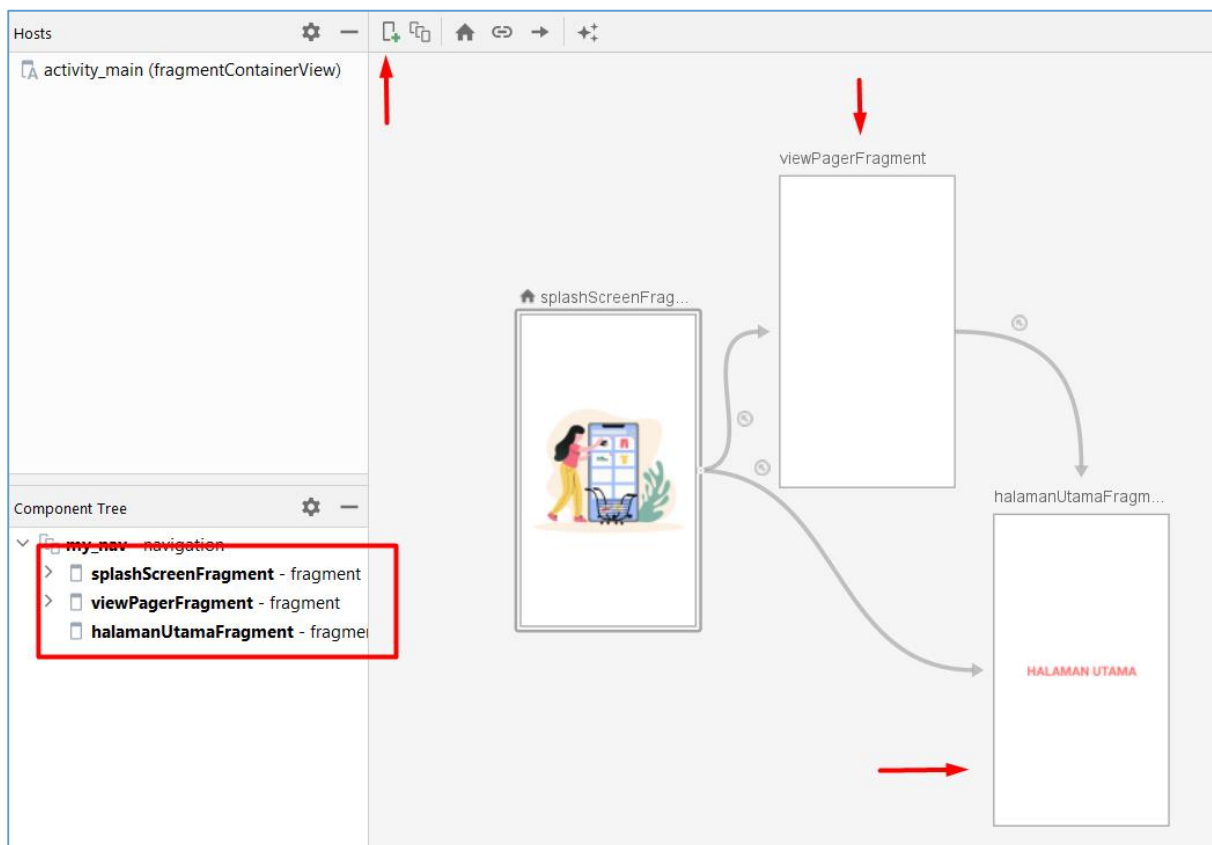
Gambar 30. Menghubungkan antar halaman

Kemudian pada tab “**Attributes**”, perhatikan bagian “**Pop Behavior**” dan lakukan hal berikut ini. Jika berhasil maka akan muncul tanda panah kecil didalam lingkaran (seperti Gambar 32)



Gambar 31. Setting pada Pop Behavior

Hubungkan juga **splashScreenFragment** dengan **HalamanUtamaFragment**, dan set **popUpTo** menjadi “**splashScreenFragment**”. Untuk **viewPagerFragment** ke **HalamanUtamaFragment**, maka set **popUpTo** menjadi “**viewPagerFragment**” (perhatikan alur Gambar 32).



