

Nama : Muhammad Najmy Wardhana

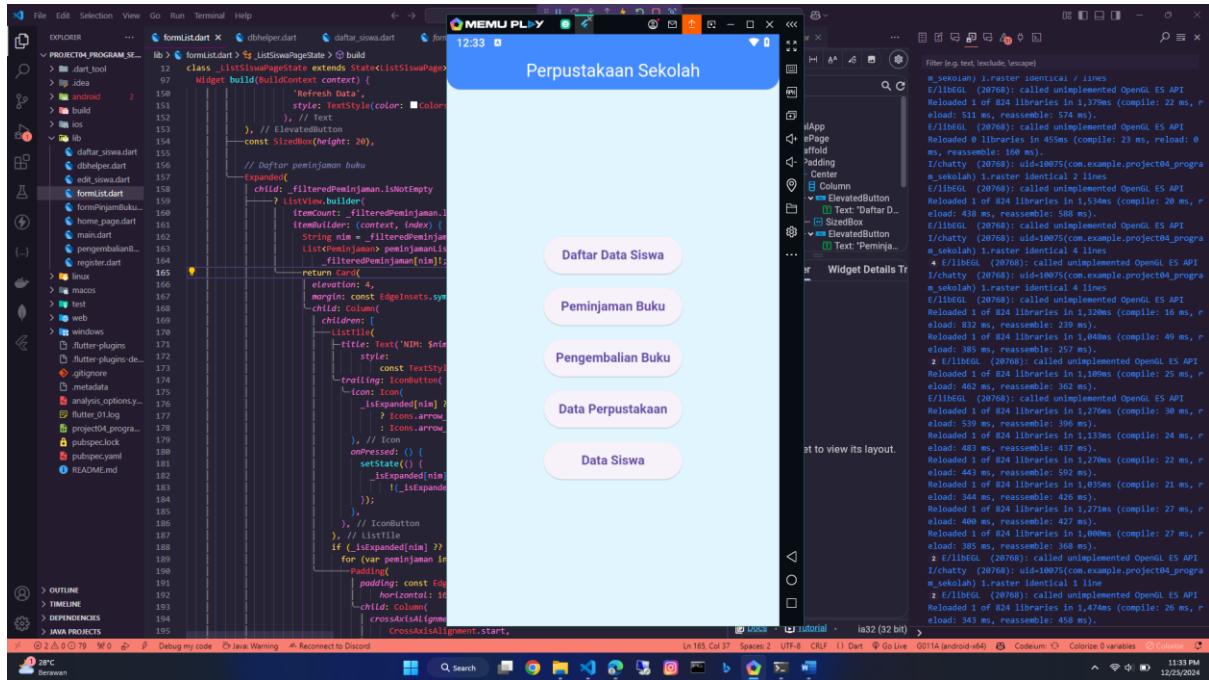
NPM : 2210010212

Kelas : 5B TI Reguler Pagi BJB

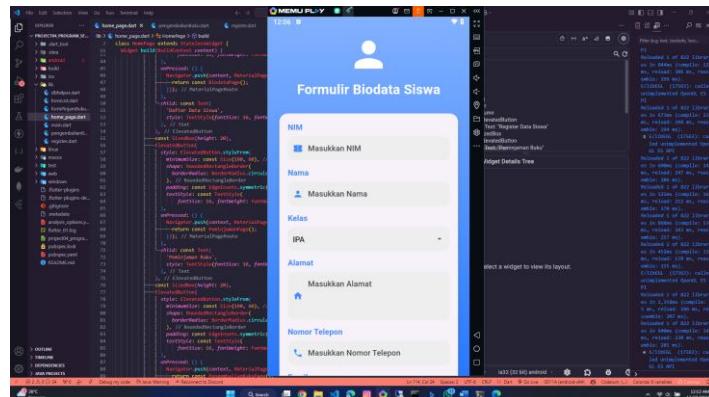
Nomor Akhir Absen 11 (1)

