

**KK4**  
**LEMBAR KERJA PESERTA DIDIK**  
**TEKNIK PENGOLAHAN INPUT USER**

---

Nama : Daffa Naufal Athallah

Kelas/Absen : XI RPL 3 / 11

Assignment:

**Ketentuan:**

- 1.** Kumpulkan dengan format file: WORD dengan Judul file: **No.Absen\_NAMA\_KELAS**
  
- 2.** Tugas Praktik dan Laporan:
  - a. Buat aplikasi disertai dengan penggunaan **Fragment** dengan Menggunakan **Android Studio**
  - b. Ikuti langkah pembuatan Fragment dengan menggunakan referensi youtube:
    - Bottom Navigation with Fragment - Kotlin:  
<https://youtu.be/ngNuDce9FPY>
  - c. Ketentuan Format Nama Project:  
**Fragment\_Nama\_No.Absen**
  - d. Program boleh dimodifikasi dan ditambahkan sesuai kreativitas (**tambahkan desain layout di setiap masing halaman menu**)
  - e. Penggeraan secara individu (diperbolehkan diskusi dengan kelompok)

Kumpulkan link **PUBLIC DRIVE** screenshot project di Android Studio, dan hasil *running* aplikasi

## Contoh Aplikasi



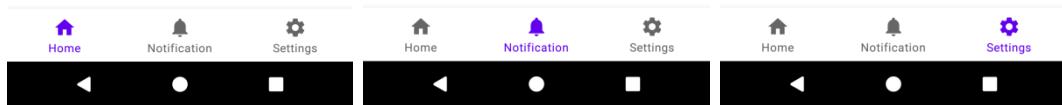
Wonton Restaurant



Notification Fragment

Settings Fragment

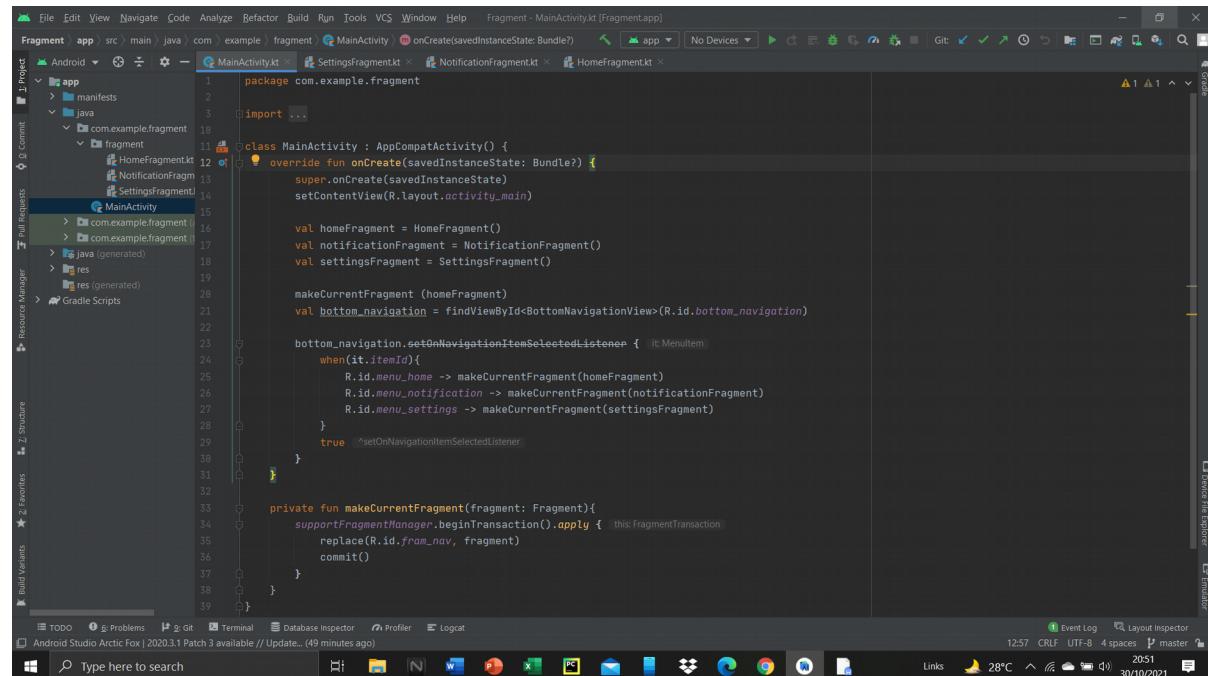
Neque porro quisquam est qui  
dolorem ipsum quia dolor sit amet,  
consectetur, adipisci velit...