Gambar 32. Halaman my\_nav.xml

Setelah itu, buka **“HalamanKetiga.kt”** dan buat kode program kotlin seperti pada gambar dibawah ini.

```
HalamanKetiga.kt
1 package com.example.belajarviewpager2.vPagerFragment.screens
2
3 import android.content.Context
4 import android.os.Bundle
5 import androidx.fragment.app.Fragment
6 import android.view.LayoutInflater
7 import android.view.View
8 import android.view.ViewGroup
9 import androidx.navigation.fragment.findNavController
10 import com.example.belajarviewpager2.R
11 import com.example.belajarviewpager2.databinding.FragmentHalamanKetigaBinding
12
13 class HalamanKetiga : Fragment() {
14     private lateinit var binding: FragmentHalamanKetigaBinding
15
16     override fun onCreateView(
17         inflater: LayoutInflater, container: ViewGroup?,
18         savedInstanceState: Bundle?
19     ): View? {
20         // Inflate the layout for this fragment
21         val view = inflater.inflate(
22             R.layout.fragment_halaman_ketiga,
23             container, attachToRoot: false
24         )
25         binding = FragmentHalamanKetigaBinding.bind(view)
26
27         binding.txtSelesai.setOnClickListener { it: View!
28             findNavController().navigate(
29                 R.id.action_viewPagerFragment_to_halamanUtamaFragment
30             )
31             onBoardingFinished()
32         }
33         return view
34     }
35
36     private fun onBoardingFinished() {
37         val sharedPreferences = requireActivity()
38             .getSharedPreferences("onBoarding", Context.MODE_PRIVATE)
39         val editor = sharedPreferences.edit()
40         editor.putBoolean("Selesai", true)
41         editor.apply()
42     }
43 }
```

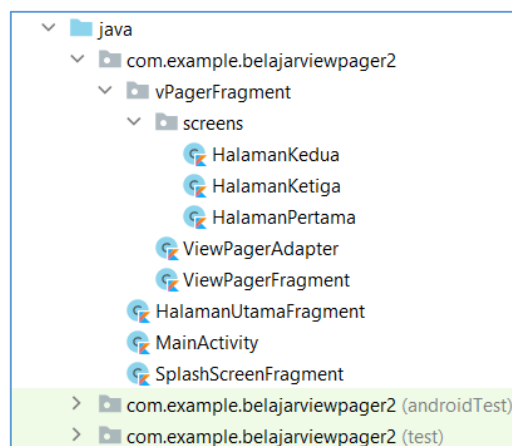
Gambar 33. Halaman Ketiga.kt

Terakhir, buka halaman „**SplashScreenFragment.kt**“ kemudian tambahkan kode kotlin berikut ini.

```
1 package com.example.belajarviewpager2
2 import android.content.Context
3 import android.os.Bundle
4 import android.os.Handler
5 import androidx.fragment.app.Fragment
6 import android.view.LayoutInflater
7 import android.view.View
8 import android.view.ViewGroup
9 import androidx.navigation.fragment.findNavController
10
11 class SplashScreenFragment : Fragment() {
12     override fun onCreateView(
13         inflater: LayoutInflater, container: ViewGroup?,
14         savedInstanceState: Bundle?
15     ): View? {
16         Handler().postDelayed({
17             if (onBoardingFinished()) {
18                 findNavController().navigate(
19                     R.id.action_splashScreenFragment_to_halamanUtamaFragment)
20             } else {
21                 findNavController().navigate(
22                     R.id.action_splashScreenFragment_to_viewPagerFragment)
23             }
24         }, delayMillis: 3000)
25         // Inflate the layout for this fragment
26         return inflater.inflate(R.layout.fragment_splash_screen, container, attachToRoot: false)
27     }
28     private fun onBoardingFinished(): Boolean {
29         val sharedPref = requireActivity()
30             .getSharedPreferences( name: "onBoarding", Context.MODE_PRIVATE)
31         return sharedPref.getBoolean( key: "Selesai", defValue: false)
32     }
33 }
```