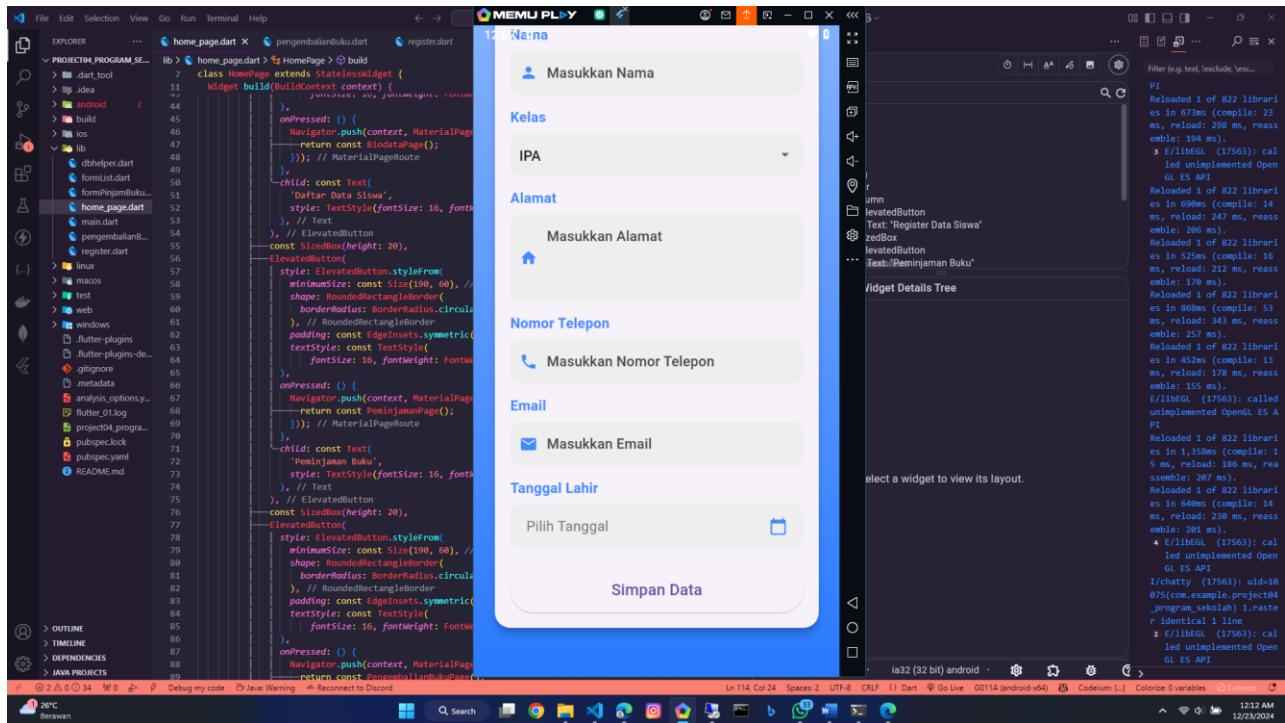
Topik Program Sekolah; Perpustakaan

Screenshot

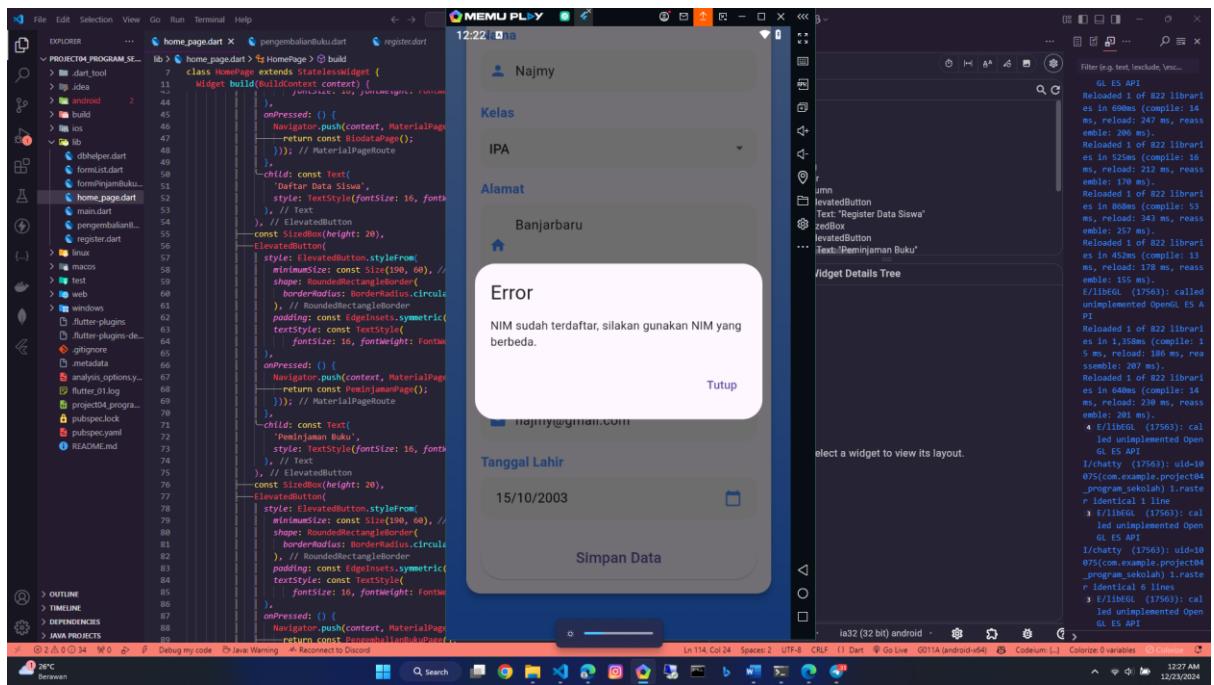


- Saya akan pilih *Register Data Siswa*

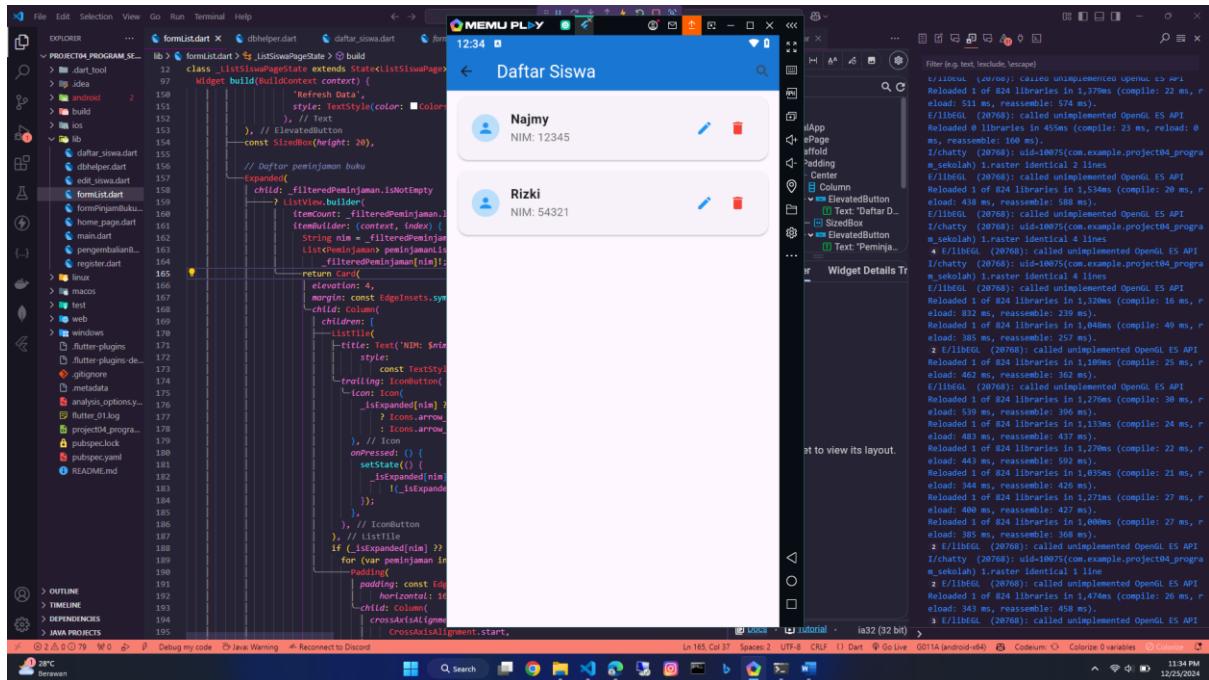




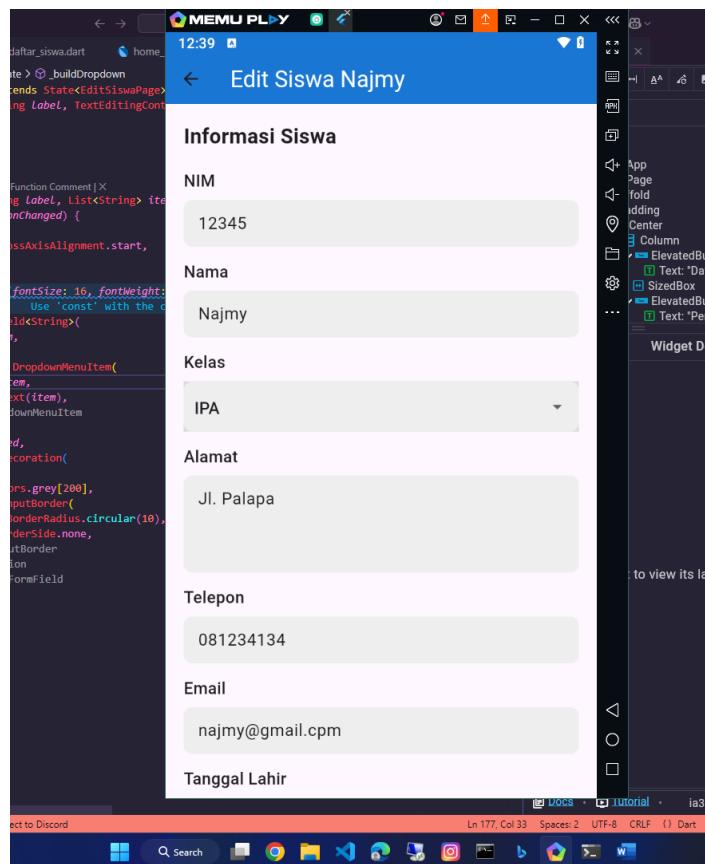
- Saya akan input NIM yang sama sebelumnya sudah didaftarkan

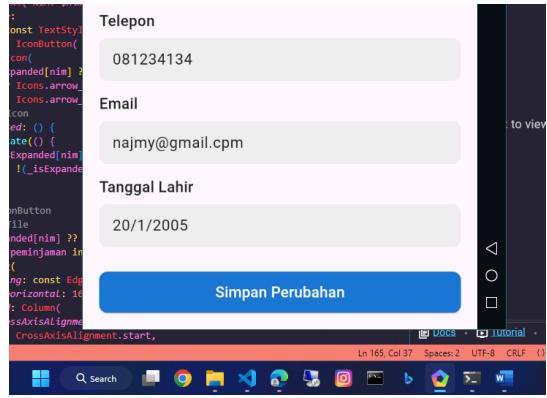


- Kembali ke halaman utama, lalu masuk ke *Data Siswa*, disini kita bisa *Menghapus Data atau Mengedit (update) data*