\*Keterangan: Modifikasi Tampilan Fragment Home, Notification, dan Settings dapat ditambahkan sesuai kreativitas.

## Bukti Screenshot FullScreen

### 1. Project pada Android Studio



===== “Semangat ya dan Selamat Belajar!” =====

The screenshot shows the Android Studio interface with the SettingsFragment.kt file open in the main editor. The code defines a Fragment subclass named SettingsFragment. It includes TODO comments for renaming parameters and changing types. The file also contains logic for inflating a layout from R.layout.fragment\_settings.

```
1 package com.example.fragment.fragment
2
3 import ...
4
5 // TODO: Rename parameter arguments, choose names that match
6 // the fragment initialization parameters, e.g. ARG_ITEM_NUMBER
7 private const val ARG_PARAM1 = "param1"
8 private const val ARG_PARAM2 = "param2"
9
10 /**
11  * A simple [Fragment] subclass.
12  * Use the [SettingsFragment.newInstance] factory method to
13  * create an instance of this fragment.
14  */
15
16 class SettingsFragment : Fragment() {
17     // TODO: Rename and change types of parameters
18     private var param1: String? = null
19     private var param2: String? = null
20
21     override fun onCreate(savedInstanceState: Bundle?) {
22         super.onCreate(savedInstanceState)
23         arguments?.let {
24             param1 = it.getString(ARG_PARAM1)
25             param2 = it.getString(ARG_PARAM2)
26         }
27     }
28
29     override fun onCreateView(
30         inflater: LayoutInflater, container: ViewGroup?,
31         savedInstanceState: Bundle?
32     ): View? {
33         // Inflate the layout for this fragment
34         return inflater.inflate(R.layout.fragment_settings, container, false)
35     }
36 }
37
38
```

The screenshot shows the Android Studio interface with the NotificationFragment.kt file open in the main editor. The code defines a Fragment subclass named NotificationFragment. It includes TODO comments for renaming parameters and changing types. The file also contains logic for inflating a layout from R.layout.fragment\_notification.

```
1 package com.example.fragment.fragment
2
3 import ...
4
5 // TODO: Rename parameter arguments, choose names that match
6 // the fragment initialization parameters, e.g. ARG_ITEM_NUMBER
7 private const val ARG_PARAM1 = "param1"
8 private const val ARG_PARAM2 = "param2"
9
10 /**
11  * A simple [Fragment] subclass.
12  * Use the [NotificationFragment.newInstance] factory method to
13  * create an instance of this fragment.
14  */
15
16 class NotificationFragment : Fragment() {
17     // TODO: Rename and change types of parameters
18     private var param1: String? = null
19     private var param2: String? = null
20
21     override fun onCreate(savedInstanceState: Bundle?) {
22         super.onCreate(savedInstanceState)
23         arguments?.let {
24             param1 = it.getString(ARG_PARAM1)
25             param2 = it.getString(ARG_PARAM2)
26         }
27     }
28
29     override fun onCreateView(
30         inflater: LayoutInflater, container: ViewGroup?,
31         savedInstanceState: Bundle?
32     ): View? {
33         // Inflate the layout for this fragment
34         return inflater.inflate(R.layout.fragment_notification, container, false)
35     }
36 }
37
38
```