Gambar 34. Halaman SplashScreenFragment.kt

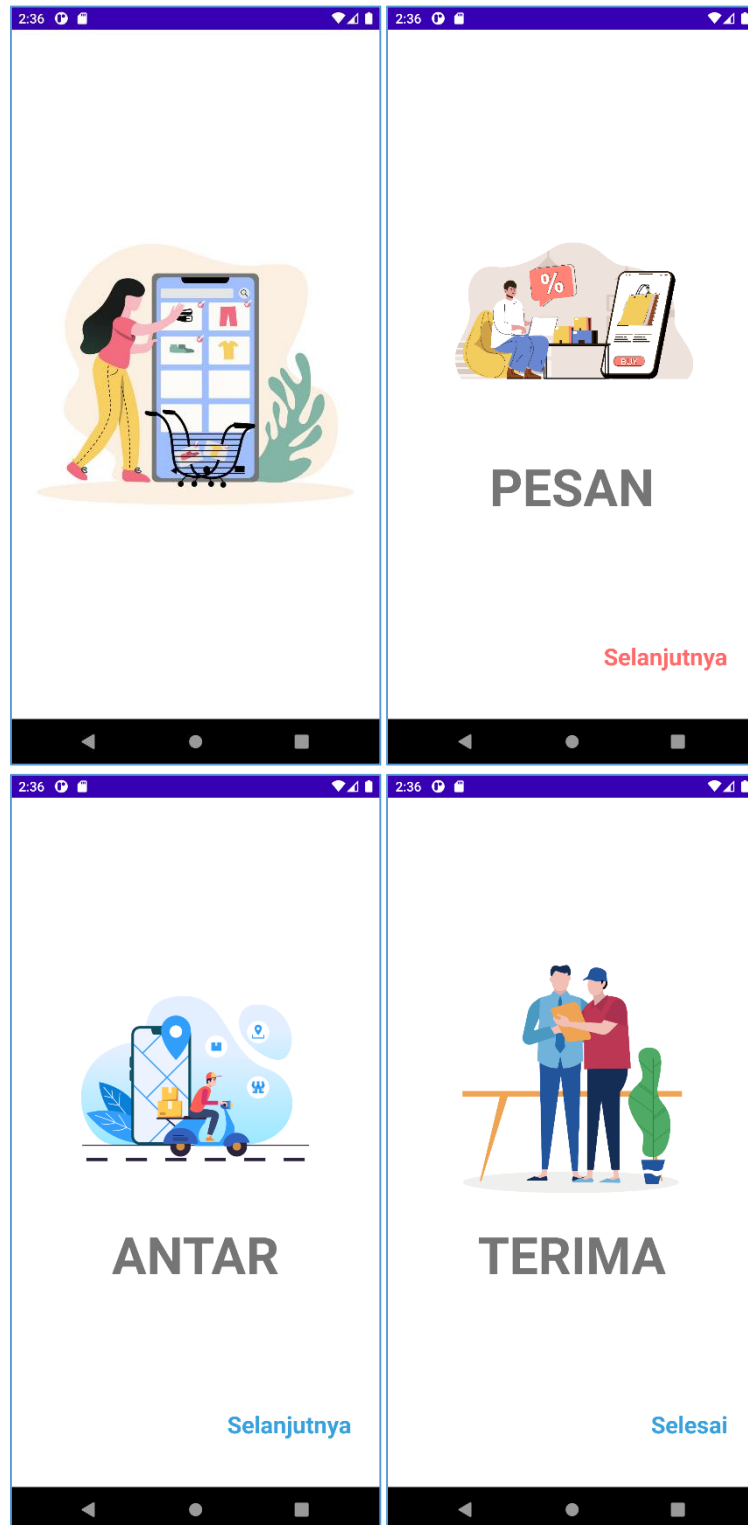
Jika selesai menambahkan semua kode program, maka hirarki proyek akan seperti pada gambar dibawah ini. Jika punya anda belum sesuai penempatan ini, maka silahkan **drag** saja *file* tersebut kedalam **package** yang sesuai.