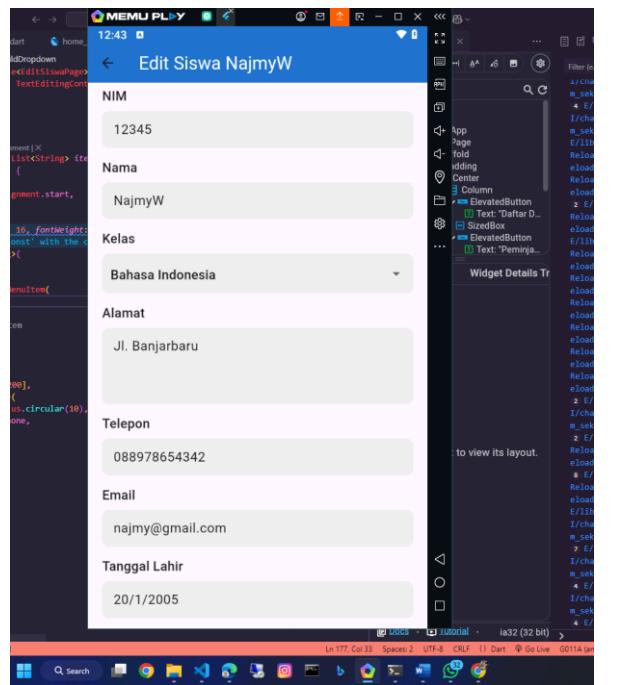


- Saya akan *Edit Data Najmy*, saya akan edit Seluruhnya



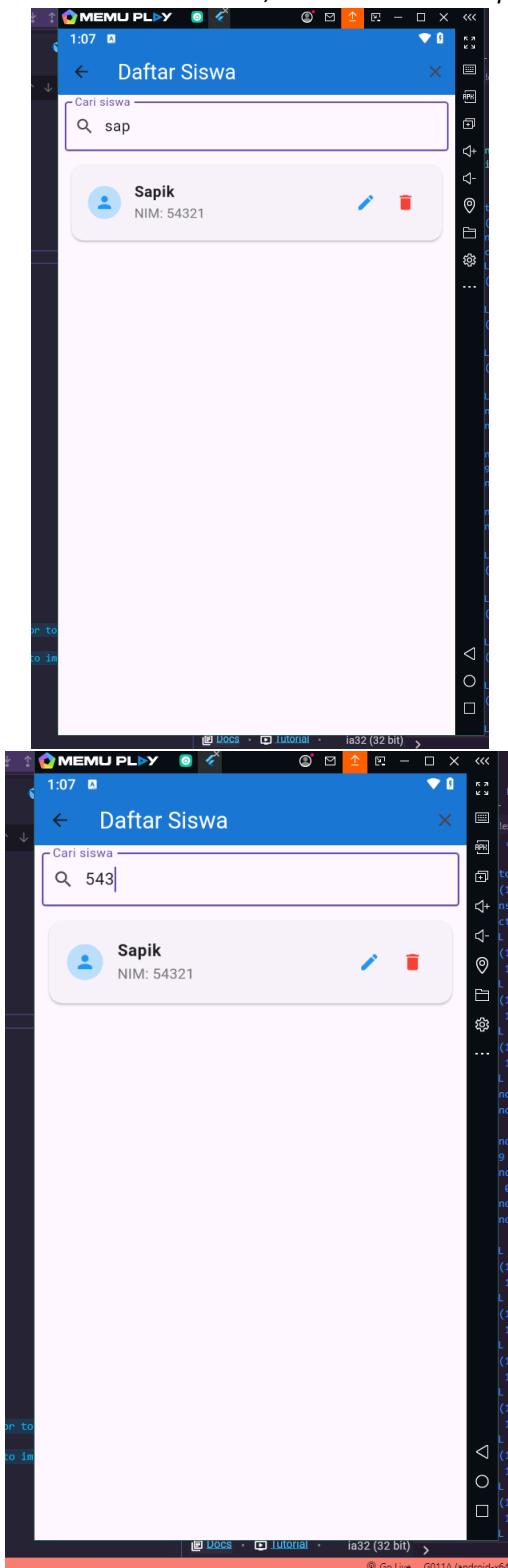


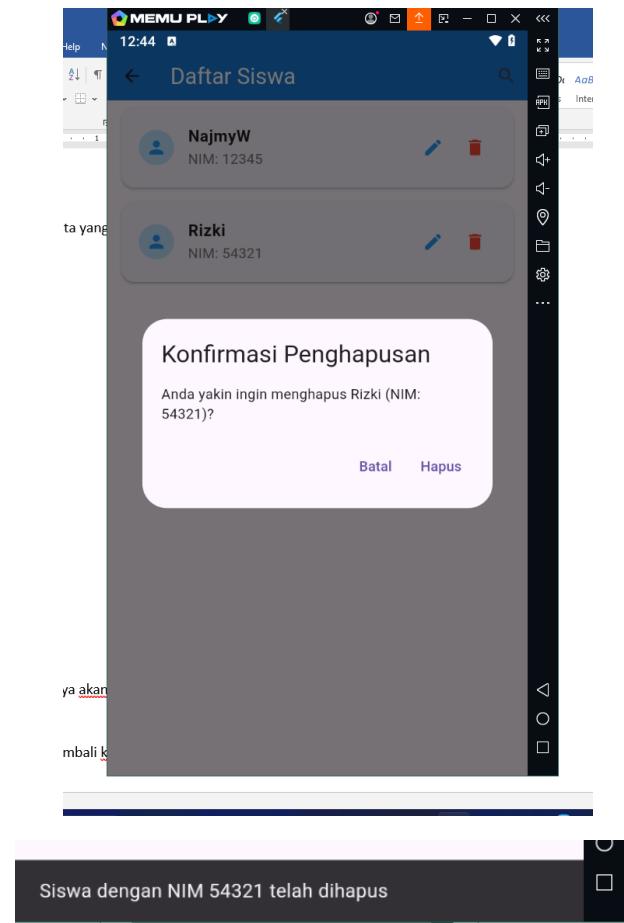
- Data yang sudah update akan tampil begini



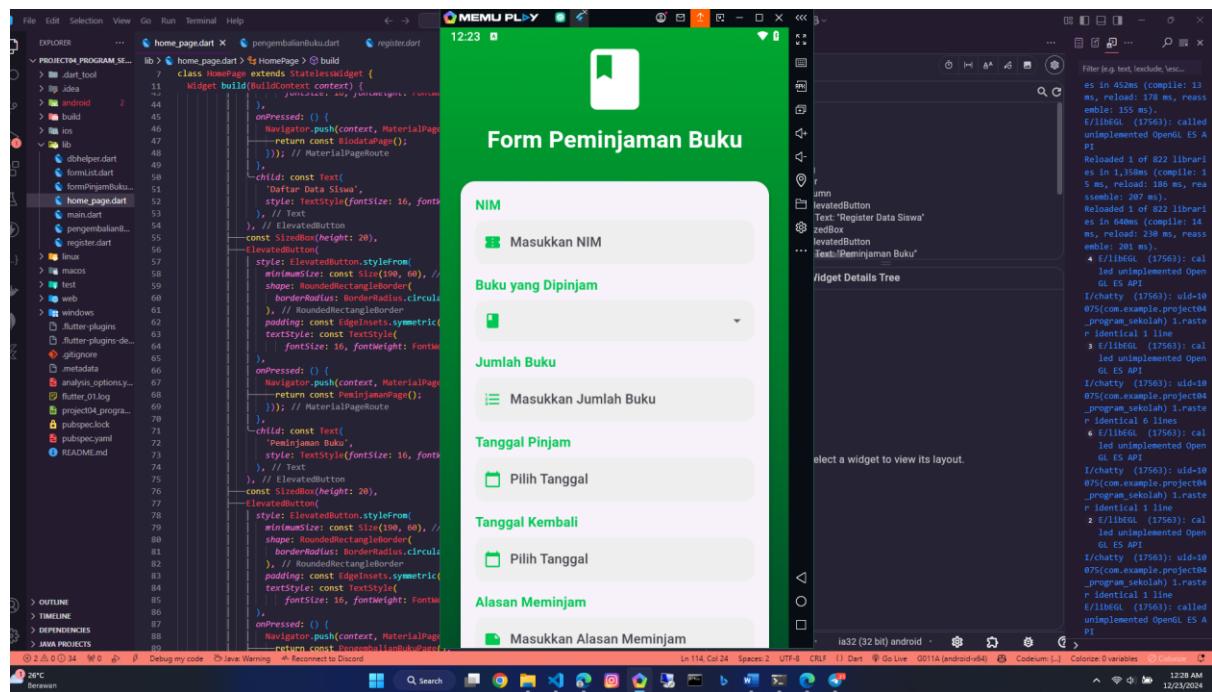
Data siswa berhasil diperbarui.