===== “Semangat ya dan Selamat Belajar!” =====

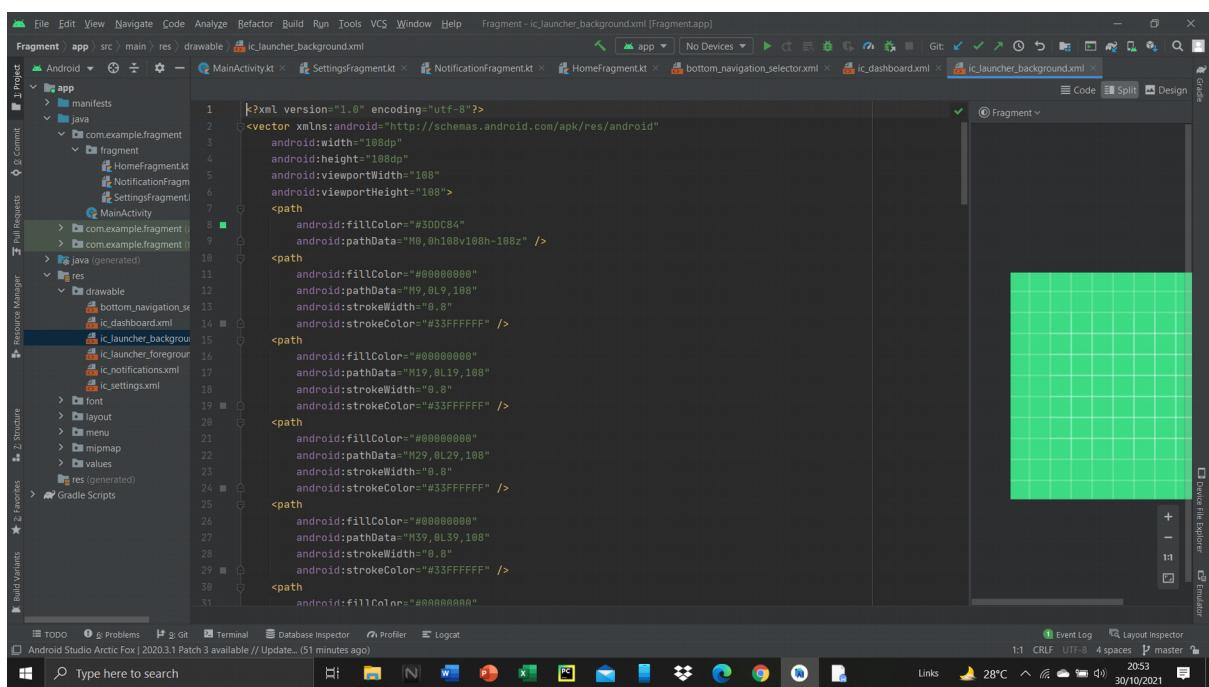
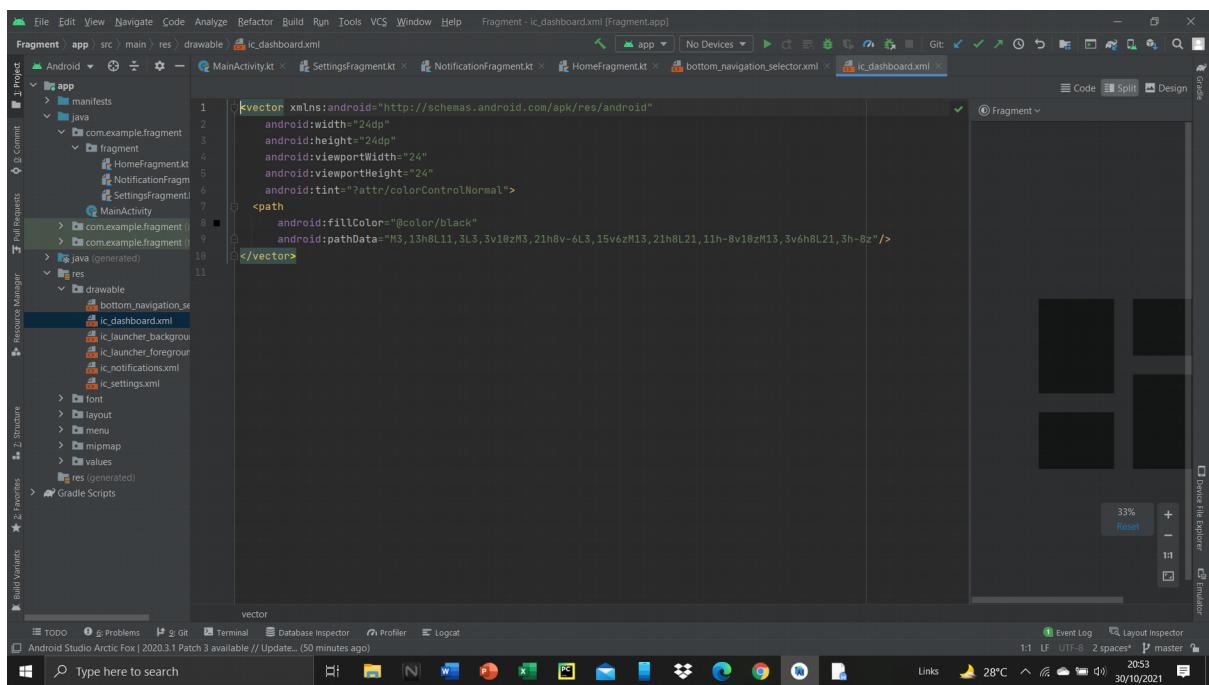
The screenshot shows the Android Studio interface with the code editor open to the `HomeFragment.kt` file. The code defines a fragment with parameters `param1` and `param2`, and inflates a layout from `R.layout.fragment_home`. The code editor includes syntax highlighting and code completion.

```
1 package com.example.fragment.fragment
2
3 import ...
4
5 // TODO: Rename parameter arguments, choose names that match
6 // the fragment initialization parameters, e.g. ARG_ITEM_NUMBER
7 private const val ARG_PARAM1 = "param1"
8 private const val ARG_PARAM2 = "param2"
9
10 /**
11  * A simple [Fragment] subclass.
12  * Use the [HomeFragment.newInstance] factory method to
13  * create an instance of this fragment.
14  */
15
16 class HomeFragment : Fragment() {
17     // TODO: Rename and change types of parameters
18     private var param1: String? = null
19     private var param2: String? = null
20
21     override fun onCreate(savedInstanceState: Bundle?) {
22         super.onCreate(savedInstanceState)
23         arguments?.let {
24             param1 = it.getString(ARG_PARAM1)
25             param2 = it.getString(ARG_PARAM2)
26         }
27     }
28
29     override fun onCreateView(
30         inflater: LayoutInflater, container: ViewGroup?,
31         savedInstanceState: Bundle?
32     ): View? {
33         // Inflate the layout for this fragment
34         return inflater.inflate(R.layout.fragment_home, container, false)
35     }
36 }
37
38 
```