Gambar 35. Hirarki proyek ViewPager2

Kemudian jalankan aplikasi. Jika berhasil maka akan menampilkan halaman splash screen ketika aplikasi di jalankan. Kemudian berselang tiga detik, maka aplikasi akan menampilkan halaman

satu untuk PESAN, lalu ketika kita klik teks selanjutnya / geser halaman, maka akan pindah ke halaman dua dan seterusnya. Ketika teks „Selesai“ di klik, maka akan masuk ke halaman utama. Ketika aplikasi ditutup dan dijalankan kembali, maka sistem akan membuka „Halaman Utama“ karena itu yang telah kita set pada navigasi. Jika aplikasi dihapus dan diinstal kembali, maka baru halaman pesan dan lain-lain bisa diakses kembali.



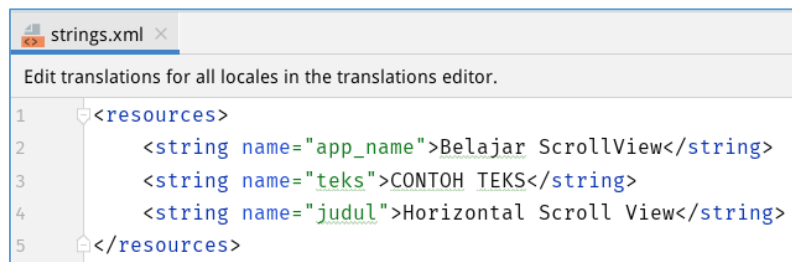
Gambar 36. Tampilan aplikasi ViewPager2



## Fragment (Belajar Horizontal Scroll View)

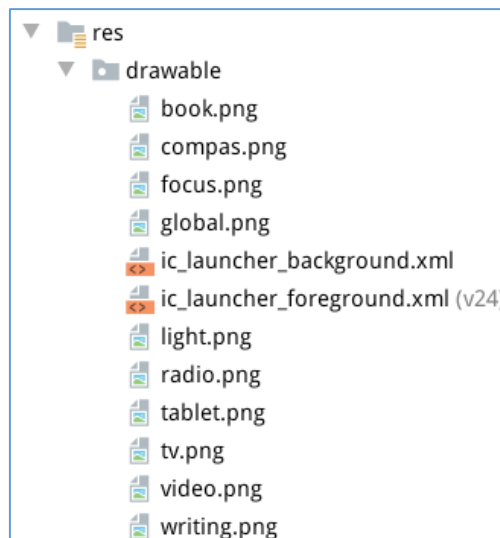
Pada modul ini kita akan mencoba membuat tampilan dengan memanfaatkan fitur scroll. Seperti yang pernah di bahas bahwa ScrollView termasuk kedalam sebuah ViewGroup. Artinya, kita dapat meletakkan sejumlah View atau Object didalam sebuah ScrollView. Kali ini kita akan mencoba menggunakan fitur scroll yang jarang dipakai, yaitu Horizontal ScrollView.

Silahkan buat sebuah projek android baru, beri nama „**BelajarHorizontalScrollView**“. Kemudian buka file **strings.xml** untuk mendaftarkan beberapa teks yang akan kita gunakan nantinya pada aplikasi.