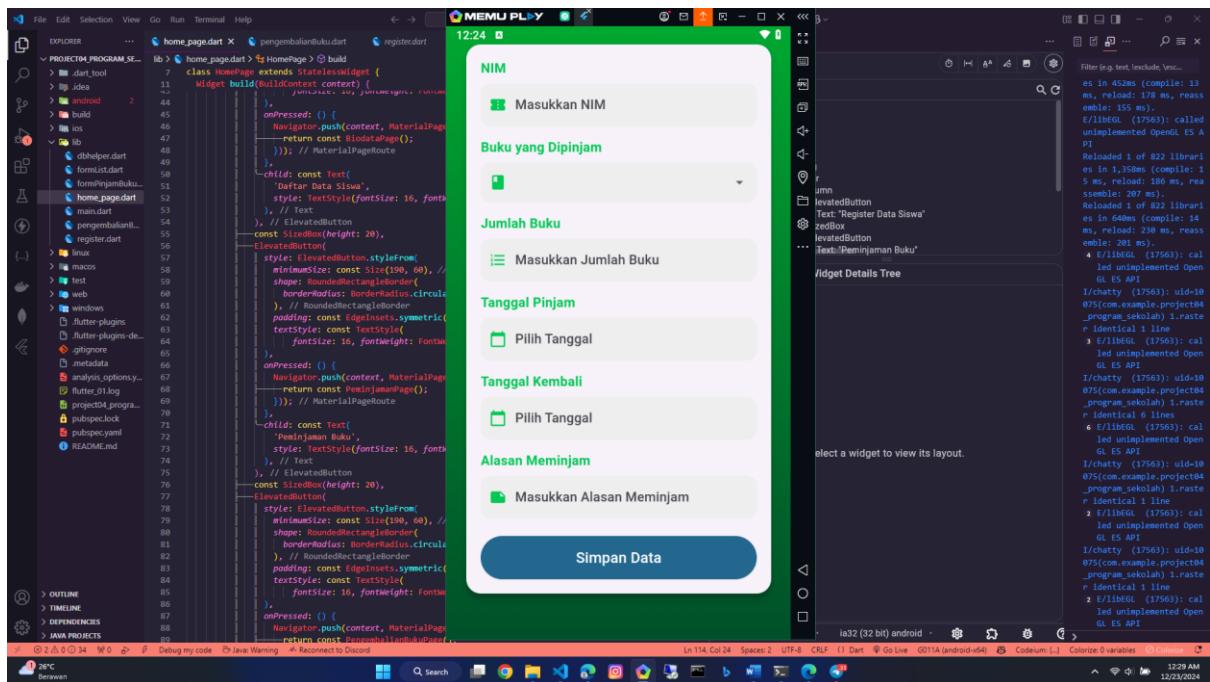
- Saya akan mencari Data sesuai Nama atau NIM, kemudian akan *Hapus* salah satu Datanya.



