Platform/Android/BluetoothPrinterService.cs

```
using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Android.Bluetooth;
using Java.Util;

namespace RestoSamsu.Platforms.Android
{
   public class BluetoothPrinterService
   {
      private static readonly UUID SPP_UUID =
UUID.FromString("00001101-0000-1000-8000-00805f9b34fb");
      private BluetoothSocket _socket;
      private BluetoothDevice _device;


      public async Task<bool> ConnectToPrinterAsync(string deviceName)
      {
         try
         {
            BluetoothAdapter bluetoothAdapter = BluetoothAdapter.DefaultAdapter;
            if (bluetoothAdapter == null)
            {
               Debug.WriteLine("Bluetooth tidak tersedia.");
               return false;
            }

            if (!bluetoothAdapter.IsEnabled)
            {
               Debug.WriteLine("Bluetooth belum diaktifkan.");
               return false;
            }

            _device = bluetoothAdapter.BondedDevices.FirstOrDefault(d => d.Name ==
deviceName);
            if (_device == null)
            {
               Debug.WriteLine($"Perangkat {deviceName} tidak ditemukan.");
               return false;
            }

            // **Tutup koneksi lama sebelum membuat koneksi baru**
            if (_socket != null && _socket.IsConnected)
            {
               _socket.Close();
               _socket = null;
```

```
            Debug.WriteLine("Koneksi lama ditutup.");
        }

        // Buat koneksi baru
        _socket = _device.CreateRfcommSocketToServiceRecord(SPP_UUID);
        await _socket.ConnectAsync();

        Debug.WriteLine("Terhubung ke printer!");
        return true;
    }
    catch (Exception ex)
    {
        Debug.WriteLine($"Gagal terhubung ke printer: {ex.Message}");
        return false;
    }
}


public async Task PrintAsync(string text)
{
    if (_socket == null || !_socket.IsConnected)
    {
        Debug.WriteLine("Printer belum terhubung.");
        return;
    }

    try
    {
        byte[] buffer = Encoding.GetEncoding(437).GetBytes(text + "\n\n\n");

        // Kirim data dalam potongan kecil (512 byte per kirim)
        for (int i = 0; i < buffer.Length; i += 512)
        {
            int size = Math.Min(512, buffer.Length - i);
            _socket.OutputStream.Write(buffer, i, size);
            await Task.Delay(50);  // Delay untuk mencegah buffer overflow pada printer
        }

        _socket.OutputStream.Flush();
        Debug.WriteLine("Struk berhasil dikirim ke printer.");
    }
    catch (Exception ex)
    {
        Debug.WriteLine($"Gagal mencetak: {ex.Message}");
    }
}

public void Disconnect()
{
    if (_socket != null)
    {
```

```
            _socket.Close();
            _socket = null;
            Debug.WriteLine("Koneksi ke printer ditutup.");
        }
    }
  }
}
```

Platform/Android/AndroidBlueToothService.cs

```
using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Android.Bluetooth;
using Java.Util;
using RestoSamsu.Platforms.Android;
using RestoSamsu;

[assembly: Dependency(typeof(AndroidBlueToothService))]

namespace RestoSamsu.Platforms.Android
{
   public class AndroidBlueToothService : IBluetoothService
   {
      BluetoothAdapter bluetoothAdapter = BluetoothAdapter.DefaultAdapter;
      public IList<string> GetDeviceList()
      {
         var btdevice = bluetoothAdapter?.BondedDevices
            .Select(i => i.Name).ToList();
         return btdevice;
      }

      public async Task Print(string deviceName, string text)
      {
         BluetoothDevice device = (from bd in bluetoothAdapter?.BondedDevices where
bd?.Name == deviceName select bd).FirstOrDefault();

         try
         {
            await Task.Delay(3000);
            BluetoothSocket bluetoothSocket =
device?.CreateRfcommSocketToServiceRecord(UUID.FromString("00001101-0000-1000-8000-
00805f9b34fb"));

            bluetoothSocket?.Connect();
            byte[] buffer = Encoding.UTF8.GetBytes(text);
            bluetoothSocket?.OutputStream.Write(buffer, 0, buffer.Length);
```

```
            bluetoothSocket.Close();
        }
        catch (Exception ex)
        {
            Debug.WriteLine(ex.Message);
            throw ex;
        }
    }
  }
}
```

Platform/Android/MainActivity.cs

```
using Android;
using Android.App;
using Android.Content.PM;
using Android.OS;
using AndroidX.Core.App;
using AndroidX.Core.Content;

namespace RestoSamsu;

[Activity(Theme = "@style/Maui.SplashTheme", MainLauncher = true, ScreenOrientation =
ScreenOrientation.Landscape,LaunchMode = LaunchMode.SingleTop, ConfigurationChanges =
ConfigChanges.ScreenSize | ConfigChanges.Orientation | ConfigChanges.UiMode |
ConfigChanges.ScreenLayout | ConfigChanges.SmallestScreenSize | ConfigChanges.Density)]
public class MainActivity : MauiAppCompatActivity
{

  const int RequestBluetoothPermission = 1;

  protected override void OnCreate(Bundle savedInstanceState)
  {
    base.OnCreate(savedInstanceState);

    RequestBluetoothPermissions();
  }

  void RequestBluetoothPermissions()
  {
    if (Build.VERSION.SdkInt >= BuildVersionCodes.S)
    {
      if (ContextCompat.CheckSelfPermission(this, Manifest.Permission.BluetoothConnect) !=
Permission.Granted ||
          ContextCompat.CheckSelfPermission(this, Manifest.Permission.BluetoothScan) !=
Permission.Granted)
      {
        ActivityCompat.RequestPermissions(this, new string[]
        {
          Manifest.Permission.BluetoothConnect,
```

```
                Manifest.Permission.BluetoothScan
            }, RequestBluetoothPermission);
        }
    }
  }
}
```

MauiProgram.cs

```
using CommunityToolkit.Maui;
using Microsoft.Extensions.Logging;
using Plugin.LocalNotification;
using The49.Maui.BottomSheet;
using Microsoft.Extensions.DependencyInjection;
using RestoSamsu;
namespace RestoSamsu;

public static class MauiProgram
{
        public static MauiApp CreateMauiApp()
        {
                var builder = MauiApp.CreateBuilder();
                builder
                        .UseMauiApp<App>()

                        .UseMauiCommunityToolkit()
                        .UseBottomSheet()
                        .UseLocalNotification()

                        .ConfigureFonts(fonts =>
                        {
                                fonts.AddFont("OpenSans-Regular.ttf", "OpenSansRegular");
                                fonts.AddFont("OpenSans-Semibold.ttf", "OpenSansSemibold");
                        });



#if DEBUG
        builder.Logging.AddDebug();
#endif
#if ANDROID
        // Registrasi khusus untuk Android
        builder.Services.AddSingleton<IBluetoothService,
RestoSamsu.Platforms.Android.AndroidBlueToothService>();
#endif
        //builder.Services.AddSingleton<PrintPageViewModel>();
        builder.Services.AddSingleton<Struk.Print1>();
#if DEBUG
        builder.Logging.AddDebug();
#endif
```

```
            return builder.Build();
        }
}
```