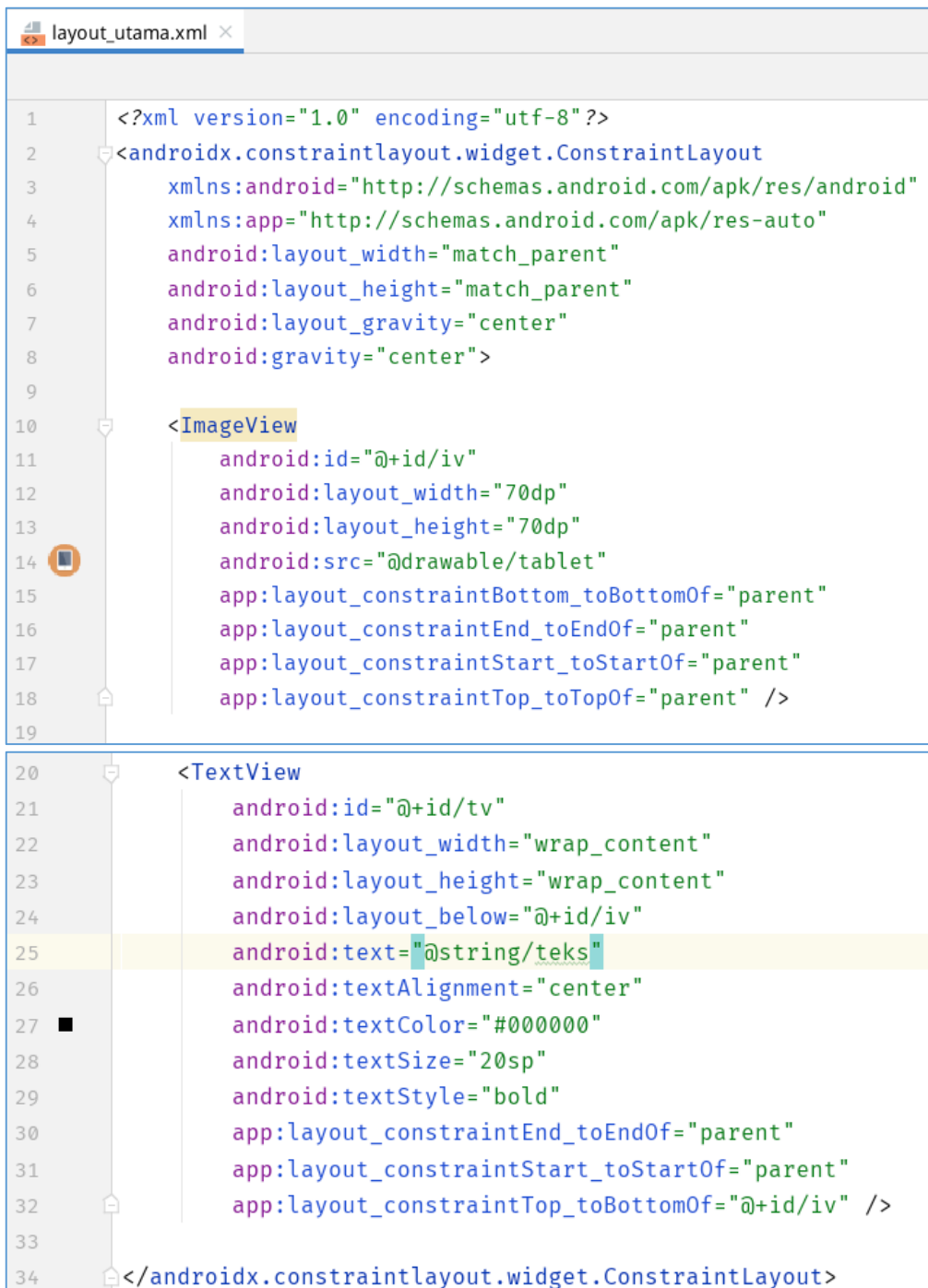
Gambar 37. Halaman string.xml

Selanjutnya anda dapat menggunakan ide bebas tentang gambar apa yang ingin ditampilkan dengan menggunakan **ScrollView**. Silahkan **unduh gambar** nya di internet, kemudian masukkan ke dalam folder **Drawable**. Apapun nama file yang digunakan, maka itu akan berpengaruh di pemanggilan pada file kotlin. Jadi silahkan diperhatikan penamaan gambar, usahakan buat semudah mungkin. Pada contoh ini, terdapat 10 jenis gambar atau *icon* berbeda yang digunakan pada aplikasi.