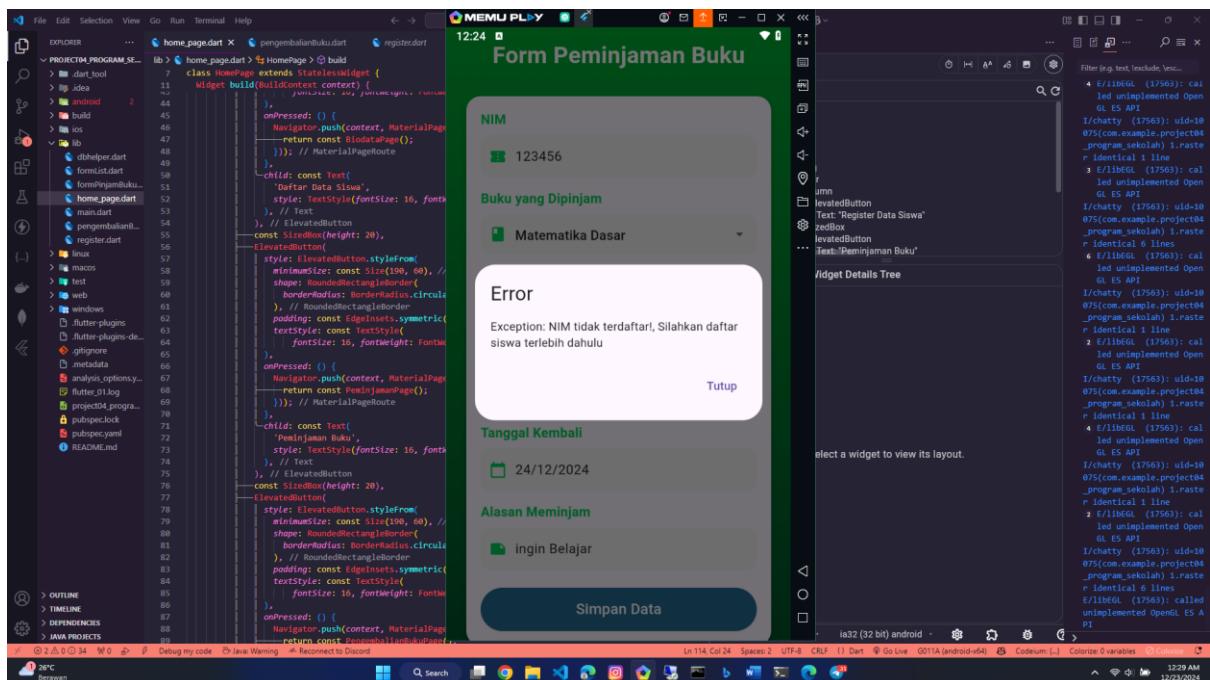


- Kembali ke halaman utama, lalu masuk ke *Peminjaman Buku*

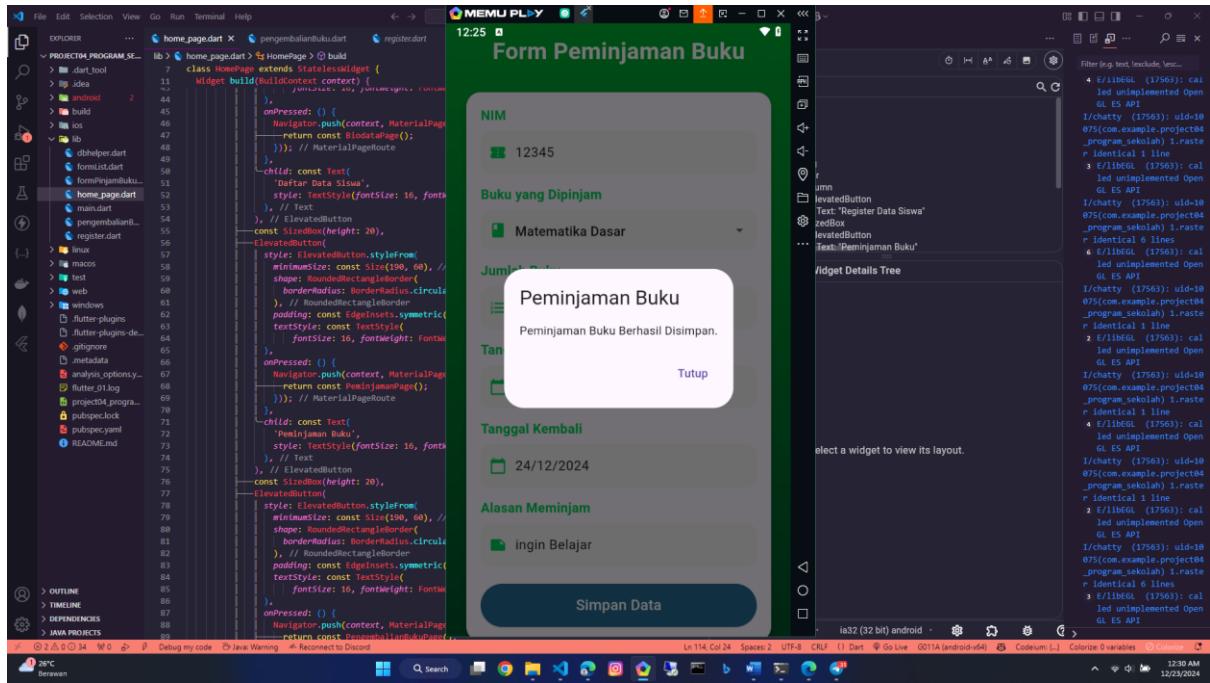




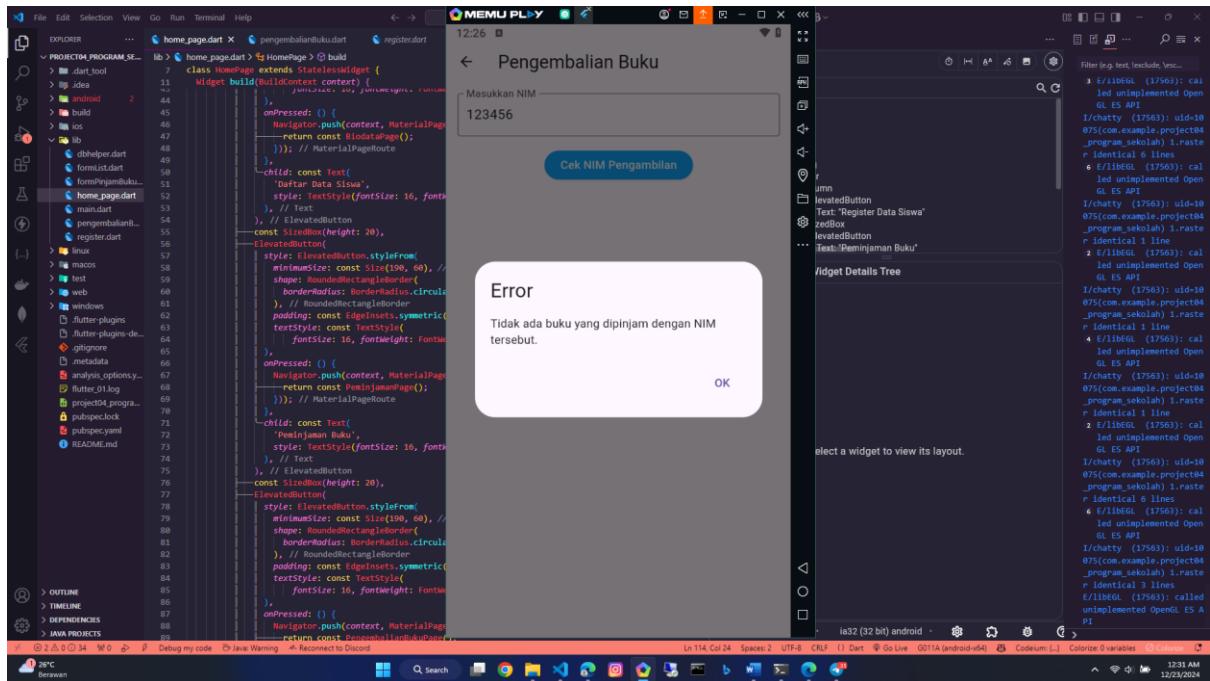
- Saya akan mengisi seluruh inputan dan mengisi NIM yang belum terdaftar



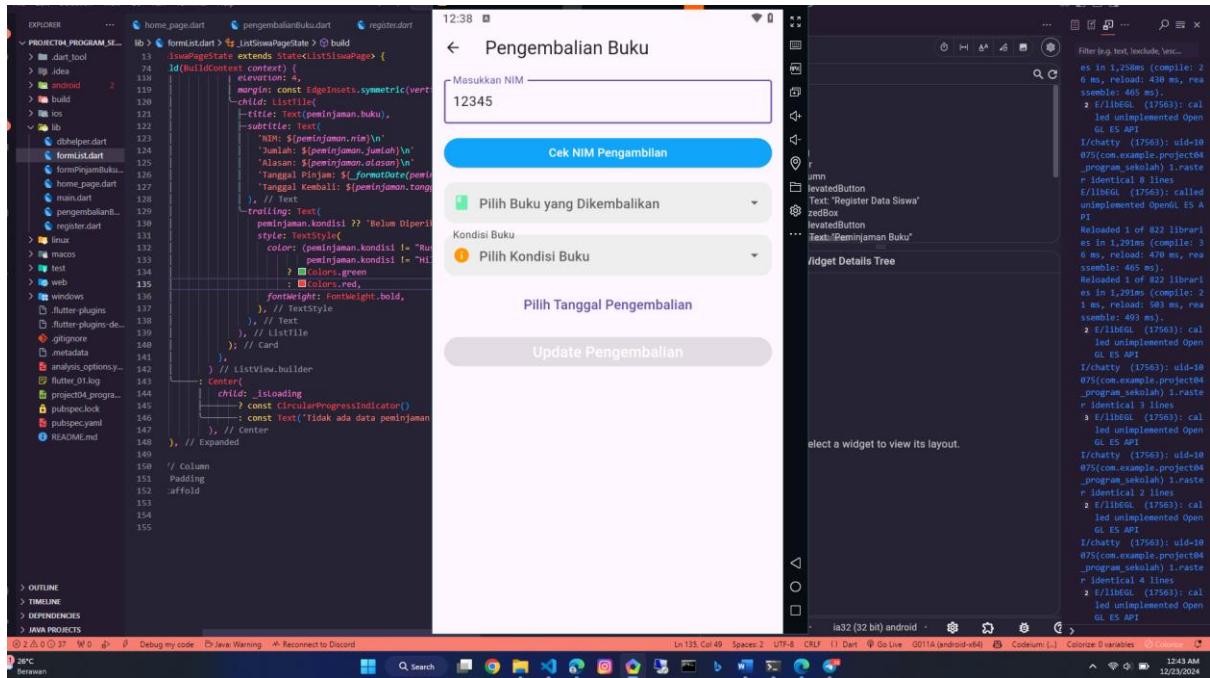
- Saya mengisi kembali dengan NIM yang sudah didaftarkan



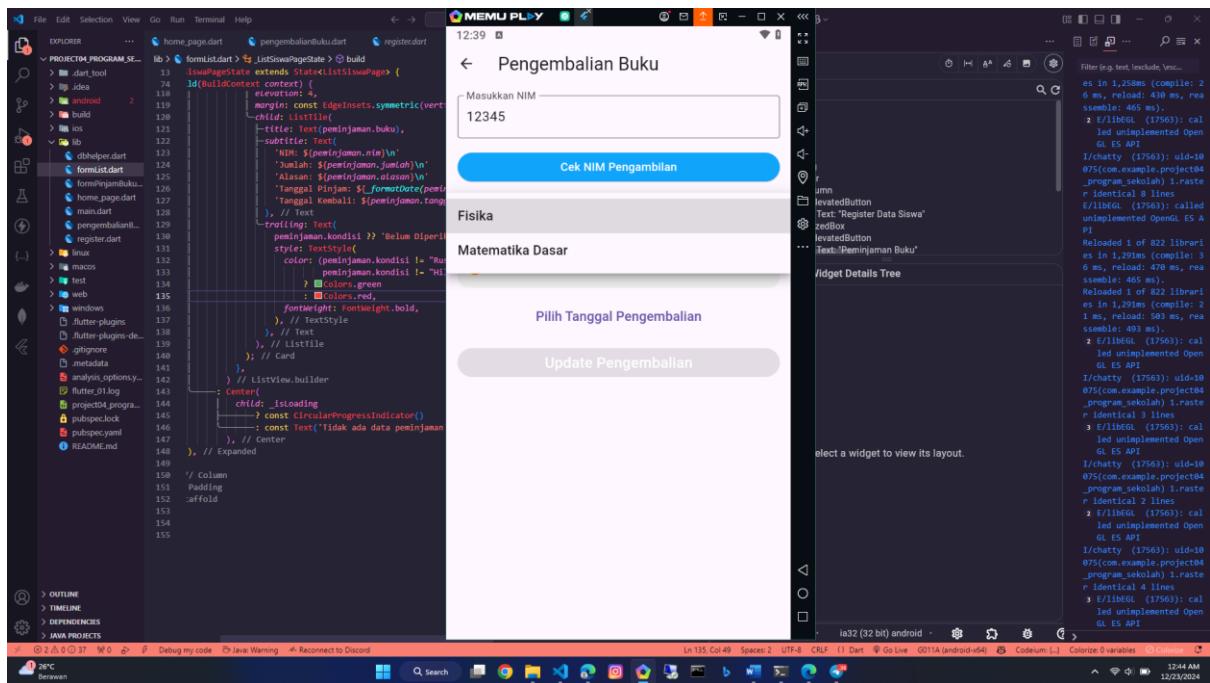
- Kembali ke Halaman Utama, lalu memilih *Form Pengembalian Buku* dan mengisi NIM yang tidak terdaftar/belum ada peminjaman buku