The screenshot shows the Android Studio interface with the code editor open to the `bottom_navigation_selector.xml` file. The XML selector defines items for a bottom navigation bar, with one item having a checked state and a specific color. The code editor includes syntax highlighting and code completion.

```
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:state_checked="true" android:color="#0071FF" />
    <item android:color="#000000" />
</selector>
```

===== “Semangat ya dan Selamat Belajar!” =====



===== “Semangat ya dan Selamat Belajar!” =====

The screenshot shows the Android Studio interface with the following details:

- Project Structure:** The project is named "Fragment - ic\_notifications.xml [Fragment.app]". The "app" module contains Java files for HomeFragment, NotificationFragment, and SettingsFragment, and XML files for bottom\_navigation\_selector, ic\_dashboard, ic\_launcher\_background, ic\_launcher\_foreground, and ic\_notifications.
- Edit Tab:** The code editor displays the content of ic\_notifications.xml:

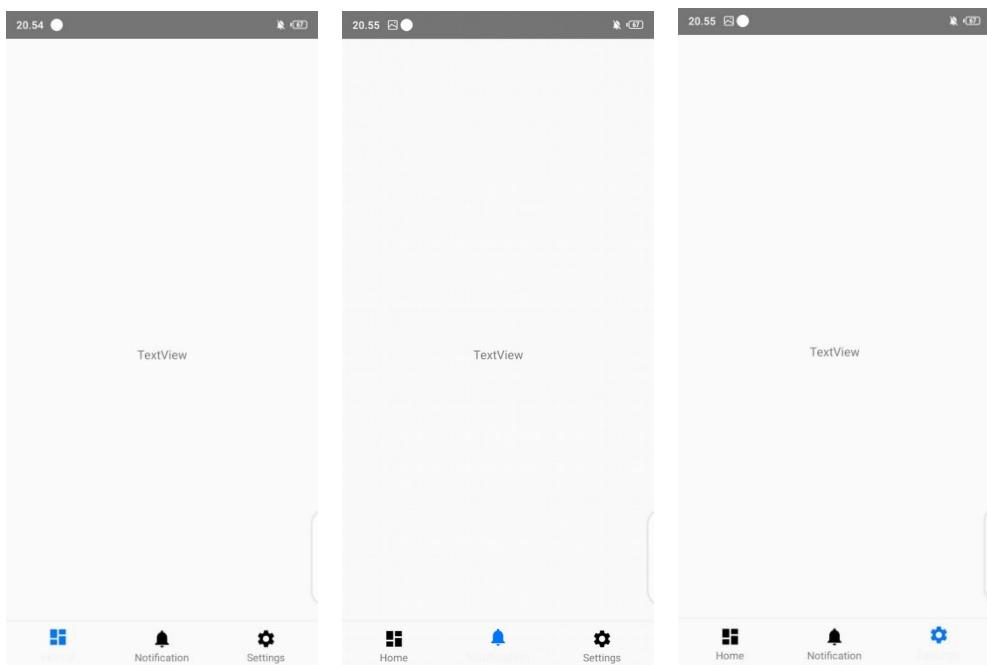
```
<vector xmlns:android="http://schemas.android.com/apk/res/android"  
    android:width="24dp"  
    android:height="24dp"  
    android:viewportWidth="24"  
    android:viewportHeight="24"  
    android:tint="?attr/colorControlNormal">  
    <path  
        android:fillColor="@color/black"  
        android:pathData="M12,22c1.1,0 2,-0.9 2,-2h-4c0,1.1 0.89,2 2,2zM18,16v-5c0,-3.07 -1.64,-5.64 -4.5,-6.32L13.5,4c-1.1,0 2,-0.9 2,-2h-4c0,1.1 0.89,2 2,2z"/>
```
- Preview Tab:** A large preview window on the right shows a black bell icon with a white outline, representing the notification icon.
- Bottom Bar:** The toolbar includes icons for TODO, Problems, Git, Terminal, Database Inspector, Profiler, Logcat, Event Log, Layout Inspector, and a search bar.
- Status Bar:** The status bar at the bottom shows the date (30/10/2021), time (20:54), battery level (28%), and network signal.

===== “Semangat ya dan Selamat Belajar!” =====

```
<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="24dp"
    android:height="24dp"
    android:viewportWidth="24"
    android:viewportHeight="24"
    android:tint="?attr/colorControlNormal">
    <path
        android:fillColor="@color/black"
        android:pathData="M19.14,12.94c0.04,-0.3 0.06,-0.61 0.06,-0.94c0,-0.32 -0.02,-0.64 -0.07,-0.94l2.03,-1.58c0.18,-0.14 0.25,-0.41 0.12,-0.61l-1.92,-1.58"/>
</vector>
```

The screenshot shows the Android Studio interface with the Fragment - ic\_settings.xml (FragmentApp) tab selected. The Project tool window on the left shows the app module structure, including Java files like HomeFragment.kt, NotificationFragment.kt, and SettingsFragment.kt, and XML files like ic\_settings.xml. The bottom status bar indicates the device is an Emulator running API level 2054 at 28°C.

## 2. Hasil Running Aplikasi pada Emulator/Handphone



===== “Semangat ya dan Selamat Belajar!” =====