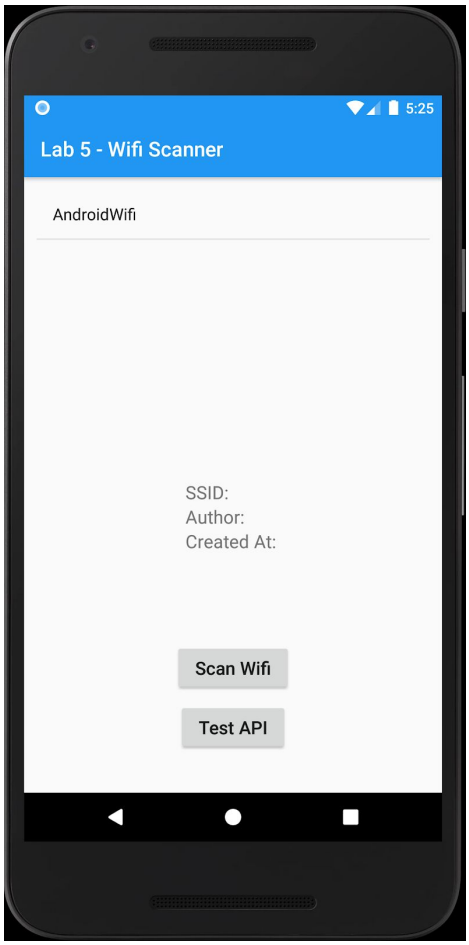
TK Teknologi Perangkat Lunak - Mobile Development

Lab 5

Disusun Oleh:
Usamah Nashirul Haq - 1606917954
30 November 2020
https://github.com/usamah1707/learn-tktpl-1606917954/tree/lab-5

A. Deskripsi Aplikasi

Aplikasi yang dibuat pada Lab 5 kali ini adalah aplikasi *wifi scanner* sederhana. Saya menggunakan hanya menggunakan satu *activity* pada aplikasi di lab kali ini. Cara bekerja aplikasi pada umumnya hanyalah melakukan *scanning* terhadap wifi yang ada disekitar kita dan melakukan *listing* terhadap SSID/nama dari wifi tersebut. Selanjutnya ada fitur untuk mengirimkan daftar SSID Wifi yang sudah kita scan dalam bentuk API.



B. Metode

Aplikasi ini harus memberikan beberapa izin agar semua proses berjalan dengan lancar, terutama untuk lokasi device. Apabila izin tidak diberikan maka aplikasi tidak dapat berjalan. Pertama kita perlu menuliskan izin berikut pada file AndroidManifest.xml

```
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.WAKE_LOCK" />
<uses-permission android:name="android.permission.INTERNET" />
```

Selanjutnya untuk melakukan *scanning* pada wifi yang ada, saya menggunakan WifiManager. WifiManager akan melakukan *scan* untuk semua wifi yang tersedia. Hasil *scan* wifi akan dikirimkan ke broadcastReceiver. Pada broadcastReceiver saya akan menuliskan semua SSID dari wifi yang sudah terdeteksi. Di dalam broadcastReceiver SSID wifi kemudian akan diproses oleh adapter untuk ditampilkan di dalam ListView.

```
@Suppress("DEPRECATION")
private fun scanWifi() {
   checkPermission()
   wifiArrayList.clear()
   Toast.makeText(this, "Scanning WiFi...", Toast.LENGTH_SHORT).show()
   wifiManager.startScan()
   registerReceiver(wifiReceiver,
IntentFilter(WifiManager.SCAN_RESULTS_AVAILABLE_ACTION))
}
```

```
val wifiReceiver = object : BroadcastReceiver() {
   override fun onReceive(context: Context?, intent: Intent?) {
       listResultWifi = wifiManager.scanResults
       unregisterReceiver(this)
       for (scan: ScanResult in listResultWifi) {
           wifiArrayList.add(scan.SSID)
           adapter.notifyDataSetChanged()
       }
   }
}
```

Selanjutnya untuk mengirimkan semua data tersebut menjadi suatu API, saya menggunakan https://pipedream.com/ untuk menyediakan link fake API. *Link* API dapat diakses di https://81326856f1545607d1e7aba4d8f379fb.m.pipedream.net.

Untuk melakukan post, saya menggunakan HttpURLConnection. Saya membuat list dari wifi SSID terlebih dahulu yang kemudian dikonversi menjadi ByteArray dengan format UTF-8. Setelah itu, saya mendefinisikan koneksi dengan url di paragraf sebelumnya. Selanjutnya saya melakukan konfigurasi untuk koneksi tersebut untuk mengatur method yang digunakan seperti *method*, *content-type*, dan *content-length*. Setelah selesai melakukan konfigurasi pada pengaturan, saya menggunakan *DataOutputStream* milik koneksi sebelumnya untuk melakukan *write parameter* yang ingin saya *post*. Berikut adalah potongan kode dan hasil pada *link* API yang saya sambungkan.

```
private fun postWifiApi() {
    val url = URL("https://81326856f1545607d1e7aba4d8f379fb.m.pipedream.net")
    var param = ""
    var counter = 0

    for (output: String in wifiArrayList) {
        param += "${++counter}. ${output}; "
    }

    val connection = url.openConnection() as HttpURLConnection
    connection.requestMethod = "POST"
    connection.readTimeout = 10000
    connection.connectTimeout = 15000
    connection.doOutput = true

    var postParam = param.toByteArray(StandardCharsets.UTF_8)

    connection.setRequestProperty("charset", "UTF-8")
    connection.setRequestProperty("content-type", "application/json")
    connection.setRequestProperty("content-length", param.length.toString())

    try {
        val outputStream: DataOutputStream =
DataOutputStream(connection.outputStream)
        outputStream.write(postParam)
        outputStream.flush()
    } catch (exception: Exception) {

    }

    if (connection.responseCode != HttpURLConnection.HTTP_OK &&
```

```
connection.responseCode != HttpURLConnection.HTTP_CREATED) {
       try {
           val inputStream: DataInputStream =
DataInputStream(connection.inputStream)
           val reader: BufferedReader =
BufferedReader(InputStreamReader(inputStream))
           val output: String = reader.readLine()

           println("There was error while connecting the chat $output")
           System.exit(0)

       } catch (exception: Exception) {
           throw Exception("Exception while push the notification
$exception.message")
       }
   }
}
```