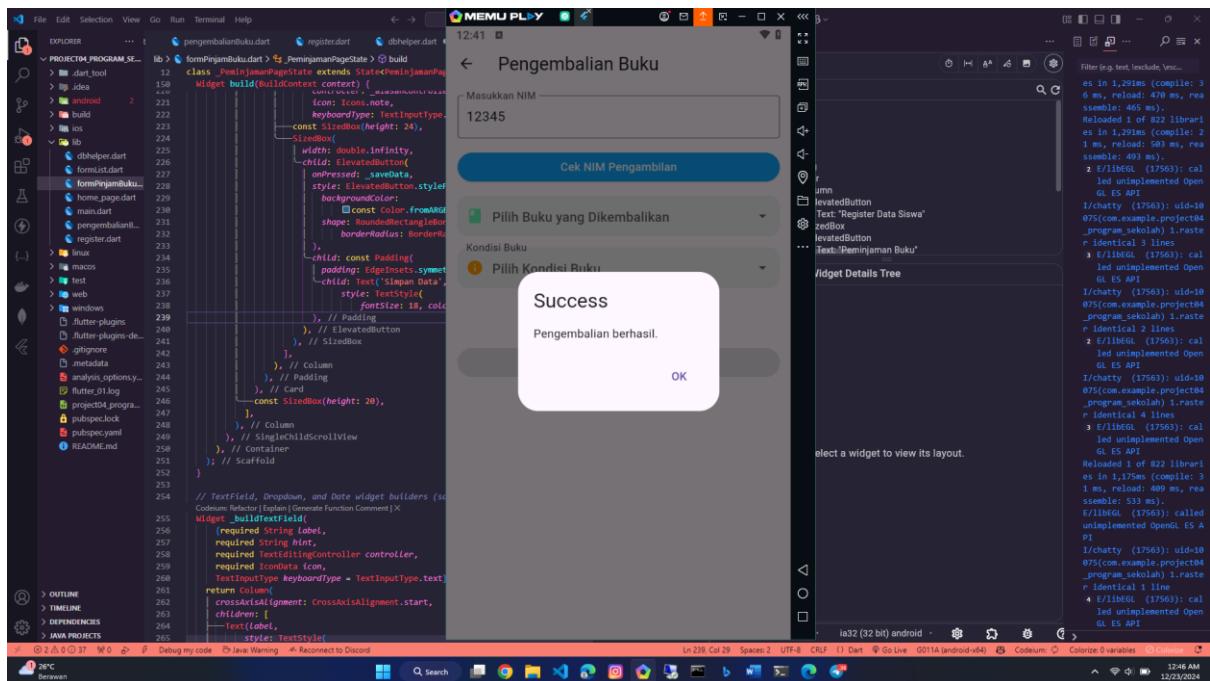
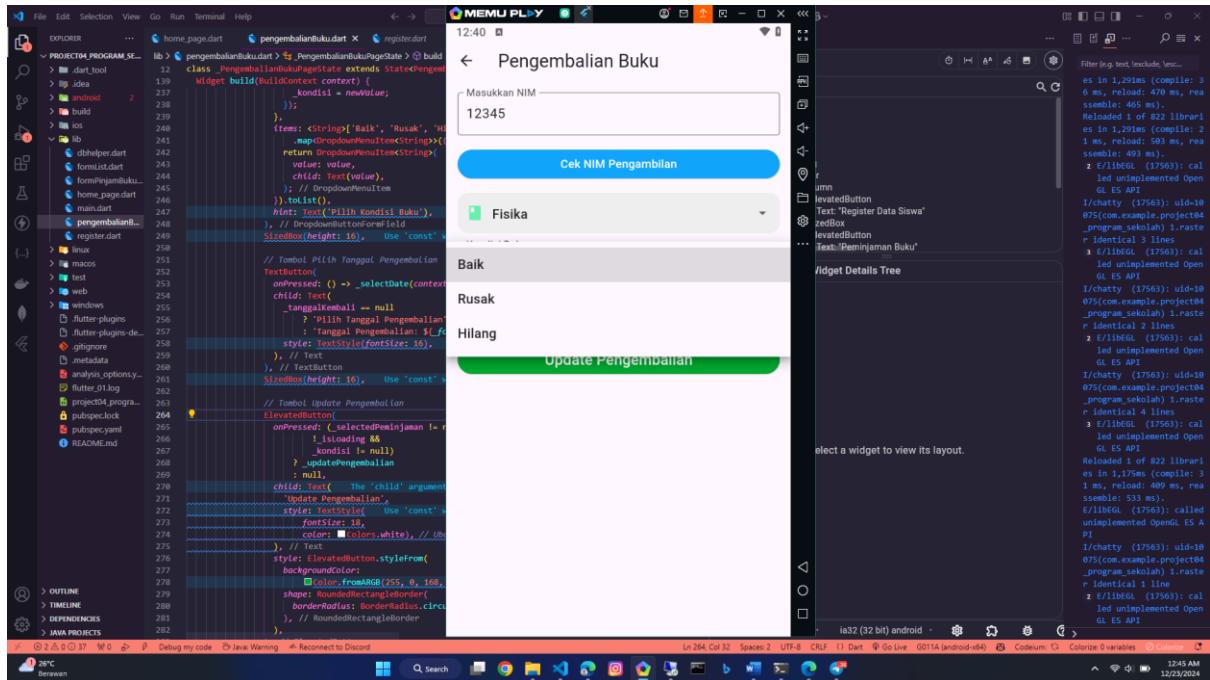
- Saya Kembali memasukan NIM yang sudah ada pinjaman buku



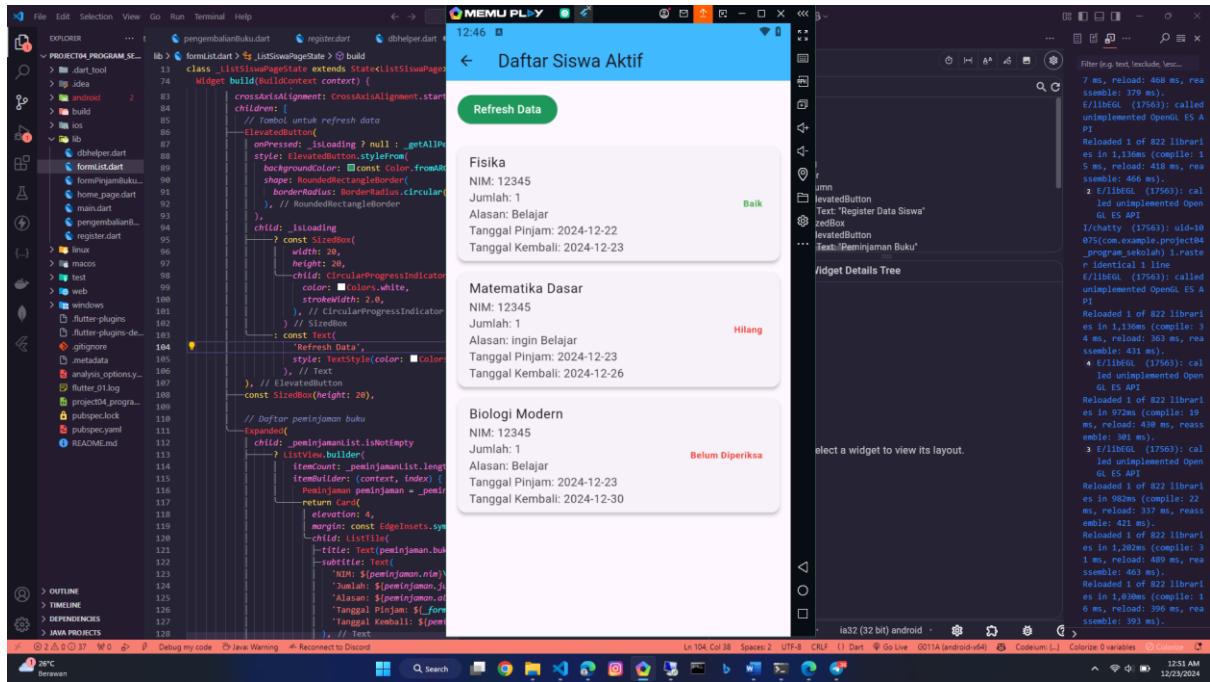
- Terlihat ada 2 Buku yang saya pinjam diperpustakaan sekolah.