Gambar 38. Konten gambar di Drawable

Selanjutnya, silahkan tambahkan sebuah layout bernama **“layout\_utama.xml”**. Layout ini akan berfungsi sebagai tampilan dasar untuk gambar yang akan ditampilkan dengan menggunakan ScrollView. Kemudian silahkan ketikkan program xml di bawah ini.



```
1      <?xml version="1.0" encoding="utf-8"?>
2      <androidx.constraintlayout.widget.ConstraintLayout
3          xmlns:android="http://schemas.android.com/apk/res/android"
4          xmlns:app="http://schemas.android.com/apk/res-auto"
5          android:layout_width="match_parent"
6          android:layout_height="match_parent"
7          android:layout_gravity="center"
8          android:gravity="center">
9
10         <ImageView
11             android:id="@+id/iv"
12             android:layout_width="70dp"
13             android:layout_height="70dp"
14             android:src="@drawable/tablet"
15             app:layout_constraintBottom_toBottomOf="parent"
16             app:layout_constraintEnd_toEndOf="parent"
17             app:layout_constraintStart_toStartOf="parent"
18             app:layout_constraintTop_toTopOf="parent" />
19
20         <TextView
21             android:id="@+id/tv"
22             android:layout_width="wrap_content"
23             android:layout_height="wrap_content"
24             android:layout_below="@+id/iv"
25             android:text="@string/teks"
26             android:textAlignment="center"
27             android:textColor="#000000"
28             android:textSize="20sp"
29             android:textStyle="bold"
30             app:layout_constraintEnd_toEndOf="parent"
31             app:layout_constraintStart_toStartOf="parent"
32             app:layout_constraintTop_toBottomOf="@+id/iv" />
33
34     </androidx.constraintlayout.widget.ConstraintLayout>
```

Gambar 39. Halaman layout\_utama.xml

Setelahnya, kita akan menambahkan Horizontal Scroll View kedalam file **activity\_main.xml**. Silahkan buka halaman tersebut, kemudian ketikkan kode xxml di bawah ini.



Gambar 40. Halaman activity\_main.xml

Setelah selesai membuat file activity\_main.xml, maka selanjutnya kita akan menambahkan logika program di file **MainActivity.kt**. Pastikan untuk memanggil nama gambar yang sesuai dengan punya anda masing-masing. Sesuaikan juga teks atau judul dari masing-masing gambar pada variable menu.

```
package com.example.belajarscrollview

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.LayoutInflater
import android.view.View
import android.widget.ImageView
import android.widget.LinearLayout
import android.widget.TextView

class MainActivity : AppCompatActivity() {
    private var linearLayout: LinearLayout? = null
    private val menu = arrayOf("Compas", "Bulb", "TV", "Radio", "Tablet", "Book", "Focus", "Global", "Writing", "Video")
    private val gambar = intArrayOf(R.drawable.compas, R.drawable.light, R.drawable.tv, R.drawable.radio,
    R.drawable.tablet, R.drawable.book, R.drawable.focus, R.drawable.global, R.drawable.writing, R.drawable.video)

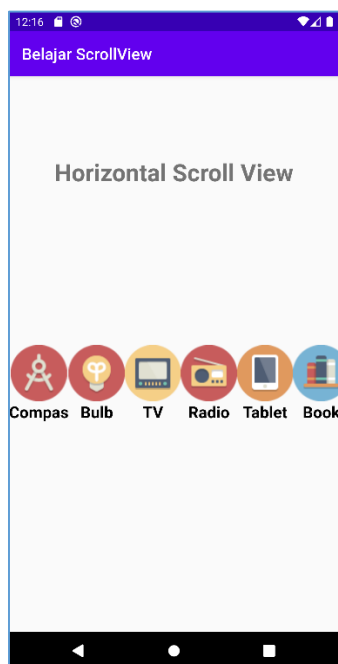
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        linearLayout = findViewById(R.id.linear1)
        val inflater : LayoutInflater = LayoutInflater.from( context: this)

        for (i :Int in menu.indices){
            val view: View = inflater.inflate(R.layout.layout_utama, linearLayout, attachToRoot: false)
            val imageView :ImageView = view.findViewById<ImageView>(R.id.iv)
            imageView.setImageResource(gambar[i])
            val tv :TextView = view.findViewById<TextView>(R.id.tv)
            tv.text = menu[i]
            linearLayout?.addView(view)
        }
    }
}
```

Gambar 41. Halaman MainActivity.kt

Aplikasi siap untuk dijalankan. Silahkan coba jalankan aplikasi tersebut, jika berhasil maka tampilan akan membentuk list gambar yang dapat di scroll ke arah kiri dan kanan seperti pada contoh di bawah ini.



Gambar 42. Tampilan Proyek BelajarHorizontalScrollView