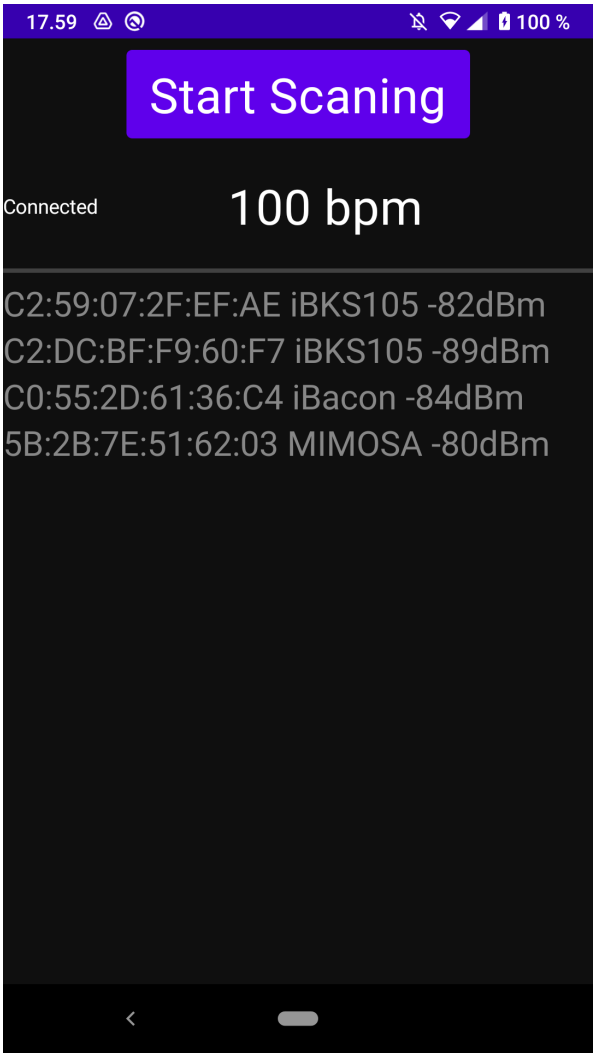
Exercise Communicating with Bluetooth LE devices

In this exercise we make a connection to a Bluetooth LE device which is able to provide Heart Rate Service. After the connection is made, the application enables the Notifications from the Service and starts receiving and displaying them on the screen.



Hint 1a: Microsoft 10 starting from the Creators Update (version 1803) supports Bluetooth Peripheral in its Bluetooth API. Therefore it is possible to emulate those Bluetooth LE peripheral operations in Windows. Windows store has an application called Bluetooth LE Explorer which is able to simulate different kind of Bluetooth GATT profiles as a peripheral. Heart Rate Service profile is one of those. Select Virtual Peripheral, then Heart Rate Service and remember to switch Advertise as connectable, Include service in advertisement and publish advertisement selections to on. Note!: Sometimes Bluetooth LE Explorer fails to start Heart Rate Service. Uncheck and check the Publish advertisement toggle a couple of times in order to wake up the Virtual Peripheral. (You can check that the Virtual Peripheral works by using nRF Connect Android application)

Hint 1b: For MacOS, there is LightBlue application which can be used to create virtual Bluetooth peripherals.

Hint 2: From the Android Play store you can install a useful application, BLE Scanner, which is able to detect Bluetooth LE devices around the phone and show their profiles. There is also an another scanner, nRF Connect for Mobile, which you are able to emulate heart beat measurement sensor also.

Hint 3: Remember to enable the Notifications by writing ENABLE_NOTIFICATION_VALUE to the CLIENT_CHARACTERISTIC_CONFIG_UUID descriptor. Otherwise Heart Rate Service does not send you the heart rate notification values.

Hint 4: You still need to scan available Bluetooth devices in your Main Activity (as you did in the previous exercise)

Hint 5: Data transfer from the ViewModel can be done using LiveData like

```
val mBPM = MutableLiveData<Int>(0)
        .
        .
        .
override fun onCharacteristicChanged(gatt: BluetoothGatt,
                                     characteristic: BluetoothGattCharacteristic
) {
   val bpm = characteristic.getIntValue(BluetoothGattCharacteristic.FORMAT_UINT16, 1)
   Log.d("DBG", "BPM: $bpm")
   mBPM.postValue(bpm)
}
```

Hint 6: Heart Rate Measurement characterisic data value is spesified in https://www.bluetooth.com/wp-content/uploads/Sitecore-Media-Library/Gatt/Xml/Characteristics/org.bluetooth.characteristic.heart_rate_measurement.xml