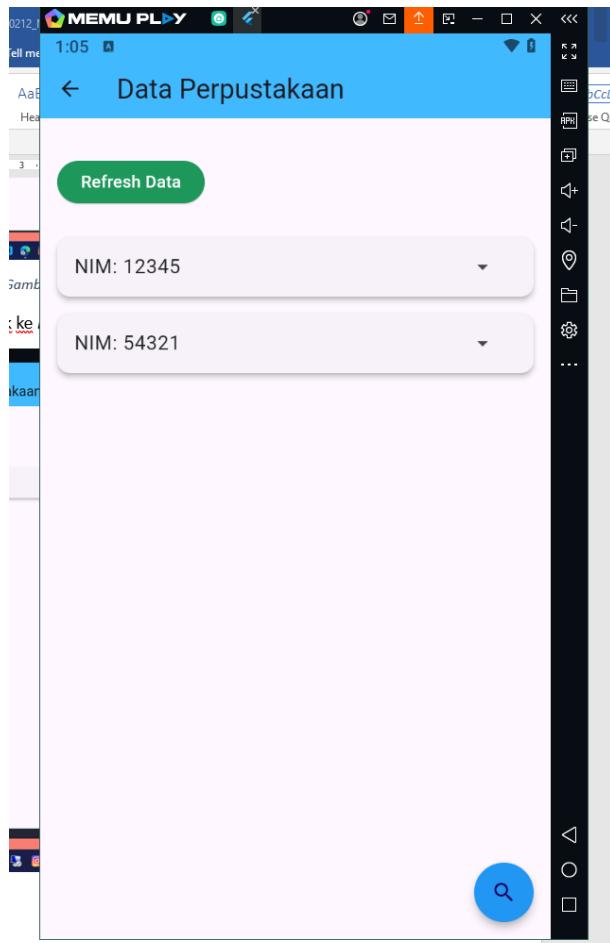
- Saya memilih salah satu buku yang ingin saya kembalikan ke perpustakaan, lalu saya input *Tanggal Pengembalian*, *Kondisi Buku* dan klik tombol *Update Pengembalian*



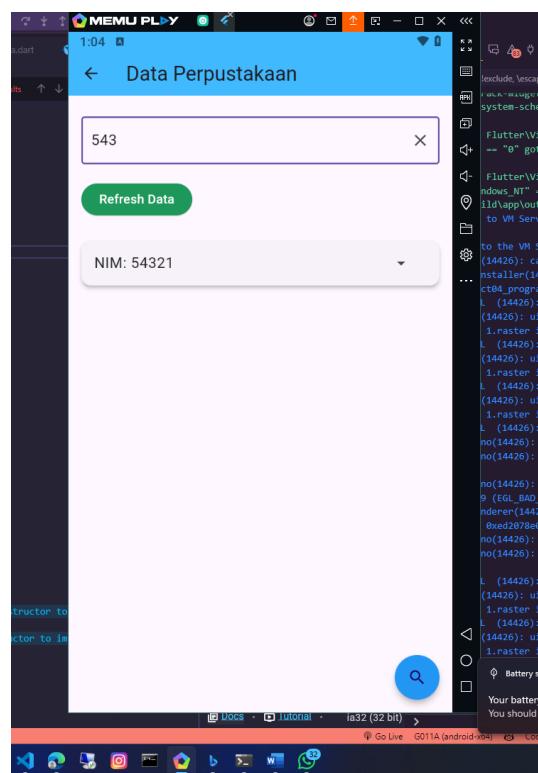
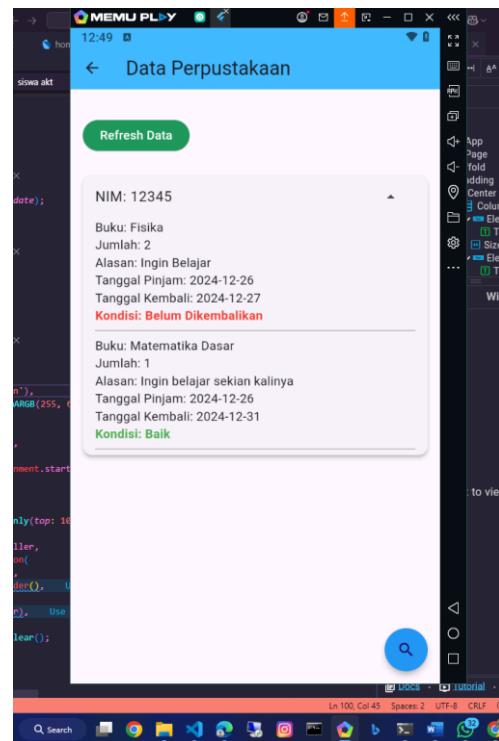
- Kembali ke Halaman Utama, saya akan memilih *Form Daftar Siswa Aktif*.



- Kembali ke halaman utama, lalu masuk ke *Data Perpustakaan*



- Jika saya klik user NIM tersebut, maka akan tampil seluruh kegiatan Siswa tersebut. Lalu saya akan mencoba search data sesuai NIM.



Code

The image shows two screenshots of a developer's workspace, likely from a Mac OS X environment, displaying code editors for Dart projects.

Screenshot 1 (Top): The code editor displays a Dart file named `daftar_siswa.dart`. The code implements a state for a list view builder, handling items and their removal. It uses various Dart libraries like `Text`, `TextStyle`, `Icon`, and `MaterialPageRoute`. The right side of the screen shows a detailed log of OpenGL ES API calls, indicating multiple reloads of libraries and numerous unimplemented API calls, such as `EGLGetConfig` and `EGLCreateWindowSurface`.

```

class _DaftarSiswaState extends State {
    ...
    Widget build(BuildContext context) {
        ...
        child: Text(
            'Tidak ada data siswa',
            style: TextStyle(color: Colors.grey[600], fontSize: 18),
        ), // Text
    } // Center
    itemCount: _siswaList.length,
    itemBuilder: (context, index) {
        ...
        child: _listItem(_siswaList[index]);
    }
}
return Card(
    elevation: 2,
    margin: const EdgeInsets.symmetric(
        vertical: 8, horizontal: 10), // EdgeInsets.symmetric
    shape: RoundedRectangleBorder(
        borderRadius: BorderRadius.circular(15),
    ), // RoundedRectangleBorder
    child: ListTile(
        contentPadding: const EdgeInsets.symmetric(
            horizontal: 20, vertical: 10), // EdgeInsets.symmetric
        leading: CircleAvatar(
            backgroundColor: Colors.blue[100],
            child: const Icon(Icons.person,
                color: Colors.blue), // Icon
        ), // CircleAvatar
        title: Text(
            siswa.name,
            style: const TextStyle(
                fontWeight: FontWeight.bold, fontSize: 16), // TextStyle
        ), // Title
        subtitle: Text(
            'NIM: ${siswa.nim}',
            style: TextStyle(color: Colors.grey[600]),
        ), // Text
        trailing: Row(
            mainAxisAlignment: MainAxisAlignment.end,
            children: [
                IconButton(
                    icon: const Icon(Icons.edit,
                        color: Colors.blue), // Icon
                    onPressed: () async {
                        await Navigator.of(context).push(
                            MaterialPageRoute(
                                builder: (context) =>
                                    EditSiswaPage(siswa: siswa),
                            ), // MaterialPageRoute
                        );
                    }
                )
            ],
        )
    )
)
}

```

Screenshot 2 (Bottom): The code editor displays a Dart file named `dhelper.dart`. This file contains logic for updating student data in a database. It includes functions for inserting new students and updating existing ones, handling conflicts with existing data. The code uses `Future<void>` for asynchronous operations and `Database` objects for database interactions.

```

if (siswa == null) {
    throw Exception(
        'NIM tidak terdaftar, Silahkan daftarkan siswa terlebih dahulu');
}

await db.insert(
    peminjaman,
    peminjaman.toMap(),
    conflictAlgorithm: ConflictAlgorithm.replace,
);
}

Future<void> updateSiswa(Siswa siswa) async {
    final db = await database;
    await db.update(
        'siswa',
        siswa.toMap(),
        where: 'nim = ?',
        whereArgs: [siswa.nim],
    );
}

// Update pengembalian buku
Future<void> updatePengembalian(Peminjaman peminjaman) async {
    final db = await database;
}

// Update tanggal kembali dan kondisi
await db.update(
    'peminjaman',
    {
        'tanggalKembali': peminjaman.tanggalKembali.toIso8601String(),
        'kondisi': peminjaman.kondisi,
    },
    where: 'nim = ? AND buku = ?',
    whereArgs: [peminjaman.nim, peminjaman.buku],
);

// Mengambil semua data peminjaman
Future<List<Peminjaman>> getAllPeminjaman() async {
    final db = await database;
    final List<Map<String, dynamic>> maps = await db.query('peminjaman');

    return List.generate(maps.length, (i) {
        return Peminjaman.fromMap(maps[i]);
    });
} // List.generate
}

```

Screenshot of two Android Studio code editors showing Dart code for 'edit_siswa.dart' and 'formlist.dart'.

edit_siswa.dart

```

class EditSiswaPageState extends State<EditSiswaPage> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: SingleChildScrollView(
        padding: const EdgeInsets.all(20.0),
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            const Text(
              'Informasi Siswa',
              style: TextStyle(fontSize: 20, fontWeight: FontWeight.bold),
            ),
            const SizedBox(height: 20),
            _buildTextField('NIM', _nimController, TextInputType.text, false),
            const SizedBox(height: 15),
            _buildTextField('Nama', _nameController, TextInputType.name, false),
            const SizedBox(height: 15),
            _buildDropdown('Kelas', _kelasList, _selectedKelas, newValue) {
              setState(() {
                _selectedKelas = newValue;
              });
            },
            const SizedBox(height: 15),
            _buildTextfield(
              'Alamat', _alamatController, TextInputType.streetAddress, true),
            const SizedBox(height: 15),
            _buildTextfield(
              'Telepon', _teleponController, TextInputType.phone, false),
            const SizedBox(height: 15),
            _buildTextfield(
              'Email', _emailController, TextInputType.emailAddress, false),
            const SizedBox(height: 15),
            _buildTextfield('Tanggal Lahir', _tanggallahirController,
              TextInputType.datetime, false),
            const SizedBox(height: 30),
            ElevatedButton(
              onPressed: _updateSiswa,
              child: const Text('Simpan Perubahan',
                style: TextStyle(fontSize: 16, color: Colors.white)),
            ), // Elevatedbutton
          ],
        ),
      ),
    );
  }
}

```

formlist.dart

```

class ListSiswaPage extends StatefulWidget {
  const ListSiswaPage({super.key});

  @override
  _ListSiswaPageState createState() => _ListSiswaPageState();
}

class _ListSiswaPageState extends State<ListSiswaPage> {
  Future<void> getAllPeminjaman() async {
    setState(() {
      _isLoading = true;
      _groupedPeminjaman = {};
      _filteredPeminjaman = {};
    });
    final Map<String, bool> _isExpanded = {};
    final TextEditingController _searchController = TextEditingController();
    final String _showSearch = false;

    try {
      List<Peminjaman> peminjaman = await BHelper().getAllPeminjaman();
      for (var peminjaman in peminjaman) {
        if (_groupedPeminjaman.containsKey(peminjamanItem.nim)) {
          _groupedPeminjaman[peminjamanItem.nim].add(peminjaman);
        } else {
          _groupedPeminjaman[peminjamanItem.nim] = Map<String, Peminjaman>();
          _groupedPeminjaman[peminjamanItem.nim].add(peminjaman);
        }
      }
      _filteredPeminjaman = Map.from(_groupedPeminjaman);
      setState(() {
        for (var nim in _groupedPeminjaman.keys) {
          _isExpanded[nim] = false;
        }
      });
    } catch (e) {
      showAlertDialog('Terjadi kesalahan: $e');
    } finally {
    }
  }
}

```

The image shows two side-by-side screenshots of the Android Studio IDE. Both screens display code editors with syntax highlighting and code navigation features.

Left Screen (Code Editor for 'formliststart'):

- File Path:** lib > formliststart > PenjemuanPage
- Code:** The code is a Flutter widget named PenjemuanPage that extends StatelessWidget. It contains logic for handling pinjam and kembali events, and a final list of strings representing book titles.
- Annotations:** Includes comments like // ignore: library_private_types_in_public_api and @override.
- Toolbars:** Standard Android Studio toolbars for file, edit, selection, view, go, run, terminal, help, and various icons for project navigation and build.

Right Screen (Code Editor for 'home_page.dart'):

- File Path:** lib > home_page.dart > HomeRoute
- Code:** The code defines a MaterialPageRoute builder for the home page. It includes logic for navigating between different screens based on the route parameters (id, title, and type).
- Annotations:** Includes comments like // Dafur Silsya Atif, // ElevatedButton, and // Dafur Silsua.
- Toolbars:** Standard Android Studio toolbars for file, edit, selection, view, go, run, terminal, help, and various icons for project navigation and build.

Screenshot 1 (Top): Screenshot of a Codeium IDE window showing the main.dart file of a Flutter project. The code defines the root widget as a MaterialApp with a HomePage. The terminal shows numerous OpenGL ES API calls, indicating a performance bottleneck.

```

import 'package:flutter/material.dart';
import 'package:project04_program_sekolah/home_page.dart';

void main() {
  runApp(const MyApp());
}

// This widget is the root of your application.
class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
  Widget build(BuildContext context) {
    return const MaterialApp(
      home: HomePage(),
      debugShowCheckedModeBanner: false,
    );
  }
}

```

Screenshot 2 (Bottom): Screenshot of a Codeium IDE window showing the pengembalianBuku.dart file of a Flutter project. The code implements a StatefulWidget for managing book loans. The terminal shows OpenGL ES API calls, similar to the first screenshot.

```

class PengembalianBukuPage extends StatefulWidget {
  const PengembalianBukuPage({super.key});

  @override
  // ignore: library_private_types_in_public_api
  _PengembalianBukuPageState createState() => _PengembalianBukuPageState();
}

class _PengembalianBukuPageState extends State<PengembalianBukuPage> {
  TextEditingController _nimController = TextEditingController();
  List<Peminjaman> _peminjamanList = [];
  Peminjaman? _selectedPeminjaman;
  bool _isLoading = false;
  String? _kondisi; // Variabel untuk menyimpan kondisi buku

  // Fungsi untuk mengambil daftar peminjaman berdasarkan NIM yang belum dikembalikan
  Future<void> fetchPeminjaman() async {
    final nim = _nimController.text.trim();
    if (nim.isEmpty) {
      _showAlertDialog('Silakan masukkan NIM terlebih dahulu.');
      return;
    }

    if (!int.tryParse(nim) == null) {
      _showAlertDialog('NIM harus berupa angka.');
      return;
    }

    setState(() {
      _isLoading = true;
      _peminjamanList = [];
      _selectedPeminjaman = null;
      _kondisi = null; // Reset kondisi saat fetch peminjaman baru
    });

    try {
      final peminjaman = await DBHelper().getPeminjamanByNim(nim);
      setState(() {
        _peminjamanList = peminjaman;
      });
    } catch (e) {
      print(e);
    }
  }
}

```